Learning with Structured Data: Applications to Computer Vision

vorgelegt von
Sebastian Nowozin, Dipl.-Inf. M.Eng.
aus Berlin

Von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation

Promotionsausschuß:
Vorsitzender: Prof. Dr. H. Ehrig
Berichter: Prof. Dr.-Ing. O. Hellwich
Berichter: Prof. Dr. B. Schölkopf

Tag der wissenschaftlichen Aussprache: 23.10.2009

Berlin 2009
D83

Sebastian Nowozin

Learning with Structured Data: Applications to Computer Vision

SELF-PUBLISHED BY THE AUTHOR

First printing, November 2009

Dedicated to my parents.

Contents

Abstract

In this thesis we address structured machine learning problems. Here "*structured*" refers to situations in which the input or output domain of a prediction function is non-vectorial. Instead, the input instance or the predicted value can be decomposed into parts that follow certain dependencies, relations and constraints. Throughout the thesis we will use hard computer vision tasks as a rich source of structured machine learning problems.

In the first part of the thesis we consider structure in the input domain. We develop a general framework based on the notion of *substructures*. The framework is broadly applicable and we show how to cast two computer vision problems — class-level object recognition and human action recognition — in terms of classifying structured input data. For the class-level object recognition problem we model images as labeled graphs that encode local appearance statistics at vertices and pairwise geometric relations at edges. Recognizing an object can then be posed within our substructure framework as finding discriminative matching subgraphs. For the recognition of human actions we apply a similar principle in that we model a video as a sequence of local motion information. Recognizing an action then becomes recognizing a matching subsequence within the larger video sequence. For both applications, our framework enables us to finding the discriminative substructures from training data. This first part contains as a main contribution a set of abstract algorithms for our framework to enable the construction of powerful classifiers for a large family of structured input domains.

The second part of the thesis addresses structure in the output domain of a prediction function. Specifically we consider image segmentation problems in which the produced segmentation must satisfy global properties such as connectivity. We develop a principled method to incorporate global interactions into computer vision random field models by means of linear programming relaxations. To further understand solutions produced by general linear programming relaxations we develop a tractable and novel concept of solution stability, where stability is quantified with respect to perturbations of the input data.

This second part of the thesis makes progress in modeling, solving and understanding solution properties of hard structured prediction problems arising in computer vision. In particular, we show how previously intractable models integrating global constraints with local evidence can be well approximated. We further show how these solutions can be understood in light of their stability properties.

Zusammenfassung

Die vorliegende Arbeit beschäftigt sich mit strukturierten Lernproblemen im Bereich des maschinellen Lernens. Hierbei bezieht sich "strukturiert" auf Prädiktionsfunktionen, deren Definitions- oder Zielmenge nicht wie sonst üblich in Vektorform dargestellt werden kann. Stattdessen kann die Eingabeinstanz oder der prädizierte Wert in Teile zerlegt werden, die gewissen Abhängigkeiten, Relationen und Nebenbedingungen genügen. Im Forschungsfeld der Computer Vision gibt es eine Vielzahl von strukturierten Lernproblemen, von denen wir einige im Rahmen dieser Dissertation diskutieren werden.

Im ersten Teil der Arbeit behandeln wir strukturierte Definitionsmengen. Basierend auf dem Konzept der *Unterstrukturen* entwickeln wir ein flexibel anwendbares Schema zur Konstruktion von Klassifikationsfunktionen und zeigen, wie zwei wichtige Probleme im Bereich der Computer Vision, das Objekterkennen auf Klassenebene und das Erkennen von Aktivitäten in Videodaten, darauf abgebildet werden können. Beim Objekterkennen modellieren wir Bilder als Graphen, deren Knoten lokale Bildmerkmale repräsentieren. Kanten in diesem Graphen kodieren Informationen über die paarweise Geometrie der adjazenten Bildmerkmale. Die Aufgabe der Objekterkennung lässt sich in diesem Schema auf das Auffinden diskriminativer Untergraphen reduzieren. Diesem Prinzip folgend können auch Videos als Sequenz zeitlich und räumlich lokaler Bewegungsinformationen modelliert werden. Das Erkennen von Aktivitäten in Videos kann somit analog zu den Graphen auf das Auffinden von passenden Untersequenzen reduziert werden. In beiden Anwendungen ermöglicht unser Schema die Identifikation einer geeigneten Menge von diskriminativen Unterstrukturen anhand eines gegebenen Trainingsdatensatzes.

In diesem ersten Teil besteht der Forschungsbeitrag aus unserem Schema und passenden abstrakten Algorithmen, die es ermöglichen, leistungsfähige Klassifikatoren für strukturierte Eingabemengen zu konstruieren.

Im zweiten Teil der Arbeit diskutieren wir Lernprobleme mit strukturierten Zielmengen. Im Speziellen behandeln wir Bildsegmentierungsprobleme, bei denen die prädizierte Segmentierung globalen Nebenbedingungen, zum Beispiel Verbundenheit klassengleicher Pixel, genügen muss. Wir entwickeln eine allgemeine Methode, diese Klasse von globalen Interaktionen in *Markov Random Field* (MRF) Modelle der Computer Vision mit Hilfe von linearer Programmierung und Relaxationen zu integrieren. Um diese Relaxationen besser zu verstehen sowie Aussagen über die prädizierten Lösungen machen zu können, entwickeln wir ein neuartiges Konzept der Lösungsstabilität unter

Störungen der Eingabedaten.

Der Hauptbeitrag zum Forschungsfeld dieses zweiten Teils liegt in der Modellierung, den Lösungsalgorithmen und der Analyse der Lösungen komplexer strukturierter Lernprobleme im Feld der Computer Vision. Im Speziellen zeigen wir die Approximierbarkeit von Modellen, die sowohl globale Nebenbedingungen als auch lokale Evidenz berücksichtigen. Zudem zeigen wir erstmals, wie die Lösungen dieser Modelle mit Hilfe ihrer Stabilitätseigenschaften verstanden werden können.

Acknowledgements

This thesis would have been impossible without the help of many. First of all, I would like to thank Bernhard Schölkopf, for allowing me to pursue my PhD at his department. His great leadership sustains a wonderful research environment and carrying out my PhD studies in his department has been a great pleasure. I am grateful to Olaf Hellwich for agreeing to review my work and for his continuing support.

I especially thank Gökhan Bakır for convincing me to start my PhD studies. I am deeply grateful for his constant encouragement and advice during my first and second year. I thank Koji Tsuda for his advice and mentoring, and for fruitful research cooperation together with Hiroto Saigo. Peter Gehler deserves special thanks for taking the successful lead on many joint projects. I would like to express my deepest gratitude to Christoph Lampert, head of the Computer Vision group. He always had an ear to listen to even the most wackiest idea and provided the honest critical feedback that is so necessary for success. His guidance made every member of the MPI computer vision group a better researcher. Both Christoph and Peter read early versions of this thesis; their input has improved the thesis significantly. I would like to thank Stefanie Jegelka for all the effort she put in our research project.

My PhD studies were funded by the EU project CLASS (IST 027978).

Open discussions, honest and critical feedback are essential for sorting out the few good ideas from the many. I thank all my colleagues for this; I thank Matthias Hein, Matthias Franz, Kwang In Kim, Matthias Seeger, Mingrui Wu, Olivier Chapelle, Stefan Harmeling, Ulrike von Luxburg, Arthur Gretton, Joris Mooij, Jeff Bilmes and Yasemin Altun. Especially I would like to thank Suvrit Sra for his feedback and for asking me to jointly organize a workshop. For their support in all technical and organizational issues I would like to thank Sebastian Stark and Sabrina Nielebock. I thank Jacquelyn Shelton for proofreading my thesis and Agnes Radl for improvements to the introduction.

My fellow PhD students have been a rich source of motivation and I thank all of them. In particular I thank Wolf Kienzle, Matthew Blaschko, Frank Jäkel, Florian Steinke, Hannes Nickisch, Michael Hirsch, Markus Maier, Christian Walder, Sebastian Gerwinn, Jakob Macke and Fabian Sinz.

The support of my family motivated me during my studies. I dedicate my thesis to my parents, for their love and for fostering all my academic endeavors; I thank my brothers Benjamin and Tobias for their support.

Most important of all, I thank my wife Juan Gao. Her love, encouragement and tolerance made possible everything. Thank you.

Papers included in the Thesis

The following publications are included in part or in an extended form in this thesis.

- Sebastian Nowozin, Koji Tsuda, Takeaki Uno, Taku Kudo and Gökhan Bakır, "Weighted Substructure Mining for Image Analysis", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*.
- Sebastian Nowozin, Gökhan Bakır and Koji Tsuda, "Discriminative Subsequence Mining for Action Classification", *IEEE Computer Society International Conference on Computer Vision (ICCV 2007)*.
- Hiroto Saigo, Sebastian Nowozin, Tadashi Kadowaki, Taku Kudo and Koji Tsuda, "gBoost: A Mathematical Programming Approach to Graph Classification and Regression", *Machine Learning Journal, Springer, Volume 75, Number 1, 2009, pages 69–89*.
- Sebastian Nowozin and Christoph H. Lampert, "Global Connectivity Potentials for Random Field Models", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*.
- Sebastian Nowozin and Stefanie Jegelka, "Solution Stability in Linear Programming Relaxations: Graph Partitioning and Unsupervised Learning", *26th Annual International Conference on Machine Learning (ICML 2009)*.
- Sebastian Nowozin and Christoph Lampert, "Global Interactions in Random Field Models: A Potential Function Ensuring Connectedness", submitted, *SIAM Journal on Imaging Sciences*.

Papers not included in the Thesis

The following publications are outside the scope of the thesis but have been part of my PhD research.

- Sebastian Nowozin and Gökhan Bakır, "A Decoupled Approach to Exemplar-based Unsupervised Learning", *25th International Conference on Machine Learning (ICML 2008)*.
- Paramveer S. Dhillon, Sebastian Nowozin and Christoph H. Lampert, "Combining Appearance and Motion for Human Action Classification in Videos", *Max Planck Institute for Biological Cybernetics Techreport TR-174*.

- Sebastian Nowozin and Koji Tsuda, "Frequent Subgraph Retrieval in Geometric Graph Databases", *IEEE International Conference on Data Mining (ICDM 2008)*.
- Sebastian Nowozin and Koji Tsuda, "Frequent Subgraph Retrieval in Geometric Graph Databases", *Max Planck Institute for Biological Cybernetics Techreport TR-180*, extended version of the ICDM 2008 paper.
- Peter Gehler and Sebastian Nowozin, "Infinite Kernel Learning", *Max Planck Institute for Biological Cybernetics Techreport TR-178*.
- Peter Gehler and Sebastian Nowozin, "Let the Kernel Figure it Out; Principled Learning of Pre-processing for Kernel Classifiers", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*.
- Paramveer S. Dhillon, Sebastian Nowozin, and Christoph Lampert, "Combining Appearance and Motion for Human Action Classification in Videos", *1st International Workshop on Visual Scene Understanding (ViSU 09)*.
- Peter Gehler and Sebastian Nowozin, "On Feature Combination Methods for Multiclass Object Classification", *IEEE International Conference on Computer Vision (ICCV 2009)*.

Introduction

> Beware of the man of one method or one instrument, either experimental or theoretical. He tends to become method-oriented rather than problem oriented. The method-oriented man is shackled: the problem-oriented man is at least reaching freely toward what is most important.
>
> John R. Platt (1963)

Overview

Throughout this thesis we address *structured* machine learning problems. In supervised machine learning we learn a mapping $f : \mathcal{X} \to \mathcal{Y}$ from an input domain $\mathcal{X}$ to an output domain $\mathcal{Y}$ by means of a given set of training data $\{(x_i, y_i)\}_{i=1,\ldots,N}$, with $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. A typical well-known setting is binary classification where we have $\mathcal{Y} = \{-1, 1\}$.

In *structured machine learning* the domain $\mathcal{X}$ or $\mathcal{Y}$, or both, have associated with it a non-trivial formalizable structure. For example, $\mathcal{X}$ might be a combinatorial set such as "the set of all English sentences", or "the set of all natural images". Clearly, being able to learn a function taking as input such objects and making meaningful predictions is highly desirable.

When the structure is in the output domain $\mathcal{Y}$, the problem of learning f is often referred to as *structured prediction* or *structured output learning*. A typical example of a structured output domain $\mathcal{Y}$ is in image segmentation, where each pixel of an image must be labeled with a class such as "person" or "background" and $\mathcal{Y}$ therefore is the "set of all possible image segmentations". Because the label decisions are not independent across the pixels, the dependencies in $\mathcal{Y}$ should be modeled by imposing further structure on $\mathcal{Y}$.

In this thesis we address the challenging problem of learning f. Furthermore, we will use computer vision problems to demonstrate the applicability of our developed methods.

1. Substructure poset framework

Our key contributions in this direction are threefold. *First*, we propose a novel framework for structured input learning that we call the "substructure poset framework". The proposed framework applies to a broad class of input domains $\mathcal{X}$ for which a natural generalization of the subset relation exists, such

as for sets, trees, sequences and general graphs. *Second*, for structured prediction we discuss Markov random field models with global non-decomposable potential functions. We propose a novel method to efficiently evaluate f in this setting by means of constructing linear programming relaxations. *Third*, we develop a novel method to quantify the *solution stability* in general linear programming relaxations to combinatorial optimization problems, such as the ones arising from structured prediction problems.

In the remainder of this introduction we describe in more detail the two main parts of this thesis.

Part I: Learning with Structured Input Data

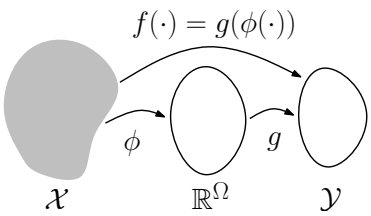


Figure 1: Schematic illustration of $f : \mathcal{X} \to \mathcal{Y}$ as composition $g(\phi(\cdot))$.

The first part of this thesis addresses the input domain $\mathcal{X}$ in learning $f : \mathcal{X} \to \mathcal{Y}$. When $\mathcal{X}$ consists of non-vectorial data it is not obvious how f can be constructed. In general, computers are limited to process numbers and we can therefore reduce the problem of learning f into two steps. *First*, a set of suitable statistics $\phi = \{\phi_\omega : \mathcal{X} \to \mathbb{R} | \omega \in \Omega\}$ has to be defined over a domain Ω. *Second*, the statistics $\phi : \mathcal{X} \to \mathbb{R}^\Omega$ serve as a proxy to reason about the true input domain $\mathcal{X}$, such that f can now be defined as $f(x) = g(\phi(x))$ for some function $g : \mathbb{R}^\Omega \to \mathcal{Y}$. This construction is illustrated in Figure 1.

This set of accessible statistics is the *feature space* or *feature map*, a single statistic is also called *feature*.

In the first chapter we review two existing approaches, *propositionalization* and *kernels*, for solving the problem of learning with structured input domains.

We argue in favor of rich feature spaces that preserve most of the information from the structured domain. Learning a linear classifier $f : \mathcal{X} \to \{-1, 1\}$ using such feature space consists of assigning a weight to each feature. Because the dimension of the feature space can be very large, we either need an *aggregated* representation of the weights or use *sparse* linear classifiers that assign a non-zero weight to only a small number of features.

Kernel methods represent the weight vector implicitly within the span of the feature vectors of the training instances. They can therefore use a rich feature space at the cost of an implicit representation of the classification function.

In contrast, *Boosting* can achieve *sparse* weight vectors. Each feature is treated as a "weak learner" and the classification function optimally combines a small set of weak learners in order to minimize a loss function on the training set predictions. Because we will use Boosting extensively in later chapters we describe a general Boosting algorithm in detail in the first chapter.

In the second chapter we introduce our novel framework to define feature spaces for structured input domains which we call *substructure poset framework*.

Within the framework, we consider statistics of the form

$$\phi_t : \mathcal{X} \to \{0,1\}, \qquad \phi_t(x) = \begin{cases} 1 & \text{if } t \subseteq x \\ 0 & \text{otherwise} \end{cases},$$

for $t \in \mathcal{X}$, i.e., we have $\Omega = \mathcal{X}$. The only necessary assumption for this construction to work is the existence of a natural *partial order*, the *substructure relation* $\subseteq: \mathcal{X} \times \mathcal{X} \to \{\top, \bot\}$ relating pairs of structures. Such a relation exists naturally for sets, but we show how to define suitable relations for other structured domains such as graphs and sequences.

This *substructure-induced feature space* has several nice properties which we analyze in detail. For one, the features *preserve* all information about a structure, essentially because $\phi_x(x) = 1$ holds. Additionally, linear classifiers within this feature space have an infinite VC-dimension, that is, any given pair of finite sets $S, T \subseteq \mathcal{X}$ with $S \cap T = \emptyset$ can be strictly separated by means of a function that is linear in the features.

To enable the learning of linear classifiers we show how the Boosting algorithm introduced in the first chapter can be applied in this feature space. In particular, we describe an algorithm to solve the Boosting subproblem of finding the best weak learner within the substructure poset framework.

In the third and fourth chapter of the first part, we demonstrate the versatility of the substructure poset framework by applying it to computer vision problems.

In the third chapter we address the problem of incorporating geometry information into bag-of-words models for class-level object recognition systems. In class-level object recognition we are given a natural image and have to determine whether an object of a known class — such as "bird", "car", or "person" — is present in the image. During training time we have access to a large collection of annotated natural images. The goal of solving class-level object recognition problems is important on its own for the purpose of indexing and sorting images by the objects shown on them. But it is also a fundamental building block to the larger goal of *visual scene understanding*, that is, to be able to semantically reason about an entire scene depicted on an image.

One popular family of approaches to the class-level object recognition problem are bag-of-words models that summarize local image information in a bag. Each element in the bag represents a match of local appearance information to a specific template from a larger template pattern set. The matches are unordered in the sense that they can happen anywhere in the image. Surprisingly, classifiers built on top of this simple representation perform well for the class-level object problem.

The bag-of-words representation is robust, but it discards a large amount of information contained in the *geometry* between local appearance matches. Therefore, in computer vision an alternative line of models that explicitly model the geometric relationships between parts has been pursued. In the

third chapter we provide an in-depth literature survey of these *part-based models*.

The remaining part of the third chapter then demonstrates how our substructure poset framework can be applied to the problem of modeling pairwise geometry between local appearance information. We evaluate the proposed model on the PASCAL VOC 2008 data set, a difficult benchmark data set for object class-level recognition.

In the fourth chapter of the first part we apply the substructure poset framework to human activity recognition in video data. Recognizing and understanding human activities is an important problem because its solution enables monitoring, indexing, and searching of video data by its semantic content.

For activity recognition bag-of-words models are again popular but they discard the temporal ordering of local motion information. We first survey the literature on human activity recognition, distinguishing the main families of approaches. We then proceed to show that by using sequences as structures in the substructure poset framework we can preserve the temporal ordering relation between local motion cues. Through the addition of a robust subsequence relation inducing a subsequence-based feature space we can learn a classifier to recognize human motions that uses the temporal ordering information.

The chapter ends with a benchmark evaluation and discussion of the approach on the popular KTH human activity recognition dataset.

The main novelty in this first part is the principled development of a framework for structured input learning. The last two chapters further fill this framework with life and show how it can be applied to graphs and sequences.

Part II: Structured Prediction

The second part of this thesis is concerned with structured prediction models and consists of three chapters. In order to build a structured prediction model $f : \mathcal{X} \to \mathcal{Y}$ one needs to formalize the notion of structure in $\mathcal{Y}$ and thus make clear the assumptions that are part of the model. In the first chapter we survey the literature of structured prediction models with a focus on *undirected graphical models* and their application to computer vision problems.

Undirected graphical models — also known as Markov networks — make explicit a set of *conditional independence assumptions* by means of a graph having as vertices the set of input and output variables. Groups of edges linking vertices encode local *interactions* between variables. We discuss in detail the currently popular models together with training and inference procedures.

In some applications of these models there are additional solution properties that depend jointly on the state of *all* variables in the model. We consider one example in the second chapter of this part, where the global property

is a topological invariant stating that all vertices which share a common label must form a connected component in the graph. This constraint on the solution does not decompose and incorporating it into a Markov network is unnatural: the graph would become complete and the usual training and inference algorithms no longer remain tractable.

We overcome this difficulty by directly formulating a *linear programming relaxation* to the maximum a posteriori estimation problem of this model. The key observation we make is that global interactions can naturally be incorporated by techniques from the field of *polyhedral combinatorics*: approximating the convex hull of all feasible solution points. Our construction allows us to obtain polynomial-time solvable *relaxations* to the original problem. This in turn enables efficient learning and estimation procedures; however, we lose the probabilistic interpretation of the model and can no longer compute quantities such as marginal probabilities.

In the last chapter of this part we propose solution stability as a non-probabilistic alternative to describe properties of the predicted solution. Intuitively, a solution that is *stable* under perturbations of the input data is preferable over an unstable solution. We formalize the concept of solution stability for the case of linear programming relaxations and propose a general novel method to compute the stability.

Unlike the probabilistic setting where computing marginals might be more difficult than computing a MAP estimate, our method is always applicable when the canonical MAP estimation problem can be solved. Again we make extensive use of linear programming relaxations to combinatorial optimization problems. For such linear programming relaxations we prove that our method is conservative and never overestimates the true solution stability in the unrelaxed problem.

The second part presents in the first chapter a survey of the known literature, and the *novel contributions* are in the second and third chapters.

PART I

Learning with Structured Input Data

> The combination of some data and an aching desire for an answer does not ensure that a reasonable answer can be extracted from a given body of data.
>
> John Wilder Tukey

Introduction

In many application domains the data is non-vectorial but *structured*: a data item is described by parts and relations between parts, where the description obeys some underlying rules. For example, a natural language document has a linear order of sections, paragraphs, and sentences and these parts decompose hierarchically from the entire document down to single words or even characters. Another example of structured data are chemical compounds, typically modeled as graphs consisting of atoms as vertices and bonds as edges, relating two or more atoms. One consequence of structured input data is that the usual techniques for classifying numerical data are not directly applicable.

In this chapter we first give a brief overview of approaches to classification of structured input data. Then we provide an introduction to Boosting, as a prequisite to the following chapter. Our viewpoint on Boosting is particularly simple and general, avoiding many of the drawbacks of early Boosting algorithms.

Approaches to Structured Input Classification

We now discuss two general approaches to handle structured input data. These are propositionalization and kernel methods.

Propositionalization

The simplest and traditionally popular method to handle structured input data is by first transforming it into a numerical feature vector, a step called *propositionalization*[1]. As a popular example, documents are often transformed into sparse *bag-of-words* vectors, encoding the presence of all words in the document[2]. Another example is in chemical compound classification and

[1] Stefan Kramer, Nada Lavrac, and Peter Flach. Propositionalization approaches to relational data mining. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 262–291. Springer, September 2001. ISBN 3-540-42289-7

[2] Thorsten Joachims. *Learning to Classify Text using Support Vector Machines*. Kluwer Academic Publishers, 2002

quantitative structure-activity relationship analysis, where for a given molecule certain derived properties such as their electrostatic fields are estimated using models possessing domain knowledge[3].

[3] Huixiao Hong, Hong Fang, Qian Xie, Roger Perkins, Daniel M. Sheehan, and Weida Tong. Comparative molecular field analysis (comfa) model using a large diverse set of natural, synthetic and environmental chemicals for binding to the androgen receptor. *SAR QSAR Environmental Research*, 14(5-6):373–388, 2003

Propositionalization can be an effective approach if sufficient domain knowledge suggests a small set of discriminative features relevant to the task. However, in general there are two main drawbacks to propositionalization.

First, because the features are generated explicitly, we are limited to using a small set of features. Usually, this results in an *information loss* as more than one element from $\mathcal{X}$ is mapped to the same feature vector, i.e., the feature mapping is non-injective. This can be seen, for example, in the bag-of-words model: a document can always be mapped uniquely to its bag-of-words representation, but given a bag-of-words vector it is not possible to recover the document because the ordering between words has been lost. Therefore, using a small number of features can limit the capacity of the function class in the original input domain $\mathcal{X}$ when a classifier is applied to the propositionalized data.

Second, the design of suitable features that are both informative and discriminative can be difficult. Within the same application domain there might be different tasks, each requiring its own set of features for the same input domain $\mathcal{X}$. Even to the domain expert it might not be a priori clear which features can be expected to work best.

In summary, the success of an approach based on propositionalization depends very much on the application domain, task, and on the existing domain knowledge. In the best case, the derived numerical features are well suited to the task and all relevant information important for obtaining good predictive performance is preserved. In the worst case, the resulting numerical feature vectors do not contain the discriminative information present in the original input representation.

Kernels for Structured Input Data

Structured input data can be incorporated into kernel classifiers in a straightforward way. In kernel classifiers a function $f : \mathcal{X} \to \mathcal{Y}$ is learned by accessing each instance exclusively through a *kernel function* $k : \mathcal{X} \times \mathcal{X} \to \mathcal{Y}$. Informally the kernel function can be thought of as measuring similarity between two instances. The use of a kernel function has a far-reaching consequence: it *separates* the algorithm from the representation of the input domain[4]. Therefore, when changing the structured input domain $\mathcal{X}$, we do not need to change the classification algorithm but only provide a new suitable kernel function.

[4] Bernhard Schölkopf and Alexander J. Smola. *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001

First of all, a suitable kernel function needs to be a *valid* kernel. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a valid kernel if and only if it corresponds to an inner product in some Hilbert space $\mathcal{H}$. This condition is equivalent to the existence of a feature map $\phi : \mathcal{X} \to \mathcal{H}$, such that $k(x, x') = \langle \phi(x), \phi(x') \rangle$ for all $x, x' \in \mathcal{H}$. The existence of a feature map is guaranteed if k is a positive definite function[5].

[5] Nachman Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950

Beyond being valid, a "good kernel" considers all information contained in an instance by having an injective feature map ϕ. Such kernel is said to be *complete* and satisfies $(k(x,\cdot) = k(x',\cdot)) \Rightarrow x = x'$ for all $x, x' \in \mathcal{X}$. Gärtner[6] further defines two properties a good kernel should have — *correctness* and *appropriateness* — but these already depend on the specific function class used by the classifier and we therefore do not discuss them here.

[6] Thomas Gärtner. A survey of kernels for structured data. *SIGKDD Explorations*, 5(1):49–58, 2003

In the following we briefly discuss three popular approaches to derive kernels for structured input domains: Fisher kernels, marginalized kernels, and convolution kernels. For a more in-depth survey, see Gärtner[7].

[7] Thomas Gärtner. A survey of kernels for structured data. *SIGKDD Explorations*, 5(1):49–58, 2003

FISHER KERNELS, proposed by Jaakkola and Haussler[8], are based on a generative parametric model of the data. Suppose that for the input domain $\mathcal{X}$ we have a model $p(X|\theta)$ with parameters $\theta \in \mathbb{R}^d$. The model could for example be learned from a large unsupervised training set. Markov networks such as Hidden Markov Models (HMM) are another popular example.

[8] Tommi S. Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *NIPS*. 1999

Given a single instance $x \in \mathcal{X}$, the so called *Fisher score* of the example is defined to be the gradient of the log-likelihood function of the model,

$$U_x = \nabla_\theta \log p(X = x|\theta),$$

with $U_x \in \mathbb{R}^d$. The expectation of the outer product of U_x over $\mathcal{X}$ is the *Fisher information matrix*,

$$I(\theta) = \mathbb{E}_{x\sim p(x|\theta)}\left[U_x U_x^\top\right],$$

so that $(I(\theta))_{i,j} = \mathbb{E}_{x\sim p(x|\theta)}[\frac{\partial}{\partial\theta_i}\log p(x|\theta)\frac{\partial}{\partial\theta_j}\log p(x|\theta)]$. Jaakkola and Haussler define the Fisher kernel $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ as proportional to

$$k(x,x') \propto U_x^\top I(\theta)^{-1} U_{x'}. \tag{1}$$

In the limit of maximum likelihood estimated models $p(x|\theta)$ we have asymptotic normality of $I(\theta)$ and therefore can approximate (1) as

$$k(x,x') \propto U_x^\top U_x.$$

The function defined in (1) can be shown to always be a valid kernel, to be invariant under invertible transformations of the parameter space θ, and to be a *good* kernel in the sense that if $p(x|\theta) = \sum_{y\in\mathcal{Y}} p(x,y|\theta)$ has a latent variable Y denoting a class label, then a kernel-based classifier with kernel (1) will asymptotically be at least as good as the maximum a posteriori estimate $y^* = \text{argmax}_{y\in\mathcal{Y}} p(x,y|\theta)$ for a given x.

In summary, for structured input domains $\mathcal{X}$ where there exist generative models, the Fisher kernel is an elegant method to reuse the model in a discriminative kernel classifier.

MARGINALIZED KERNELS, proposed by by Tsuda et al.[9], generalize the Fisher kernels considerably. The idea of marginalized kernels is the following. Let

[9] Koji Tsuda, Taishin Kin, and Kiyoshi Asai. Marginalized kernels for biological sequences. In *ISMB*, pages 268–275, 2002

each instance be composed as $z = (x, y) \in \mathcal{X} \times \mathcal{Y}$, where x is an observed part and y corresponds to a latent part that is never observed during training and testing. If we would fully observe (x, y), we could define a *joint kernel* $k_z : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R}$ on both parts. Marginalized kernels now assume that we have a model $p(y|x)$ relating the observed to the latent variables. Using this model, the *marginalized kernel* $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is defined as

$$\begin{aligned} k(x, x') &= \sum_{y \in \mathcal{Y}} \sum_{y' \in \mathcal{Y}} p(y|x) p(y'|x') k_z((x, y), (x', y')) \\ &= \mathbb{E}_{y \sim p(y|x)} \mathbb{E}_{y' \sim p(y'|x')} \left[k_z((x, y), (x', y')) \right]. \end{aligned} \tag{2}$$

The marginalized kernel (2) is a strict generalization of the Fisher kernel (1). This can be seen by taking the joint kernel to be

$$k_z((x, y), (x', y')) = \nabla_\theta \log p(x, y|\theta)^\top I(\theta)^{-1} \nabla_\theta \log p(x', y'|\theta)$$

and using the identity

$$\nabla_\theta \log p(x|\theta) = \sum_{y \in \mathcal{Y}} p(y|x, \theta) \nabla_\theta \log p(x, y|\theta)$$

to obtain by (2)

$$\begin{aligned} k(x, x') &= \sum_{y \in \mathcal{Y}} \sum_{y' \in \mathcal{Y}} p(y|x) p(y'|x') \nabla_\theta \log p(x, y|\theta)^\top I(\theta)^{-1} \nabla_\theta \log p(x', y'|\theta) \\ &= \nabla_\theta \log p(x|\theta)^\top I(\theta)^{-1} \nabla_\theta \log p(x'|\theta) \\ &= U_x^\top I(\theta)^{-1} U_{x'}, \end{aligned}$$

which is precisely the original Fisher kernel (1).

In contrast with the Fisher kernel, the marginalized kernel separates the joint kernel from the probabilistic model, making the *design* of kernels for structured data easier.

One example of the flexibility gained by the marginalized kernel formulation is exhibited by Kashima et al.[10], who defined a marginalized kernel for labeled graphs. They achieve this by letting the hidden domain $\mathcal{Y}$ correspond to the set of all random walks in the graph. For this choice of $\mathcal{Y}$ a simple closed form solution exists for $p(y|x)$. The joint kernel compares the ordered labels for a given pair of paths y and y'. Due to the closed form distribution of random walks on a graph, the computation of (2) is tractable.

Kernels for graphs have been further analyzed and generalized in Ramon and Gärtner[11], where it was shown that the marginalized graph kernel of Kashima is not complete and that any complete graph kernel is necessarily NP-hard to compute.

CONVOLUTION KERNELS, proposed by Haussler[12], are a general class of kernels applicable when the instances can be decomposed into a fixed number of parts that can be compared with each other in a meaningful way.

[10] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Marginalized kernels between labeled graphs. In *ICML*, 2003

[11] Jan Ramon and Thomas Gärtner. Expressivity versus efficiency of graph kernels. In *First International Workshop on Mining Graphs, Trees and Sequences (MGTS-2003)*, pages 65–74, September 2003

[12] David Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, University of California at Santa Cruz, Santa Cruz, CA, USA, July 1999

Haussler defines a *decomposition* of an instance $x \in \mathcal{X}$ by means of a relation $R : \mathcal{X}_1 \times \cdots \times \mathcal{R}_D \times \mathcal{X} \to \{\top, \bot\}$ such that $R(x_1, \ldots, x_D, x)$ is true if $x_1, \ldots, x_D$ are parts of x, each part having domain $\mathcal{X}_d$. The inverse relation $R^{-1} : \mathcal{X} \to 2^{\mathcal{X}_1 \times \cdots \times \mathcal{X}_D}$ is defined as

$$R^{-1}(x) = \{(x_1, \ldots, x_D) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_D | R(x_1, \ldots, x_D, x)\}.$$

For a specific application, the definition of R can be used to encode allowed decompositions into parts and the particular invariances that exist between parts. The *convolution kernel* is defined as

$$k(x, x') = \sum_{(x_1,\ldots,x_D) \in R^{-1}(x)} \sum_{(x'_1,\ldots,x'_D) \in \mathbb{R}^{-1}(x')} \prod_{d=1}^{D} k_d(x_d, x'_d), \tag{3}$$

where $k_d : \mathcal{X}_d \times \mathcal{X}_d \to \mathbb{R}$ is a kernel measuring the similarity between the parts x_d and x'_d. This general definition is shown by Haussler to contain many well-known kernels such as RBF kernels. He uses (3) to define kernels for strings. However, it seems that the use of the relation R and the fixed number D of parts make it difficult to apply (3) to a novel structured input domain.

SUMMARIZING, kernels for structured input data separate the classification algorithm from the representation of the input domain. When designed properly they are efficient and provide a large feature space. Due to the constraint of being positive-definite it can be difficult to create or modify a kernel for a new structured input domain.

In the remaining part of this chapter we give an introduction to Boosting. As with kernel methods, Boosting allows tractable learning in large feature spaces. In the next chapter we will introduce a family of feature spaces for structured input domains that can naturally be combined with the Boosting classifiers introduced in this section. Like in kernel methods we achieve the separation of the Boosting learning algorithm from the actual input domain.

Boosting Methods

Boosting is commonly understood as the combination of many weak decision functions into a single strong one. This general idea can be motivated, understood and realized in many different ways and indeed both the success of practical Boosting methods and the intuitive appeal of the method have led to diverse research efforts in the area. Unfortunately, Boosting is often understood only as an iterative procedure.

In this thesis, we will take a simple, general and fruitful approach to Boosting methods. Our approach is based on formulating a single optimization problem over all possible decision functions from a hypothesis space. This problem *can* be solved iteratively and in that case well-known methods such as AdaBoost are recovered.

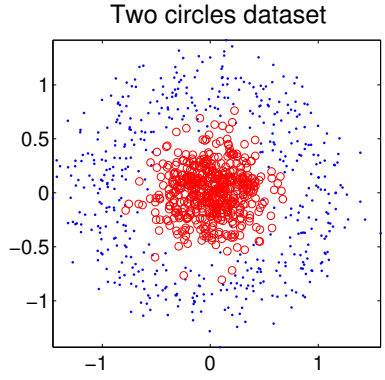


Figure 2: Two class classification training data. It is not possible to separate the instances using linear decision functions.

As an example, consider a two-class classification problem with per-class distributions as shown in Figure 2. The distributions are radially-symmetric and we want to learn to separate the two classes by means of a function $h : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X} = \mathbb{R}^2$ is the input space in this case and $\mathcal{Y} = \{-1, 1\}$ are the class labels.

Let us choose a particularly simple *function class* $\mathcal{H} : \Omega \to \mathcal{Y}^{\mathcal{X}}$, with $\Omega = \{(\omega_1, \omega_2, \omega_3) : \omega_1 \in \{1, 2\}, \omega_2 \in \mathbb{R}, \omega_3 \in \{-1, 1\}\}$. We consider functions of the form

$$h(\boldsymbol{x}; \omega) = \begin{cases} \omega_3 & \text{if } x_{\omega_1} \leq \omega_2 \\ -\omega_3 & \text{otherwise.} \end{cases}$$

This class $\mathcal{H}$ of decision functions is known as *decision stumps*. A decision stump $h(\boldsymbol{x}; (\omega_1, \omega_2, \omega_3))$ simply looks at a single dimension ω_1 of the sample $\boldsymbol{x}$, compares it with a fixed value ω_2 and returns ω_3 or $-\omega_3$, depending on whether the value is smaller or larger than the threshold.

Obviously, no $\omega \in \Omega$ will yield a good decision function for the dataset shown in Figure 2, because the hypothesis set is too weak. Still, for some parameters we can produce a function which performs better than chance performance.

If we consider *all possible hypotheses* $h \in \mathcal{H}$, it should be possible to improve the classification accuracy by considering *weighted combinations* of multiple $h_1, \ldots, h_M \in \mathcal{H}$. To this end, we define a new classification function $F : \mathcal{X} \to \mathbb{R}$ as

$$F(\boldsymbol{x}; \boldsymbol{\alpha}) = \sum_{\omega \in \Omega} \alpha_\omega h(\boldsymbol{x}; \omega), \tag{4}$$

with mixture weights α_ω, satisfying

$$\alpha_\omega \geq 0, \qquad \forall \omega \in \Omega \tag{5}$$

$$\sum_{\omega \in \Omega} \alpha_\omega = C, \tag{6}$$

where $C > 0$ is a given constant. Thus, F evaluates a linear combination of hypotheses from $\mathcal{H}$. Clearly, F represents a much larger set of hypotheses, the set

$$\mathcal{F} = \{F(\cdot; \boldsymbol{\alpha}) | \boldsymbol{\alpha} \text{ satisfies (5) and (6)}\}.$$

This includes the set $\mathcal{H}$: each hypothesis $h(\cdot; \omega') \in \mathcal{H}$ is recovered by setting $\alpha_{\omega'} = C$ and $\alpha_\omega = 0$ for all $\omega \in \Omega \setminus \{\omega'\}$.

For our example dataset, $\mathcal{F}$ is powerful enough to separate the points, as shown in Figure 3 and 4. This holds in more generality: if each point in the set of samples is unique, there exists a hypothesis in $\mathcal{F}$ able to separate the samples perfectly. The hypothesis set $\mathcal{F}$ is said to have an infinite Vapnik-Chervonenkis dimension[13].

Summarizing from our example: one way to understand Boosting is the construction of a powerful hypothesis set $\mathcal{F}$ from a weak hypothesis set $\mathcal{H}$ by considering mixtures from $\mathcal{H}$.

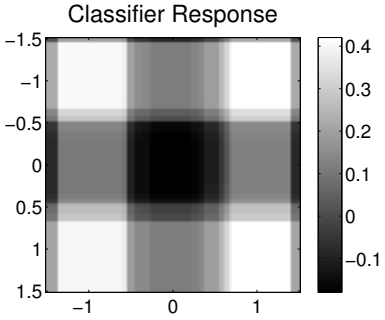


Figure 3: Response of the combined function $F : \mathcal{X} \to \mathbb{R}$. While artifacts due to axis-aligned decisions are still visible, the resulting separation is very good.

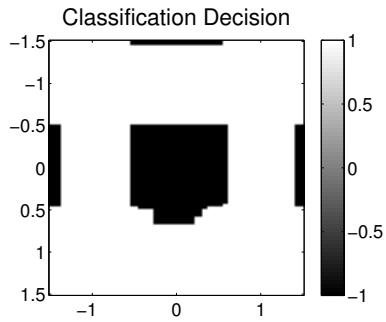


Figure 4: Hard decision of the combined function, i.e., $\text{sign}(F(\cdot))$.

[13] Vladimir N. Vapnik and Alexey Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971

Regarding the set $\mathcal{H}$, we refer to the individual elements $h \in \mathcal{H}$ as *weak learner* or *hypothesis*, but equivalently they can be seen as feature functions. Then, F is a linear model in a high dimensional feature space $\mathcal{H}$. Thus, *another* way to understand Boosting is to fit a linear model in a large implicitly defined feature space.

In the remaining part of this chapter we first make a comment on the generality of Boosting techniques and then formalize a general Boosting model and an efficient Boosting algorithm, followed by a discussion of the history of Boosting and current developments. We will then see how the Boosting idea lends itself ideally to structured input data: structured data often has a natural *substructure-superstructure relation* which defines a hypothesis space.

Boosting as Linearization

The consequences of viewing Boosting as learning a linear model are profound: the construction underlying Boosting is *not* restricted to supervised learning. In the above view, Boosting simultaneously achieves two things, i) extending the function class, and ii) *linearizing* its representation. Thus, in general, in a larger model, a possibly non-linear function can be simultaneously replaced by a more powerful one and made linear in a new parametrization.

In the above example, the elements of $\mathcal{H}$ depend non-linearly on ω, yet the new class $\mathcal{F}$ depends only linearly on $\boldsymbol{\alpha}$. This is achieved by instantiating all values in Ω and taking the convex mixture of the resulting parameter-free functions.

This general construction is the underlying principle of the *inner linearization* and generalized *Dantzig-Wolfe decomposition*. For an introduction into this literature, see Geoffrion[14].

[14] Arthur M. Geoffrion. Elements of large-scale mathematical programming: Part i: Concepts. *Management Science*, 16 (11):652–675, 1970; and Arthur M. Geoffrion. Elements of large-scale mathematical programming: Part ii: Synthesis of algorithms and bibliography. *Management Science*, 16(11):676–691, 1970

Formalization

We now formalize the above discussion. In the general setting we consider a family $\mathcal{H}$ of functions $h : \mathcal{X} \to \mathbb{R}$, where the elements of the family are indexed by a set Ω. The family is thus of the form

$$h(\cdot; \omega) : \mathcal{X} \to \mathbb{R}.$$

Given N training examples samples $\{(\boldsymbol{x}_n, y_n)\}_{n=1,\ldots,N}$, with $(\boldsymbol{x}_n, y_n) \in \mathcal{X} \times \{-1, 1\}$, we want to learn a classification function

$$F(\boldsymbol{x}; \boldsymbol{\alpha}) = \sum_{\omega \in \Omega} \alpha_\omega h(\boldsymbol{x}; \omega),$$

which generalizes to the entire input domain $\mathcal{X}$.

To achieve this, we minimize a loss function with the addition of a regularization term. For a loss function $L : \mathbb{R} \to \mathbb{R}_+$, and regularization function $R : \mathbb{R}^\Omega \to \mathbb{R} \cup \{\infty\}$ the task is to minimize the regularized empirical risk

function

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{N}\sum_{n=1}^{N} L(y_n F(\boldsymbol{x}_n;\boldsymbol{\alpha})) + R(\boldsymbol{\alpha}).$$

We now discuss two popular Boosting methods based on this regularized empirical risk function, AdaBoost and LPBoost.

[15] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997

[16] Chunhua Shen and Hanxi Li. A duality view of boosting algorithms. *CoRR*, abs/0901.3590, 2009

AdaBoost[15] was the first practical Boosting algorithm. It is arguably the most well known Boosting method and still popular for its simplicity. Shen and Li[16] show that the optimization problem that AdaBoost solves incrementally can be equivalently rewritten as the following convex mathematical program, the *AdaBoost primal*.

$$\min_{\boldsymbol{\alpha},z} \quad \log \sum_{n=1}^{N} \exp(z_n) \tag{7}$$

$$\text{sb.t.} \quad z_n = -y_n \sum_{\omega\in\Omega} \alpha_\omega h(\boldsymbol{x}_n;\omega) : \lambda_n, \qquad n = 1,\dots,N, \tag{8}$$

$$\alpha_\omega \geq 0, \qquad \forall \omega \in \Omega,$$

$$\sum_{\omega\in\Omega} \alpha_\omega = \frac{1}{T} : \gamma, \tag{9}$$

where λ_n and γ are Lagrange multipliers and the parameter $T > 0$ is a regularization parameter which is implicitly chosen in the original AdaBoost algorithm by means of stopping the algorithm after a fixed number of iterations. Here, large values of T correspond to strong regularization, small values to a better fit on the training data.

[17] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004

The convex problem (7) can be dualized[17] to obtain the following AdaBoost dual problem.

$$\max_{\gamma,\boldsymbol{\lambda}} \quad \frac{1}{T}\gamma - \sum_{n=1}^{N} \lambda_n \log \lambda_n \tag{10}$$

$$\text{sb.t.} \quad \sum_{n=1}^{N} \lambda_n y_n h(\boldsymbol{x}_n;\omega) \leq -\gamma, \qquad \forall \omega \in \Omega, \tag{11}$$

$$\lambda_n \geq 0, \qquad n = 1,\dots,N,$$

$$\sum_{n=1}^{N} \lambda_n = 1.$$

The two problems (7) and (10) form a primal-dual pair of convex optimization problems and can be solved efficiently using standard convex optimization solvers. AdaBoost uses the exponential loss function and we now discuss alternatives to this choice. It will turn out that for different choices of loss functions we will obtain slightly different dual problems (10) and we can formulate a single algorithm for all of them.

[18] Ayhan Demiriz, Kristin P. Bennett, and John Shawe-Taylor. Linear programming boosting via column generation. *Journal of Machine Learning*, 46:225–254, 2002

An alternative to AdaBoost is the so called Linear Programming Boosting (LPBoost) proposed by Demiriz et al.[18] Compared to AdaBoost there are

two notable differences. First, instead of minimizing the exponential loss as in (7) the Hinge loss is minimized. Second, in LPBoost the *margin* between samples is maximized explicitly.

We can generalize the Hinge loss to a p-norm Hinge loss, and thus obtain a family of generalized LPBoost procedures. Given the p-norm Hinge loss parameter $p > 1$, the loss is simply ξ_n^p, the p-exponentiated margin violation of the instance. The loss is visualized for $p = 1.5$ and $p = 2$ in Figure 5.

Together with an additional regularization parameter $D > 0$ the generalized LPBoost primal problem can be formulated as follows.

$$\min_{\boldsymbol{\alpha},\rho,\boldsymbol{\xi}} \quad -\rho + D\sum_{n=1}^{N}\xi_n^p \tag{12}$$

$$\begin{aligned}\text{sb.t.}\quad & y_n \sum_{\omega\in\Omega}\alpha_\omega h(\boldsymbol{x}_n;\omega) + \xi_n \geq \rho : \lambda_n, \qquad n = 1,\ldots,N, \\ & \xi_n \geq 0, \qquad n = 1,\ldots,N, \\ & \alpha_\omega \geq 0, \qquad \forall\omega\in\Omega, \\ & \sum_{\omega\in\Omega}\alpha_\omega = \frac{1}{T} : \gamma,\end{aligned} \tag{13}$$

where again λ_n and γ are Lagrange multipliers of the respective constraints. As for AdaBoost we obtain the Lagrangean dual problem of (12).

$$\max_{\gamma,\boldsymbol{\lambda}} \quad \frac{1}{T}\gamma - \frac{(q-1)^{q-1}}{q(Dq)^{q-1}}\sum_{n=1}^{N}\lambda_n^q \tag{14}$$

$$\begin{aligned}\text{sb.t.}\quad & \sum_{n=1}^{N}\lambda_n y_n h(\boldsymbol{x}_n;\omega) \leq -\gamma : \alpha_\omega, \qquad \forall\omega\in\Omega, \\ & \lambda_n \geq 0, \qquad n = 1,\ldots,N, \\ & \sum_{n=1}^{N}\lambda_n = 1 : \rho,\end{aligned} \tag{15}$$

where $q = \frac{p}{p-1}$ for $p > 1$ such that q is the dual norm of the p-norm in (12), i.e., we have $\frac{1}{p} + \frac{1}{q} = 1$.

From the above primal and dual mathematical programs we see that problem (10) and (14) are the same, except for the objective function. If we separate out the part of the dual objective which differs as

$$R_{\text{AdaBoost}}(\boldsymbol{\lambda}) = \sum_{n=1}^{N}\lambda_n \log\lambda_n$$

for (10), and likewise[19] for (14)

$$R_{\text{GLPBoost}}(\lambda;q,D) = \frac{(q-1)^{q-1}}{q(Dq)^{q-1}}\sum_{n=1}^{N}\lambda_n^q,$$

then we can use a unified dual problem to solve both the original AdaBoost optimization problem, as well as the generalized linear programming Boosting problem.

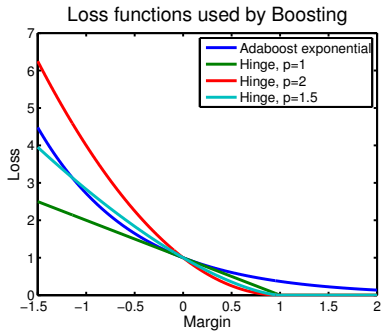


Figure 5: Different loss functions used by AdaBoost and generalized linear programming boosting.

[19] The q-norm can be interpreted as Tsallis entropy:

Constantino Tsallis. Possible generalization of boltzmann-gibbs statistics. *Journal of Statistical Physics*, 52(1–2):479–487, 1988

Additionally, we define the dual regularization function corresponding to a variant[20] of Logitboost as

$$R_{\text{Logitboost}}(\boldsymbol{\lambda}) = \sum_{n=1}^{N} (\lambda_n \log \lambda_n + (1-\lambda_n)\log(1-\lambda_n)).$$

[20] When the standard Logitboost primal is dualized, the resulting dual problem is not of the form (16). However, the distribution constraint (18) can be added and a meaningful primal problem can be rederived. The primal Logitboost problem which yields a proper distribution over $\boldsymbol{\lambda}$ in the dual is of the form $\min_{\boldsymbol{\alpha},\rho,z} \sum_{n=1}^{N} \log(1+\exp z_n) - \rho$, subject to $z_n = \rho - y_n \sum_{\omega\in\Omega} \alpha_\omega h(\boldsymbol{x}_n;\omega)$ for $n = 1,\dots,N$, and $\sum_{\omega\in\Omega} \alpha_\omega = \frac{1}{T}$, and $\alpha_\omega \geq 0$ for all $\omega \in \Omega$.

A general totally corrective Boosting algorithm

From the above discussion we see that the structure of the dual problem remains the same for the exponential loss, the p-norm Hinge loss and the logistic loss. We can therefore obtain a single dual problem, which we call the *general totally corrective Boosting dual problem*. It is given as follows.

$$\max_{\gamma,\boldsymbol{\lambda}} \quad \frac{1}{T}\gamma - R(\boldsymbol{\lambda}) \tag{16}$$

$$\text{sb.t.} \quad \sum_{n=1}^{N} \lambda_n y_n h(\boldsymbol{x}_n;\omega) \leq -\gamma \;:\; \alpha_\omega, \qquad \forall \omega \in \Omega, \tag{17}$$

$$\lambda_n \geq 0, \qquad n = 1,\dots,N,$$

$$\sum_{n=1}^{N} \lambda_n = 1, \tag{18}$$

where α_ω is the Lagrange multiplier corresponding to the constraint (17). For the above three regularization functions R_{AdaBoost}, R_{GLPBoost} and $R_{\text{Logitboost}}$, any solution to the above program (16) satisfies the constraint $\sum_{\omega\in\Omega} \alpha_\omega = \frac{1}{T}$.

The overall totally corrective Boosting algorithm is shown in Algorithm 1. Notice how it is different from classical Boosting algorithms.

First, unlike AdaBoost and Gentleboost it is *totally corrective* in that in each iteration all weights $\boldsymbol{\alpha}_{\Omega'}$ are adjusted to optimality with respect to the subspace indexed by Ω'.

Second, in each iteration an arbitrary large set of hypotheses — indexed by Γ in Algorithm 1 — can be added to the problem, as long as each hypothesis corresponds to a violated constraint in the master problem. This property improves the rate of convergence considerably in practice if multiple good weak learners can be provided. Whether it is possible to do so efficiently depends on the structure of the weak hypothesis set $\mathcal{H}$.

Third, we give a convergence criterion based on the constraint violation of (17).[21]

[21] If the exact best hypothesis can be found in each iteration, it is possible to compute an alternative convergence criterion from the duality gap.

For these reasons, in practice the TCBoost algorithm is preferable over other Boosting algorithms in almost all situations. Empirically it makes more efficient use of the weak learners, has orders of magnitude fewer outer iterations, can exploit the ability to return multiple hypotheses and allows different regularization functions.

The master problem (16) can be solved efficiently using interior-point methods[22]. The problem is well structured: for all the considered regularization functions, the Hessian of the Lagrangian is diagonal, all constraints are dense and linear.

[22] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer, second edition, 2006. ISBN 0-387-30303-0

Algorithm 1 TCBoost: general Totally Corrective Boosting

$\boldsymbol{\alpha} = \text{TCBoost}(X, Y, R, T, \epsilon)$
Input:
  $(X, Y) = \{(\boldsymbol{x}_n, y_n)\}_{n=1,\ldots,N}$ training set, $(\boldsymbol{x}_n, y_n) \in \mathcal{X} \times \{-1, 1\}$
  $R : \mathbb{R}^N \to \mathbb{R}_+$ regularization function
    (one of R_{AdaBoost}, R_{GLPBoost} or $R_{\text{Logitboost}}$)
  $T > 0$ regularization parameter
  $\epsilon \geq 0$ convergence tolerance
Output:
  $\boldsymbol{\alpha} \in \mathbb{R}^\Omega$ learned weight vector
Algorithm:
$\boldsymbol{\lambda} \leftarrow \frac{1}{N}\mathbf{1}$ {Initialize: uniform distribution}
$\gamma \leftarrow -\infty$
$(\Omega', \boldsymbol{\alpha}) = (\emptyset, \mathbf{0})$
loop
  $\Gamma \leftarrow \{\omega_1, \omega_2, \ldots, \omega_M\} \subset \Omega$, where
    $\forall m = 1, \ldots, M : \sum_{n=1}^N \lambda_n y_n h(\boldsymbol{x}_n; \omega_m) + \gamma \leq 0$ {Subproblem}
  $\text{maxviolation} \leftarrow \max_{\omega \in \Gamma}(\sum_{n=1}^N \lambda_n y_n h(\boldsymbol{x}_n; \omega) + \gamma)$
  $\Omega' \leftarrow \Omega' \cup \Gamma$ {Enlarge restricted master problem}
  $$(\gamma, \boldsymbol{\lambda}, \boldsymbol{\alpha}_{\Omega'}) \leftarrow \begin{cases} \underset{\gamma, \boldsymbol{\lambda}}{\text{argmax}} & \frac{1}{T}\gamma - R(\boldsymbol{\lambda}) \\ \text{sb.t.} & \sum_{n=1}^N \lambda_n y_n h(\boldsymbol{x}_n; \omega) \leq -\gamma : \alpha_\omega, \quad \forall \omega \in \Omega' \\ & \lambda_n \geq 0, \qquad n = 1, \ldots, N, \\ & \sum_{n=1}^N \lambda_n = 1. \end{cases}$$
  if maxviolation $\leq \epsilon$ **then**
    break {Converged to tolerance}
  end if
end loop

Boosting Subproblem

During the course of Algorithm TCBoost, the following subproblem needs to be solved.

Problem 1 (Boosting Subproblem) *Let* $(X, Y) = \{(\boldsymbol{x}_n, y_n)\}_{n=1,\ldots,N}$ *with* $(\boldsymbol{x}_n, y_n) \in \mathcal{X} \times \{-1, 1\}$ *be a given set of training samples, and* $\boldsymbol{\lambda} \in \mathbb{R}^N$ *be given, satisfying* $\sum_{n=1}^N \lambda_n = 1$, $\lambda_n \geq 0$ *for all* $n = 1, \ldots, N$. *Given a family of functions* $\mathcal{H} : \Omega \to \mathbb{R}^\mathcal{X}$ *indexed by a set* Ω, *the* Boosting subproblem *is the problem of solving for* ω^* *such that*

$$\omega^* = \underset{\omega \in \Omega}{argmax} \sum_{n=1}^N \lambda_n y_n h(\boldsymbol{x}_n; \omega). \tag{19}$$

The subproblem is an optimization problem over variables defined by the set of weak learners, maximizing the inner product between a given coefficient vector and the weak learner response. Throughout this chapter we assume

the Boosting subproblem can be solved exactly. There are methods which can deal with the case when the subproblem can only be solved approximately, see Meir and Rätsch[23].

The Boosting subproblem will take an important part in what follows. We will derive a family of feature spaces for structured data which share the property that the subproblem (19) can be solved efficiently. Moreover, the feature space is a natural one, and a large body of literature of data mining algorithms working in the same feature space exists. Most of these algorithms can be easily adapted to solve the Boosting subproblem.

Before we discuss the structured feature spaces, let us briefly reconcile on the historical development of Boosting approaches.

History of Boosting

We briefly discuss the development of Boosting in chronological order. For a detailed introduction covering recent trends see Meir and Rätsch[24].

The origins of Boosting are commonly attributed to an unpublished note[25] in which Kearns defined the *hypothesis boosting problem*: "[Does] an efficient learning algorithm that outputs an hypothesis whose performance is only slightly better than random guessing implies the existence of an efficient learning algorithm that outputs a hypothesis of arbitrary accuracy?".

Schapire[26] provided an affirmative answer in the form of a polynomial-time algorithm. The first *practical* Boosting algorithms appeared a few years later, *AdaBoost* due to Freund and Schapire[27], and *Arcing* due to Breiman[28]. Where AdaBoost optimizes an exponential loss function, Arcing directly maximizes the minimum margin.

The empirical success of predictors trained using AdaBoost and the simplicity of implementation of the original AdaBoost algorithm led to a flurry of research activity and empirical evidence in favor of the approach: in the late 1990's, Boosting and the then recently introduced *kernel machines* invigorated the machine learning community.

The empirical success was partially explained by Friedman et al.[29] and Mason et al.[30], who viewed Boosting as incremental fitting procedure of a linear model by means of coordinate-descent in the space of all weak learners. The Boosting subproblem becomes a descent-coordinate identification problem. In the unified *Anyboost* algorithm proposed by Mason, the learned function at iteration t is updated according to

$$F^{t+1} = F^t + \alpha_{\omega^t} h(\cdot; \omega^{t+1}),$$

where $h(\cdot; \omega^{t+1}) : \mathcal{X} \to \mathbb{R}$ is the weak learner produced at iteration t and $\alpha_{\omega^{t+1}}$ is its weight. The weight is optimized over by solving a one-dimensional line search problem. The algorithm can be shown to have a strong convergence guarantee[31].

[23] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging. In *Advanced Lectures on Machine Learning*, pages 119–184. Springer, 2003

[24] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging. In *Advanced Lectures on Machine Learning*, pages 119–184. Springer, 2003

[25] Michael Kearns. Thoughts on hypothesis boosting. (Unpublished), December 1988. URL `http://www.cis.upenn.edu/~mkearns/papers/boostnote.pdf`

[26] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5: 197–227, 1990

[27] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EUROCOLT*, 1994; Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proc. 13th International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996; and Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997

[28] Leo Breiman. Prediction games and arcing algorithms. Technical report, December 1997. Technical Report 504, University of California, Berkeley

[29] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–374, 2000

[30] Llew Mason, Jonathan Baxter, Peter L. Bartlett, and Marcus R. Frean. Boosting algorithms as gradient descent. In *NIPS*, pages 512–518. The MIT Press, 1999

[31] Tong Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Transactions on Information Theory*, 49(3):682–691, 2003

Although in the literature Boosting is most often viewed as procedure that fits into the Anyboost framework, this view has a number of shortcomings, i) a poor convergence rate, ii) inability to add more than one weak learner per iteration, iii) repeated generation of the same weak learner, iv) inability to incorporate additional constraints into the learning problem, v) inefficient adjustment of weights of previously generated weak learners (not totally-corrective), and vi) a fixed number of iterations and absence of a convergence criterion. All the above points are overcome in the TCBoost algorithm described earlier in this chapter.

The functional gradient view has been instrumental in generalizing Boosting to regression[32] and unsupervised learning tasks[33]. Recently, an interesting discussion around the different views on Boosting emerged from contradicting empirical evidence[34]. This discussion provides further interesting research directions on Boosting.

[32] Gunnar Rätsch, Ayhan Demiriz, and Kristin P. Bennett. Sparse regression ensembles in infinite and finite hypothesis spaces. *Machine Learning*, 48(1-3):189–218, 2002

[33] Gunnar Rätsch, Sebastian Mika, Bernhard Schölkopf, and Klaus-Robert Müller. Constructing boosting algorithms from SVMs: An application to one-class classification. *IEEE Trans. Pattern Anal. Mach. Intell*, 24(9):1184–1199, 2002

[34] David Mease and Abraham Wyner. Evidence contrary to the statistical view of boosting. *Journal of Machine Learning Research*, 9:131–156, February 2008

Conclusion

In this chapter we first discussed propositionalization and kernels as two possible methods to learn with structured input data. We then discussed Boosting as an efficient method to fit linear models in large feature spaces. By designing a feature space that captured all relevant information about the input domain we showed that it is possible to use Boosting to learn a classifier for structured input data. In the next chapter we will introduce our general approach to construct such a complete feature space.

Substructure Poset Framework

Structured data is abundant in the real-world. In order to perform predictions on structured data, the learning method has to be able to access statistics about the data that contain discriminative information. The set of accessible statistics about the data constitutes the *feature space*.

This chapter introduces a novel framework called *substructure poset framework* for building classification functions for structured input domains. The basic modeling assumption made in the framework is that the input domain has natural *substructure relation* "$\subseteq$".

The substructure relation can capture natural inclusion properties within a part-based representation of an object. For example, when classifying documents, this could mean that given a sentence s and a document t the expression $s \subseteq t$ states whether s appears in t or not. For chemical compounds the relation could be defined as to test whether certain functional groups are present in the compound or not, as illustrated in Figure 6.

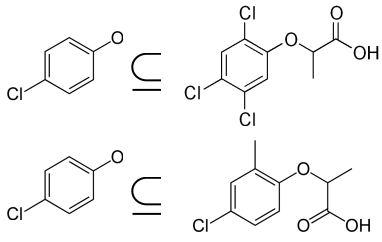


Figure 6: Example substructure relation for chemical compounds: the functional group on the left is present within the larger molecules on the right side.

Based on this substructure assumption we derive a feature space and a set of abstract algorithms for building linear classifiers in this feature space. In later chapters we make these abstract algorithms concrete for structured input domains such as sequences and labeled graphs.

Within our feature space we learn a classification function using *Boosting* by combining a large number of weak classification functions in order to obtain a single strong classifier.

We first define substructures and then examine properties of the associated feature space. In the latter part of this chapter we discuss in detail how the *Boosting subproblem* can be solved efficiently in our framework.

The main contribution of this chapter is the substructure poset framework. A limited form of the framework was originally proposed by Kudo et al.[35] and Saigo et al.[36], our generalization adds a theoretical analysis as well as two abstract constructions for efficient enumeration algorithms of which all the previous works are special instances.

[35] Taku Kudo, Eisaku Maeda, and Yuji Matsumoto. An application of boosting to graph classification. In *NIPS*, 2004

[36] Hiroto Saigo, Sebastian Nowozin, Tadashi Kadowaki, Taku Kudo, and Koji Tsuda. gboost: A mathematical programming approach to graph classification and regression. *Machine Learning*, 75(1): 69–89, 2009

Substructures

We first define what we mean by *structure* in the input space. Although our definition is flexible, it does not encompass all of structured input learning. In particular, all cases included by our definition can naturally be used with the Boosting learning method.

Definition 1 (Substructure Poset) *Given a set $\mathcal{S}$ of structures and a binary relation $\subseteq: \mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$, the pair $(\mathcal{S}, \subseteq)$ is called* substructure poset *(partially ordered set) if it satisfies,*

- *there exists a unique* least element $\emptyset \in \mathcal{S}$ *for which* $\emptyset \subseteq s$ *for any* $s \in \mathcal{S}$,
- $\subseteq$ *is* reflexive*:* $\forall s \in \mathcal{S} : s \subseteq s$,
- $\subseteq$ *is* antisymmetric*:* $\forall s_1, s_2 \in \mathcal{S} : (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1) \Rightarrow (s_1 = s_2)$,
- $\subseteq$ *is* transitive*:* $\forall s_1, s_2, s_3 \in \mathcal{S} : (s_1 \subseteq s_2 \wedge s_2 \subseteq s_3) \Rightarrow (s_1 \subseteq s_3)$.

In other words, $\subseteq$ *is a* partial order *on* $\mathcal{S}$ *and* $(\mathcal{S}, \subseteq)$ *is a* partially ordered set *(poset) with a unique* least element $\emptyset \in \mathcal{S}$.

In this thesis we will consider three families of substructure posets $(\mathcal{S}, \subseteq)$, where the elements in $\mathcal{S}$ correspond to sets of integers, labeled sequences and labeled undirected graphs, respectively. For the case of sets, $\subseteq$ corresponds to the usual subset relation, but for sequences and graphs we will have to explicitly define the relation.

We will now use the substructure relation $\subseteq$ to define a *covering relation*. The covering relation will later play an important role in devising algorithms to enumerate the elements of $\mathcal{S}$. It is defined as follows.

Definition 2 (Covering Relation $\sqsubset$) *Given a substructure poset $(\mathcal{S}, \subseteq)$, define $\sqsubset: \mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$, such that for all $s, t \in \mathcal{S}$ we have $s \sqsubset t$ iff*

$$s \subseteq t \text{ and } \nexists\, u \in (\mathcal{S} \setminus \{s, t\}) : s \subseteq u, u \subseteq t.$$

Given the definition of substructure poset, we now derive an induced feature space.

Definition 3 (Substructure-induced Feature) *Given a substructure poset $(\mathcal{S}, \subseteq)$ and an element $s \in \mathcal{S}$, define $x_s : \mathcal{S} \to \{0, 1\}$ as*

$$x_s(t) = \begin{cases} 1 & \text{if } t \subseteq s, \\ 0 & \text{otherwise.} \end{cases}$$

An example of the feature function associated to sets is shown in Figure 7. The substructure induced feature space has some interesting properties that we now examine in detail. We first show that the feature mapping preserves all information about a structure.

	$x_s(\{2\})$	$x_s(\{1,3\})$		
	$x_s(\{1\})$	$x_s(\{3\})$	$x_s(\{1,2,3\})$	
$s = \{1,3,5\}$	1	0	1	1	0

Figure 7: Example of substructure induced features for the case of sets.

Lemma 1 (Structure Identification) *Given a substructure poset $(\mathcal{S}, \subseteq)$, an unknown element $s \in \mathcal{S}$ and its feature representation $x_s \in \mathbb{R}^{\mathcal{S}}$, we can identify s from x_s uniquely.*

Proof. Consider the set $T = \{t | x_s(t) = 1\}$. Because $s \in \mathcal{S}$, we have $x_s(s) = 1$ and hence $s \in T$. Let $U = \{u \in T | \forall t \in T : t \subseteq u\}$. We show that $U = \{s\}$.

First, existence, i.e., $s \in U$: we have $s \in T$ and $t \subseteq s$ for all $t \in T$, by definition. Next, uniqueness: let $u_1, u_2 \in U$. By definition of U it holds that $u_1 \subseteq u_2$ and $u_2 \subseteq u_1$. By antisymmetry of $\subseteq$ we have $u_1 = u_2$. Therefore U contains exactly one element, the original structure s. □

In the next section we first discuss how the substructure-induced features can be used to find frequent substructures in a database. In the section following it we introduce substructure Boosting for identifying discriminative substructures.

Frequent Substructure Mining

Given a set of observed structures, an important task is to identify substructures that occur frequently. We first define the *frequency* of a substructure, then define the frequent substructure mining problem.

Definition 4 (Frequency of a Substructure) *Given a substructure poset $(\mathcal{S}, \subseteq)$, a set of N instances $X = \{s_n\}_{n=1,\ldots,N}$, and an element $t \in \mathcal{S}$, the* frequency of t with respect to X *is defined as*

$$freq(t, X) = \sum_{n=1}^{N} x_{s_n}(t).$$

We have the following simple but important lemma about frequencies.

Lemma 2 (Anti-monotonicity of Frequency) *The frequency of a fixed element $t \in \mathcal{S}$ with respect to X is a monotonically decreasing function under $\subseteq$, that is*

$$\forall\, t_1, t_2 \in \mathcal{S}, t_1 \subseteq t_2 : freq(t_1, X) \geq freq(t_2, X).$$

Proof. We have

$$\begin{aligned}
\text{freq}(t_1, X) &= \sum_{n=1}^{N} x_{s_n}(t_1) \\
&= \sum_{n=1}^{N} I[t_1 \subseteq x_{s_n}] \\
&= \sum_{n=1}^{N} (I[t_1 \subseteq x_{s_n}] + \underbrace{I[t_2 \subseteq x_{s_n}] - I[t_1 \subseteq x_{s_n} \wedge t_2 \subseteq x_{s_n}]}_{=0}) \\
&= \sum_{n=1}^{N} (I[t_2 \subseteq x_{s_n}] + \underbrace{I[t_1 \subseteq x_{s_n}] - I[t_1 \subseteq x_{s_n} \wedge t_2 \subseteq x_{s_n}]}_{\geq 0}) \\
&\geq \sum_{n=1}^{N} I[t_2 \subseteq x_{s_n}] \\
&= \text{freq}(t_2, X),
\end{aligned}$$

where $I(\text{pred})$ is 1 if the predicate is true and 0 otherwise.

The definition of frequency of substructures with respect to a set of structures already allows us to define an interesting problem, the *frequent substructure mining* problem.

Problem 2 (Frequent Substructure Mining) *Given a substructure poset $(\mathcal{S}, \subseteq)$, a set of N instances $X = \{s_n\}_{n=1,\dots,N}$ with $s_n \in \mathcal{S}$, and a frequency threshold $\sigma \in \mathbb{N}$, find the set $F(\sigma, X) \subseteq \mathcal{S}$ of all σ-frequent substructures, i.e., the largest set such that $\forall t \in F(\sigma, X) : freq(t, X) \geq \sigma$.*

The frequent substructure mining problem is an important problem in the data mining community because substructures which appear more frequently in a dataset are often more interesting for the task at hand.[37] Due to the importance of the frequent substructure mining problem, a large number of methods for different structures such as sets, sequences, trees, graphs, etc. have been proposed[38].

[37] The original *frequent itemset mining* methods were invented to do *basket analysis* of customers. There, products that are frequently bought *together* might reveal customer behavior.

[38] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *ICDM*, 2002; Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Jianyong Wang, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. Mining sequential patterns by pattern-growth: The prefixspan approach. *IEEE Trans. Knowl. Data Eng*, 16(11):1424–1440, 2004; and Takeaki Uno, Masashi Kiyomi, and Hiroki Arimura. LCM ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *FIMI*, volume 126 of *CEUR Workshop Proceedings*, 2004

Substructure Boosting

We now consider learning a function $F : \mathcal{S} \to \{-1, 1\}$. For applying the substructure-induced feature space in the Boosting context, we need two ingredients. First, we need to define the family $\omega \in \Omega$ of weak learners $h(\cdot; \omega) : \mathcal{S} \to \mathbb{R}$. Second, we need to provide a means to solve the Boosting subproblem $\omega^* = \text{argmax}_{\omega \in \Omega} \sum_{n=1}^{N} \lambda_n y_n h(x_{s_n}; \omega)$.

We define the family of substructure weak learners as follows.

Definition 5 (Substructure Boosting Weak Learner) *We define $\Omega = \mathcal{S} \times \{-1, 1\}$ and $\omega = (t, d) \in \Omega$, with*

$$h(\cdot; \omega) : \mathcal{S} \to \{-1, 1\}, \qquad h(s; (t, d)) = \begin{cases} d & \text{if } x_s(t) = 1, \\ -d & \text{otherwise.} \end{cases}$$

The family is then given as $\mathcal{H} = \{h(\cdot; (t, d)) | (t, d) \in \Omega\}$.

This definition of weak learner is natural in the substructure-induced feature space. Both the presence ($x_s(t) = 1$) and absence ($x_s(t) = 0$) of a substructure t can cause a response into positive or negative direction.

Moreover, the weak learners can be linearly combined. The linear combination of a finite number of weak learners is sufficient to linearly separate any given finite training set. This is formalized in the next theorem.

Theorem 1 (Capacity and Strict Linear Separability) *Given a substructure poset $(\mathcal{S}, \subseteq)$, a set of N labeled instances $X = \{(s_n, y_n)\}_{n=1,\dots,N}$ with $(s_n, y_n) \in \mathcal{S} \times \{-1, 1\}$ and uniqueness over labels, $\forall s_{n_1}, s_{n_2}$, $n_1, n_2 \in \{1, \dots, N\}$: $s_{n_1} = s_{n_2} \Rightarrow y_{n_1} = y_{n_2}$, and given the set $\mathcal{H}$ of substructure weak learners, it is possible to build a function $F(\cdot; \boldsymbol{\alpha}) : \mathcal{S} \to \mathbb{R}$ such that there exists an $\epsilon > 0$ with*

$$\forall n \in \{1, \dots, N\} : \; y_n F(x_{s_n}; \boldsymbol{\alpha}) \geq \epsilon.$$

That is, a hard margin of ϵ is achieved.

Proof. We give an explicit construction for F. For a fixed constant $\rho > 0$, let $\beta \in \mathbb{R}^{\mathcal{S}}$ be defined as

$$\beta_{s_n} = y_n \rho - \sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \subseteq s_n}} \beta_{s_{n'}},$$

with $\beta_s = 0$ for all $s \notin X$, including $\beta_\emptyset = 0$. The coefficients α_ω are derived from β as

$$\alpha_{(t,d)} = |\beta_{s_n}|, \qquad t = s_n, \quad d = \operatorname{sign}(\beta_{s_n}).$$

First, we show that for the above construction of β and the derived $\boldsymbol{\alpha}$ we have $F(s_n; \alpha) y_n = \rho$ for all $s_n \in X$. Then we show that $\alpha_{(t,d)} \leq N\rho$ and thus normalization of α leads to a margin of at least $\frac{1}{N^3}$. From the definition of β and the identity $y_n^2 = 1$ we have

$$\begin{aligned}
& \beta_{s_n} = y_n \rho - \sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \subseteq s_n}} \beta_{s_{n'}} \\
\Leftrightarrow \quad & \rho = \Big(\sum_{\substack{s_{n'} \in X, \\ s_{n'} \subseteq s_n}} \beta_{s_{n'}} \Big) y_n \\
\Leftrightarrow \quad & \rho = F(s_n; \boldsymbol{\alpha}) y_n.
\end{aligned}$$

Now, we show that $\alpha_{(t,d)} \leq N^2 \rho$. To see this, note that

$$\begin{aligned}
\alpha_{(s_n,d)} = |\beta_{s_n}| \quad &= \quad |y_n \rho - \sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \subseteq s_n}} \beta_{s_{n'}}| \\
&\leq \quad |y_n \rho| + |\sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \subseteq s_n}} \beta_{s_{n'}}|
\end{aligned}$$

The last sum can alternatively be expressed as a sum of $F(\cdot; \boldsymbol{\alpha})$ evaluations:

$$\sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \subseteq s_n}} \beta_{s_{n'}} \quad = \quad \sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \sqsubset s_n}} F(s_{n'}; \boldsymbol{\alpha}) - \sum_{\substack{s_p \in X \setminus \{s_n\}, \\ s_p \subseteq s_n, s_p \not\subseteq s_n}} \tau_{s_p} F(s_p; \boldsymbol{\alpha}),$$

where $s_p \sqsubset s_q$ is the covering relation, i.e., $s_p \sqsubset s_q$ iff $s_p \neq s_q$, and $s_p \subseteq s_q$ and $\neg \exists s_k \in X \setminus \{s_p, s_q\} : s_p \subseteq s_k \subseteq s_q$. The coefficients $\tau_{s_p} \geq 0$ are the number of times the respective terms of β need to be removed, i.e., how often they are duplicated by the first F-terms. Let $k(s_n) = \sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \sqsubset s_n}} 1$ denote the number of F-terms under s_n, i.e., the number of terms in the first part of the decomposition. We have $k(s_n) \leq N - 1$ for all $s_n \in X$. From the poset ordering we further have

$$\sum_{\substack{s_p \in X \setminus \{s_n\}, \\ s_p \subseteq s_n}} \tau_{s_p} \leq (N - k(s_n)) k(s_n) + k(s_n) \leq N k(s_n).$$

Now, we can further bound

$$\begin{aligned}
|\beta_{s_n}| &\leq \rho + |\sum_{\substack{s_{n'} \in X \setminus \{s_n\}, \\ s_{n'} \sqsubset s_n}} F(s_{n'}; \boldsymbol{\alpha}) - \sum_{\substack{s_p \in X \setminus \{s_n\}, \\ s_p \subseteq s_n, s_p \not\sqsubset s_n}} \tau_{s_p} F(s_p; \boldsymbol{\alpha})| \\
&\leq \rho + k(s_n)\rho + |\sum_{\substack{s_p \in X \setminus \{s_n\}, \\ s_p \subseteq s_n, s_p \not\sqsubset s_n}} \tau_{s_p} F(s_p; \boldsymbol{\alpha})| \\
&\leq \rho + k(s_n)\rho + Nk(s_n)\rho \\
&\leq N^2\rho.
\end{aligned}$$

Therefore, we can normalize $\boldsymbol{\alpha}' = \frac{1}{\|\boldsymbol{\alpha}\|_1}\boldsymbol{\alpha}$ and have

$$\begin{aligned}
y_n F(x_n; \boldsymbol{\alpha}') &= y_n \left(\frac{1}{\|\boldsymbol{\alpha}\|_1} F(x_n; \boldsymbol{\alpha}) \right) \\
&= \frac{1}{\|\boldsymbol{\alpha}\|_1} \underbrace{y_n F(x_n; \boldsymbol{\alpha})}_{\rho} \\
&= \frac{1}{\sum_{s_n \in X} |\beta_{s_n}|} \rho \\
&\geq \frac{1}{\sum_{s_n \in X} N^2 \rho} \rho \\
&= \frac{1}{N^3}.
\end{aligned}$$

This completes the proof: every sample has a strictly positive margin with $\epsilon = \frac{1}{N^3}$. □

Note that the theorem does not state anything about the generalization performance of the constructed classification function. It simply asserts that the feature space has enough capacity to separate any given set of instances.

We now turn to the Boosting problem and how to solve it for our chosen weak learners. The key result that allows efficient solution of the subproblem is a monotonic upper bound on the Boosting subproblem objective due to Morishita[39] and later Kudo et al.[40]. We first state the bound, then describe how to use it for solving the Boosting subproblem over $\mathcal{H}$.

[39] Shinichi Morishita. Computing optimal hypotheses efficiently for boosting. In *Progress in Discovery Science*, volume 2281, pages 471–481. Springer, 2002. URL `http://citeseer.ist.psu.edu/492998.html`

[40] Taku Kudo, Eisaku Maeda, and Yuji Matsumoto. An application of boosting to graph classification. In *NIPS*, 2004

Theorem 2 (Bound on the Subproblem Objective (Morishita, Kudo)) *Given a substructure poset $(\mathcal{S}, \subseteq)$, a training set $X = \{(s_n, y_n)\}_{n=1,\ldots,N}$, with $(s_n, y_n) \in \mathcal{S} \times \{-1, 1\}$ and weight vector $\boldsymbol{\lambda} \in \mathbb{R}^N$ over the samples. Then*

$$\forall t \in \mathcal{S} : \forall (q, d) \in \Omega, q \subseteq t : \quad \sum_{n=1}^{N} \lambda_n y_n h(x_n; (q, d)) \leq \mu(t; X, \boldsymbol{\lambda}),$$

holds, where the upper bound $\mu : \mathcal{S} \to \mathbb{R}$ is defined as

$$\mu(t; X, \boldsymbol{\lambda}) = \max \left\{ 2 \sum_{\substack{n=1, \\ y_n=1, t \subseteq x_n}}^{N} \lambda_n - \sum_{n=1}^{N} \lambda_n y_n, \quad 2 \sum_{\substack{n=1, \\ y_n=-1, t \subseteq x_n}}^{N} \lambda_n + \sum_{n=1}^{N} \lambda_n y_n \right\}.$$

Proof. We have for an arbitrary $(t, d) \in \Omega$ that

$$\begin{aligned}\sum_{n=1}^{N} \lambda_n y_n h(x_n; (t,d)) &= \sum_{n=1}^{N} \lambda_n y_n (2I(t \subseteq x_n) - 1)d \\ &= \sum_{n=1}^{N} 2d\lambda_n y_n I(t \subseteq x_n) - \sum_{n=1}^{N} \lambda_n y_n d \\ &= 2d \sum_{\substack{n=1, \\ t \subseteq x_n}}^{N} \lambda_n y_n - \sum_{n=1}^{N} \lambda_n y_n d.\end{aligned}$$

Fixing $d = 1$ gives

$$= 2 \sum_{\substack{n=1, \\ t \subseteq x_n}}^{N} \lambda_n y_n - \sum_{n=1}^{N} \lambda_n y_n \leq 2 \sum_{\substack{n=1, \\ y_n = 1, t \subseteq x_n}}^{N} \lambda_n - \sum_{n=1}^{N} \lambda_n y_n = \mu_1(t; X, \boldsymbol{\lambda}).$$

Likewise, fixing $d = -1$ gives

$$= -2 \sum_{\substack{n=1, \\ t \subseteq x_n}}^{N} \lambda_n y_n + \sum_{n=1}^{N} \lambda_n y_n \leq 2 \sum_{\substack{n=1, \\ y_n = -1, t \subseteq x_n}}^{N} \lambda_n + \sum_{n=1}^{N} \lambda_n y_n = \mu_{-1}(t; X, \boldsymbol{\lambda}).$$

Both $\mu_1(t; X, \boldsymbol{\lambda})$ and $\mu_2(t; X, \boldsymbol{\lambda})$ are monotonically decreasing with respect to the partial order in their first terms. $\mu_1(t; X, \boldsymbol{\lambda})$ bounds the subproblem objective for all weak learners of the form $h(\cdot; (q, 1))$ with $q \subseteq t$, whereas $\mu_{-1}(t; X, \boldsymbol{\lambda})$ bounds the subproblem objective for all learners of the form $h(\cdot; (q, -1))$ with $q \subseteq t$. Thus, the overall bound is the maximum of the two, and by combining $\mu(t; X, \boldsymbol{\lambda}) = \max\{\mu_1(t; X, \boldsymbol{\lambda}), \mu_{-1}(t; X, \boldsymbol{\lambda})\}$ we obtain the result. □

We can use the upper bound $\mu(t; X, \boldsymbol{\lambda})$ to find the most discriminative weak learner if we can enumerate elements of $\mathcal{S}$ in such a way that we respect the partial ordering relationship, starting from $\emptyset$. We discuss enumeration of substructures in the next section.

Enumerating Substructures

For enumerating elements from $\mathcal{S}$ that satisfy the property we are interested in such as being discriminative or frequent, we will use the *reverse search* framework, a general construction principle for solving exhaustive enumeration problems. Avis and Fukuda[41] proposed the algorithm and applied it successfully to a large variety of enumeration problems such as enumerating all vertices of a polyhedron, all spanning trees of a graph and all subgraphs of a graph. Because we are interested in enumerating elements from $\mathcal{S}$, from now on we assume that $\mathcal{S}$ is countable.

[41] David Avis and Komei Fukuda. Reverse search for enumeration. *Discrete Appl. Math.*, 65:21–46, 1996

Definition 6 (Enumeration, Efficient Enumeration) *Given a substructure poset $(\mathcal{S}, \subseteq)$, and a function $g : \mathcal{S} \to \{\top, \bot\}$ satisfying anti-monotonicity,*

$$\forall s, t \in \mathcal{S} : (s \subseteq t \wedge g(t)) \Rightarrow g(s),$$

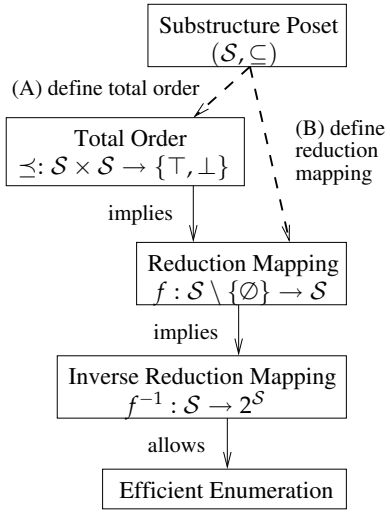


Figure 8: Dependencies for the substructure approach. The dashed arcs indicate possible alternatives: (A) we can either define a *total order* $\preceq$ which implies a reduction mapping, or (B) define the reduction mapping f directly. Once the reduction mapping is defined, its inverse f^{-1} and an efficient enumeration scheme follow.

the problem of listing all elements from the set

$$T_{(\mathcal{S},\subseteq)}(g) := \{s \subseteq \mathcal{S} : g(s)\}$$

is the enumeration problem *for g. An algorithm producing $T_{(\mathcal{S},\subseteq)}(g)$ is an* enumeration algorithm. *It is said to be* efficient *if its runtime is bounded by a polynomial in the output size, i.e., if there exists a $p \in \mathbb{N}$ such that its runtime is in $O(|T_{(\mathcal{S},\subseteq)}(g)|^p)$.*

THE IDEA OF REVERSE SEARCH is to *invert* a *reduction mapping* $f : \mathcal{S} \setminus \{\emptyset\} \to \mathcal{S}$. The reduction mapping reduces any element from $\mathcal{S}$ to a "simpler" one in the neighborhood of the input element. By considering the inverted mapping $f^{-1} : \mathcal{S} \to 2^{\mathcal{S}}$, an enumeration tree rooted in the $\emptyset$ element can be defined. Traversing this tree from its root to its leaves enumerates all elements from $\mathcal{S}$ exhaustively.

With an efficient enumeration scheme in place, we can solve interesting problem such as the frequent substructure mining problem, as well as the Boosting subproblem for substructure weak learners.

In order to apply reverse search to substructure posets a suitable reduction mapping needs to be defined. We take two alternative approaches to defining the reduction mapping. This is illustrated in Figure 8. First, given a substructure poset $(\mathcal{S}, \subseteq)$ we can choose to define the reduction mapping directly as shown as option (B) in the figure. Alternatively, we can instead define a total ordering relation on the set $\mathcal{S}$ which implies a canonical reduction mapping.

Depending on the kind of substructure it will be convenient to choose one option over the other. Later we we will use the total order definition for sets and graphs and the direct definition of the reduction mapping for labeled sequences.

But before we explain the total order construction, let us formalize the requirements to the reduction mapping in our context.

Definition 7 (Reduction Mapping) *Given a substructure poset $(\mathcal{S}, \subseteq)$, a mapping $f : \mathcal{S} \setminus \{\emptyset\} \to \mathcal{S}$ is a* reduction mapping *if it satisfies*

1. *covering:* $\forall s \in \mathcal{S} \setminus \{\emptyset\} : f(s) \sqsubset s$,
2. *finiteness:* $\forall s \in \mathcal{S} \setminus \{\emptyset\} : \exists k \in \mathbb{N}, k > 0 : f^k(s) = \emptyset$.

Thus the reduction mapping is defined such that when it is applied repeatedly, every element is eventually reduced to $\emptyset$.

Given f, the *inverse* of the reduction mapping is already well defined. Explicitly, we define it as follows.

Definition 8 (Inverse Reduction Mapping) *Given a substructure poset $(\mathcal{S}, \subseteq)$ and a reduction mapping $f : \mathcal{S} \setminus \{\emptyset\} \to \mathcal{S}$, the* inverse reduction mapping *$f^{-1} : \mathcal{S} \to 2^{\mathcal{S}}$ is*

$$f^{-1}(t) = \{s \in \mathcal{S} | f(s) = t\}.$$

We now describe how we can use a total order on $\mathcal{S}$ to construct f and f^{-1} for substructure posets, and then describe the general reverse search algorithm.

Constructing the Reduction Mapping from a Total Order

If we are given a total order $\preceq\colon \mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$, we show how we can use it to define a canonical reduction mapping. A total order on $\mathcal{S}$ satisfies the following total order assumption.

Assumption 1 (Total Order Assumption) *Given a substructure poset $(\mathcal{S}, \subseteq)$ we assume we are given a* total order $\preceq\colon \mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$. *A total order satisfies for all $s, t, u \in \mathcal{S}$,*

1. $s \preceq t \wedge t \preceq s \Rightarrow s = t$ *(antisymmetry),*
2. $s \preceq t \wedge t \preceq u \Rightarrow s \preceq u$ *(transitivity),*
3. $s \preceq t \vee t \preceq s$ *holds (totality).*

The total order assumption allows us to define a *reduction mapping* which maps structures from $\mathcal{S}$ to successively "simpler" structures.

Definition 9 (Reduction Mapping derived from $(\mathcal{S}, \subseteq)$ and $\preceq$) *Given a substructure poset $(\mathcal{S}, \subseteq)$ and a total order $\preceq\colon \mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$ satisfying the finite preimage property*

$$\forall s \in \mathcal{S} : |\{t \in \mathcal{S} : t \subseteq s\}| < \infty,$$

we define a reduction mapping $f : (\mathcal{S} \setminus \{\emptyset\}) \to \mathcal{S}$ as

$$f(s) = \{t \in \mathcal{S} : (t \sqsubset s \text{ and } \forall u \sqsubset s : t \preceq u)\}.$$

The mapping f is well-defined. For the case $s \neq \emptyset$, the expression $t \sqsubset s$ with $\forall u \sqsubset s : t \preceq u$ yields a unique element $t \in \mathcal{S}$ because $\preceq$ is a total order, hence if there exists a $t \sqsubset s$, there exist a unique minimal one. But there always exists a $t \sqsubset s$ because $\emptyset \subseteq s$ for all s and $\subseteq$ is a partial order. Furthermore, assuming $\mathcal{S}$ is countable, by recursively applying f we eventually reach the $\emptyset$ element.

We illustrate this construction for the case of sets. Assume a *finite* set of base elements, $\Sigma = \{1, 2, 3\}$. Now set $\mathcal{S} = 2^\Sigma$ to be the power set. The usual subset relation $\subseteq$ is a partial order and can be visualized in terms of a Hasse diagram, as shown in Figure 9. We define a total order $\preceq$ as follows.

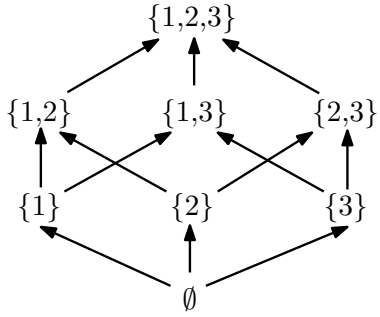


Figure 9: Hasse diagram of the $\subseteq$ relation over the set $\mathcal{S} = 2^\Sigma$ with $\Sigma = \{1, 2, 3\}$.

Example 1 (Total Order for Sets) *Given a finite alphabet Σ with canonical total order $\leq\colon \Sigma \times \Sigma \to \{\top, \bot\}$ and let $\mathcal{S} = 2^\Sigma$. Then we define $\preceq\colon \mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$ to be a total order defined on sets as lexicographic order applied to the ordered concatenation of elements from Σ. That is, for any $s, t \in \mathcal{S}$, define $s \preceq t$ true if*

$$(s_1, s_2, \ldots, s_{|s|}) \preceq (t_1, t_2, \ldots, t_{|t|}),$$

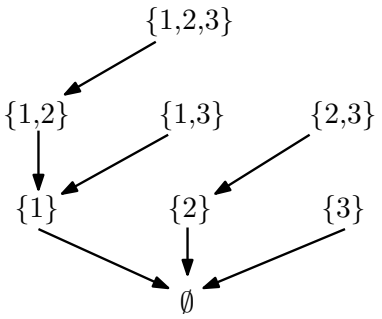


Figure 10: Reduction mapping $f : (\mathcal{S} \setminus \{\emptyset\}) \to \mathcal{S}$ induced by $(\mathcal{S}, \subseteq)$ and the total order $\preceq$.

where $(s_1, s_2, \ldots, s_{|s|})$, *and* $(t_1, t_2, \ldots, t_{|t|})$, *are the ordered elements of* s *and* t, *respectively, and* $\preceq$: $\Sigma^* \times \Sigma^* \to \{\top, \bot\}$ *is the* lexicographic order *defined as* $(s_1, s_2, \ldots, s_{|s|}) \preceq (t_1, t_2, \ldots, t_{|t|})$ *being true if*

- $\exists k, 1 \leq k \leq \min\{|s|, |t|\} : \forall i < k : s_i = t_i$ *and* $s_k \leq t_k$, *or*
- $|t| \geq |s|$, *and* $\forall k, 1 \leq k \leq |s| : s_k = t_k$.

For example, the structures shown in Figure 9 would be ordered according to $\emptyset \preceq \{1\} \preceq \{1,2\} \preceq \{1,2,3\} \preceq \{1,3\} \preceq \{2\} \preceq \{2,3\} \preceq \{3\}$.

We now have all ingredients in order to apply the above definition to derive a reduction mapping.

The reduction mapping is visualized in Figure 10. Each element $s \in 2^U$ except for the empty set is mapped to a unique element t such that $t \sqsubset s$. As discussed above this induces a tree rooted in $\emptyset$.

The reduction mapping $f : (\mathcal{S} \setminus \{\emptyset\}) \to \mathcal{S}$ reduces an element such that it eventually becomes the $\emptyset$ element. The inverse reduction mapping $f^{-1} : \mathcal{S} \to \mathcal{S}$ expands an element $t \in \mathcal{S}$ to the set of possible extensions $t \sqsubset s$.

Inverse Reduction Mapping Derived From a Total Order

The *inversion* of the reduction mapping derived from the total order follows from the total order itself. Because it is an important ingredient in the reverse search scheme when using the total order construction, we define it explicitly.

Lemma 3 (Inverse Reduction Mapping given a Total Order) *Given a substructure poset* $(\mathcal{S}, \subseteq)$ *and a total order* $\preceq$: $\mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$, *the inverse reduction mapping*

$$f^{-1}(t) = \{s \in \mathcal{S} | f(s) = t\}$$

can equivalently be defined as

$$f^{-1}(t) = \{s \in \mathcal{S} | t \sqsubset s \text{ and } \forall u \sqsubset s : t \preceq u\}.$$

Proof. From its definition the inverse of the reduction mapping needs to satisfy the following two conditions.

1. $\forall t \in \mathcal{S} : \forall s \in f^{-1}(t) : t = f(s)$, and
2. $\forall s \in \mathcal{S} \setminus \{\emptyset\} : t = f(s) \Rightarrow s \in f^{-1}(t)$.

The above mapping satisfies both properties. To see the first point, fix $t \in \mathcal{S}$ arbitrarily, choose any $s \in f^{-1}(t)$. We have for s that $t \sqsubset s$ and $\forall u \sqsubset s : t \preceq u$, and therefore by definition $t = f(s)$. To see the second point, choose $s \in \mathcal{S} \setminus \{\emptyset\}$ and let $t = f(s)$. Then we have again $t \sqsubset s$ and $\forall u \sqsubset s : t \preceq u$, so $s \in f^{-1}(t)$. □

The inverse mapping is visualized in Figure 11. It corresponds to reversing the direction of all arcs shown in Figure 10.

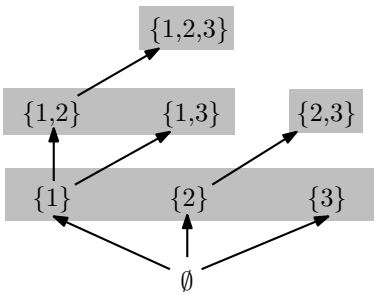


Figure 11: Illustration of the inverse reduction mapping $f^{-1} : \mathcal{S} \to 2^{\mathcal{S}}$. Each element $s \in \mathcal{S}$ is mapped to a set of larger elements satisfying $s \sqsubset t$. The inverse mapping induces an enumeration tree rooted in $\emptyset$. The elements within one gray box are the output of the inverse reduction mapping applied to their parent.

Algorithm 2 Enumerate All Property-Satisfying Elements in $\mathcal{S}$

ReverseSearch$((\mathcal{S}, \subseteq), f^{-1}, s_0, g)$
Input:
  $(\mathcal{S}, \subseteq)$, substructure poset
  $f^{-1} : \mathcal{S} \to 2^{\mathcal{S}}$, inverse reduction mapping
  $s_0 \in \mathcal{S}$, root element for which $g(s_0) = \top$
  $g : \mathcal{S} \to \{\top, \bot\}$, property, anti-monotone with respect to $\subseteq$
Output:
  $T \subseteq 2^{\mathcal{S}}$, the set of all substructures $s \in \mathcal{S}$ for which $g(s)$ holds
Algorithm:
output s_0
for $t \in \{s \in f^{-1}(s_0) | g(s) = \top\}$ **do**
  ReverseSearch$((\mathcal{S}, \subseteq), f^{-1}, t, g)$
end for
return

Reverse Search Algorithm

The general *reverse search* algorithm is shown in Algorithm 2. When invoked as ReverseSearch$((\mathcal{S}, \subseteq), f^{-1}, \emptyset, g)$, the algorithm enumerates all elements from $\mathcal{S}$ that satisfy the given predicate g. To see the correctness of the algorithm, note that recursing along f^{-1} generates each element in $\mathcal{S}$ exactly once. Pruning subtrees at s when $g(s) = \bot$ does not skip over elements for which g would be true, because g is anti-monotone with respect to $\subseteq$.

We now show how Algorithm 2 can be used to solve the frequent substructure mining problem. We also show how to find *discriminative* substructures that solve the Boosting subproblem.

First, the Frequent Substructure Mining Problem (Problem 2). Given a substructure poset $(\mathcal{S}, \subseteq)$ and a set of structures $X = \{s_n\}_{n=1,\ldots,N}$ with $s_n \in \mathcal{S}$ we define g as

$$g_{\text{fsm}}(s; X, \sigma) = (\text{freq}(s, X) \geq \sigma). \tag{20}$$

We see that g_{fsm} is anti-monotone with respect to $\subseteq$. Running Algorithm 2 as ReverseSearch$((\mathcal{S}, \subseteq), f^{-1}, \emptyset, g_{\text{fsm}})$ will thus enumerate exactly all σ-frequent substructures.

Second, the discriminative substructure mining problem (Problem 1 for the Substructure Boosting Weak Learner). Given a substructure poset $(\mathcal{S}, \subseteq)$ and a labeled training set $X = \{(s_n, y_n)\}_{n=1,\ldots,N}$ with $(s_n, y_n) \in \mathcal{S} \times \{-1, 1\}$, and given a weight vector $\boldsymbol{\lambda} \in \mathbb{R}^N$, we define g as

$$g_{\text{dsm}}(s; X, \boldsymbol{\lambda}) = (\mu(s; X, \boldsymbol{\lambda}) \geq \sigma(t)), \tag{21}$$

where $\sigma(t)$ is a monotonically increasing minimum required gain. For example, if during the course of the algorithm a set of substructures $\{q_1, q_2, \ldots, q_k\}$

has been produced as output, $\sigma(t)$ could be defined as

$$\sigma(t) = \max_{i=1,\ldots,k} \left\{ \sum_{n=1}^{N} \lambda_n y_n h(\boldsymbol{x}_{q_i}; \omega_i) \right\}.$$

In this case, the algorithm would prune subtrees at s for which the bound $\mu(s; X, \boldsymbol{\lambda})$ states that it is impossible to exceed the gain of the best found substructure so far. The algorithm is guaranteed to output the substructure with the best gain.

In the next two chapters we will use the above algorithms in a concrete fashion for classifying graphs and sequences. Using the above bound and enumeration method during Boosting we can efficiently find discriminative weak learners.

Online Generation of f^{-1}, An Example

In the reverse search algorithm, the set $\{s \in f^{-1}(t) | g(s) = \top\}$ of enlarged substructures needs to be generated. In principle this can be achieved by first generating $f^{-1}(t)$ and then filtering out all elements which do not satisfy $g(s) = \top$. However, when the set $f^{-1}(t)$ is large and the condition encoded in g is stringent this can be inefficient. It is therefore better to *directly* generate the filtered set.

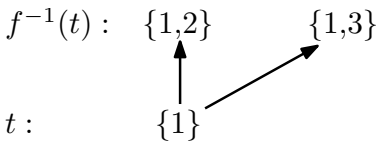


Figure 12: Extension of $t = \{1\}$ to $f^{-1}(t) = \{\{1,2\},\{1,3\}\}$.

Direct generation requires an algorithm which can use the structure present in g. We show how this can be achieved for the example of sets. Consider the situation shown in Figure 12. We have

$$\begin{aligned} \Sigma &= \{1,2,3\}, \\ X &= (\{1\},\{1,2\},\{1,2,3\},\{3\}), \\ t &= \{1\}, \\ f^{-1}(t) &= \{\{1,2\},\{1,3\}\}, \end{aligned}$$

and let

$$g(s) = (\text{freq}(s; X) \geq 2).$$

Thus, the set of interest is

$$\{s \in f^{-1}(t) | g(s) = \top\} = \{\{1,2\},\{1,3\}\}.$$

To generate $f^{-1}(t)$ from the definition and the total order $\preceq$, we have

$$\begin{aligned}
f^{-1}(t) &= \{s \in \mathcal{S} | t \sqsubset s \text{ and } \forall u \sqsubset s : t \preceq u\} \\
&= \{s \in \mathcal{S} | t \sqsubset s \wedge \forall u \sqsubset s : \\
&\qquad [(\exists k, 0 \leq k \leq |t| : \forall i < k : t_i = u_i \wedge t_k \leq u_k) \\
&\qquad\quad \vee (|u| \geq |t| \wedge \forall k, 0 \leq k \leq |t| : t_k = u_k)]\} \\
&= \{s \in \mathcal{S} | t \sqsubset s \wedge \forall u \sqsubset s : \\
&\qquad \exists k, 0 \leq k \leq |t| : \forall i < k : t_i = u_i \wedge t_k \leq u_k\} \\
&= \{t \cup \{e\} | e \in (\Sigma \setminus t) \wedge \forall e' \in (t \cup \{e\}) : e' \leq e\} \\
&= \{t \cup \{e\} | e \in \Sigma \text{ and } e > \max_{j \in t} j\},
\end{aligned}$$

such that $f^{-1}(t)$ simply enlarges t by one element from the ground set Σ. The additional element must be strictly larger than the largest element already in t. In the figure, the elements $2 \in \Sigma$ and $3 \in \Sigma$ satisfy this. The condition g can now be incorporated into the inverse reduction mapping as follows.

$$\begin{aligned}
& \{s \in f^{-1}(t) | g(s) = \top\} \\
= \; & \{s \in \{t \cup \{e\} | e \in \Sigma \text{ and } e > \max_{j \in t} j\} | g(s) = \top\} \\
= \; & \{t \cup \{e\} | e \in \Sigma \text{ and } e > \max_{j \in t} \text{ and } \text{freq}(t \cup \{e\}; X) \geq 2\} \\
= \; & \{t \cup \{e\} | e \in \Sigma \text{ and } e > \max_{j \in t} j \text{ and } \text{freq}(t; X) \geq 2 \text{ and} \\
& \qquad \sum_{\substack{n=1,\ldots,N, \\ t \subseteq s_n}} I(e \in s_n) \geq 2\} \\
= \; & \{t \cup \{e\} | e \in \Sigma \text{ and } e > \max_{j \in t} j \text{ and } \sum_{\substack{n=1,\ldots,N, \\ t \subseteq s_n}} I(e \in s_n) \geq 2\}
\end{aligned}$$

Now it is clear how to enlarge the structure t to produce the subset of $f^{-1}(t)$ which satisfies g. We have to consider the structures in X for which t is already frequent and for this set find all elements in Σ which are both larger than the highest value in t and frequent. Depending on the data structure used, it is possible to obtain only the frequent elements. This is not possible in the original filter approach, where all sets in $f^{-1}(t)$ need to be first generated explicitly.

Further Improvements

Although we focus here on the general framework for substructure-based classification, we want to note that further improvements on Algorithm 2 are possible. First, note that for the discriminative substructure mining problem we are using a surrogate bound on the gain of a substructure, the true quantity of interest being the gain. In case we explore parts of the enumeration tree where there is no discriminative substructure we can only prune in case the

bound is tight enough. Ideally, we would know the tightest possible bound, the true gain-maximizing substructure in the respective substree.

This observation allows the first improvement: we first use an inexact method such as a greedy depth-first traversal or a beam search on the enumeration tree in order to obtain a good lower bound $\sigma(0)$ on the achievable gain. Thereafter an exact method can be run using the greedy solution to provide a global lower bound on the gain.

The second idea to improve the algorithm is related: the traversal order can be modified to reach a high-gain discriminative substructure early. This is in contrast to the frequent substructure mining problem: there, the traversal order is not important and all frequent substructures are of interest. Because all frequent substructures are traversed exactly once we cannot gain anything by choosing a different enumeration order.

This is different for discriminative mining, where it helps to discover a high-gain substructure *early* in the enumeration as this allows efficient pruning. This can be achieved by extending the above algorithm from simple enumeration to keeping and updating a search frontier in promising directions. In Nowozin et al.[42] we successfully applied this idea by using $A*$-enumeration and iterative deepening $A*$ enumeration[43]. Using a search frontier allows one to extend different parts of the enumeration tree in parallel and once a high-gain substructure is observed, a large part of the current search frontier can be pruned. The scheme works well in practice because often the most discriminative substructures turn out to be rather small. The search frontier scheme typically searches through the small set first and thus obtains a good bound early.

[42] Sebastian Nowozin, Gökhan Bakır, and Koji Tsuda. Discriminative subsequence mining for action classification. In *ICCV 2007: Proceedings of the 2007 IEEE Computer Society International Conference on Computer Vision*, 2007

[43] Nils J. Nilsson. *Artificial Intelligence: A New Synthesis*. Morgan Kaufmann Publishers, San Francisco, 1998. ISBN 1558604677

Conclusion

In this chapter we introduced substructures and defined an associated feature space in which each possible substructure is represented by a binary feature. The problem of applying Boosting in this feature space was then discussed and a general algorithmic framework for identifying discriminative substructures has been proposed.

In the next two chapters we apply the framework to two computer vision tasks, class-level object recognition in still images and action recognition in videos. The applications use graphs and sequences as substructures and the concepts of the current chapter are further illustrated by them.

Graph-based Class-level Object Recognition

> The more we look for patterns, the more likely we are to find them, particularly when we don't begin with a particular question.
>
> Peter Austin

THE SUBSTRUCTURE POSET FRAMEWORK introduced in the previous chapter allows feature induction in large, structured feature spaces. This chapter is about applying the framework to images in order to decide the presence or absence of objects of a particular class.

The key contributions of this chapter is a principled way of incorporating higher order geometric relations between local parts into class-level object recognition models. This is achieved by means of the substructure poset framework. Furthermore, the proposed approach is assessed experimentally.

Introduction

Images of natural scenes contain a lot of structure. For one, there is the fundamental structure contained in the statistics of the signal, such as the characteristic distribution of image gradients in natural images. But also, on the high semantic level there is a structure inherent in objects, textures, geometry, context, and scene composition. This high-level structure is not a result of the image formation process, but instead exists in the real world.

Class-level object recognition is the problem of detecting the existence and possibly additional spatial information of objects in images, where the objects to be recognized are not particular instances ("my bicycle") but are members of a class ("all bicycles"). Whereas the problem of recognizing particular instances is largely solved in computer vision, recognizing objects on a class level remains a difficult problem.

The larger part of the difficulty of class-level object recognition is due to the variability of objects in the real world. My bicycle might look quite different from another bicycle, and no dog looks like another. What is shared by *all instances* of an object class is often less the visual appearance than abstract attributes describing functional purpose, compositionality and geometry, physical properties or generative history. For example, a bicycle is defined in

[44] http://wordnet.princeton.edu/

WordNet[44] as "a wheeled vehicle that has two wheels and is moved by foot pedals" and a dog is defined in WordNet as "a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds". Both definitions do not describe visual properties of the object but accurately describe the members in these object classes. Therefore, for class-level object recognition the visual properties observed in images can merely serve as a proxy to the true semantic properties that define an object class.

Moreover, even for visually very similar objects *there are* differences in visual appearance caused by changes in lighting, color, texture, size and shape of objects and the scene.

MODELS FOR OBJECT RECOGNITION face these difficulties. It is fair to say that while no best-practice model has emerged the typical model consists of a fixed part incorporating domain knowledge and a machine learning part adapting to different instances of the problem, such as different object classes. For example, in the fixed part many models use image features which incorporate knowledge about properties that remain invariant under various lighting conditions. Another model part that often remains fixed is the model structure, representing dependence assumptions and simplifications between parts of the model. The machine learning part is often a parametrized function representing either a distribution or classification function.

A consistent trend in models for class-level object recognition is the use of object *parts*, reusable and transferable descriptions of parts of objects. Similar parts appearing in multiple objects can be jointly learned and flexibly combined with other parts to yield an overall object description. We will discuss the advantages of part-based models in detail in a later section, but in essence the use of part-based representations allows expressive but compact models.

In the machine learning part of the model the modeling decisions made determine a tradeoff between the feasibility of approximation, estimation and optimization of the resulting model[45]. *Approximation* refers to the expressiveness of the model, the ability to accurately represent the problem data. *Estimation* is the ability to statistically estimate the parameters of the model from a finite amount of observed training data. Finally, *optimization* is the tractability of the resulting model: even if its possible to estimate the correct model parameters, is it computationally tractable?

[45] Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *NIPS*, 2007

To give an example, a simple linear classification function on a small set of simple image features will not yield a very expressive model but its parameters can be estimated from few training instances and the optimization is very efficient even for large data sets. In contrast, a deep convolutional neural network covers a much larger set of classification functions but its many parameters and model symmetry make it difficult to assess estimation properties and the non-convexity of its training objective make optimization

difficult.

OUTLINE. In the remaining part of this chapter we first motivate part-based models for object recognition and then give an extensive literature survey. Then we introduce graph-based object recognition using the substructure poset framework of the previous chapter and describe in detail the algorithms necessary to perform learning in a feature space defined by subgraph features. The remaining part of the chapter is an extensive experimental evaluation on the PASCAL VOC 2008 data set and describes how we transform images into graph structures. We end the chapter with conclusions.

Related Work: Part-based Object Recognition

The idea that natural everyday objects can be visually decomposed into meaningful parts is as old as the attempts to understand the human vision system. Biederman[46] gives a summary of the early psychology literature related to this idea.

[46] Irving Biederman. Recognition by components - a theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987

Extensive experiments by Biederman and others suggest that object recognition in humans uses a mechanism that, i) does not require absolute or precise quantitative information, ii) is invariant with respect to changes in orientation, and iii) continues to function when the object is partially occluded or is a new type within the object class, resembling other previously seen instances only partially.

As of today humans are still vastly outperforming computers on almost all visual recognition tasks. Therefore, besides the biological motivation for understanding and modeling the human visual system, understanding the human visual system might also shed light on fundamental principles that could aid in designing computer vision systems.

THE ABOVE THREE REQUIREMENTS motivate the design of *statistical*, *part-based* models for recognizing objects in images as follows. First, the model should be *statistical* because no component of the model is free from noise and ambiguities; the input image is noisy, statistics in the form of image features are noisy, and intermediate states or final decisions of the model are never completely certain. Detecting objects, that is, reasoning about the input data in order to make a decision about the presence of an object requires an inference which takes into account uncertainty at *all* levels, the very definition of a statistical model.

Second, the model should be *part-based*. While it is difficult to find a satisfying definition of "part" we understand as *part-based model* a system which explicitly or implicitly can take into accounts groups of image statistics in a non-additive manner, i.e., the influence of a *group* of image statistics depends non-linearly on the individual statistics within the group. Note that under this broad definition essentially all successful general object recognition

systems are part-based, as they include nonlinearities at the feature extraction or classification stage.

The number of proposed statistical, part-based models in the computer vision literature is large. In the remainder of this section we provide an overview of the most important models, but first we digress briefly to discuss the issues of *label granularity* and *training* of these models.

LABEL GRANULARITY refers to the level of detail of the available annotation for the training data. Some training procedures for part-based models require very careful annotation of a number of pre-specified parts of the objects shown in the training images. For example, it must be prespecified that "a car in sideview has two visible wheels" and the "wheel"-parts must be labeled by the user. Other models require weaker labels only, such as a bounding box around the object instances shown in the images. The weakest annotation contains only the information that an object instance is shown somewhere in the image. The weaker the annotation, the more is demanded from the model. In essence, training the model might mean to simultaneously recognize the location of the object, a set of suitable parts and their appearance for all images in the training set. In the machine learning literature the problem of weak labels has partially been discussed as the *multiple instance learning* problem.

THE TRAINING PROCEDURE is essential in judging a model because an expressive model that cannot be trained in a tractable way is essentially useless. This does not mean that efficiency should be the primary design goal but that a model that does not scale to today's datasets will impose unnecessary limits on what it can learn in practice, even if it could do so in principle. For this reason, many approaches deal with tractable approximations to a more desirable model that is intractable.

Literature Survey

We survey and categorize the proposed models for part-based object recognition. Table 1 summarizes the surveyed approaches into a set of properties, defined as follows.

- **explicit parts**: the ability of the model to represent and identify parts explicitly with a single portion of the image,
- **multiple objects**: the ability of the model to naturally handle multiple objects of a given class within one image, without referring to sliding window wrapper methods,
- **prediction output**: the final prediction output of the model, i.e., whether only the presence of an object is indicated or a precise part localization is delivered,

- **parts selected by learning**: whether the identity of parts is established during the training phase,
- **scale invariance**: whether the approach can handle multi scale detections without referring to explicit scaling of the image,
- **variable number of parts**: whether the number of parts is variable during training and detection,
- **label granularity**: what level of details is required for the labels during training,
- **geometry between parts**: whether the approach incorporates geometry between parts,
- **pairwise relations**: whether the approach encodes pairwise part-to-part geometry information,
- **higher-order-relations**: whether the approach can encode higher-than-pairwise information, for example a constellation of triples of parts,
- **comparison with baseline**: whether the publication compares the approach against a baseline not within the model family.

This classification scheme is not exhaustive but covers the most relevant aspects of the compared models.

Literature Survey: Constellation Models

Burl, Weber and Perona[47] propose a joint probabilistic model integrating local part similarity with a global shape prior. The local appearance is modeled by means of matched filters obtained from manual part-level annotations. The shape prior is a Gaussian fitted to shape statistics obtained from an annotated training set. The proposed joint criterion for recognition turns out to be hard to optimize so the authors propose a set of heuristics. Experimental evaluation is performend on the task of recognizing faces by means of facial parts.

[47] Michael C. Burl, Markus Weber, and Pietro Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *ECCV*, pages 628–641, 1998

Weber, Welling and Perona[48] extend the model of Burl et al. by addressing the problem of weak annotation in a thorough probabilistic model. Given a set of images known to contain either objects of a single unknown class or background only, Weber proposes a model that can simultaneously learn the object class as a combination of parts and their constellation. The unobserved states of object presence and part selection are treated by means of expectation maximization (EM), providing a local maximum of the likelihood of the observed states, the image and its parts. Parts are modeled sparsely at interest points. Each part is represented as normalized correlation filter responses of a small set of filters produced by clustering training data patches. Shape is modeled by assigning each part a 2D Gaussian distribution encoding the

[48] Markus Weber, Max Welling, and Pietro Perona. Unsupervised learning of models for recognition. In *ECCV*, 2000

relative coordinates with respect to a reference part. Thus, although robust to small changes, the shape representation does not encode pairwise relations.

Li, Fergus and Perona[49] use a similar approach as Weber et al., but focus on the problem of learning an object class when only very few labeled training instances are available. To this end, Li et al. propose a generative graphical model where object classes are represented by parametric probabilistic models and a shared prior is represented as a distribution on the parameters of the class models. The assumption that a joint prior can allow generalization across object classes is demonstrated experimentally, however, as with the constellation model of Weber et al., the model is limited to a small number of local features (≈ 40) and an even smaller number of parts (≈ 5). The work is particularly interesting for its principled use of Bayesian techniques to faithfully represent uncertainty arising from the limited training data.

[49] Fei-Fei Li, Robert Fergus, and Pietro Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *ICCV*, 2003

Fergus, Perona and Zisserman[50] extend the constellation model of Burl et al. and Weber et al. in two ways. First, the appearance of a part is modeled as a multivariate Gaussian distribution in an appearance space created by the first ten principal components of small image patches. The distribution parameters are hidden and learned using expectation maximization. Second, Fergus et al. achieve scale-invariance learning by detecting candidate parts using a scale-invariant interest point detector, extracting a fixed small number (≈ 30) of interesting image regions. The model is shown to work well experimentally on six object classes, including non-rigid classes.

[50] Robert Fergus, Pietro Perona, and Andrew Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages 264–271, 2003

Felzenszwalb and Huttenlocher[51] directly extend the pictorial structures model of Fischler and Elschlager in three important directions. First, the model of Fischler is made statistical by representing a distribution of all possible part configurations, allowing analysis of the posterior of all possible part configurations. Felzenszwalb and Huttenlocher carry out this analysis by means of sampling in order to find multiple likely configurations as well as finding multiple objects within one image. Second, facilitated by the statistical view, Felzenszwalb shows that during parameter estimation by means of maximum likelihood the parts model decouple and can be learned separately. Moreover, when limited to tree structured part distributions, the tree structure can be learned as well using a modified Chow-Liu tree procedure[52]. Third, the authors identify an important class of restricted deformation potentials for which the MAP estimation problem can be solved in $O(nh)$ time complexity for n parts and h possible individual part positions. The original general algorithm of Fischler and Elschlager had a complexity of $O(nh^2)$. The restricted potentials are of the form $\psi(x,y) = (x-y)^\top D(x-y)$, where $D \succeq 0$ is diagonal and x, y denote the vectorial coordinates of two parts sharing an edge. Felzenszwalb and Huttenlocher evaluate their system on face detection and human pose estimation tasks, demonstrating the models' robustness to noise. Moreover, the authors demonstrate that the model learns intuitively plausible part layouts.

[51] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005

[52] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968

Crandall, Felzenszwalb and Huttenlocher[53] propose a flexible family of

[53] David J. Crandall, Pedro F. Felzenszwalb, and Daniel P. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *CVPR*, 2005

constellation models called *k*-fans which have a graphical structure as shown in Figure 13.

Publication	Year	Explicit parts	Multiple objects	Prediction output	Parts selected by learning	Scale invariance	Variable number of parts	Label granularity	Geometry between parts	Pairwise relations	Higher-order relations	Comparison with baseline
Fischler, Elschlager	1973	yes	no	part positions (L)	no	(no)	no	superv., part-label	yes	yes	no	no
Burl, Weber, Perona	1998	yes	no	part positions (L)	no	no	no	superv., part-label	yes	no	no	no
Weber, Welling, Perona	2000	yes	no	object presence (C)	yes	yes	(yes)	unsuperv., one-class	yes	no	no	no
Li, Fergus, Perona	2003	yes	no	object presence (C)	yes	yes	(yes)	unsuperv., one-class	yes	no	no	no
Fergus, Perona, Zisserman	2003	yes	no	object presence (C)	yes	yes	no	unsuperv., one-class	yes	no	no	no
Felzenszwalb, Huttenlocher	2005	yes	yes	part positions (L)	no	no	no	superv., part-label	yes	(tree)	no	no
Crandall, Felzenszwalb, Huttenlocher	2005	yes	no	part positions (L)	yes	no	no	superv., part-label	yes	yes	yes	(no)
Quattoni, Collins, Darrell	2004	yes	no	object presence (C)	yes	(yes)	no	superv., image label	yes	(yes)	no	no
Winn, Shotton	2006	(yes)	yes	segmentation (S)	no	no	no	superv., segmentations	yes	(yes)	(yes)	no
Hoiem, Rother, Winn	2007	(yes)	yes	segmentation (S)	no	no	no	superv., segmentations	yes	(yes)	(yes)	no
Schneiderman, Kanade	1998	no	(yes)	object position (L)	no	no	no	superv., bbox	yes	no	no	(yes)
Papageorgiou, Poggio	2000	no	yes	object position (L)	(yes)	no	(no)	superv., bbox	yes	no	no	yes
Viola, Jones	2001	no	yes	object position (L)	(yes)	no	(no)	superv., bbox	yes	no	no	yes
Felzenszwalb, McAllester, Ramanan	2008	yes	no	part positions (L)	(no)	no	no	superv., bbox	yes	no	no	yes
Krempp, Geman, Amit	2002	yes	yes	object position (L)	yes	no	yes	superv., bbox	yes	no	no	no
Agarwal, Awan, Roth	2004	yes	yes	object position (L)	(yes)	(yes)	yes	superv., bbox	yes	(yes)	no	yes
Lazebnik, Schmid, Ponce	2005	yes	no	object presence (C)	(yes)	yes	yes	superv., image label	yes	yes	yes	yes
Nowozin, Tsuda, Uno, Kudo, Bakır	2007	no	yes	object presence (C)	yes	yes	yes	superv., image label	yes	yes	yes	yes

Table 1: Popular part-based object recognition approaches from the computer vision literature. The *predicted output* is one of (C), (L), (S), where (C) is the binary decision of deciding the presence of an object on the image, (L) is a predicted image location — for example by means of a bounding box — for the object, and (S) is providing a per-pixel image segmentation into object/background classes. The *label granularity* of the training labels is either unsupervised (no labels) or supervised. The supervised training annotations are either per-image labels, bounding box (bbox) annotations or specific part annotations. Attributes (yes) and (no) denoted in brackets are partially satisfied and do not completely match the attribute description.

A small number of reference parts are fully connected to each other, whereas the remaining parts have their position determined relative to the reference parts only. Thus, denoting by l_i the location of the i'th part and by l_R the set of locations of all reference parts, where R is the set of reference parts, the joint probability $p(L)$ of all parts L factorizes according to $p(L) = p(l_R) \cdot \prod_{i \in V \setminus R} p(l_i|l_R)$. This special structure allows efficient inference for the case when K is small. For an n-part model with $k \leq n$ reference parts and h possible part locations in the image, Crandall et al. show how exact inference can be performed in $O(nh^{k+1})$ time complexity. Experimentally, the higher order spatial constraints enforced by the model are shown to improve detection performance on aeroplane and bicycle objects, using simple edge-map features.

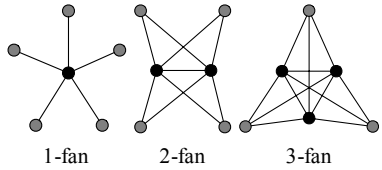


Figure 13: Crandall's k-fan models of increasing complexity. Conditioned on the k reference parts (black), the remaining parts (gray) become independent of each other. (Reproduced from Crandall, Felzenszwalb and Huttenlocher's original paper.)

Fischler and Elschlager[54], almost forty years ago considered in a very general setting the problem of recognizing objects in images, where a deformable parts model and a scoring function define the quality of a located object.

The scoring function considers both the matching of local appearance as well as overall consistent geometry. The optimal configuration of parts which minimizes the scoring function for a given image is found by means of a dynamic programming procedure, much alike the max-product message passing procedure for undirected Markov networks. Fischler and Elschlager differentiate between a tree-structured graph of springs and general graphs containing cycles. For the latter, they propose a linear-time complexity heuristic, iteratively fixing one variable at a time. To appreciate this influential paper further, some additional remarks are necessary.

[54] Martin A. Fischler and Robert A. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Computer*, 22(1):67–92, January 1973

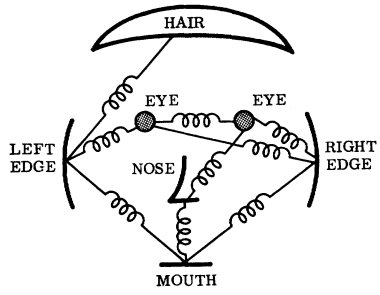


Figure 14: Fischler and Elschlager's spring model (1973) for object recognition. Each part (eye, mouth, etc.) has its own appearance model. A deformation model consisting of pairwise deformation potentials (springs) require the parts to have a consistent layout within the scene. (Figure reproduced from Fischler and Elschlager's original paper.)

First, the direct minimization of a scoring function in order to find a good configuration, now a very popular technique in computer vision named *energy minimization*, is broadly motivated and possible criticism anticipated when Fischler states,

> "...without a noise and distortion model, there is no theoretically valid way to derive or predict the error performance of a selected procedure prior to its actual application."

And indeed up to today it appears difficult to explicitly state a noise and distortion model suitable to high-level vision tasks such as object recognition. Fischler and Elschlager realize that it is not necessary to do so explicitly.

Second, Fischler and Elschlager's model is a precursor to the advanced Markov random field (MRF) models which now permeate many subfields of computer vision research. In fact, their model is *exactly* a MRF with pairwise potentials coming from the deformation costs. Their inference procedure is *exactly* max-product message-passing for tree-structured models.

Third, they provide a list of five criteria an object representation for the task of object recognition should possess: completeness, compactness, transformability, incremental changeability, and simplicity of translation. By *completeness*, the representation should allow the solution of all the tasks of interest. *Compactness* requires the representation to be non-redundant. *Transformability*

demands easy and efficient manipulation of the information encoded in the representation. By *incremental changeability* Fischler and Elschlager require small changes in the world to translate into small changes in the representation. By the last property, *accuracy and simplicity of translation*, it should be simple to derive an accurate representation of a real world object. Starting from these requirements, the authors criticize linguistic and symbolic approaches as unable to accurately represent the real world in the context of object recognition problems. This is a remarkable early comment as the majority of the symbolic line of computer vision work happened afterwards in the 70's and early 80's.

In summary, the paper of Fischler and Elschlager was ahead of its time and influenced all later part-based recognition systems.

Literature Survey: CRF-based Approaches

Quattoni, Collins and Darrell[55] use discriminative models in the form of conditional random fields to learn to recognize objects from a given training set. The objects are decomposed into parts, which are modeled as patches around interest points. Each part is assigned a hidden variable and feature vector. Interactions between parts are reduced to tree-structure form by means of a minimum spanning tree approximation on top of the image coordinates of pairwise parts, the assumption being that parts close to each other have a stronger dependency. All model parameters are estimated by maximizing the marginal likelihood of the observed binary label, the presence of an object. Thus, the hidden variables are marginalized out. This operation can be performed efficiently because the model is tree-structured. However, the objective function is no longer concave thus only a local maximum is obtained. The proposed model is evaluated on the task of detecting cars.

[55] Ariadna Quattoni, Michael Collins, and Trevor Darrell. Conditional random fields for object recognition. In *NIPS*, 2004

Winn and Shotton[56] propose the "layout CRF" model to jointly detect and segment partially occluded objects from a known object class. The basic idea of the layout CRF model is to enforce label consistency among a dense set of parts which cover the object instance in a grid-like order. Each part has its own discrete label and thus simple pairwise orientation preferences between parts can be modeled as pairwise potential functions in a conditional random field model. The dense positioning of parts over the object allows to distinguish the border of the object from the interior. Therefore it is possible to perform inference of occlusion patterns such as object-object occlusions and object-background occlusions. The model is trained by cross validation on the training set. Experimentally, for cars and faces, the model is shown to accurately detect instances despite severe occlusions. Additionally it labels the parts consistently with the training layout.

[56] John M. Winn and Jamie Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *CVPR*, pages 37–44, 2006

Hoiem, Rother and Winn[57] extend the layout CRF proposed by Winn and Shotton to handle multiple views by means of a rough 3D model of the object class. Additionally, Hoiem explicitly models instance-level properties such

[57] Derek Hoiem, Carsten Rother, and John M. Winn. 3D layoutCRF for multi-view object class recognition and segmentation. In *CVPR*, 2007

as the color distribution of an object instance, leading to high-order potential functions. Both the used image features and the joint inference procedure are sophisticated. The test-time inference is no longer guaranteed optimal, an effect of the incorporating the per-instance features. Experimentally Hoiem et al. show excellent recognition and segmentation performance on cars from multiple views. However, as with the layout CRF of Winn and Shotton the model is only suited for rigid object classes.

Literature Survey: Viola-Jones Style Approaches

Schneiderman and Kanade[58] consider the task of frontal and profile face detection and propose to estimate the appearance probabilities for a set of fixed size parts within a detection window. The appearance model of each part uses quantized responses of projections onto the first twelve PCA components. For each response a class-conditional probability is estimated and additionally a spatial prior is estimated within the detection window for all discrete responses which appear frequently enough in the training data. The proposed method is evaluated on several face detection datasets and shows better performance than the previous methods. However, compared to the methods later proposed by Papageorgiou and Poggio and also Viola and Jones the performance is severely limited due to the discretization and the generative nature of the model.[59]

[58] Henry Schneiderman and Takeo Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *CVPR*, pages 45–51, 1998

Papageorgiou and Poggio[60] first describe what is now a popular approach to build object detection systems. For a given image and a fixed size bounding box, Papageorgiou and Poggio determine a large, overcomplete set of normalized multiscale Haar wavelet responses within the bounding box. Using a large bounding box annotated training image set which includes a set of background images, a binary classifier is trained on this feature representation. Detection is performed by sliding a bounding box over the image, classifying each feature vector produced from the image within the bounding box as either positive (object) or negative (background). While the approach is still severely limited — the training data must be precisely annotated, the features are fixed and manually designed, and extensive sliding window evaluation is necessary at test time — it is particularly interesting for its simplicity, high accuracy for some object classes such as cars and pedestrians and its influence on later object detection systems.

[59] Although Schneiderman and Kanade refer to their model as discriminative, they explicitly model $p(r|\text{has object})$ and $p(r|\text{has no object})$, where r is the appearance description of a region within the detection model.

[60] Constantine Papageorgiou and Tomaso Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000

Viola and Jones[61] describe in a series of papers an object detection system much like the one of Papageorgiou and Poggio — sliding windows with fixed Haar-wavelet features — but improve on the computational complexity in three directions. First, Viola and Jones introduce *integral images* for fast computation of Haar-wavelet features. Second, instead of using a nonlinear SVM as Papageorgiou and Poggio did, they use AdaBoost[62], incrementally selecting single discriminative wavelet features. This allows to use a much larger set of features. Third, they introduce *cascade classifiers* for efficient early rejection of

[61] Paul A. Viola and Michael Jones. Robust Real-Time face detection. In *ICCV*, pages 747–747, 2001; Paul A. Viola and Michael J. Jones. Robust real time object detection. In *Workshop on Statistical and Computational Theories of Vision*, 2001; and Paul A. Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2): 137–154, 2004

[62] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997

unlikely object hypotheses. Together, these three changes drastically reduce the test-time complexity, allowing real-time full resolution object detection systems. The Viola and Jones system has considerably influenced computer vision research and since 2001 a large number of derived systems have been proposed.

Felzenszwalb, McAllester and Ramanan[63] propose an iterative algorithm for training linear SVMs where part of the training sample vectors is *latent*, that is, unknown at both training and test time. These latent parts represent the appearance and positions of object parts and their value is defined by choosing a single element from the set of all possible part positions. Any feasible setting of the latent variables for a negative sample not containing an object defines a negative training sample for the SVM classifier. Positive samples are represented by a bounding box on the image plane. For each such bounding box, we know that at least one object is contained within the box. Felzenszwalb represents the positive instances by the latent variable setting which achieves the *highest* possible classifier response. By iteratively refining the classifier and latent variables for the positive instances, the classifier learns the appearance and likely position of object parts. The appearance is represented by histograms of oriented gradients (HoG) features[64]. At test time, detection is performed by means of sliding a detection window across the image at multiple scales. The approach is extensively evaluated on the PASCAL VOC 2007 object classification challenge and a preliminary version of the described system won the 2007 VOC object detection challenge. While motivated from first principles, many decisions in the system are largely heuristic: the aspect ratio and size of the classification window, the final sliding-window detection procedure, the initialization procedure, etc. However, the overall latent variable modeling approach holds considerable promise at improving object detection systems and this paper is likely to have some influence on further research.

[63] Pedro F. Felzenszwalb, David A. McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008

[64] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005

Literature Survey: Other Notable Approaches

Krempp, Geman and Amit[65] focus on the problem of how parts should be learned and reused in the case of many object classes. Krempp suggests a *sequential* learning procedure in which classes are added iteratively such that when a new class is added the number of reused parts is maximized. The sequential learning is realized by means of a greedy heuristic and the evaluation is on the artificial task of recognizing mathematical symbols.

Agarwal, Awan and Roth[66] describe a fixed size encoding which contains information about salient parts and their pairwise spatial relations. The parts are detected by extracting and vector quantizing small image patches around interest points. Their pairwise relations encode relative distance and angle information, quantized to a total of 20 discrete labels. For each fixed sized window in the image a vectorial representation is created by binary

[65] Samuel Krempp, Donald Geman, and Yali Amit. Sequential learning of reusable parts for object detection. Technical report, 2002

[66] Shivani Agarwal, Aatif Awan, and Dan Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. Pattern Anal. Mach. Intell*, 26(11):1475–1490, 2004

encoding the presence of each part-type and part-relation, yielding a large binary vector. Object localization is performed by first computing the classifier output densely in successively downsized versions of the image. In this densely evaluated scale-space an iterative non-maximal suppression scheme is used to output found objects. Agarwal et al. evaluate the approach on a newly introduced UIUC cars dataset on the task of detecting cars in side-view, achieving precision-recall error rates of 23.5% and 60.4% for fixed scale and multiscale test sets, respectively.[67] The proposed approach is completely heuristic and achieves low performance, but is representative of approaches which first convert geometric relations into fixed-size vectorial representations.

Lazebnik, Schmid and Ponce[68] propose a logistic regression model with features derived from "semi-local parts". The semi-local parts encode a set of local image features, thus modeling co-occurrence of these features. Additionally a pairwise feature encoding the overlap of individual features is used. Lazebnik et al. apply the model to both texture classification and object classification tasks. For the task of texture classification they report no significant improvement over a simple naive Bayes baseline model. For object classification a slight improvement is reported. Overall the model is particularly simple in that geometric parts simply become features, whereas the classification function is still linear in these features.

We now introduce our substructure-based framework for object recognition.

[67] Since then, the results have been improved to 1.5% and 1.4%, respectively, using a flat training technique, see

Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008; and Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *PAMI*, 2009

[68] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. A maximum entropy framework for part-based texture and object recognition. In *ICCV*, 2005

Graph-based Object Recognition

The notion that objects are composed of parts related by geometry lends itself ideally to a graph-based description of objects. The plentiful literature examples of the previous section illustrates this. Graphs are structured representations and as such we can try to apply our substructure poset framework.

The key issue when doing so is how the graph representation is created from an image. For many other application domains there is a natural graph representation of the objects of interest. For example, in chemical compound classification the graph is simply the molecule itself, composed of atoms and bonds of different types. Another example would be documents, which are often already well structured into a hierarchical graph representation, composed of chapters, sections, and paragraphs. In contrast, images do not have such natural graph structure.[69] We will come back to this issue later.

We first define graphs and subgraphs, then give specialized algorithms for subgraph based classification in the substructure poset framework. The specific details on how images are represented as such labeled graphs are provided in a later section.

[69] Although, one might argue that a 2D image naturally is a planar grid graph this is not a natural representation for object recognition as any other measurement layout would provide the same information.

Labeled Graph Structures

We apply the substructure poset framework introduced in the previous chapter to the classification of undirected, connected and labeled graphs. For this, we define a substructure poset $(\mathcal{S}, \subseteq)$ as follows.

Definition 10 (Labeled Graph) *A graph $g = (V, E, \Sigma_V, \Sigma_E, \ell_V, \ell_E)$ consists of a set $V \subset \mathbb{N}$ of vertices, a set of undirected edges $E \subseteq V \times V$, an alphabet of vertex labels Σ_V, an alphabet of edge labels Σ_E, and labeling functions $\ell_V : V \to \Sigma_V$, $\ell_E : E \to \Sigma_E$ assigning each vertex and edge a label from the respective alphabet. The graph must be simple and connected.*

We denote by $V(g)$, $E(g)$, $\Sigma_V(g)$, $\Sigma_E(g)$ the respective tuple elements of g and by ℓ_V^g, ℓ_E^g the respective labeling functions.

Definition 11 (Set of All Graphs $\mathcal{S}$) *Let $\mathcal{S}$ be the set of all graphs satisfying the above definition.*

Definition 12 (Subgraph-supergraph relation $\subseteq$) *The $\subseteq$: $\mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$ relation is defined as $g_1 \subseteq g_2$ true iff $\exists$ injective $\gamma : V(g_1) \to V(g_2)$ such that*

- $\forall v \in V(g_1) : \ell_V^{g_1}(v) = \ell_V^{g_2}(\gamma(v))$,
- $\forall (v_1, v_2) \in E(g_1) :$
  $(\gamma(v_1), \gamma(v_2)) \in E(g_2) \wedge \ell_E^{g_1}((v_1, v_2)) = \ell_E^{g_2}(\gamma(v_1), \gamma(v_2))$.

Then g_1 is called a subgraph of g_2 and g_2 is called a supergraph of g_1.

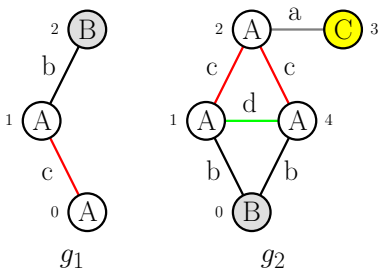


Figure 15: $g_1 \subseteq g_2$ as there exist two injective vertex mappings $\gamma_1, \gamma_2 : V(g_1) \to V(g_2)$ with $\gamma_1 = \{2 \to 0, 1 \to 1, 0 \to 2\}$ and $\gamma_2 = \{2 \to 0, 1 \to 4, 0 \to 2\}$, such that g_1 is a subgraph of g_2. The different vertex labels from the alphabet $\Sigma_V = \{A, B, C\}$ and edge labels from the alphabet $\Sigma_E = \{a, b, c, d\}$ are drawn in different colors for clarity.

Figure 15 shows an example of a subgraph-isomorphism. It turns out that in general evaluating $g_1 \subseteq g_2$ is NP-complete. However, for small graphs and sparse graphs appearing frequently in applications efficient algorithms have been devised.

In the previous chapter we have seen that for efficient enumeration of the substructure poset $(\mathcal{S}, \subseteq)$ we can define a *total order* on $\mathcal{S}$. This total order then implicitly defines the reduction mapping and thus the enumeration tree. For labeled graphs it is non-trivial to define a total order; this was first achieved by Yan and Han in their gSpan algorithm[70]. They propose to map each graph to a *canonical label* such that two graphs are isomorphic to each other if and only if they have the same canonical label. The canonical label comes with a natural total order. In the remainder of this section we describe the canonical label as used in gSpan.

[70] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *ICDM*, 2002

Depth First Search

For defining the total ordering, we first need the notion of a depth-first traversal of a graph. Because our graphs are assumed to be connected and undirected such that they form a single connected component, we can reach all vertices of the graph by starting from an arbitrary vertex and moving along edges.

Algorithm 3 DFSLabel: Depth-First-Search Labeling of a Graph

1: $\tau = \text{DFSLABEL}(g)$
2: **Input:**
3: $g \in \mathcal{S}$ labeled graph
4: **Output:**
5: $\tau : V(g) \to \mathbb{N}$ vertex traversal order
6: **Algorithm:**
7: $\tau(v) \leftarrow -1$ for all $v \in V(g)$ {Initialize: all vertices unvisited}
8: Choose a starting vertex $v_0 \in V(g)$
9: $\tau(v_0) \leftarrow 0$
10: $\tau \leftarrow \text{DFS}(g, v_0, v_0, \tau)$
11: **return** τ

Depth-first-search (DFS) starts from a vertex of the graph and systematically lists all edges and vertices in the order of traversal. For a good introduction to depth-first-search algorithms on graphs and their properties, see Sedgewick[71].

[71] Robert Sedgewick. *Algorithms in C: Part 5: Graph algorithms*. Addison-Wesley, 3rd edition, 2002. ISBN 0-201-31663-3

The overall DFS algorithm is shown in Algorithm 3, the recursion in Algorithm 4. The algorithm maintains an assignment $\tau : V(g) \to \mathbb{Z}$ over vertices, which has $\tau(v) = -1$ if v has not been visited yet and $\tau(v) \in \mathbb{N}$ if the vertex v has already been visited.

If $v, w \in V$, $\tau(v) \neq -1$, $\tau(w) \neq -1$, the ordering of $\tau(v)$,$\tau(w)$ corresponds to the visiting order of the vertices. In the DFS algorithm, each time the algorithm reaches a new vertex v (line 17) the vertex is assigned a new index $\tau(v)$ and the procedure recurses (line 19). The edge set adjacent to v is partitioned into B and F, the *backward edge set* and the *forward edge set*, respectively. The backward edgeset leads to vertices $w \in V(g)$ which have been visited already (line 10), whereas the forward edgeset leads to new unexplored vertices (line 14). Every edge seen is outputted (line 18 for forward edges, line 12 for backward edges).

There are two degrees of freedom in the DFS traversal, the choice of starting vertex v_0 (Algorithm 3, line 8), and the total ordering $\kappa : V(g) \times V(g) \to \{\top, \bot\}$ (Algorithm 4, line 16). Depending on the choice of v_0 and κ, different DFS traversals are produced.

Figures 16(b) to (d) illustrate three different DFS traversals for the labeled graph shown in Figure 16(a).

DFS code α	DFS code β	DFS code γ
(0,1,X,a,Y)	(0,1,Y,a,X)	(0,1,X,a,X)
(1,2,Y,b,X)	(1,2,X,a,X)	(1,2,X,a,Y)
(2,0,X,a,X)	(2,0,X,b,Y)	(2,0,Y,b,X)
(2,3,X,c,Z)	(2,3,X,c,Z)	(2,3,Y,d,Z)
(3,1,Z,b,Y)	(3,0,Z,b,Y)	(2,4,Y,b,Z)
(1,4,Z,d,Y)	(0,4,Y,d,Z)	(4,0,Z,c,X)

Table 2: Three different DFS codes α, β and γ for the graphs shown in Figure 16.

Algorithm 4 DFS: Depth-First-Search Recursion

$\tau = \text{DFS}(g, v, p, \tau)$
Input:
  $g \in \mathcal{S}$ labeled graph
  $v \in V(g)$ current vertex
  $p \in V(g)$ previous vertex
  $\tau : V(g) \to \mathbb{Z}$ vertex traversal order
Output:
  $\tau : V(g) \to \mathbb{N}$ vertex traversal order
Algorithm:
$B \leftarrow \{w | (v, w) \in E(g), w \neq p, \tau(w) \geq 0\}$ {Back-edges to already visited vertices}
for $w \in \text{SORT}(B, \{(v, w) \in V(g) \times V(g) | \tau(v) \leq \tau(w)\})$ **do**
  output $"(\tau(v), \tau(w), \ell_V^g(v), \ell_E^g(v, w), \ell_V^g(w))"$
end for
$F \leftarrow \{w | (v, w) \in E(g), \tau(w) = -1\}$ {Forward-edges to unvisited vertices}

{Traverse forward edges using total order κ}
for $w \in \text{SORT}(F, \kappa)$ **do**
  $\tau(w) \leftarrow (\max_{w \in V} \tau(w)) + 1$
  output $"(\tau(v), \tau(w), \ell_V^g(v), \ell_E^g(v, w), \ell_V^g(w))"$
  $\tau \leftarrow \text{DFS}(g, w, v, \tau)$
end for
return τ

Each DFS traversal generates a different sequence of **output**-calls. If the output is concatenated in order, then each DFS traversal leads to a unique code, shown in Table 2. The DFS traversal depends on v_0 and κ, the total order on the edges.

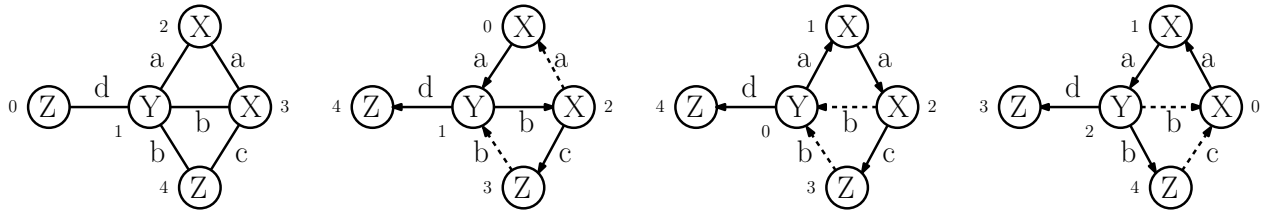
Definition 13 (DFS Code of a Graph) *Given a graph g, the sequence*

$$(a_0, a_1, \ldots, a_{|V(g)|})$$

of elements $a_i \in \mathbb{N} \times \mathbb{N} \times \Sigma_V \times \Sigma_E \times \Sigma_V$ is called DFS code of the graph *g if there exists an initial vertex $v_0 \in V(g)$, and a total order $\kappa : E(g) \times E(g) \to \{\top, \bot\}$ such that Algorithm* DFSLABEL *produces the sequence. Given a DFS code γ of the graph g, denote by $\mathcal{G}(\gamma) \in \mathcal{S}$ with $g = \mathcal{G}(\gamma)$ the original graph.*

By selecting among all possible DFS traversals the one that produces the minimum DFS code according to a total order $\preceq$ defined for DFS codes, we can uniquely associate a *canonical label* to each graph $g \in \mathcal{S}$.

Definition 14 (Canonical Label of a Graph) *For a given labeled graph g let $\psi(g)$ be its* canonical label, *where $\psi : \mathcal{S} \to (a_0, a_1, \ldots, a_{|V(g)|})$, with $a_i \in \mathbb{N} \times \mathbb{N} \times \Sigma_V \times \Sigma_E \times \Sigma_V$ is the DFS code that is minimal over all valid DFS codes representing g. It is minimal according to a total order $\preceq$ defined on DFS codes.*



(a) Labeled graph for which multiple DFS traversals exist.

(b) DFS traversal that generates code α in Table 2.

(c) DFS traversal that generates code β in Table 2.

(d) DFS traversal that generates code γ in Table 2.

Figure 16: Different DFS codes for the same labeled graph

The total order $\preceq$ is derived by lexicographically extending total orders $(\leq_{\mathbb{N}}, \leq_{\Sigma_V}, \leq_{\Sigma_E})$ on $\mathbb{N}$, Σ_V and Σ_E, respectively, to define $\preceq$ on the set $\mathbb{N} \times \mathbb{N} \times \Sigma_V \times \Sigma_E \times \Sigma_V$ as the concatenation $(\leq_{\mathbb{N}}, \leq_{\mathbb{N}}, \leq_{\Sigma_V}, \leq_{\Sigma_E}, \leq_{\Sigma_V})$.

For example, if we assume $\leq_{\Sigma_V}$ to be $X \leq_{\Sigma_V} Y \leq_{\Sigma_V} Z$, and $\leq_{\Sigma_E}$ to be $a \leq_{\Sigma_E} b \leq_{\Sigma_E} c \leq_{\Sigma_E} d$, then the three codes shown in Table 2 are ordered by $\gamma \preceq \alpha \preceq \beta$. In fact, γ is the minimal DFS code of g and thus its canonical label. We therefore have $\gamma = \psi(g) = ((0,1,X,a,X), (1,2,X,a,Y), (2,0,Y,b,X), (2,3,Y,d,Z), (2,4,Y,b,Z), (4,0,Z,c,X))$.

Regarding the choice of κ in Algorithm 4, if our goal is to produce only the minimum DFS code, then the choice of κ can be restricted to those orders on $\Sigma_V \times \Sigma_E \times \Sigma_V$ which respect the order $(\leq_{\Sigma_V}, \leq_{\Sigma_E}, \leq_{\Sigma_V})$. However, it can be the case that two different edges in the original graph $(v_i, v_j), (w_k, w_l) \in E(g)$ are identical under this order, i.e., that we have

$$\kappa((v_i,v_j),(w_k,w_l)) = \Big[(\ell_V^g(v_i), \ell_E^g((v_i,v_j)), \ell_V^g(v_j)) = (\ell_V^g(w_k), \ell_E^g((w_k,w_l)), \ell_V^g(w_l))\Big].$$

In this case, both orders need to be tried and the minimum DFS code is chosen a posteriori. In general, the number of orderings that may have to be tried is exponential and this ambiguity makes finding the minimum DFS code for a labeled graph a NP-complete problem[72]. Despite this negative result, real world graphs are usually sparse and have discriminative labels. Both properties help to limit the number of DFS codes that need to be generated in order to find the minimal one.

[72] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *ICDM*, 2002

Generating f^{-1}

In the previous chapter we have discussed how the substructure poset $(\mathcal{S}, \subseteq)$ and the total order $\preceq$ together define the reduction mapping f and, more importantly, its inverse f^{-1}. The reduction mapping allows efficient enumeration of frequent and discriminative substructures. Recall the definition of f^{-1} as

$$f^{-1}(t) = \{s \in \mathcal{S} | t \sqsubset s \text{ and } \forall u \sqsubset s : t \preceq u\}.$$

Generating the subset of $f^{-1}(t)$ for which a condition g holds was the central subproblem of Algorithm 2, where we considered as conditions g the frequency or discriminative value of a substructure. For the case of sets we briefly described how the condition-satisfying subset of $f^{-1}(t)$ can be generated efficiently. For labeled graphs this is again possible by the following theorem, due to Yan and Han[73].

[73] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *ICDM*, 2002

Theorem 3 (DFS Code Prefix Ordering (Yan and Han)) *For a given graph $t \in \mathcal{S}$ with canonical label $\psi(t)$, the extended set $f^{-1}(t)$ is exactly the set of subgraphs enlarged by one edge over t whose canonical label contains $\psi(t)$ as prefix, i.e.,*

$$f^{-1}(t) = \{\mathcal{G}(\gamma) \mid \psi(\mathcal{G}(\gamma)) = \gamma = (\psi(t), a), a \in \mathbb{N} \times \mathbb{N} \times \Sigma_V \times \Sigma_E \times \Sigma_V\}.$$

Proof. This is stated in different form in Theorem 4 of Yan and Han. □

The fact stated in the theorem can be used to build the set $f^{-1}(t)$ directly in the DFS code representation by extending the canonical label $\psi(t)$ of t towards candidate graphs with DFS codes of the form $(\psi(t), a)$. The extended graphs represented by $(\psi(t), a)$ need to satisfy the following two conditions.

1. $(\psi(t), a) = \psi(\mathcal{G}((\psi(t), a)))$, i.e., the DFS code needs to be the canonical label of the graph $\mathcal{G}((\psi(t), a))$, and

2. $g(\mathcal{G}((\psi(t), a))) = \top$, i.e., the (optional) condition needs to be satisfied.

Checking condition 1. involves testing the minimality of $(\psi(t), a)$. Algorithm 3 can be adapted to this end, for details of what optimizations are possible in the minimality check see the discussion in Section 5.1 of Yan and Han.

Condition 2. can be asserted by considering only extensions $a \in \mathbb{N} \times \mathbb{N} \times \Sigma_V \times \Sigma_E \times \Sigma_V$ for which the condition will hold. For example, if g is the minimum frequency condition (20), then iff a is a frequent edge in X with respect to the current subgraph-isomorphisms into X, so will $\mathcal{G}((\psi(t), a))$ be frequent in X.

The above method of generating $f^{-1}(t)$ can be summarized as follows. First, we only work directly in the minimal DFS code representation. Second, the operation of extending a graph by an edge must preserve the current minimal DFS code prefix; if it does not, the extended graph will be enumerated elsewhere and is not in $f^{-1}(t)$. Third, the condition g can be naturally accommodated as we always know the current subgraph-isomorphisms into the graph database X.

THE ABOVE DEFINITIONS AND ALGORITHMS SUFFICE to apply the substructure poset framework to undirected labeled graphs. That is, using the Boosting method from the previous chapter we can now learn a classification function on labeled graphs. In the next section we describe how images can be represented as labeled graphs.

Images as Graphs

We first describe how the structure of the graph is defined, then provide details of how we introduce the discrete vertex and edge labels.

Graph Structure

We use a *superpixel segmentation*[74] to define a low complexity partitioning of the image into a small number of superpixels. Each superpixel becomes a node in a graph and the partition boundaries in the image plane define edges between superpixels in that graph.

There are various popular methods to obtain superpixel segmentations for a given image. The most popular methods are mean-shift segmentation[75], spanning tree based segmentations[76] and normalized cuts[77]. We use normalized cuts because it produces a quite regular decomposition of the image into roughly equal-sized partitions. For an example of superpixel representations, see Figure 17.

K-way normalized cuts[78] is a clustering objective on weighted undirected graphs. For a fixed number K of desired partitions the objective balances the total within-cluster edge weights to the overall edge weights of all nodes within the cluster. This leads to a K-partitioning of the graph. More formally, let there be an image $\mathcal{I}$ with N pixels. We define a symmetric weight matrix $W \in \mathbb{R}_+^{N \times N}$ with non-negative weights between nearby pairs of pixels of the image. These are produced by measuring similarity between the pixels, for example similarity in color and texture of the immediate surrounding of the pixel. Nearby pixels i and j that are very similar receive a large weight $w_{i,j} = w_{j,i} > 0$, whereas pixels with different properties receive a weight close to zero, i.e., $w_{i,j} \approx 0$. Let $D = \mathrm{diag}(W\mathbf{1}_N)$ be the diagonal matrix which has on the diagonal the total sum of weights of each pixel. Then $D_{i,i}$ contains the *degree* of a pixel, the total sum of weights of the edges connecting to the pixel.

Using this notation, the K-way normalized cuts objective can be stated as the following mathematical program.

$$\max_X \quad \frac{1}{K} \sum_{\ell=1}^{K} \frac{X_\ell^\top W X_\ell}{X_\ell^\top D X_\ell} \tag{22}$$

$$\text{sb.t.} \quad X\mathbf{1}_K = \mathbf{1}_N, \tag{23}$$

$$X \in \{0,1\}^{N \times K}, \tag{24}$$

where $X_{i,k} = 1$ denotes that pixel i is assigned to the k'th partition. The above problem is NP-hard in general but a good approximate solution can be obtained even for large problems ($N > 10^6$) by first solving a spectral relaxation in the continuous domain and afterwards applying an iterative rounding procedure. For details see Yu and Shi. The procedure provides a partition label for each pixel in the image.

[74] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, 2003

[75] Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *ICCV*, pages 1197–1203, 1999

[76] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004

[77] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, 2003; and Greg Mori. Guiding model search using segmentation. In *ICCV*, 2005

[78] Stella X. Yu and Jianbo Shi. Multiclass spectral clustering. In *ICCV*, pages 313–319, 2003

The advantages of using superpixels stem from three directions. First, superpixels restrict the hypothesis space by coarsening the image representation into meaningful groups. This leads to lower computational complexity as the number of basic elements is reduced, say, from $\approx 10^6$ pixels to ≈ 100 superpixels. Moreover, overfitting can be reduced, a benefit we will later come back to in a chapter dealing with image segmentation. Second, the unsupervised pixel grouping that superpixels provide allows *pooling* of image statistics within meaningful regions. This can increase the robustness of features such as histograms; for example, a color histogram within a superpixel region is a more robust statistic than a color histogram in an arbitrary square box region of the image. Third, superpixels relate to visually consistent parts of the image. Thus, part-based representations can be constructed on top of superpixels. For example, note how the body parts such as legs and hands are recovered in the superpixel segmentations shown in Figure 17.

The use of superpixels has some disadvantages. First, the technique compresses the the image structure considerably, and thus possibly useful information might get lost. For example, segmentation errors where one superpixel crosses the object boundary are impossible to correct. Second, it is a purely unsupervised preprocessing step producing an intermediate image representation. In principle, it would be preferable to incorporate the representation only as additional information in an end-to-end learning system. And third, although some progress has been made recently[79], creating the superpixel representation is computationally expensive and takes a few minutes per image.

[79] Alastair P. Moore, Simon Prince, Jonathan Warrell, Umar Mohammed, and Graham Jones. Superpixel lattices. In *CVPR*, 2008; and Bryan Catanzaro, Narayanan Sundaram, Bor-Yiing Su, Yunsup Lee, Mark Murphy, and Kurt Keutzer. Damascene: Highly parallel image contour detection, March 2009. URL `http://www.gigascale.org/pubs/1510.html`

Given the superpixel segmentation in terms of $X \in \{0,1\}^{N\times K}$, let $P(i) \in \{1,2,\ldots,K\}$ be a unique partition label assigned to each pixel i by $P(i) = \text{argmax}_{k=1,\ldots,K} X_{i,k}$. We define an undirected connected simple graph $G = (V, E)$ with vertex set $V = \{1,2,\ldots,K\}$ consisting of the superpixels. The edge set is constructed such that if two superpixels are adjacent in the image, there is an edge linking them. Formally, $E \subseteq V \times V$ with $(k,l) \in E$ iff $\exists i \in \{1,\ldots,N\} : P(i) = k$ and $\exists j \in \mathcal{N}(i) : P(j) = l$, where $\mathcal{N}(i)$ is a neighborhood set around pixel i. We use the 4-neighborhood.

Graph Labels

As described in the previous section, we use labeled graphs in which each vertex and edge is assigned a discrete label from an alphabet. We now describe how the labels are chosen for vertices and edges.

Vertex labels. We extract 30,000 SURF image features[80] densely and randomly per image and additionally a few thousand using the SURF box-filter interest point operator. SURF features are gradient histogram features, akin to the popular SIFT features. From the training set, a random subset of features is taken and k-means clustered to produce a codebook with 500 codewords.

[80] Herbert Bay, Tinne Tuytelaars, and Luc J. Van Gool. SURF: Speeded up robust features. In *ECCV*, pages 404–417, 2006

Figure 17: Examples of superpixel segmentations for the PASCAL VOC 2008 images. The top row images are decomposed into approximately 100 superpixels, the bottom row shows the same images decomposed into approximately 300 superpixels. Note that the very coarse granularity of 100 superpixels often suffices to accurately describe the object boundaries. In some cases, such as the person shown in the top left image a finer partitioning into 300 superpixels improves the object boundaries (second row, leftmost image).

Each SURF feature is quantized to its nearest codeword vector, such that for each image we have an average of 38,000 "XYC-tuples" of the form (x, y, c), where (x, y) is the pixel position of the feature and $c \in \{1, \ldots, 500\}$ is the codeword identifier. For each superpixel we create a histogram of codeword assignments of the features whose center position (x, y) is covered by the superpixel. We normalize the histogram to have a 1-norm of one. Finally, for each superpixel we have obtained a normalized histogram vector in $\mathbb{R}^{500}$. For the entire training set we collect all these histogram vectors and k-means cluster them into codebooks of sizes 32, 64, 128, and 256 codewords. By vector quantizing each histogram into the nearest codeword we obtain for each codebook size one discrete label for each superpixel.

Edge labels. The edge labels are set according to one of the following three schemes. In the first scheme ("constant"), all edgelabels are set to the same constant. This provides only the connectivity information between superpixels but no further information about properties of the edge. In the second scheme ("edgewidth-k"), the size of the shared edge e between the adjacent superpixels in the image is discretized into one of k labels according to the formula $\lceil k \frac{w_e}{\max_{f \in E} w_f} \rceil$, where w_e is the width in pixels of the edge e. This encoding provides not only connectivity information but also some quantification of the amount of adjacency of the two superpixels. We use values of $k \in \{4, 10\}$. In the third scheme ("angular-k"), we encode pairwise geometry information by discretizing the orientation of a straight line between the mean pixel coordinates of the adjacent superpixel regions. The encoding is according to the formula $\lceil k \frac{\gamma_e}{\pi} \rceil$, where $\gamma_e \in [0; \pi]$ is the undirected orientation of the straight line between the mean image coordinates of the superpixel regions. We again use $k \in \{4, 10\}$ to define two possible quantization choices. This edge labeling scheme encodes pairwise geometry relations such as "is adjacent in vertical direction". In total for the three schemes and the parameter choices there are five possible edge labeling methods.

USING THE ABOVE CONSTRUCTION, by varying the vertex codebook sizes and edge labeling parameters we have a family of 20 possible graph construction schemes. In the experiments we will perform model selection to identify which one is best for each class.

Experiments and Results

We now evaluate the proposed approach experimentally. For this we first provide details about the benchmark data set we use. Then we describe the baseline models we compare against. Finally we explain the experimental setup and provide the experimental results. As both our proposed approach and the baseline models use exactly the same image features we can assess their true performance in a fair manner.

Throughout this section we seek to answer the following three questions.

1. Is a discrete graph-based representation suitable for class-level object recognition problems?

2. Can substructure based methods which have been used successfully in other domains be applied on noisy vision data?

3. Does geometry help for high-level class-level object recognition?

PASCAL VOC 2008 data set

The PASCAL Visual Object Classes (VOC) Challenge[81] is an annual computer vision challenge held since 2005. We describe the classification task of the 2008 challenge. The 2008 data set contains a large number of photographic images obtained from flickr.com. Each image contains one or more objects from a set of 20 popular object classes, such as bicycles, cars, cats and persons. The overall image set is split into training, validation and testing data and human ground truth annotation is made available only for the training and validation data. A list of object classes as well as image count statistics in the training and validation set are shown in Table 3.

[81] http://pascallin.ecs.soton.ac.uk/challenges/VOC/

Image set	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
train	119	92	166	111	129	48	243	159	177	37
val	117	100	139	96	114	52	223	169	174	37
	diningt.	dog	horse	mbike	person	plant	sheep	sofa	train	tv
train	53	186	96	102	947	85	32	69	78	107
val	52	202	102	102	1055	95	32	65	73	108

Table 3: PASCAL VOC 2008 database image count statistics for the classification task. Shown are the number of images with at least one positive object instance of the respective class.

The provided annotations have three granularities. The coarsest annotation is a simple per-image binary label for each object class which tells us whether this image contains at least one object of the respective object class. A finer annotation is provided in terms of bounding boxes for each object instance. For each object instance appearing in an image the bounding box coordinates in image space, the object class, a rough object orientation and the information whether the object is occluded or truncated is provided. The finest annotation is available only for some images and contains a per-pixel segmentation of the entire image into object classes and object instances. In this chapter we will only use the coarsest image-level annotation and will not use the bounding box and segmentation labels. Later, in the structured output learning part of this thesis we will separately make use of the VOC 2008 data set and its segmentation labels.

Some example images for each object class with bounding boxes are shown in Figure 18 to 22. The data set is known to be very difficult due to severe variations in appearance. It thus better captures the difficulty of class-level

object recognition than other popular data sets such as the Caltech 101 object categories data set.

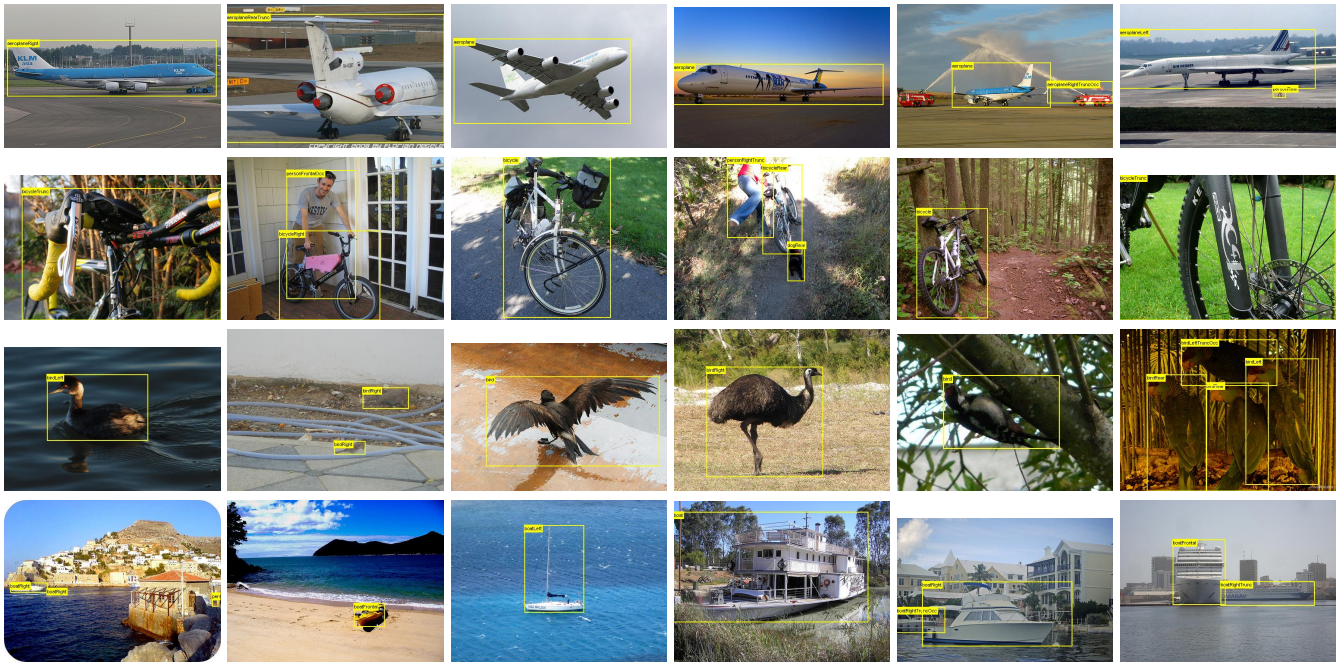


Figure 18: Examples of the PASCAL VOC 2008 object classes, row-wise: aeroplane, bicycle, bird and boat.

Experimental Setup

Each class in the VOC classification set is treated individually such that we obtain 20 individual binary classification tasks. Because the test set labels are unavailable, for the purpose of evaluation we train exclusively on the `train` data and evaluate the model performance once on the `val` validation set. In principle this reduces the overall performance compared to training on the entire `trainval` set and evaluating on the `test` set, as is done in the competition. However, we are interested in the relative model performance.

FOR THE PERFORMANCE CRITERION we choose the area under the Receiver Operating Characteristic curve (ROC AUC). The ROC curve plots the true positive rate[82] as a function of the false positive rate of a classifier, evaluated on a holdout sample set. The true positive rate and false positive rate are defined as

$$TPR(\theta) = \frac{TP(\theta)}{POS}, \qquad FPR(\theta) = \frac{FP(\theta)}{NEG},$$

where POS and NEG is the total number of positive and negative samples in the holdout set, respectively. The scalar $\theta \in \mathbb{R}$ defines a classification threshold, such that when $f(\boldsymbol{x}) \geq \theta$, the sample $\boldsymbol{x}$ is classified positive and negative

[82] The true positive rate is also known as *sensitivity*.

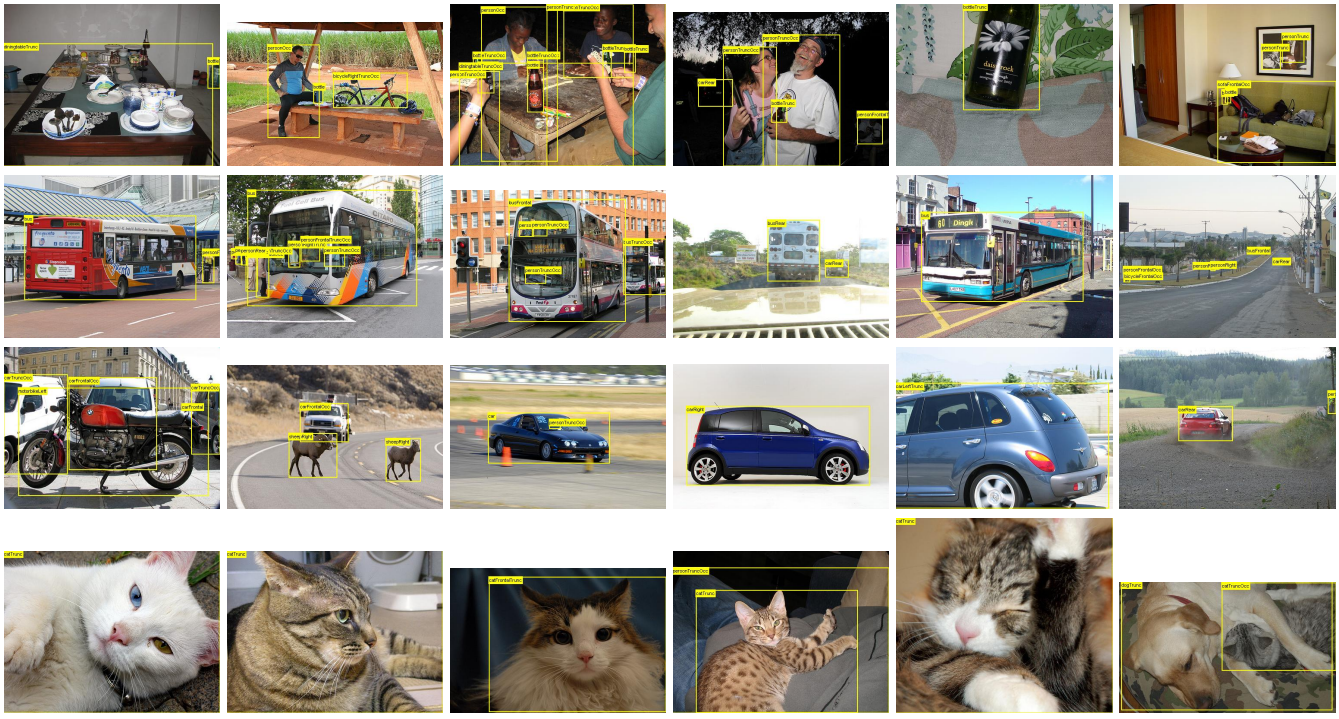


Figure 19: More object classes, row-wise: bottle, bus, car and cat.

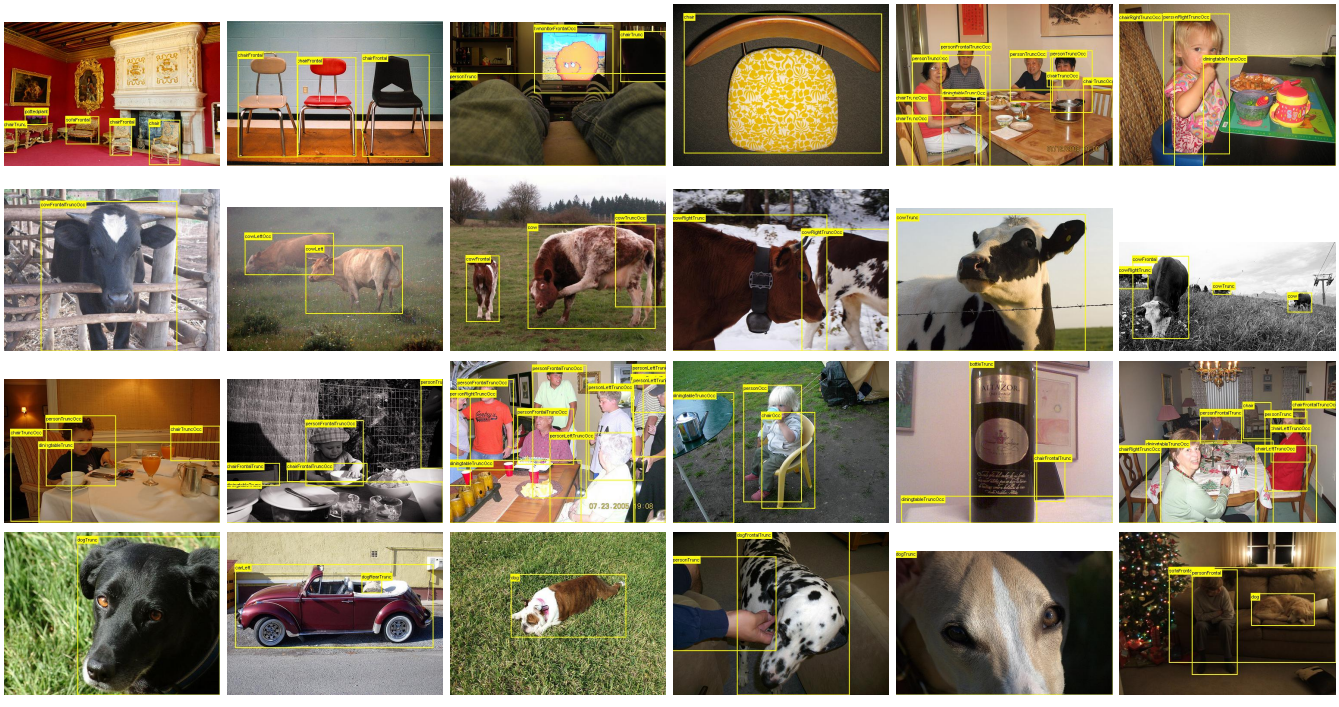


Figure 20: More object classes, row-wise: chair, cow, diningtable and dog.

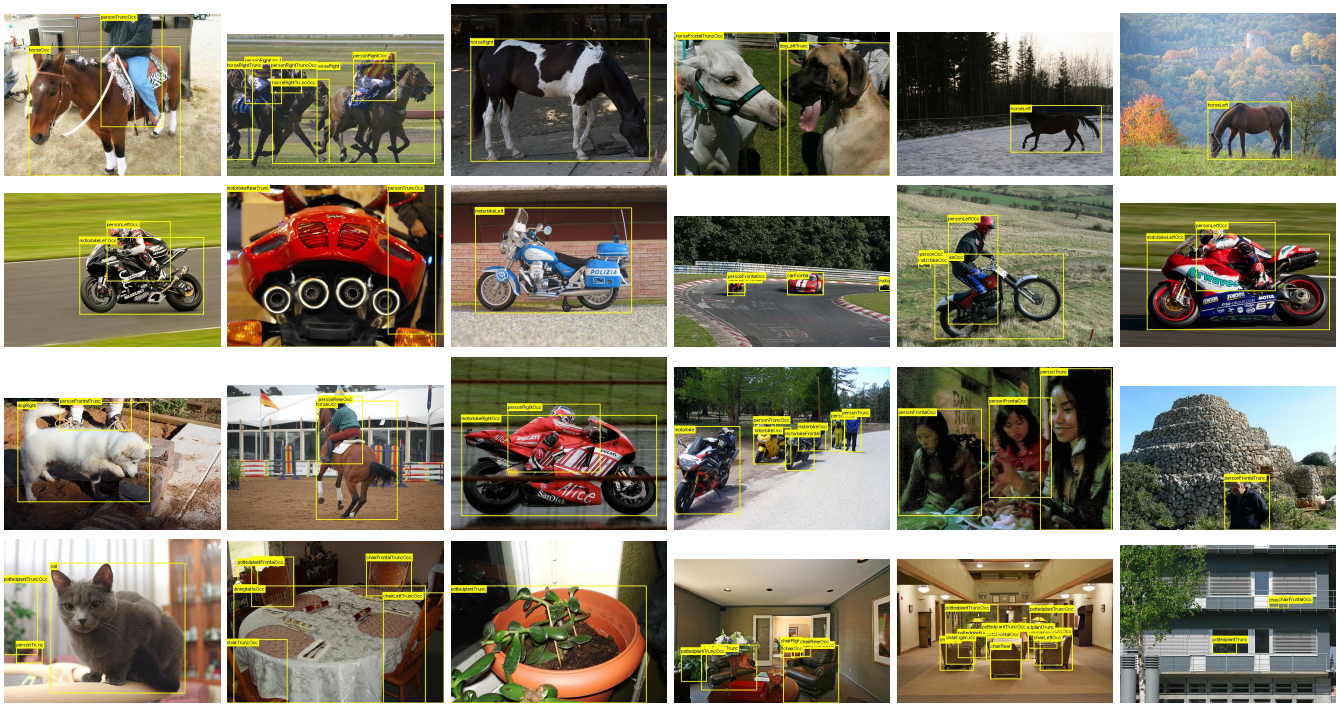


Figure 21: More object classes, row-wise: horse, motorbike, person and potted plant.

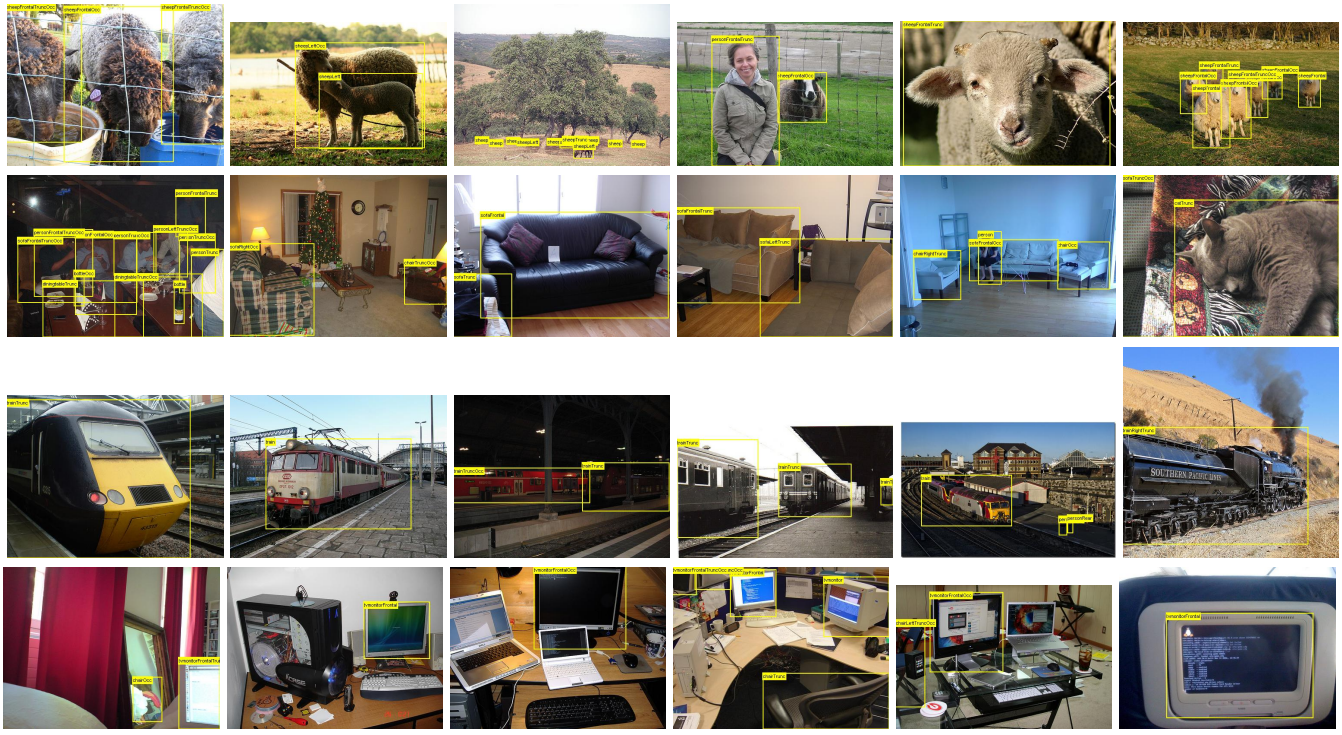


Figure 22: More object classes, row-wise: sheep, sofa, train and TV/monitor.

otherwise. Then true positive count $TP(\theta)$ is the number of positive samples from the holdout set which are actually classified as positive. Likewise, the false positive count $FP(\theta)$ is the number of negative samples actually classified as negative by the thresholded classifier. For all values of θ, we have $0 \leq TPR(\theta) \leq 1$ and $0 \leq FPR(\theta) \leq 1$. By plotting the set of points $(FPR(\theta), TPR(\theta))$ as θ varies, the ROC curve is obtained. A random classifier would achieve an expected area under the ROC curve of 0.5, whereas a perfect classifier would obtain 1.0. The ROC AUC measure is useful to evaluate the model performance in our setting because it is invariant under class imbalance. In the VOC data set some classes have far more negative samples than positive ones.

We additionally provide also the mean average precision (MAP) measure used in the official VOC challenge. The measure is a uniform average of eleven points on the precision-recall curve and is described in detail in the official VOC report[83]. However, the MAP measure is not invariant under class imbalance and we therefore prefer the ROC AUC measure.

[83] Mark Everingham, Luc Van Gool, Christopher K.I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2008 Results. http://www.pascal-network.org/challenges/VOC/voc2008/

MODEL SELECTION is performed on the `train` set only. The `train` set is split once and at random in proportions 70% to 30%, where the larger set of 70% is used for training and the 30% set is used for estimating the holdout performance of the trained model. For each model class and each possible parameter setting a classifier is trained and its performance estimated. The parameter setting that achieves the best performance is fixed and the classifier is trained once on the entire `train` set. This one classifier per model class is evaluated on the `val` set and its performance is reported.

Methods

In order to assess the true performance of our proposed graph-based model and to the relative influence of modeling decision, we evaluate the following four baseline models versus the proposed approach "graph".

LR-unnorm. A linear logistic regression classifier on the original XYC histograms, without normalization. The only free regularization parameter C is model selected over the set $\{0.0001, 0.001, \ldots, 1000, 10000\}$. For a given training set $\{(\boldsymbol{x}_n, y_n)\}_{n=1,\ldots,N}$ and regularization parameter $C > 0$ training the logistic regression classifier minimizes a regularized logistic loss as

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C \sum_{n=1}^{N} \log(1 + \exp(-y_n \boldsymbol{w}^\top \boldsymbol{x}_n)).$$

This model is the standard "bag-of-words" model.

LR-norm. The same as LR-unnorm but with additional one-norm normalization on the histogram. The value of C is determined by model selection from the same set as before.

LR-super-unnorm. Linear logistic regression classifier on the superpixel

label histogram, the histogram of the discrete label assigned to each superpixel by the graph construction scheme. The free parameters are the codebook size for the superpixel quantization, which is selected from the set $\{32, 64, 128, 256\}$, and the regularization parameter C, which is selected from the set $\{0.0001, 0.001, \ldots, 1000, 10000\}$. The total set of models from which the best is selected by the model selection procedure is $4 \cdot 180 = 720$ models.

LR-super-norm. The same as LR-super-unnorm but with additional one-norm normalization on the superpixel histogram. The parameters selected are from the same set as for the LR-super-unnorm model.

graph. A totally corrective AdaBoost classifier learned in the space of all subgraph weak learners, as explained in the structured input chapter. The regularization parameter T is part of the model selection and taken from the set $\{1, 0.25, 0.1, 0.05\}$. In each iteration the subgraph weak learners are found using the the gSpan traversal order on the DFS code tree and the final classifier consists of a set of graphs with associated signed weights. A new image represented as graph is classified by checking for subgraph-isomorphism of the discriminative graphs and adding all weights of matched graphs.

Results

Tables 4 and 5 show the ROC AUC and mean average precision scores achieved by the baseline models and the proposed method "graph". We first state the results of the baseline models, then make the comparison to the proposed approach.

WITHIN THE BASELINE MODELS the LR-norm has higher test performance than the unnormalized version LR-unnorm. The superpixel label histogram baselines (LR-super-unnorm and LR-super-norm) have roughly the same performance as the bag of words models, with the exception of some classes such as "bus", "cat", "mountain bike" and "train", where the bag of words model fares better. In other classes such as "bottle", "car" and "sheep" the superpixel models perform better.

The proposed graph-based approach does not offer a performance increase, with the exception of the classes "chair" and "sofa", where it outperforms the baseline models. For some classes such as "cat", "dinging table" and "mountain bike" it achieves performance on the level of the superpixel baselines. For other classes such as "boat", "bottle", "cow", "dog", "sheep" and "train" there is a steep drop in performance compared to the superpixel baseline models.

Discussion

Part of the bad results of the graph based method can be explained by the second discretization step needed to label the superpixels. This can be recognized by observing that for some classes such as "cat", "dining

Approach	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
LR-unnorm	0.9057	0.7129	0.7100	0.7988	0.6279	0.7863	0.6806	0.6780	0.6832	0.7378
LR-norm	**0.9271**	**0.7471**	**0.7453**	**0.8689**	0.6795	**0.8362**	0.7613	**0.7544**	0.6905	0.7398
LR-super-unnorm	0.9139	0.7110	0.7360	0.8517	**0.6932**	0.7865	**0.7737**	0.7065	0.7006	0.7289
LR-super-norm	0.9145	0.7129	0.7357	0.8542	0.6822	0.7900	0.7669	0.7092	0.6972	0.7260
graph	0.9000	0.7152	0.7118	0.8170	0.6478	0.7730	0.7532	0.6936	**0.7429**	0.6372
	diningt.	dog	horse	mbike	person	plant	sheep	sofa	train	tv
LR-unnorm	0.7611	0.6302	**0.7756**	0.7307	0.7045	0.5757	0.7279	0.7182	0.7539	0.8050
LR-norm	**0.7754**	**0.6949**	0.7658	**0.7440**	**0.7323**	**0.6067**	0.7376	0.7117	**0.8158**	**0.8362**
LR-super-unnorm	0.7363	0.6416	0.7486	0.6793	0.7200	0.5619	0.7575	0.6947	0.7445	0.8212
LR-super-norm	0.7379	0.6505	0.7316	0.6449	0.7161	0.5974	**0.7742**	0.7092	0.7633	0.8186
graph	0.6940	0.5973	0.7014	0.6518	0.6849	0.5766	0.7037	**0.7505**	0.6560	0.7964

Table 4: PASCAL VOC 2008 classification ROC AUC results of the VOC `val` set (2227 images). Model selection was performed on the VOC `train` set (2113 images).

Approach	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
LR-unnorm	0.4816	0.1345	0.2438	0.2345	0.0782	0.0917	0.1925	0.1807	0.1699	**0.0569**
LR-norm	**0.5463**	**0.2009**	**0.3134**	**0.2839**	0.0852	0.1178	0.2729	**0.2564**	0.1780	0.0390
LR-super-unnorm	0.5272	0.1328	0.2763	0.2615	0.0891	0.0777	0.2735	0.1551	0.2153	0.0401
LR-super-norm	0.5310	0.1213	0.2812	0.2659	0.0857	0.0833	0.2797	0.1595	0.1425	0.0403
graph	0.4371	0.1516	0.1952	0.2377	0.0794	**0.1635**	0.2551	0.1847	**0.2654**	0.0273
	diningt.	dog	horse	mbike	person	plant	sheep	sofa	train	tv
LR-unnorm	0.0577	0.1331	0.1477	0.1244	0.6825	0.0953	0.0504	0.0836	0.1891	0.2342
LR-norm	0.0710	**0.2505**	0.1391	**0.1339**	0.7078	**0.1457**	0.0538	0.0611	**0.2275**	0.2410
LR-super-unnorm	0.1463	0.1420	0.1282	0.0782	0.7039	0.0692	0.0561	0.0603	0.0941	0.2679
LR-super-norm	0.1468	0.1448	0.1239	0.0822	0.6981	0.0683	0.0601	0.0617	0.1104	0.2656
graph	0.0479	0.1305	0.1469	0.0817	0.6631	0.0594	0.0520	**0.1176**	0.0749	0.2165

Table 5: PASCAL VOC 2008 classification mean average precision (MAP) results of the same models as shown in Table 4.

table" and "mountain bike" the performance drop is about the same for all superpixel based models (LR-super-unnorm, LR-super-norm, graph).

For a large part of the classes, the information loss due to the additional discretization cannot be the reason for the inferior performance of the graph approach. In particular, for the "boat", "bottle", "cow", "dog", "sheep" and "train" classes the superpixel baselines fare quite well while the graph based approach achieves only a lower AUC.

In fact, the feature space used in the LR-super-unnorm classifier is a small subset of the features available to the graph classifier. Hence, we believe that for these classes the decrease in performance by enlarging the feature space is due to two reasons. First, it could be that for these classes there is little or

no discriminative information contained in the edge attributes. Second, the feature space is too large or the 1-norm regularization on the feature weights is not well suited to avoid overfitting the training set.

For two classes, "chair" and "sofa" the performance of the graph-based approach is visibly improved over all baseline methods. Because the used test set is quite large (2227 images) we believe that the reported estimates are indeed a reliable indicator of the model performance but we did not find an immediate reason for the improved performance.

Conclusion

We now come back to the initial questions we posed.

Is a discrete graph-based representation suitable for class-level object recognition problems? Discretization causes an information loss but it is hard to quantify the amount of discriminative information that is lost. From the experiments it seems the loss due to discretization is small. Our graph-based representation that includes geometric information does not seem to provide an improvement in class-level object recognition performance, with the exception of two object classes. This lack of improvement in performance despite the intuitive appeal of including pairwise information such as co-occurrence and geometry seems consistent with the larger part of the literature that reported baseline comparisons.

Can substructure based methods which have been used successfully in other domains be applied on noisy vision data? In light of the obtained results the substructure based method does not seem well suited in addressing the large amount of variation, clutter and noise in the image features. This conclusion might not hold for more artificial objects such as symbols which have a clear structure and repeatable image features. We believe substructure based methods are best suited for hard classification tasks in which the definition of the graph structure is naturally obtained from domain knowledge, the basis features have low noise level, but the discriminative information is contained in higher order patterns. This is consistent with our observations on the domain of chemical compound classification.[84]

Does geometry help for high-level class-level object recognition? From our experiments but also from the literature review we believe that at the current weak performance levels of class-level object recognition systems it does not seem to help to incorporate geometric information beyond what is implicitly contained in standard image features.

[84] Hiroto Saigo, Sebastian Nowozin, Tadashi Kadowaki, Taku Kudo, and Koji Tsuda. gboost: A mathematical programming approach to graph classification and regression. *Machine Learning*, 75(1): 69–89, 2009; and Sebastian Nowozin and Koji Tsuda. Frequent subgraph retrieval in geometric graph databases. In *ICDM*, 12 2008

Activity Recognition using Discriminative Subsequence Mining

In the previous chapter we have considered the problem of classifying static images as to whether they contain objects of a certain class or not. In this chapter we take a step further and consider the problem of recognizing human activities from video data. We will continue to apply our structured input framework to derive classifiers for structured input data in a principled way.

The contributions of this chapter are, i) a new sequential representation for video data which encodes the temporal ordering among locally informative appearance patterns, and ii) a concretization of the substructure poset concept to this sequential representation by a suitable definition of a *subsequence* relation.

Human Activity Recognition

Human activity recognition and classification systems can provide useful semantic information to solve higher-level tasks, for example to summarize or index videos based on their semantic content. Robust activity classification is also important for video-based surveillance systems, which should act intelligently, such as alerting an operator of a possibly dangerous situation.

Building such a general activity recognition and classification system is a challenging task, because of variations in the environment, objects and actions. Variations in the *environment* can be caused by cluttered or moving background, camera motion, occlusion, weather- and illumination changes. Variations in the *objects* are due to differences in appearance, size or posture of the objects or due to self-motion which is not itself part of the activity. Variations in the *action* can make it difficult to recognize semantically equivalent actions as such, for example imagine the many ways to jump over an obstacle or different ways to throw a stick.

In current computer vision research, it is common to represent each data instance (i.e., video or image) as a histogram of visual words, see for example the recent PASCAL VOC2008 object classification challenge[85]. However, due to the variations stated above, not all visual words are informative for classification.

[85] Mark Everingham, Luc Van Gool, Christopher K.I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2008 Results. http://www.pascal-network.org/challenges/VOC/voc2008/

Thus, *feature selection* is important both for robustness against variations and for the interpretability of the learned classification rule. However, simply removing visual words based on some statistics (e.g., correlation to the class label) might be harmful, because, if combined with other features, a visual word can possibly become an important feature. In this light, finding the optimally discriminative *combination of features* is a combinatorial optimization problem, leading to an exponentially large feature space. The problem of high dimensionality of such feature space can partially be overcome using kernel methods, which allows one to learn a classification function implicitly. However, the cost is that the resulting classification function is not interpretable.

THE SUBSTRUCTURE BOOSTING FRAMEWORK could potentially address both the issue of a rich enough feature space to achieve high recognition performance while remaining interpretable.

In this chapter we apply the substructure Boosting framework to a sequence representation of videos. A natural subsequence relationship induces a rich feature space suitable for classifying sequences by recognizing discriminative subsequences.

The sequence representation will be built from sparse spatio-temporal "video words" encoding local appearance around interesting movements. The use of these sparse spatio-temporal interest points is a recent trend in action classification. However, most of the recent approaches have used a simple histogram representation, discarding the temporal order among features. Our assumption is that this ordering information can contain important information about the action itself. For example, consider the sport disciplines of *hurdle race* and *long jump*, where the global temporal order of motions (running, jumping) is important to discriminate between the two.

Therefore, we propose a sequential representation which retains this temporal order. Using the substructure Boosting framework on top of this sequence representation then amounts to simultaneously learning a classification function and performing feature selection in the space of all possible feature sequences. The resulting classifier linearly combines a small number of interpretable decision functions, each checking for the presence of a single discriminative pattern.

The remaining part of this chapter is structured as follows. We first give a survey of current approaches to action recognition in videos. Then we formalize our notion of sequence in terms of the substructure poset framework of the first chapter. The next section describes how a video with sparse spatio-temporal interest points can be represented in our sequence format. We continue by evaluating the classifier learned using substructure Boosting on the KTH action recognition benchmark dataset against other state-of-the-art approaches. Finally the results give rise to a discussion and we conclude by discussing further research directions.

Related Work

We now discuss two main groups of approaches popular in the literature, part-based representations and holistic representations.

Part-based Representations

Part-based representations based on interest point detectors, combined with robust descriptor methods have been used very successfully for object classification tasks, see for example the approaches submitted to the PASCAL VOC2006 challenge[86].

Recently, representations based on sparse local features have become popular also for human action classification. Laptev[87] proposed to assign each voxel in a spatio-temporal volume a saliency value and extract descriptors from the neighborhood of local saliency maxima. Schüldt et al.[88] used these features successfully for human action classification by discretizing them into codewords and producing a histogram of the occurring words for each video. The histograms are treated as fixed-length vectors to train a classification function. A visualization of sparse interest points detected in a video volume is shown in Figure 23.

Dollár et al.[89] argue in principle for the same approach but suggest to use a denser sampling of the spatio-temporal volume by only requiring each interest point to be a local maxima in the spatial directions instead of *both* spatial and temporal dimensions. They justify this change by increased classification performance on the same dataset.

Niebles et al.[90] train an unsupervised probabilistic topic model on the same features as Dollár and obtain comparable classification performance. Another approach is due to Ke at al.[91], who use a forward feature selection procedure to train a classifier on volumetric features.

[86] Mark Everingham, Andrew Zisserman, Chris Williams, and Luc Van Gool. The pascal visual object classes challenge 2006 (VOC2006) results. Technical report, 2006

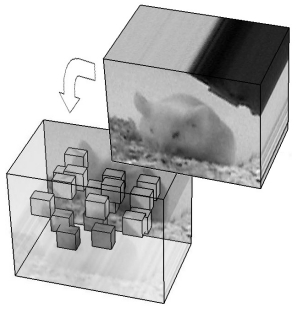


Figure 23: Sparse interest points defined on the video volume. Figure taken from Dollár et al.

[87] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005

[88] Christian Schüldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local SVM approach. In *ICPR (3)*, pages 32–36, 2004

[89] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005

[90] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *British Machine Vision Conference*, page III:1249, 2006

[91] Yan Ke, Rahul Sukthankar, and Martial Hebert. Efficient visual event detection using volumetric features. In *ICCV*, pages 166–173, 2005

Holistic Representations

Holistic representations contrast part-based representations. Bobick et al.[92] proposed *motion history images* (MHI) as a meaningful way to encode short spans of motion. For each frame of the input video the MHI is the gray scale image which records the location of motion, where recent motion has high intensity values and older motion produces lower intensities. An example of a motion history image is shown in Figure 24.

For each frame of the input video, a MHI is produced from the motion in the current frame and the MHI of the previous frame: the MHI of the previous frame is multiplied by a scalar smaller than one and the new motion is added on top of it. Thus, older motions are assigned lower values in the MHI. The MHI representation can be matched efficiently using global statistics, such as

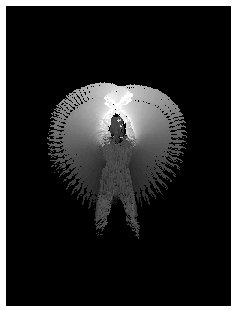


Figure 24: Motion History Image, where motion causes a response which decays temporally. Figure due to Bobick et al.

[92] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(3):257–267, 2001

moment features.

Weinland et al.[93] extended the idea to *motion history volumes* by means of multiple cameras. By using such controlled environment a high classification accuracy and desirable invariances can be achieved. However, for most practical cases, Weinland's environment with five cameras around the scene is too expensive or difficult to setup.

Efros et al.[94] created stabilized spatio-temporal volumes for each object whose action is to be classified. For each volume a smoothed dense optical flow field is extracted and used as descriptor. Their method is particularly well suited for classifying the actions of distant objects where detailed information is unavailable.

Yilmaz and Shah[95] again use a spatio-temporal volume, but only project the contour of each frame into the volume. Descriptors encoding direction, speed and local shape of the resulting surface are generated by measuring local differential geometrical properties.

Zelnik-Manor and Irani[96] describe features derived from space-time gradients at multiple temporal scales. To compare two sequences of these features, they use a sliding-window of fixed size and the distance between two such windows is calculated as χ^2-distance or Mahalanobis distance. Their method works well to cluster a single long video sequence into similar actions, as well as to recognize actions in real-time.

There is a large body of work which first recover the posture of the human actor by means of tracking and fitting a detailed model of the human body. Action classification can then be performed by using the intrinsic model parameters as features, providing great robustness and invariance. Representatively, let us cite the work of Yacoob and Black[97], Ramanan and Forsyth[98], Agarwal and Triggs[99] and for an unsupervised method, Song et al.[100].

COMPARING PART-BASED AND HOLISTIC REPRESENTATION, part-based representations treat the video as a set of independent features, where each feature is equally important, and by discarding the position information they are robust against changes in both space and time dimensions. A practical drawback of part-based representations is the variable size of the resulting representations, which is often overcome by producing a histogram of fixed-length. Naturally, part-based representations do not require tracking and are often more resistant to clutter, as only few parts may be occluded.

Holistic representations derive a fixed-length description vector for each object whose action is to be classified. Approaches using these representations often require more preprocessing of the input data, such as object tracking, registration, shape fitting or optical flow field calculation. Provided the environment conditions can be controlled, these approaches perform very well.

Each of the above methods has its particular strength but is also limited in its application. In particular, the part-based methods discussed discard the

[93] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2-3):249–257, 2006

[94] Alexei A. Efros, Alexander C. Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *ICCV*, pages 726–733, 2003

[95] Alper Yilmaz and Mubarak Shah. Actions sketch: A novel action representation. In *CVPR*, pages 984–989. IEEE Computer Society, 2005. ISBN 0-7695-2372-2

[96] Lihi Zelnik-Manor and Michal Irani. Event-based analysis of video. In *CVPR*, pages 123–130. IEEE Computer Society, 2001. ISBN 0-7695-1272-0

[97] Yaser Yacoob and Michael J. Black. Parameterized modeling and recognition of activities. *Computer Vision and Image Understanding*, 73(2):232–247, 1999

[98] Deva Ramanan and David A. Forsyth. Automatic annotation of everyday movements. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *NIPS*. MIT Press, 2003. ISBN 0-262-20152-6

[99] Ankur Agarwal and Bill Triggs. Learning to track 3D human motion from silhouettes. In *ICML*. ACM, 2004

[100] Yang Song, Luis Goncalves, and Pietro Perona. Unsupervised learning of human motion. *IEEE Trans. Pattern Anal. Mach. Intell*, 25(7):814–827, 2003

temporal order of the parts, which contains useful information to disambiguate actions. For example, consider the disciplines of *hurdle race* and *long jump*, where the global temporal order of motions (running, jumping) is important to discriminate between the two. Therefore, in this work we use a part-based view but *preserve* information about the relative temporal order of spatio-temporal words by proposing a classifier for a sequential representation.

In the next section we introduce labeled sequence structures as a specialization of the substructure poset framework introduced in the first chapter.

Labeled Sequence Structures

In order to apply our structured input framework, we first define the substructure poset, then a total order and the associated reduction mapping.

Definition 15 (Sequence) *Given a ground alphabet set Σ, a sequence $s \in (2^\Sigma)^*$, $s = (s_1, s_2, \ldots, s_\ell)$ is an ordered list of* elements *s_i, that are finite subsets of Σ, i.e., $s_i \subseteq \Sigma$. Let $\mathcal{S}$ be the set of all sequences and $\emptyset = ()$ be the empty sequence. Let $\ell : \mathcal{S} \to \mathbb{N}$ be the* length of a sequence, *i.e., the number of elements of the sequence.*

Some example sequences are shown in Figure 25.

s: ({a,b},{c},{a,b})
t: ({b},{a})
u: ({c},{c})
v: ({a},{a})
w: ({a,b},{a,c},{c},{a,b,c})
y: ({d},{a,d},{a,c})

Figure 25: Example sequences: each sequence is composed of elements, each of which is a subset of the alphabet $\Sigma = \{a, b, c, d\}$.

Definition 16 (Subsequence) *We define a partial order $\subseteq$: $\mathcal{S} \times \mathcal{S} \to \{\top, \bot\}$ such that for any $s, t \in \mathcal{S}$ we have $s \subseteq t$ iff*

$$\exists (i_1, \ldots, i_{\ell(s)}) \text{ with } i_p > i_q \text{ for all } p > q,$$
$$i_k \leq \ell(t) \quad \forall k, \text{ such that } \forall k = 1, \ldots, \ell(s) : s_k \subseteq t_{i_k}.$$

Note that the subsequence relation is defined such that a sequence matches into a longer sequence if the individual elements of the shorter sequence can be assigned in order to elements of the longer sequence, such that they are subsets. The assignment can create arbitrary long gaps; only the order is required.

Figure 26 shows examples of the subsequence relation for the sequences shown in Figure 25. For example, we have $v \subseteq s$ by matching the two $\{a\}$ elements of v to the first and third element of s, respectively.

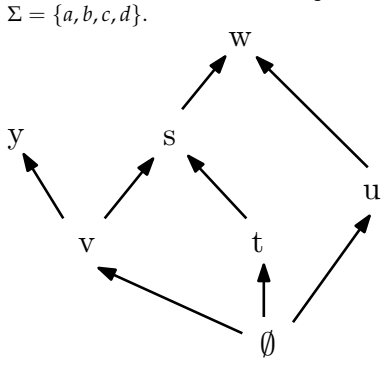


Figure 26: Hasse diagram of the subsequence relation poset structure for the example sequences shown in Figure 25. For example, $t \subseteq s \subseteq w$.

The above definitions form a substructure poset.

Lemma 4 (Sequence Poset) *$(\mathcal{S}, \subseteq)$ is a substructure poset.*

Proof. We have $\emptyset = () \subseteq s$ for all $s \in \mathcal{S}$. The relation $\subseteq$ is i) *antisymmetric*; for this take $s, t \in \mathcal{S}$ and assume $s \subseteq t$, $t \subseteq s$. From this, we must have $\ell(s) \leq \ell(t)$ and $\ell(t) \leq \ell(s)$ and thus $\ell(s) = \ell(t)$. Due to index monotonicity we have for all $i = 1, \ldots, \ell(s)$ that $s_i \subseteq t_i$ and $t_i \subseteq s_i$, therefore $s_i = t_i$ and $s = t$. The relation is ii) transitive; take $s, t, u \in \mathcal{S}$ and let $s \subseteq t$ with $(i_1, \ldots, i_{\ell(s)})$ and $t \subseteq u$ with $(j_1, \ldots, j_{\ell(t)})$. Then we also have $s \subseteq u$ with $(j_{i_1}, j_{i_2}, \ldots, j_{i_{\ell(s)}})$. The relation is iii) reflexive; for all $s \in \mathcal{S}$ we have $s \subseteq s$ with $(1, 2, \ldots, \ell(s))$ mapping. Thus $(\mathcal{S}, \subseteq)$ is a substructure poset. □

The substructure poset guarantees a high-capacity substructure-induced feature space. However, for applying the substructure Boosting framework we need to be able to enumerate $\mathcal{S}$ efficiently. To this end, if we choose option (B) of Figure 8 and directly define a reduction mapping f then the implied inverse reduction mapping allows efficient enumeration.

Definition 17 (Reduction Mapping for Sequences) *Given $(\mathcal{S}, \subseteq)$ defined on a ground alphabet Σ, and given a total order $\leq\colon \Sigma \times \Sigma \to \{\top, \bot\}$, we define $f : \mathcal{S} \setminus \{\emptyset\} \to \mathcal{S}$ as*

$$f(s) = \begin{cases} (s_1, s_2, \ldots, s_{\ell(s)-1}) & \text{if } s_{\ell(s)} = \emptyset, \\ (s_1, s_2, \ldots, s_{\ell(s)} \setminus \{e \in s_{\ell(s)} | e' \leq e, \forall e' \in s_{\ell(s)}\}) & \text{otherwise.} \end{cases}$$

In Table 6 we illustrate iterative application of the reduction mapping to the sequences shown in Figure 25. The reduction mapping is straightforward to understand: remove the highest item in the last element. If there is no item in the last element, remove the element.

s	t	u	v	w	y
				$(\{a,b\},\{a,c\},\{c\},\{a,b,c\})$	
				$(\{a,b\},\{a,c\},\{c\},\{a,b\})$	
				$(\{a,b\},\{a,c\},\{c\},\{a\})$	
				$(\{a,b\},\{a,c\},\{c\},\emptyset)$	
$(\{a,b\},\{c\},\{a,b\})$				$(\{a,b\},\{a,c\},\{c\})$	$(\{d\},\{a,d\},\{a,c\})$
$(\{a,b\},\{c\},\{a\})$				$(\{a,b\},\{a,c\},\emptyset)$	$(\{d\},\{a,d\},\{a\})$
$(\{a,b\},\{c\},\emptyset)$				$(\{a,b\},\{a,c\})$	$(\{d\},\{a,d\},\emptyset)$
$(\{a,b\},\{c\})$				$(\{a,b\},\{a\})$	$(\{d\},\{a,d\})$
$(\{a,b\},\emptyset)$	$(\{b\},\{a\})$	$(\{c\},\{c\})$	$(\{a\},\{a\})$	$(\{a,b\},\emptyset)$	$(\{d\},\{a\})$
$(\{a,b\})$	$(\{b\},\emptyset)$	$(\{c\},\emptyset)$	$(\{a\},\emptyset)$	$(\{a,b\})$	$(\{d\},\emptyset)$
$(\{a\})$	$(\{b\})$	$(\{c\})$	$(\{a\})$	$(\{a\})$	$(\{d\})$
$(\emptyset)$	$(\emptyset)$	$(\emptyset)$	$(\emptyset)$	$(\emptyset)$	$(\emptyset)$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Table 6: Example reductions for the sequences shown in Figure 25.

Lemma 5 (Inverse Reduction Mapping for Sequences) *Given $(\mathcal{S}, \subseteq)$ with ground alphabet Σ, the inverse $f^{-1} : \mathcal{S} \to 2^{\mathcal{S}}$ of f is given as*

$$\begin{aligned} f^{-1}(s) &= \{t \in \mathcal{S} \setminus \{\emptyset\} | f(t) = s\} \\ &= \{(s_1, s_2, \ldots, s_{\ell(s)}, \emptyset)\} \cup \\ &\quad \{(s_1, s_2, \ldots, s_{\ell(s)} \cup \{e \in \Sigma \setminus s_{\ell(s)} | e' \leq e, \forall e' \in s_{\ell(s)}\}\}. \end{aligned}$$

Proof. Follows in a straightforward way from Definition 17. □

Unlike in the previous chapter, where we considered the case of labeled graphs, the inverse reduction mapping for sequences can be evaluated efficiently. Therefore Algorithm 2 has output polynomial time complexity.

In the following section we explain how videos can be naturally represented as labeled sequences.

Sequence Representation of Videos

As a basis of our sequence representation, we use the spatio-temporal detector of Dollár which has shown good experimental performance in Niebles et al.[101] and Dollár et al.[102] for human action classification.

In the Dollár detector, a response function $R = (I * g * h_{ev})^2 + (I * g * h_{od})^2$ is calculated at each spatio-temporal voxel (x, y, t) in the video volume I. In the spatial directions, a 2D Gaussian kernel g with bandwidth σ is used, while temporally, a quadrature pair of 1D Gabor filters

$$h_{ev}(t; \tau, \omega) = -\cos(2\pi t\omega)e^{-t^2/\tau^2}$$

and

$$h_{od}(t; \tau, \omega) = -\sin(2\pi t\omega)e^{-t^2/\tau^2}$$

is used. The Gabor filters respond strongest on temporal intensity changes that vary at the frequency ω, which has to be set in advance. Maxima of the three dimensional function R define a sparse set of points in the video volume. These maxima are the so-called interest points.

For each interest point found, we have the spatio-temporal coordinates (x, y, t) as well as the *descriptor*, the concatenated vector of voxel values in the neighborhood of the point. Typically, we have volumes of size $13 \cdot 13 \cdot 19$ voxels, so the descriptor is a 3211-dimensional vector. To reduce the dimensionality, principal components analysis is used to keep only the projections of the descriptor onto the 25 components of highest variance.

The reduced descriptors in $\mathbb{R}^{25}$ are clustered using k-means clustering to produce a codebook of prototypes. Using the codebook, a video is represented as a set of words of the form (x, y, t, w), where (x, y) are the coordinates in the video frame t and w is the codebook index.

Finally, the words are sorted ascendingly by their time components t and then grouped into *temporal bins* as shown in Figure 27, where the first frame a feature occurred is denoted *start*, the last frame is denoted *end*. Two parameters determine how the features are mapped into the temporal bins, i) the number of temporal bins B, and ii) the temporal overlap τ, with $0 \leq \tau < 1$. The length of each temporal bin is simply the overall number of frames (end-start), divided by $B/(1 + \tau)$, such that a large value of τ denotes a larger overlap. The bins are distributed equidistant over the range of found features. Since the bins are overlapping, it is possible that a word is assigned to more than

[101] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *British Machine Vision Conference*, page III:1249, 2006

[102] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005

two bins. Although for the experiments we will keep B fixed over all videos, our representation and algorithms do not require this and we could have a variable number of sequence elements.

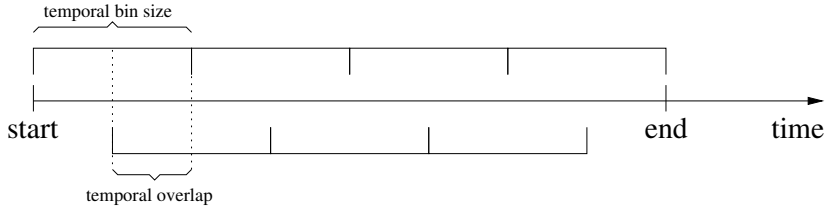


Figure 27: Temporal binning scheme: A number of overlapping temporal bins are distributed equidistantly over the video frames. Here $B = 7$, $\tau = 0.5$.

Now each video is encoded as a labeled sequence of sets of integers, such that it fits our definition of sequence (Definition 15).

Classifier

Action recognition is a multiclass classification problem in general, but first we focus on the binary classification problem. Let us denote the training data as $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{\ell}$, where $\boldsymbol{x}_n$ is the sequence corresponding to a video and $y_n \in \{-1, 1\}$ is a class label. We use the TCBoost algorithm (Algorithm 1) to construct the classification function as a linear combination of weak hypothesis functions. Our hypothesis functions are the substructure Boosting weak learners, defined earlier (Definition 5).

Therefore we have the parameter domain of the weak learners as $\Omega = \mathcal{S} \times \{-1, 1\}$ and a final learned classification function of the form

$$F(s; \boldsymbol{\alpha}) = \sum_{(t,d)\in\Omega} \alpha_{t,d} h(s; (t, d)),$$

with

$$h(s; (t, d)) = \begin{cases} d & \text{if } t \subseteq s, \\ -d & \text{otherwise.} \end{cases}$$

Learning therefore consists of producing a parameter vector $\boldsymbol{\alpha} \in \mathbb{R}^{\Omega}$. After learning we can classify a new sequence u by evaluating $F(u; \boldsymbol{\alpha})$.

To learn $\boldsymbol{\alpha}$ in the experiments we will use TCBoost with the original Hinge loss formulation of LPBoost, corresponding to the limit of the generalized linear programming Boosting formulation (12) for $p \to 1$.

Using TCBoost as structure classifier allows us to learn two-class decision functions. To solve a *multiclass* learning problem we use a 1-vs-1 class decomposition in the form of a *decision directed acyclic graph (DDAG)*[103], producing for k classes $\frac{k(k-1)}{2}$ 1-vs-1 problems. While this is similar to the usual 1-vs-1 decomposition, the DDAG offers the additional advantage that we do not have to resolve ties during test time. Instead, for decision DAGs, the DAG structure is not unique. We use the fixed decomposition as described in Platt et al.

We now evaluate the approach experimentally.

[103] John C. Platt, Nello Cristianini, and John Shawe-Taylor. Large margin DAGs for multiclass classification. In *NIPS*, pages 547–553. The MIT Press, 1999

Experiments and Results

To evaluate our substructure approach, we use the KTH human action classification data set of Schüldt et al.[104], available online[105]. It consists of 25 individuals, each performing six activities (boxing, hand-clapping, handwaving, jogging, running and walking) under four different environment conditions. Together, with one broken video file removed, the data set totals 599 video clips. We used the training, validation and testing splits as proposed in Schüldt et al., such that the sets contain 191, 192 and 216 samples, respectively.

[104] Christian Schüldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local SVM approach. In *ICPR (3)*, pages 32–36, 2004

[105] `http://www.nada.kth.se/cvap/actions/`

Typical frames from the six actions in the KTH data set are shown in Figure 28.

Figure 28: KTH Action Classification dataset with six actions and a total of 599 video sequences. The actions are shown in alphabetical order: boxing, hand-clapping, handwaving, jogging, running, walking.

The spatio-temporal features were extracted as described in the previous section using the toolbox[106] provided by Piotr Dollár with the default settings.[107]

[106] `http://vision.ucsd.edu/~pdollar/research/research.html`

[107] Parameters to `stfeatures` function: $\sigma = 2, \tau = 3$, thresh $= 0.0002$, overlap_r $= 1.85$, shr_spt $= 2$, tau_spt $= 1$ and we use $\omega = \frac{1}{2\tau}$ for all experiments.

Model selection is performed on the training and validation sets followed by a single training run on the combined training+validation set with the best parameters of the validation phase. The final reported classification accuracy is the one evaluated once on the test set. Codebooks of sizes 128, 192, 256, 384, 512, 768 and 1024 codewords are created from the training set descriptors. In all experiments, the same features and codebooks are used to produce sequences as well as the histograms, such that all benchmarked approaches use exactly the same features.

For model selection, the number of bins is varied from $B = 1$ to $B = 15$; the temporal overlap $\tau = 0.5$ remains fixed. The LPBoost regularization parameter ν is set to 0.01, 0.05, 0.1 and 0.25. All combinations of codebook sizes, B and ν have been tested.

For the model selection of the baseline classifiers, the histograms are preprocessed in one of the following two ways, i) the 1-norm of the histogram is normalized, or ii) the histogram is "*binarized*", that is all non-zero entries of the histogram are set to one. This is a common preprocessing step for

bag-of-words models in computer vision.

As SVM kernel we use the linear kernel, the RBF Gaussian kernel and the χ^2-histogram-kernel

$$K(h, h') = \exp\left(-\frac{1}{A}\left[\frac{1}{2}\sum_{\{n:h_n+h'_n>0\}}\frac{(h_n - h'_n)^2}{h_n + h'_n}\right]\right).$$

For the RBF Gaussian and χ^2-kernel the kernel width has been selected as the mean Euclidean and mean χ^2 distance between all training samples, respectively. This is a common heuristic choice known to work well in practice. As multiclass decomposition both 1-vs-rest and 1-vs-1 decompositions have been tested.

In total, for the SVM baseline all combinations of the codebook sizes, histogram preprocessing methods, multiclass decompositions and kernel choices are part of the model selection procedure. Thus the model selection for the SVM baseline is much more exhaustive than in previous works[108].

[108] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005; and Christian Schüldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local SVM approach. In *ICPR (3)*, pages 32–36, 2004

[109] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *British Machine Vision Conference*, page III:1249, 2006

[110] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005

[111] Christian Schüldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local SVM approach. In *ICPR (3)*, pages 32–36, 2004

[112] Yan Ke, Rahul Sukthankar, and Martial Hebert. Efficient visual event detection using volumetric features. In *ICCV*, pages 166–173, 2005

Results

The classification results of our Subsequence Boosting approach, the results of the baseline SVM classifiers and the results from the literature are shown in Table 7. The literature results are from Niebles et al. [109], Dollár et al. [110], Schuldt et al. [111], and Ke et al. [112].

During the model selection process a codebook with 768 codewords turned out to be consistently the best for all tested classifier types. Each of our 1-vs-1 class Subsequence Boosting classifiers selected around 20-70 active patterns, where the tendency is fewer and shorter patterns for classes that are easy to distinguish (e.g. boxing versus running), and more and longer patterns for difficult-to-separate classes.

Figure 29 visualizes the sequence of a single selected feature of a trained classifier. In Figure 30 we further illustrate how the subsequences typically match into unseen test sequences. The confusion matrix for our Subsequence Boosting classifier is shown in Figure 31.

Our features and preprocessing seem to be of high quality, given that the baseline SVM method produces better results than reported in the literature. In part, this is also due to more thorough model selection, as noted above.

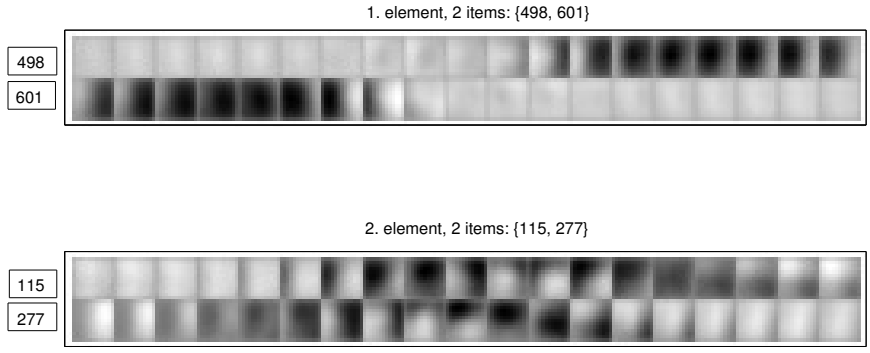


Figure 29: A discriminative pattern. Here, the pattern sequence is $\{498, 601\}\{115, 277\}$ and was selected in the jogging-vs-walking classifier. Each row in the figure shows a codebook vector as 19 frames of size 13x13 over time. The pattern was assigned a negative ω-weight, such that the presence of this pattern will influence the decision towards the walking class.

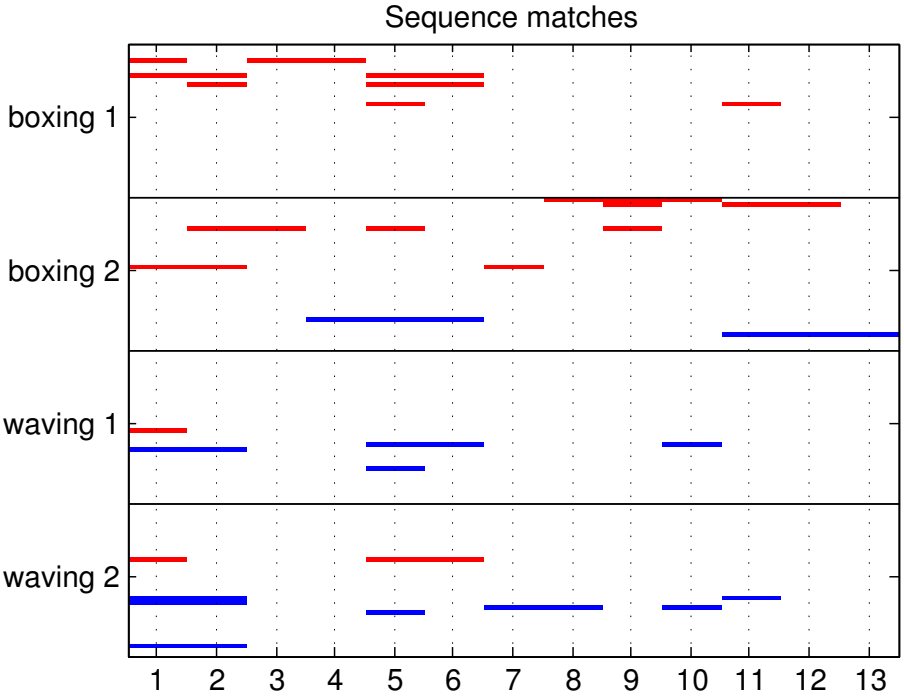


Figure 30: Visualization of how the most influential patterns match in four unseen test sequences in the boxing-vs-handwaving classifier, for the case of a 768-word codebook and 13 temporal bins. Each of the four rows shows a distinct test videos, where the first two correspond to boxing, the latter two to handwaving. We visualize the 32 pattern sequences of the decision stumps with the highest coefficient value α. Sequences voting for boxing ($\omega = 1$) are shown at the top of each row in red (•) and sequences voting for handwaving ($\omega = -1$) are shown at the bottom of each row in blue (•). All four test sequences are classified correctly.

Method	KTH accuracy
Niebles et al., BMVC 2006, LOO, pLSA	81.50
Dollár et al., 2005, LOO, SVM RBF	80.66
Schuldt et al., ICPR 2004, splits, SVM match	71.71
Ke et al., ICCV 2005, splits, forward feat.-sel.	62.94
baseline SVM linear bin, 1-vs-1	83.33
baseline SVM RBF bin, 1-vs-1	85.19
baseline SVM χ^2 bin, 1-vs-1	87.04
Subsequence Boosting, $B = 12$, splits	84.72

Table 7: Results for the KTH human action classification data set. For all the baseline SVM and Boosting results the model selection has been performed on the validation set, followed by a single training run on the joined training+validation set. The multiclass accuracy shown is the one measured on the final test set. For the baseline SVM results, the best classifier on the validation set was found with a codebook size of 768 and a regularization parameter of $C = 10$ for all kernels. The subsequence boosting result is obtained with a codebook size of 768, $B = 12$ and $\nu = 0.05$.

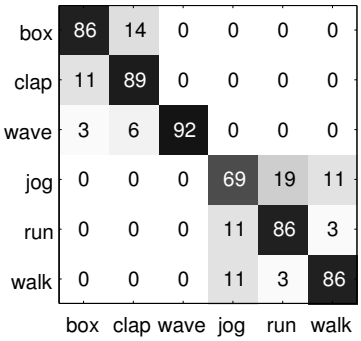


Figure 31: Confusion matrix of the Subsequence Boosting classifier on the KTH test set. The classifier was produced with a 768 element codebook, $B = 12$ and $\nu = 0.05$. Confusions happen between the boxing, hand-clapping and hand-waving classes, as well as between the jogging, running and walking classes.

Discussion

We achieved state-of-the-art classification results using our proposed algorithm and report competitive results when compared to other approached from the literature.

Our algorithm has favorable properties, such as increased interpretability of the resulting classification function, explicit feature selection, global optimal convergence and fast testing times, but in the end we did not show a clear and significant improvement of the classification accuracy over a histogram approach with a SVM classifier and nonlinear kernel.

This is quite surprising and it is not obvious why this is the case. Possibly, the KTH data set is favorable to histogram based classifiers because each action is quite homogeneous and does not involve global changes or complex behavior.

Also, as with the reported literature results, in our classifier the confusions happen in two clusters, namely i) boxing, handclapping and handwaving, and ii) jogging, running and walking. Each of these actions might be easily confused on both a local temporal scale as well as a coarse temporal scale, and we might very well do not gain much by including the temporal order of features.

Conclusion

In this chapter we proposed a novel classifier for sequence representations, suitable for action classification in videos. A goal of our work is to make efficient pattern selection algorithms and the substructure based classification framework accessible to the computer vision community. Experimentally we achieved state-of-the-art performance, but our original motivation of improving accuracy by incorporating temporal relationships has not been fulfilled.

Given this result, we would like to apply our approach to classify higher order action patterns in the future with the hope that for these actions the temporal ordering plays a more important role. Unfortunately the lack of an openly available action classification data set for such high level actions is currently a problem[113].

[113] All algorithms and experiments are made available under the GNU General Public License at `http://www.kyb.mpg.de/bs/people/nowozin/pboost/`

PART II

Structured Prediction

All models are wrong, but some are useful.

George Box

All models are wrong, and increasingly you can succeed without them.

Peter Norvig

Introduction

This chapter is concerned with prediction tasks in which the target variable $\boldsymbol{y}$ comes from a structured domain.[114] Structured in this setting is a vague notion, but usually it is assumed that the target domain satisfies one or more of the following criteria:

1. the set of possible output values $\boldsymbol{y} \in \mathcal{Y}$ and its dimensionality depends on the instance $\boldsymbol{x}$, i.e., the target domain $\mathcal{Y}(\boldsymbol{x})$ is a function of the instance $\boldsymbol{x}$,
2. not all of the representable target values are allowed, i.e., there exist constraints on what values are feasible predictions,
3. there exists some formalizable "structure" on the output space, for example a semi-metric distance function on the target domain.[115]

[114] An alternative name is structured output learning.

[115] A semi-metric distance satisfies non-negativity $d(y, y') \geq 0$, identity of indiscernibles $d(y, y') = 0 \Leftrightarrow y = y'$, and symmetry $d(y, y') = d(y', y)$.

Typical machine learning problems like classification and regression do not satisfy these criteria because the output space is small and fixed and does not have a structure which is particularly problem-dependent. In this thesis we limit ourselves to the case where the target variable comes from a finite but possibly very large set. Many problems related to structured prediction such as inference and learning then become combinatorial optimization problems.

The purpose of this chapter is to provide a partial literature overview of structured prediction methods with a focus on techniques popularly used in computer vision. It does not contain novel research results.

When dealing with a structured domain, it is natural to represent beliefs as to what value is the correct prediction as a probability distribution over the elements of the underlying feasible set. However, because this set is large[116] concise *representation* of this distribution becomes an issue. Such a representation need not only be compact, but it should allow efficient manipulation and computation for a number of desirable tasks.

[116] Imagine as an example an image labeling task where each pixel has one of two states. The number of possible labelings grows as $O(2^n)$ in the number n of pixels.

Graphical Models

PROBABILISTIC GRAPHICAL MODELS[117] are models addressing these issues. They allow efficient representation as well as manipulation and computation of interesting quantities related to the specific distribution or family of distributions which they represent. We will use them in this and the subsequent chapter.

[117] Steffen L. Lauritzen. *Graphical Models*. Clarendon Press, Oxford, 1996. ISBN 0-19-852219-3; and Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006

Graphical models come in two flavors. *Directed graphical models*, also known as Bayesian networks and *undirected graphical models*, also known as Markov networks or Markov random fields (MRF). Both represent a family of joint distributions over the target domain. They differ in their factorization and conditional independence relations, which specify the way the distribution can be decomposed into smaller parts and constrain the relationship between these parts. Because we will eventually apply an undirected graphical model to solve computer vision problems, we restrict ourselves to undirected graphical models only, which are more popular and better suited for computer vision applications.[118]

[118] Some researchers disagree for practical purposes, see e.g.

Justin Domke, Alap Karapurkar, and Yiannis Aloimonos. Who killed the directed model? In *CVPR*. IEEE Computer Society, 2008

UNDIRECTED GRAPHICAL MODELS, also known as Markov networks, specify a family of probability distributions by means of an undirected, simple graph $G = (V, E)$. The graph encodes a set of conditional independence assumptions about all distributions in the family; by making use of this conditional independence the distribution can be efficiently represented and efficient algorithms can be derived.

In this thesis we will denote random variables by uppercase letters, their values by the corresponding lowercase ones. For example, if X is a random variable taking values on a finite set $\mathcal{X}$, then $x \in \mathcal{X}$ is a specific value and $p(X = x) = p(x)$ is the probability.

For discrete random variables X,Y,Z conditional independence of X and Y given Z, written as $X \perp\!\!\!\perp Y|Z$, simply states that the conditional joint probability of X and Y factorizes into the separate conditional probabilities of X and Y, i.e., we have for all x, y, z that

$$P(X = x, Y = y|Z = z) = P(X = x|Z = z)P(Y = y|Z = z).$$

To make clear the independence assumptions encoded by the graph we now define two properties known as *Markov properties*. For this, assume a given set of random variables defined over the nodes, $(X_i)_{i\in V}$ taking values in a probability space $(\mathcal{X}_i)_{i\in V}$. The joint space is denoted by $\mathcal{X}$, i.e., we have $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_{|V|}$ and the vector of random variables $X \in \mathcal{X}$ is denoted by $X = (X_1, \ldots, X_{|V|})$. Let us further denote by X_A the subset of random variables indexed by $A \subseteq V$, and by $\boldsymbol{x}_A$ the subset of elements of a vector $\boldsymbol{x} \in \mathcal{X}$ restricted to A. Likewise, if $A = \{i\}$ we simply write X_i and x_i.

The following two properties are defined by means of the given graph G.

[119]

1. *Pairwise Markov Property*: we have $\forall i, j \in V : i \neq j \wedge (i,j) \notin E$: $i \perp\!\!\!\perp j \,|\, V \setminus \{i,j\}$.
2. *Global Markov Property*: we have for all disjoint sets $I \subset V$, $J \subset V$, $S \subset V$ with S being a vertex-separator set of I, J in G that $I \perp\!\!\!\perp J \,|\, S$.

The global Markov property implies the pairwise Markov property by taking $I = \{i\}$, $J = \{j\}$ and $S = V \setminus \{i,j\}$.

It is natural to ask how, given a graph G, a probability distribution satisfying the above two properties with respect to G can be specified. It turns out that all distributions which can be represented by a factorization respecting the graph structure automatically satisfy the global Markov property. Such factorization is of the form

$$p(X = \boldsymbol{x}) = p(\boldsymbol{x}) = \frac{1}{Z} \prod_{\substack{A \subseteq V \\ A \text{ complete}}} \psi_A(\boldsymbol{x}_A), \tag{25}$$

where a subset A of the vertex set is said to be *complete* if $A \times A = E_A$, such that for each pair $(i,j) \in A \times A$ we have $(i,j) \in E$. Further, we have non-negative factors $\psi_A : \mathcal{X}_A \to \mathbb{R}$, also known as *potential functions*[120] and a normalization constant referred to as *partition function*,

$$Z = \sum_{x \in \mathcal{X}} \prod_{\substack{A \subseteq V \\ A \text{ complete}}} \psi_A(x_A).$$

When a probability distribution can be described by (25) it is said to *factorize according to* G. The factorization is not necessarily unique and during modeling one often *starts* by specifying the factorization directly such that it best suits the task at hand.

A factorization according to G implies the global Markov property with respect to G which in turn implies the pairwise Markov property with respect to G. For a proof of existence of this factorization and its relations to the Markov properties see Proposition 3.8 in Lauritzen[121].

The above argument is an implication: a distribution of the form (25) satisfies the Markov properties with respect to G. For the other direction, if a given distribution $p(x)$ satisfies the pairwise Markov property with respect to a given graph G *and* it has $p(x) > 0$ for all $x \in \mathcal{X}$ then the converse is also true, i.e., it factorizes according to G. This result is known as the Hammersley-Clifford theorem.[122] Additionally, not only there exists a factorization of the form (25), but moreover a limited form (25) restricted to maximal cliques[123] is guaranteed to exist, i.e., a factorization which has $\psi_A(x_A) = 1$ whenever the subgraph induced by A is *not* a maximal clique. The distribution can therefore be represented as

$$p(X = \boldsymbol{x}) = p(\boldsymbol{x}) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C),$$

[119] There exists another Markov property called the *local Markov property*, see the Lauritzen book on graphical models.

[120] Some authors, e.g. Wainwright, use the word potential function for functions in the exponential.

[121] Steffen L. Lauritzen. *Graphical Models*. Clarendon Press, Oxford, 1996. ISBN 0-19-852219-3

[122] Although not the convention, some authors limit their definition of "random field" to distributions which satisfy $p(x) > 0$ for all $x \in \mathcal{X}$. See for example section 3.1 in:

Gerhard Winkler. *Image Analysis, Random Fields, and Dynamic Monte Carlo Methods: A Mathematical Introduction*. Springer, 1995

[123] A *clique* is a dense subgraph $G' = (V', E')$ with $V' \subseteq V$, $V' \times V' = E' \subseteq E$. A clique is *maximal* if there is no superset B, with $A \subset B \subseteq V$ which is also a clique.

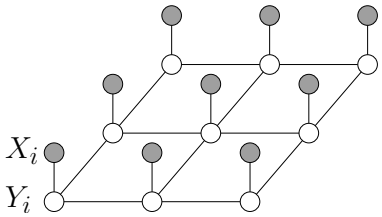


Figure 32: Typical MRF setup in computer vision: a 3-by-3 pixel grid with two random variables X_i, Y_i for each pixel i. The observation variable X_i could be the measured image intensity of the pixel, and the latent variable $Y_i \in \{0, 1\}$ could represent that the pixel is a foreground pixel.

where $\mathcal{C}$ is the set of all cliques in G. In general however, we will only assume that there exists a $\boldsymbol{x} \in \mathcal{X}$ such that $p(\boldsymbol{x}) > 0$ and there can be some $\boldsymbol{x} \in \mathcal{X}$ for which for some factors we have $\psi_A(\boldsymbol{x}_A) = 0$. From now on we will use the shorthand notation $p(\boldsymbol{x})$ to denote $p(X = \boldsymbol{x})$.

Markov Random Fields for Images

When applying undirected graphical models to images, one typically associates to each pixel i in the image two random variables: one observation variable X_i and one variable Y_i representing a latent state of interest. For example, $X_i \in \{0, 1, \ldots, 255\}$ might represent the measured pixel intensity in the image and $Y_i \in \{0, 1\}$ represents whether the pixel is part of a foreground object. The graph structure of the random field is typically derived by a fixed neighborhood relation. Figure 32 shows a Markov random field for nine pixels where neighbors in the 4-neighborhood are connected.

The central modeling assumption made in this construction is that the observation variables X are *conditionally independent* given the latent states Y. This assumption can be understood visually in the graph shown in Figure 32 by means of the global Markov property: any pair of X_i, X_j is conditionally independent on the set of latent variables Y.

In the factorized representation (25) we have not specified the functional form of the factors ψ_A. For reasons which will become clear later it is convenient to represent these factor functions as exponentials of the negative of an *energy function* E_A, i.e., to define each factor ψ_A in (25) as

$$\psi_A(x_A) = \exp\{-E_A(x_A)\}.$$

This representation is called *Boltzmann distribution* and the energy function $E_A : \mathcal{X}_A \to \mathbb{R}$ can be arbitrarily defined. Low energies correspond to likely configurations, and high energies to unlikely ones.

We now simplify the notation used in (25) by using energy functions. In the above image example we have two sets of random variables, the observations X and the latent states Y. Therefore (25) can be rewritten to make clear the two sets of variables as

$$p(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{Z} \prod_{\substack{A \subseteq V \\ A \text{ complete}}} \psi_A(x_{A_x}, y_{A_y}) = \frac{1}{Z} \prod_{\substack{A \subseteq V \\ A \text{ complete}}} \exp\{-E_A(x_{A_x}, y_{A_y})\}, \quad (26)$$

where we denote by $A_x \cup A_y = A$ the disjoint sets of indices of random variables, and by x_{A_x} and y_{A_y} the subsets of random variables themselves. The partition function is

$$Z = \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}} \prod_{\substack{A \subseteq V \\ A \text{ complete}}} \exp\{-E_A(x_{A_x}, y_{A_y})\}.$$

Because a product of exponentials is equivalent to an exponential of sums of the individual inner terms, we can define a joint energy function as

$$E(\boldsymbol{x}, \boldsymbol{y}) := \sum_{\substack{A \subseteq V \\ A \text{ complete}}} E_A(x_{A_x}, y_{A_y}), \tag{27}$$

such that (26) becomes

$$p(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{Z} \exp\{-E(\boldsymbol{x}, \boldsymbol{y})\}, \tag{28}$$

with $Z = \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}} \exp(-E(\boldsymbol{x}, \boldsymbol{y}))$. Therefore, specifying the distribution $p(\boldsymbol{x}, \boldsymbol{y})$ has been reduced to specifying the form and decomposition of the energy function.

A TYPICAL ENERGY FUNCTION for the example shown in Figure 32 would take into account the a priori probability of a pixel being a foreground pixel. It would also model the *pairwise* relations between adjacent y_i, y_j as nearby pixels are likely to be correlated in their property of being foreground, such that $y_i = 1$ would make it more likely that $y_j = 1$ and vice versa. Another part of the energy function would model the pairwise relation between the observation x_i and its latent state y_i, that is, the energy would couple the observed pixel intensity to the probability of being foreground. For example in some applications pixels with high intensity are more likely to be foreground pixels.

THE DECOMPOSITION INTO FACTORS in (25) or equivalently into subsets A in (27) can be most conveniently described with a so called *factor graph*[124].

A factor graph is a bipartite[125] graph consisting of a set of *factor nodes* and *variable nodes*. Factor graphs make the form of the factorization specific. For our example shown in Figure 32 one suitable factorization as a factor graph is shown in Figure 33.

Each square-shaped factor node represents a factor depending only on its adjacent variables. Conversely, each round node represents a random variable and is connected only to factor nodes. In our example we would have three kinds of factors,

1. $\psi_i^1 : \mathcal{Y}_i \to \mathbb{R}$, a so called *unary potential* for the a priori beliefs $p(Y_i)$,
2. $\psi_i^2 : \mathcal{X}_i \times \mathcal{Y}_i \to \mathbb{R}$, the *pairwise potential* linking observation and latent state of a pixel, and
3. $\psi_{i,k}^3 : \mathcal{Y}_i \times \mathcal{Y}_k \to \mathbb{R}$, the *pairwise potential* related to the adjacent pixels' latent states.

In terms of expressing these factors as exponentials of energy functions (27), we simply define $\psi_i^1(y_i) := \exp\{-E_i^1(y_i)\}$, $\psi_i^2(x_i, y_i) := \exp\{-E_i^2(x_i, y_i)\}$, and $\psi_{i,k}^3(y_i, y_k) := \exp\{-E_{i,k}^3(y_i, y_k)\}$.

[124] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001; and Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006

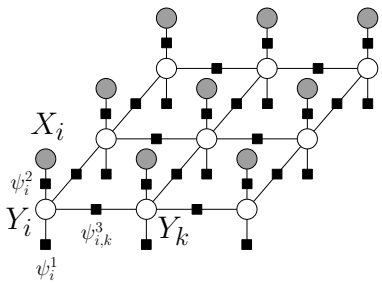


Figure 33: Typical factor graph for our MRF example. Two kind of pairwise potentials couple X_i, Y_i and Y_i, Y_k, respectively. One unary potential per pixel sets the prior probability distribution $p(y_i)$.

[125] A graph is bipartite if its vertex set can be partitioned into two sets such that there exist only edges between the two sets. For a factor graph only edges between factor nodes and variable nodes are allowed.

Inference

We will later make the exact functional form of the energies concrete. Assume for now that we found a suitable energy function for the problem and are given an observed image $\boldsymbol{x} \in \mathcal{X}$ with the task to find a latent state $\boldsymbol{y} \in \mathcal{Y}$ corresponding to $\boldsymbol{x}$. This is one example of an *inference task*: given a distribution and some observations, *infer* something about other random variables.

In our setting we are given $p(X = \boldsymbol{x}, Y)$ in terms of an energy function and the observations $\boldsymbol{x}$, and want to say something about the unobserved variables Y. We can do this by stating the conditional probability over $\boldsymbol{y} \in \mathcal{Y}$ as

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, \boldsymbol{y})}{p(\boldsymbol{x})},$$

where $p(\boldsymbol{x})$ is the same for all $\boldsymbol{y}$, hence dropping it retains proportionality, that is

$$p(\boldsymbol{y}|\boldsymbol{x}) \propto p(\boldsymbol{x}, \boldsymbol{y}).$$

If we want to find the most probable $\boldsymbol{y} \in \mathcal{Y}$ by maximizing $p(\boldsymbol{y}|\boldsymbol{x})$, we have

$$\begin{aligned} \boldsymbol{y}^* &:= \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} p(\boldsymbol{y}|\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} p(\boldsymbol{x}, \boldsymbol{y}) \\ &= \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \frac{1}{Z} \exp\{-E(\boldsymbol{x}, \boldsymbol{y})\} = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \exp\{-E(\boldsymbol{x}, \boldsymbol{y})\} \\ &= \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{Y}} E(\boldsymbol{x}, \boldsymbol{y}). \end{aligned}$$

The last step follows because $\exp : \mathbb{R} \to \mathbb{R}^+$ is a monotonically increasing function of its argument. From the derivation, the state $\boldsymbol{y}^*$ with the minimum energy $E(\boldsymbol{x}, \boldsymbol{y}^*)$ is the most probable configuration given that we have observed the image $\boldsymbol{x}$.

Finding the most likely state, i.e., the state with the *maximum a-posteriori probability* (MAP) is known as the MAP-MRF problem. Because this problem will be important in what follows, we define it separately.

Problem 3 (MAP-MRF problem) *Given a distribution $p(\boldsymbol{x}, \boldsymbol{y})$ over $\mathcal{X} \times \mathcal{Y}$ of the form*

$$p(X = \boldsymbol{x}, Y = \boldsymbol{y}) = \frac{1}{Z} \exp\{-E(\boldsymbol{x}, \boldsymbol{y})\},$$

with an energy function $E : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, and given an observation $\boldsymbol{x} \in \mathcal{X}$, the problem of finding

$$\boldsymbol{y}^* = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} p(\boldsymbol{x}, \boldsymbol{y})$$

is called the MAP-MRF problem.

The MAP-MRF problem is NP-hard in general, but later in this chapter we will describe methods to solve the problem approximately. If the graph has a

special structure, such as being a chain or a tree, the problem can be solved efficiently. For all typical models used in computer vision, this is unfortunately not the case.

Conditional Random Fields

The MRF model (28) is said to be a *generative model* because it directly specifies the *joint* distribution $p(\boldsymbol{x}, \boldsymbol{y})$. But during prediction time we are interested only in $p(\boldsymbol{y}|\boldsymbol{x})$, a *conditional* distribution. Moreover, we always observe $\boldsymbol{x}$ and therefore modeling $p(\boldsymbol{x})$ is more a burden than a degree of freedom we can use to our advantage; it is not needed for solving the MAP-MRF problem.

Conditional Random Fields (CRF), first proposed by Lafferty, McCallum and Pereira[126], directly model $p(\boldsymbol{y}|\boldsymbol{x})$. The CRF model is said to be a *discriminative model* because it does not include an explicit model of $p(\boldsymbol{x})$. As a particular MRF, CRFs are undirected graphical models.[127]

In a CRF corresponding to our MRF for images, the conditional distribution $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})$ is given as

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}) = \frac{1}{Z(\boldsymbol{x}, \boldsymbol{w})} \exp\{-E(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{w})\}, \tag{29}$$

with partition function

$$Z(\boldsymbol{x}, \boldsymbol{w}) = \sum_{\boldsymbol{y} \in \mathcal{Y}} \exp\{-E(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{w})\}. \tag{30}$$

The functional form of (29) resembles (28), the MRF joint probability. In fact, the hypothesis space considered by the two models is the same. The difference lies in the training of the two models. A CRF is trained by means of the *conditional likelihood*, a point we will elaborate on in the next section.

[126] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001

[127] An excellent introduction into Conditional Random Fields and the differences between generative and discriminative models for structured prediction can be found in:

Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In *Introduction to Statistical Relational Learning*, chapter 4. 2007

Advantages of Discriminative Models

We now discuss the advantages of the discriminative approach. The Markov random field models $p(X, Y)$ and implicitly includes a model for $p(X)$. The conditional random field models $p(Y|X = \boldsymbol{x})$ directly, without explicitly specifying a model of $p(X)$.

Intuitively the direct modeling of $p(Y|X = \boldsymbol{x})$ appeals to the *Vapnik principle*[128]: never solve a problem that is more general than what you actually need to solve.

In general, modeling of $p(\boldsymbol{x})$ is indeed difficult because the feature functions depending on $\boldsymbol{x}$ are often *highly correlated* across nodes. For example an image feature suitable for image segmentation might contain information similar to another node's feature. We would like to use the features for both nodes but they are clearly not independent. Other examples of dependent features can be found in Sutton and McCallum[129].

[128] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998

[129] Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In *Introduction to Statistical Relational Learning*, chapter 4. 2007

For dealing with this dependency, we can either choose to ignore it and thus work with a simple but wrong model, or we have to model $p(\boldsymbol{x})$, leading to intractable models. The independence assumption is encoded as missing edges between X-nodes in Figure 32. Modeling dependency would mean adding edges between these X-nodes.

[130] Tom Minka. Discriminative models, not discriminative training. Technical Report MSR-TR-2005-144, Microsoft Research (MSR), October 2005

Minka[130] provides another point of view on generative versus discriminative models: he argues that there is no such thing as the "conditional likelihood" but that by training a model using ℓ_c in (33), one implicitly trains using the standard likelihood function of a *changed model* which decouples $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})$ and a new term $p(\boldsymbol{x}|\boldsymbol{w}')$. Because $\boldsymbol{w}' \in \mathbb{R}^F$ is an additional set of parameters unrelated to $\boldsymbol{w}$, the degree of freedom of the model is enlarged. The new likelihood function decouples $\boldsymbol{w}$ and $\boldsymbol{w}'$, and by dropping the terms related to $\boldsymbol{w}'$ we obtain the "conditional likelihood". Dropping the terms is possible because $p(\boldsymbol{x}|\boldsymbol{w}')$ and thus $\boldsymbol{w}'$ is never used in computations related to $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})$.

This idea has been advanced further by introducing an explicit coupling between the generative $p(\boldsymbol{x}|\boldsymbol{w}')$ term and the discriminative term $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})$ using a *joint prior* $p(W = \boldsymbol{w}, W' = \boldsymbol{w}')$ in Lasserre et al.[131]. The resulting models are coined *generative-discriminative* hybrid models.

[131] Julia A. Lasserre, Christopher M. Bishop, and Thomas P. Minka. Principled hybrids of generative and discriminative models. In *CVPR*, pages 87–94. IEEE Computer Society, 2006

Throughout the machine learning community there is consensus that if only $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})$ is required and all training data is fully observed, then conditional random fields outperform their generative MRF counterpart.

Learning Random Field Models

The potential functions ψ_i^1, ψ_i^2, and $\psi_{i,k}^3$ and their corresponding energies from our example can be thought of as numerical tables associated to each factor in Figure 33. Each entry in the table contains the real valued non-negative potential for the corresponding states.

Because each pixel and neighborhood have the same interpretation, typically the potential functions — and therefore the tables — are replicated for each pixel and pairwise edge, such that $\psi_i^1 = \psi_j^1$ and $\psi_i^2 = \psi_j^2$ for all i, j, as well as $\psi_{i,k}^3 = \psi_{j,l}^3$, for all pairs (i, k), (j, l). In effect, this means that only one table has to be specified for each *type of potential*, independent of the image size.

In some applications, such as dense stereo reconstruction, computation of optical flow and panorama stitching, the manual design of the energy tables is a successful strategy and leads to state-of-the-art performance[132].

[132] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26 (9):1124–1137, 2004

For high-level vision tasks such as object recognition and image segmentation, however, this is not enough. There, it is often unclear how a simple observation variable like pixel intensity relates to a high-level latent state, such as "being a pixel belonging to an object of class car". Then, the manual design of energies becomes infeasible.

To overcome this limitation, a suitable potential function can be *learned* given fully observed *training data*. The basic idea to enable learning is this: specifying a potential function fixes the distribution. However, by specifying a *class of possible potential functions*, learning can be posed as the problem of selecting the right potential function from this class.

From this point of view, learning a random field boils down to two decisions to make, i) specifying the class of potential functions to use, and ii) having a method to select a good one, given the training data. We now discuss these two issues separately.

Specifying the Potential Function Class

A class of potential functions can be defined by *parametrizing* the energy functions. The parametrized energy function[133] is written as $E(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w})$ with

$$E(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w}) := \sum_{\substack{A \subseteq V \\ A \text{ complete}}} E_A(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w}), \tag{31}$$

[133] We denote this parameter by $\boldsymbol{w}$ throughout this and the following chapter.

for some convenient factorization of the graph. In the example, each factor in the factor graph of Figure 33 would be one term in the sum.

The most common method to parametrize the individual energy functions is by means of an inner product between the weight vector $\boldsymbol{w} \in \mathbb{R}^F$ and a *feature function* f. The feature function maps observations and latent states to a vector in $\mathbb{R}^F$. In our example, consider ψ_i^2, the pairwise potential between observations and latent state. We define

$$\psi_i^2(x_i, y_i) = \exp\{-E_i^2(x_i, y_i; \boldsymbol{w})\} = \exp\{-\boldsymbol{w}^\top f_i^2(x_i, y_i)\}.$$

This change frees us from having to define a fixed energy function. Instead, we only define a feature function $f_i^2 : \mathcal{X}_i \times \mathcal{Y}_i \to \mathbb{R}^F$. The output of the feature function implicitly defines the energy by means of the inner product $\boldsymbol{w}^\top f_i^2(x_i, y_i)$ and thus the potential $\psi_i^2(x_i, y_i)$ *depends* on the free parameters $\boldsymbol{w}$. We write $\psi_i^2(x_i, y_i; \boldsymbol{w})$ from now on to make this dependency clear, and also denote the joint distribution by $p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w})$.

Typically in a computer vision MRF model only a few distinct *types* of feature functions are used and these are replicated for all pixels, i.e., we would have $f_i^2 := f^2$ for all i. To design a good feature function we can incorporate features known to be relevant to the application task. This is an easier task than designing the complete energy function.

Another typical feature of parametrized MRF models is to associate a separate weight vector with each type of potential function. To illustrate this, for our example, we write the full energy as

$$E(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w}) = \sum_i w_1^\top f^1(y_i) + \sum_i w_2^\top f^2(x_i, y_i) + \sum_{(i,k) \in E} w_3^\top f^3(y_i, y_k),$$

such that each feature function has its own weight vector w_1, w_2 and w_3, as well as its own output dimension F_1,F_2, and F_3, respectively.

Maximum Likelihood Training

For training we assume a given set $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\ldots,N}$ of N training instances $(\boldsymbol{x}_n, \boldsymbol{y}_n) \in \mathcal{X} \times \mathcal{Y}$ with observed latent states $\boldsymbol{y}$. The training instances are assumed to be independent and identically distributed (iid).

The distribution specified by (31) describes a family of distributions where each member of the family is indexed by one particular value of $\boldsymbol{w} \in \mathbb{R}^F$. Suppose there exists a *true distribution* $q(\boldsymbol{x}, \boldsymbol{y})$ and we would like to estimate the parameters $\boldsymbol{w}$ in such a way that $p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w})$ best resembles $q(\boldsymbol{x}, \boldsymbol{y})$.

The Kullback-Leibler divergence $D_{KL}(q \| p; \boldsymbol{w})$ is a natural measure of similarity defined on distributions. For our case of discrete distributions it is defined as follows.

$$D_{KL}(q \| p; \boldsymbol{w}) = \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}} q(\boldsymbol{x}, \boldsymbol{y}) \log \frac{q(\boldsymbol{x}, \boldsymbol{y})}{p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w})}.$$

Finding the vector $\boldsymbol{w}^* \in \mathbb{R}^F$ which minimizes $D_{KL}(q \| p; \boldsymbol{w})$ can then be seen to produce the best approximation to q.

Unfortunately, $q(\boldsymbol{x}, \boldsymbol{y})$ is not known. But because the training set is taken to be an iid sample from q, it can be used to construct an empirical approximation to $q(\boldsymbol{x}, \boldsymbol{y})$. We have

$$\begin{aligned}
& \operatorname*{argmin}_{\boldsymbol{w} \in \mathbb{R}^F} D_{KL}(q \| p; \boldsymbol{w}) \\
= \; & \operatorname*{argmin}_{w \in \mathbb{R}^F} \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} q(\boldsymbol{x}, \boldsymbol{y}) \log \frac{q(\boldsymbol{x}, \boldsymbol{y})}{p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w})} \\
= \; & \operatorname*{argmin}_{w \in \mathbb{R}^F} \left[\underbrace{\sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} q(\boldsymbol{x}, \boldsymbol{y}) \log q(\boldsymbol{x}, \boldsymbol{y})}_{\text{constant}} - \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} q(\boldsymbol{x}, \boldsymbol{y}) \log p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w}) \right] \\
= \; & \operatorname*{argmax}_{w \in \mathbb{R}^F} \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} q(\boldsymbol{x}, \boldsymbol{y}) \log p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w}) \\
\approx \; & \operatorname*{argmax}_{w \in \mathbb{R}^F} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w}) \qquad (32) \\
= \; & \operatorname*{argmax}_{w \in \mathbb{R}^F} \prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w}).
\end{aligned}$$

The last expression is the maximum likelihood estimation problem, where the true distribution $q(\boldsymbol{x}, \boldsymbol{y})$ is approximated as empirical expectation over the training samples. From the above derivation the joint *likelihood* of a parameter $\boldsymbol{w}$ can be written as

$$\ell(\boldsymbol{w}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w}).$$

Finding the most likely parameter which generated the samples is called *maximum likelihood estimation* and can be posed as optimization problem over

$\mathbb{R}^F$ by maximizing $\ell(\boldsymbol{w})$:

$$\begin{aligned}
\boldsymbol{w}^* &= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w}) \\
&= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w}) \\
&= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} \log \frac{1}{Z(\boldsymbol{w})} \exp\{-E(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w})\} \\
&= \operatorname*{argmin}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} E(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w}) - N \log Z(\boldsymbol{w}).
\end{aligned}$$

Solving for $\boldsymbol{w}^*$ is in general difficult because of the $\log Z(\boldsymbol{w})$ term: computing this partition function exactly is NP-hard[134], but approximations to the partition function exist[135].

For some special graphs such as chain graphs and trees it is possible to compute the partition function because the summation over all states can be carried out using dynamic programming algorithms[136]. The most popular application of maximum-likelihood training for Markov random fields has therefore traditionally been limited to these models, for example the Hidden Markov Models (HMM)[137].

[134] Gerhard Winkler. *Image Analysis, Random Fields, and Dynamic Monte Carlo Methods: A Mathematical Introduction*. Springer, 1995

[135] Martin J. Wainwright, Tommi Jaakkola, and Alan S. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, 2005

[136] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006

[137] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*, volume November. John Wily & Sons, Inc., New York, second edition, 2000. ISBN 0471056693

Conditional Training

For conditional random fields the training procedure is similar to the one above, but the *conditional likelihood* is used in place of the likelihood function. Given a fully observed, iid training set $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\ldots,N}$ the conditional likelihood is given as

$$\ell_c(\boldsymbol{w}) = \prod_{n=1}^{N} p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{w}) \tag{33}$$

When using a prior distribution $p(\boldsymbol{w})$ over the parameters, we have the posterior distribution $p(\boldsymbol{w}|\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\ldots,N})$ by Bayes rule and the iid assumption given as

$$\begin{aligned}
p(\boldsymbol{w}|\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\ldots,N}) &= p(\boldsymbol{w}) \frac{p(\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\ldots,N}|\boldsymbol{w})}{p(\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\ldots,N})} \\
&= p(\boldsymbol{w}) \prod_{n=1}^{N} \frac{p(\boldsymbol{x}_n, \boldsymbol{y}_n|\boldsymbol{w})}{p(\boldsymbol{x}_n, \boldsymbol{y}_n)} \\
&= p(\boldsymbol{w}) \prod_{n=1}^{N} \frac{p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{w})}{p(\boldsymbol{y}_n|\boldsymbol{x}_n)}.
\end{aligned}$$

The optimal MAP estimate of the parameter vector $\boldsymbol{w}$ given a prior $p(\boldsymbol{w})$ can therefore be inferred by maximizing $p(\boldsymbol{w}|\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\dots,N})$, obtaining

$$\begin{aligned} \boldsymbol{w}^* &:= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \left(\prod_{n=1}^{N} \frac{p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{w})}{p(\boldsymbol{y}_n|\boldsymbol{x}_n)} \right) p(\boldsymbol{w}) \\ &= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} \log p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{w}) + \log p(\boldsymbol{w}) \\ &= \operatorname*{argmin}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) - \sum_{n=1}^{N} \log Z(\boldsymbol{x}_n, \boldsymbol{w}) - \log p(\boldsymbol{w}). \end{aligned}$$

Like for the MRF training, different prior distributions $p(\boldsymbol{w})$ lead to different regularizing functions. The difficulty of computing the partition function remains, but note that different from the maximum likelihood training of the Markov random field, the partition function *does* depend on the observation $\boldsymbol{x}_n$ of each individual instance. Therefore (30) sums only over the latent states, whereas for the MRF training the summation is over all states in $\mathcal{X} \times \mathcal{Y}$.

Regularization

Regularization can be used to avoid overfitting in case there are few training instances or many given features ($F \gg N$). The use of regularization can be derived in a sound way by specifying a prior distribution over possible values of $\boldsymbol{w}$. We assume a prior distribution $p(\boldsymbol{w})$ and an iid training set $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\dots,N}$ are given and use Bayes rule to derive the posterior distribution over parameters as

$$\begin{aligned} p(W = \boldsymbol{w}|X, Y) &= \frac{p(\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\dots,N}|\boldsymbol{w})p(\boldsymbol{w})}{p(\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\dots,N})} \\ &= \frac{\prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{y}_n|\boldsymbol{w})}{\prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{y}_n)} p(\boldsymbol{w}) \\ &\propto \left(\prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{y}_n|w) \right) p(\boldsymbol{w}). \end{aligned}$$

A Bayesian statistician is interested in the full distribution $p(W|\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\dots,N})$ and its properties. We are only interested in the maximum a-posteriori estimate $\boldsymbol{w}^*$ under our prior distribution $p(W)$ and

hence we explicitly optimize for $\boldsymbol{w}^*$ as follows.

$$\begin{aligned}
\boldsymbol{w}^* &= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \left(\prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{y}_n|\boldsymbol{w}) \right) p(\boldsymbol{w}) \\
&= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{y}_n|\boldsymbol{w}) + \log p(\boldsymbol{w}) \\
&= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} \log \frac{1}{Z(\boldsymbol{w})} \exp\{-E(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w})\} + \log p(\boldsymbol{w}) \\
&= \operatorname*{argmin}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N} E(\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{w}) - N \log Z(\boldsymbol{w}) - \log p(\boldsymbol{w}).
\end{aligned}$$

We are free to choose a prior distribution at will but a common prior distribution is the multivariate Normal distribution $\mathcal{N}(0, \sigma^2 I)$ such that

$$\begin{aligned}
-\log p(\boldsymbol{w}) &= \log \prod_{i=1}^{F} \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{ -\frac{1}{2\sigma^2} w_i^2 \right\} \\
&= \frac{1}{2\sigma^2} \|\boldsymbol{w}\|_2^2 - \underbrace{F \log \frac{1}{\sigma\sqrt{2\pi}}}_{\text{constant}}
\end{aligned}$$

and hence the function $-\log p(\boldsymbol{w})$ is *strictly convex*, making $\boldsymbol{w}^*$ unique. Alternative popular priors include the *multivariate Laplace distribution* of the form

$$p(\boldsymbol{w}; \sigma) = \frac{1}{(4\sigma^2)^F} \exp\left\{ -\frac{1}{2\sigma^2} \sum_{i=1}^{F} |w_i| \right\}. \tag{34}$$

Both the multivariate Normal and the multivariate Laplacian distribution are members of the general family of the *p-generalized Normal distributions*[138]. When the prior (34) is used to regularize the maximum likehood estimation problem it induces *sparse* weight vectors. For the regularization it can be more conveniently expressed by means of one rate parameter $\lambda > 0$ with $\lambda = \frac{1}{2\sigma^2}$ such that

$$-\log p(\boldsymbol{w}; \lambda) = \lambda \sum_{i=1}^{F} |w_i| + \underbrace{\left(\frac{2}{\lambda}\right)^n}_{\text{constant}}.$$

For the regularized maximum likelihood estimation problem the difficulty of computing $Z(\boldsymbol{w})$ remains.

In the next sections we will introduce alternative methods to infer a good parameter $\boldsymbol{w}$. One popular method is based on a generalization of Support Vector Machine (SVM) learning to structured prediction tasks. The principal advantage of the method is that it does not require the computation of the partition function but only repeated solution of MAP-MRF problems.

[138] Irwin R. Goodman and Samuel Kotz. Multivariate θ-generalized normal distributions. *Journal of Multivariate Analysis*, 3(2):204–219, June 1973; and Fabian Sinz, Sebastian Gerwinn, and Matthias Bethge. Characterization of the p-generalized normal distribution. *Journal of Multivariate Analysis*, 100(5):817–820, May 2009

Alternative Training Procedures

Although the maximum (conditional) likelihood training discussed in the previous section is arguably the most popular training procedure, for many problems arising in computer vision it is intractable. The intractability arises because for general graphs computing the partition function $Z(\boldsymbol{x}, \boldsymbol{w})$ involves the summation over all possible labelings.

Because of this difficulty a number of approximations and alternative training procedures have been invented. We now discuss in detail two popular methods well-suited to parameter learning, the structured support vector machine and pseudolikelihood training. At the end of this section we additionally provide a brief survey of the literature on training procedures and recent trends in computer vision.

Training using Structured Support Vector Machines

This section discusses a training method known as *Structured Support Vector Machine*. To use structured SVM training to train CRFs has been a recent trend in computer vision[139].

[139] Matthew B. Blaschko and Christoph H. Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008; Yunpeng Li and Daniel Huttenlocher. Learning for stereo vision using the structured support vector machine. In *CVPR*, 2008; and Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning CRFs using graph cuts. In *ECCV*, 2008

TAKING A STEP BACK, what properties should any reasonable training procedure have? First, it should produce a prediction function that *generalizes* well to unseen instances. Second, it should try to produce correct predictions on the training set.

The requirement to predict correctly on a given training instance $(\boldsymbol{x}, \boldsymbol{y}^*)$ can be formalized simply as the requirement to assign the correct prediction $\boldsymbol{y}^*$ a lower energy $E(\boldsymbol{y}^*; \boldsymbol{x}, \boldsymbol{w})$ than any other prediction $\boldsymbol{y} \in \mathcal{Y}$, i.e., to satisfy

$$E(\boldsymbol{y}^*; \boldsymbol{x}, \boldsymbol{w}) \leq E(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{w}), \qquad \forall \boldsymbol{y} \in \mathcal{Y}. \tag{35}$$

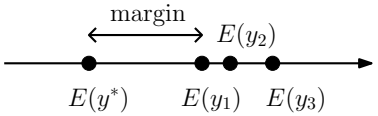


Figure 34: Desired energy configurations: the energy $E(\boldsymbol{y}^*; \boldsymbol{x}, \boldsymbol{w})$ of the true label $\boldsymbol{y}^*$ is strictly smaller than the energies of other states $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{y}_3 \in \mathcal{Y}$.

While this condition is necessary and intuitive, it is not enough: E is a linear function in $\boldsymbol{w}$ and therefore $\boldsymbol{w} = \boldsymbol{0}$ will trivially satisfy (35). What is needed is a *strictly positive margin* between the correct prediction and any other prediction. This is illustrated in Figure 34.

The constraints (35) change to

$$E(\boldsymbol{y}^*; \boldsymbol{x}, \boldsymbol{w}) + d \leq E(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{w}), \qquad \forall \boldsymbol{y} \in \mathcal{Y}, \tag{36}$$

where $d > 0$ is a constant. Each training instance $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ demands one set of constraints of the form (36).

Two issues remain. First, how should d be set in each constraint, and second, how to guarantee there exists a $\boldsymbol{w} \in \mathbb{R}^F$ which satisfies all constraints (36).

FOR SETTING THE DESIRED MARGIN d, let us consider two possible mispredictions $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$. Let us assume $\boldsymbol{y}_1$ is similar to the correct prediction $\boldsymbol{y}^*$. The notion of *similarity* depends on the task at hand. For image segmentation

it could mean that the predicted segmentation $\boldsymbol{y}_1$ is mostly correct and differs in only a few pixels from $\boldsymbol{y}^*$. Further, let $\boldsymbol{y}_2$ be quite different from $\boldsymbol{y}^*$, for example an image segmentation which differs from $\boldsymbol{y}^*$ in most pixels. If we would have the choice of which prediction is acceptable, we would choose $\boldsymbol{y}_1$ over $\boldsymbol{y}_2$. Conversely, the margin d should be larger for the energies $E(\boldsymbol{y}^*; \boldsymbol{x}, \boldsymbol{w})$ and $E(\boldsymbol{y}_2; \boldsymbol{x}, \boldsymbol{w})$ than for $E(\boldsymbol{y}^*; \boldsymbol{x}, \boldsymbol{w})$ and $E(\boldsymbol{y}_1; \boldsymbol{x}, \boldsymbol{w})$.

To incorporate this, we assume there is a natural semi-metric $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ defined which satisfies for all $(\boldsymbol{y}, \boldsymbol{y}') \in \mathcal{Y} \times \mathcal{Y}$ the following properties; symmetry $\Delta(\boldsymbol{y}, \boldsymbol{y}') = \Delta(\boldsymbol{y}', \boldsymbol{y})$, non-negativity $\Delta(\boldsymbol{y}, \boldsymbol{y}') \geq 0$, and the identity of indiscernibles $\Delta(\boldsymbol{y}, \boldsymbol{y}') = 0 \Leftrightarrow \boldsymbol{y} = \boldsymbol{y}'$. In our example above we would have $\Delta(\boldsymbol{y}^*, \boldsymbol{y}_2) > \Delta(\boldsymbol{y}^*, \boldsymbol{y}_1) > 0$. For each constraint of the form (36) we set $d = \Delta(\boldsymbol{y}^*, \boldsymbol{y})$ and thus obtain for each training sample $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ constraints of the form

$$E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) + \Delta(\boldsymbol{y}_n, \boldsymbol{y}) \leq E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}), \qquad \forall \boldsymbol{y} \in \mathcal{Y}. \tag{37}$$

FOR THE EXISTENCE of $\boldsymbol{w} \in \mathbb{R}^F$, there is in general no guarantee that the set described by (37) is not empty. To ensure feasibility, we introduce for each system (37) a slack variable $\xi_n \geq 0$. For ξ_n large enough there will always exists a feasible $\boldsymbol{w}$ in the new constraint system

$$\begin{aligned} &E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) + \Delta(\boldsymbol{y}_n, \boldsymbol{y}) \leq E(\boldsymbol{y}; \boldsymbol{x}_n, \boldsymbol{w}) + \xi_n, \qquad \forall \boldsymbol{y} \in \mathcal{Y}, \\ &\xi_n \geq 0. \end{aligned}$$

The variables ξ_n are penalized in the objective such that $\boldsymbol{w}$ is sought to violate (37) the least.

The use of slack variables avoids the extreme of infeasibility. Consider the other extreme: there is a set of vectors $\mathcal{W} \subset \mathbb{R}^F$ which *all* satisfy (37). In this case, regularization by means of adding a strictly convex function in $\boldsymbol{w}$ is used to choose a unique element from $\mathcal{W}$. For linear models, the most popular regularization function is the squared Euclidean norm $\|\boldsymbol{w}\|_2^2$.

PUTTING THE ABOVE POINTS TOGETHER, the problem of finding $\boldsymbol{w}$ can be posed as mathematical optimization problem. The problem is known as *structured support vector machine* and was formulated by Tsochantaridis et al.[140]. Given iid training data $\{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1,\ldots,N}$ with $(\boldsymbol{x}_n, \boldsymbol{y}_n) \in \mathcal{X} \times \mathcal{Y}$, we solve

$$\min_{\boldsymbol{w}, \boldsymbol{\xi}} \quad \|\boldsymbol{w}\|^2 + \frac{C}{\ell} \sum_{n=1}^{\ell} \xi_n \tag{38}$$

$$\text{sb.t.} \quad E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) + \Delta(\boldsymbol{y}_n, \boldsymbol{y}) \leq E(\boldsymbol{y}; \boldsymbol{x}_n, \boldsymbol{w}) + \xi_n, \ \forall n, \ \forall \boldsymbol{y} \in \mathcal{Y}, \tag{39}$$

$$\xi_n \geq 0, \quad n = 1, \ldots, N.$$

BECAUSE THE ENERGIES IN (39) ARE LINEAR FUNCTIONS in $\boldsymbol{w}$ and additionally the term $\Delta(\boldsymbol{y}_n, \boldsymbol{y})$ is constant, (39) is a set of linear inequalities. The objective function (38) contains linear and quadratic terms.

[140] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, September 2005

The problem is therefore a quadratic programming problem. The constant $C > 0$ specifies the tradeoff between the regularization term and the loss term. High values of C will produce a low training error but possibly generalize less, whereas small values of C typically lead to good generalization performance but lower training set performance.

The term "program" is a historic artifact: in the 1950s when mathematical optimization was developed, the term *programming* was used equivalent to the term *planning*. The activity of mathematical programming was to formulate and solve planning problems mathematically.

The set (39) of linear inequalities describes an intersection of halfspaces. If the set of linear inequalities is finite, the resulting intersection is a *polyhedron*. In our case both N and $|\mathcal{Y}|$ are finite in (39), so the constraints indeed describe a polyhedron. Despite being finite, $|\mathcal{Y}|$ might be very large, usually exponentially large in the length of the input representation. For example, for image segmentation with k pixels and binary states we would have $|\mathcal{Y}| = 2^k$. Therefore, (39) cannot be explicitly optimized over.

OPTIMIZING IMPLICITLY OVER A LARGE SET OF INEQUALITIES such as (39) is a classic technique in numerical optimization known as *delayed constraint generation*.

To understand constraint generation, we first make the following observation: assume we could optimize (38) over the entire set (39). Then, the optimal solution $(\boldsymbol{w}^*, \boldsymbol{\xi}^*)$ is binding[141] at only a subset of (39) and all constraints in in (39) which are not binding could be removed without changing the solution. Moreover, for any optimal solution a subset of $F + N$ binding linear inequalities from (39) suffices. All additional binding inequalities are *degenerate*, that is, they are linearly dependent on the set of $F + N$ constraints. See Figure 35.

[141] For a point x an inequality $a^\top x \leq c$ is said to be *binding* if $a^\top x = c$.

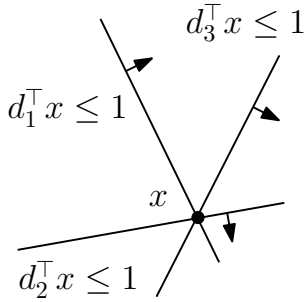


Figure 35: Degeneracy: at $x \in \mathbb{R}^2$ any 2-subset of the 3 inequalities suffices to define x.

Instead of dropping constraints *after* we obtain the optimal solution, in delayed constraint generation we *start* with no constraints and solve (38) to obtain a candidate solution. We then verify whether the candidate solution violates any of the inequalities (39). If it does, the violated inequality is explicitly generated and added to the problem and the problem is resolved. If the candidate solution turns out not to violate any inequality, then by the above reasoning the candidate solution is also the optimal solution. The incrementally growing problem is the *restricted master problem*, the problem of finding violated inequalities is the *separation problem*.

The overall procedure is summarized in Algorithm STRUCTUREDSVM. The algorithm iterates between solving the restricted master problem and generating violated constraints. The constraints found are used to tighten the master problem which is then resolved. If no violated constraints can be found, the procedure terminates. In each iteration, the maximum violation magnitude can be used to as convergence criterion and usually in practice one stops training once it is small enough. Because in our case $|\mathcal{Y}|$ is finite, the algorithm is finitely convergent, a fact proved in Tsochantaridis et al.[142].

[142] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, September 2005

Algorithm 5 Structured SVM Training

$\boldsymbol{w} = \textsc{StructuredSVM}(X, Y, C)$
Input:
  $\{(x_n, y_n)\}_{n=1,\ldots,N}$ training set, $(x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$
  $C > 0$ regularization parameter
  $\epsilon \geq 0$ convergence tolerance
Output:
  $\boldsymbol{w} \in \mathbb{R}^F$ learned weight vector
Algorithm:
$D_{w,\xi} \leftarrow \mathbb{R}^F \times \mathbb{R}^N_+$ {Initially: no constraints}
loop
  $(\boldsymbol{w}^*, \boldsymbol{\xi}^*) \leftarrow \begin{cases} \underset{w,\xi}{\text{argmin}} & \|\boldsymbol{w}\|_2^2 + C\sum_{n=1}^N \xi_n \\ \text{sb.t.} & (\boldsymbol{w}, \boldsymbol{\xi}) \in D_{w,\xi} \end{cases}$ {Solve master}
  maxviol $\leftarrow -\infty$
  for $n = 1, \ldots, N$ **do**
    $(\text{viol}, \boldsymbol{y}_v) \leftarrow (\max, \underset{\boldsymbol{y}\in\mathcal{Y}}{\text{argmax}}) \Big[E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}^*) - E(\boldsymbol{y}; \boldsymbol{x}_n, \boldsymbol{w}^*)$
      $+\Delta(\boldsymbol{y}_n, \boldsymbol{y}) - \xi_n^*\Big]$ {Solve separation problem}
    if viol > 0 **then**
      $D_{w,\xi} \leftarrow D_{w,\xi} \cap \{\boldsymbol{w}, \boldsymbol{\xi} : E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) + \Delta(\boldsymbol{y}_n, \boldsymbol{y}_v) \leq$
        $E(\boldsymbol{y}_v; \boldsymbol{x}_n, \boldsymbol{w}) + \xi_n\}$
    end if
    maxviol $\leftarrow \max\{\text{viol}, \text{maxviol}\}$
  end for
  if maxviol $> \epsilon$ **then**
    break
  end if
end loop

Pseudolikelihood Training

One simple approach to parameter learning in Markov networks from fully-observed training data is the *pseudolikelihood*, originally proposed and analyzed by Besag[143].

The pseudolikelihood is based on the following idea: the joint probability of the dependent variables, $p(Y|\boldsymbol{x}, \boldsymbol{w})$ can be approximated as a product of individual conditional probabilities over each dependent variable Y_i, where the conditioning is on all the neighbors of the variable. This assumption is

[143] Julian Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3): 179–195, 1975; and Julian Besag. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrica*, (64):616–618, 1977

written as

$$p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}) \approx p'(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}) \tag{40}$$

$$:= \prod_{i\in V} p(y_i|\boldsymbol{y}_{V\setminus\{i\}},\boldsymbol{x},\boldsymbol{w})$$

$$= \prod_{i\in V} p(y_i|\boldsymbol{y}_{\mathcal{N}(Y_i)},\boldsymbol{x},\boldsymbol{w}), \tag{41}$$

where $\boldsymbol{y}_{V\setminus\{i\}}$ is the set of all dependent random variables excluding i. Because of the Markov properties it is enough to condition on the neighbors $\mathcal{N}(y_i)$ of y_i, the so called *Markov blanket* of y_i.

The pseudolikelihood $\ell_p : \mathbb{R}^F \to \mathbb{R}$ over the parameter space is defined as follows. Given fully observed iid training data $\{(\boldsymbol{x}_n,\boldsymbol{y}_n)\}_{n=1,\ldots,N}$, with $(\boldsymbol{x}_n,\boldsymbol{y}_n) \in \mathcal{X}\times\mathcal{Y}$, the pseudolikelihood is the product of conditional probabilities of the form (40), i.e.,

$$\ell_p(\boldsymbol{w}) = \prod_{n=1}^{N} p'(\boldsymbol{y}_n|\boldsymbol{x}_n,\boldsymbol{w}) \tag{42}$$

$$= \prod_{n=1}^{N}\prod_{i\in V_n} p(y_{n,i}|\boldsymbol{y}_{n,\mathcal{N}(Y_{n,i})},\boldsymbol{x},\boldsymbol{w}),$$

where we denoted by $Y_{n,i}$ the i'th random variable in the network corresponding to the n'th training instance.

The effect of this approximation can be understood in terms of the factor graph. Take the central variable in Figure 33 and its Markov blanket consisting of its neighbors $\mathcal{N}(Y_i) = \{X_i, Y_{j_1}, Y_{j_2}, Y_{j_3}, Y_{j_4}\}$. The part of the network corresponding to only this subset of variables is shown in Figure 36.

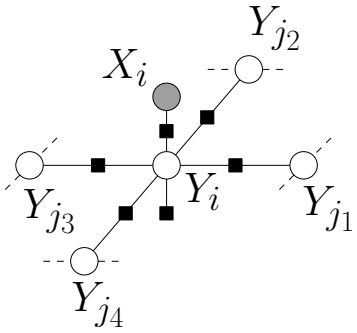


Figure 36: Markov blanket of the center variable Y_i. The shown part of the network includes all factors depending on Y_i.

Conditioned on this set of neighbor variables, Y_i is independent from all other random variables. Pseudolikelihood assumes mutual independence of all the conditional distributions, one at each variable. While this assumption is not valid in general it might provide an acceptable approximation.

Graphically, the assumption corresponds to using the *observed* training values for y_{j_1}, y_{j_2}, y_{j_3}, y_{j_4} and x_i for the computation of of the pairwise factors depending on Y_i. By instantiating the training values, the factor graph is transformed such that only unary factors remain, as shown in Figure 37.

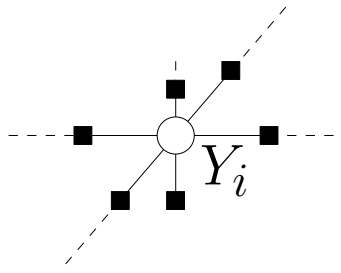


Figure 37: Remaining part of the factor graph after the pseudolikelihood assumption is made. The other variables the factors depend on are instantiated using training data so that the factor expressions become unary functions of y_i.

After this transformation the conditional distribution involves only a partial partition function $Z_i(x_i, y_{j_1}, y_{j_2}, y_{j_3}, y_{j_4})$ summing over only the states of Y_i, i.e., over $y_i \in \mathcal{Y}_i$. This is the key insight that makes pseudolikelihood training tractable and extremely efficient.

In general, the partial partition function at variable Y_i is given as

$$Z_i(\boldsymbol{x}_{\mathcal{N}(Y_{n,i})},\boldsymbol{y}_{\mathcal{N}(Y_{n,i})},\boldsymbol{w}) = \sum_{y_i\in\mathcal{Y}_i} \exp\{-E(y_i,\boldsymbol{x}_{\mathcal{N}(Y_{n,i})},\boldsymbol{y}_{\mathcal{N}(Y_{n,i})};\boldsymbol{w})\},$$

for a combined energy function depending only on the set of variables which are neighbors to y_i. In our example, this would be the sum of energies of the unary factors shown in Figure 37.

Finding the parameter $\boldsymbol{w}^*$ which optimizes the pseudolikelihood for a given training set then becomes the following optimization problem:

$$\begin{aligned}
\boldsymbol{w}^* &= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \quad \ell_p(\boldsymbol{w}) \qquad (43)\\
&= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \prod_{n=1}^{N}\prod_{i\in V_n} p(y_{n,i}|\boldsymbol{x}_{n,\mathcal{N}(Y_{n,i})}, \boldsymbol{y}_{n,\mathcal{N}(Y_{n,i})}, \boldsymbol{w})\\
&= \operatorname*{argmax}_{\boldsymbol{w}\in\mathbb{R}^F} \sum_{n=1}^{N}\sum_{i\in V_n} \Big[-E(y_{n,i}, \boldsymbol{x}_{n,\mathcal{N}(Y_{n,i})}, \boldsymbol{y}_{n,\mathcal{N}(Y_{n,i})}; \boldsymbol{w})\\
&\qquad\qquad + \sum_{y_{n,i}\in\mathcal{Y}_{n,i}} E(y_{n,i}, \boldsymbol{x}_{n,\mathcal{N}(Y_{n,i})}, \boldsymbol{y}_{n,\mathcal{N}(Y_{n,i})}; \boldsymbol{w})\Big],
\end{aligned}$$

which is tractable because only a small number of terms appear. The maximizer $\boldsymbol{w}^*$ is determined numerically by solving (43) using a continuous unconstrained optimization software such as nonlinear conjugate gradient or limited memory quasi-Newton methods such as L-BFGS. Also note that (43) lends itself ideally to a parallel and stochastic implementation as it decouples over samples and sites.

Estimating $\boldsymbol{w}^*$ by maximizing the pseudolikelihood (43) is known to converge to the true parameter in the limit of infinite data, if the true distribution is contained in the model class. This is the case if all conditional distributions in (41) are matched to the data exactly. This consistency result was proven by Gidas[144], Comets[145] and generalized to Boltzmann machines by Hyvärinen[146]. However, the assumption that the true distribution is contained in the model class is usually not satisfied, and training data is always finite and usually rare.

Nevertheless, pseudolikelihood estimation has been successfully applied and empirical studies have confirmed its efficiency when the training data is fully observed. See for example Parise and Welling[147], and also Sutton and McCallum[148]. For an application of pseudolikelihood training on images, see Vishwanathan et al.[149] and the monograph by Winkler[150].

Other Training Procedures

Because the parameter learning problem in large Markov networks is both hard and important in practice, a large number of alternative methods for parameter learning have been proposed.

For tractable models such as trees and chains, the Perceptron algorithm can be adapted to yield online algorithms which iteratively make passes through the training set, correcting the weight vector after each individual instance, see Collins[151]. The Perceptron algorithm is a member of a larger class of stochastic gradient descent algorithms, used by Vishwanathan et al.[152].

For general undirected graphs, the available methods can be divided into four groups.

[144] Basilis Gidas. Consistency of maximum likelihood and pseudo-likelihood estimators for Gibbs distributions. In *Stochastic Differential Systems, Stochastic Control Theory and Applications*. Springer, 1988

[145] Francis Comets. On consistency of a class of estimators for exponential families of Markov random fields on the lattice. *The Annals of Statistics*, 20(1):455–468, 1992

[146] Aapo Hyvärinen. Consistency of pseudolikelihood estimation of fully visible boltzmann machines. *Neural Computation*, 18(10):2283–2292, 2006

[147] Sridevi Parise and Max Welling. Learning in Markov random fields: An empirical study. In *Joint Statistical Meeting JSM2005*, 2005

[148] Charles A. Sutton and Andrew McCallum. Piecewise pseudolikelihood for efficient training of conditional random fields. In *ICML*, 2007

[149] SVN Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *ICML*, 2006

[150] Gerhard Winkler. *Image Analysis, Random Fields, and Dynamic Monte Carlo Methods: A Mathematical Introduction*. Springer, 1995

[151] Michael Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms, July 2002

[152] SVN Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *ICML*, 2006

FIRST, approximate inference methods based on belief propagation. *Belief propagation*, proposed by Pearl[153], is an exact inference method for directed and undirected graphical models that do not contain cycles. In most recent works, the algorithm is now called *sum-product* and *max-sum* algorithm, for computing marginals and and the MAP state, respectively.

Belief propagation is a dynamic programming algorithm able to compute exact marginal probabilities, the maximum a posteriori probability and the partition function. It has been generalized to be able to work for general graphs by first forming an augmented tree-structured graph[154]. Unfortunately, the augmented graph is of exponential size when the graph has a high *tree-width*, that is, it contains sufficiently many cycles. This makes exact inference intractable.

For this reason, Pearl suggested to use the belief propagation updates as an approximation, even when the graph contains cycles and the updates are therefore not exact. This is possible because the updates are still well defined, although convergence cannot be guaranteed. Subsequently, Frey and McKay[155], followed by others[156], showed that this *loopy belief propagation* scheme works surprisingly well in practice.

Since then many authors have tried to explain this efficiency. Yedidia et al.[157] show that belief propagation is a special case of a larger class of approximations discussed as "*free energies*" of systems in statistical physics. This view has subsequently lead to a number of improved algorithms. Despite these improvements, loopy belief propagation remains the most popular used inference algorithm due to its simplicity and speed.

SECOND, methods in which the partition function is bounded. *Variational methods*[158] approximate the true distribution by a family of simpler distributions and iteratively search within this simplified family for the best approximation of the true distribution. Naturally, these methods provide a *lower bound on the partition function* and the parameter learning problem becomes a saddle-point finding problem. For an application in computer vision, see Verbeek and Triggs[159].

In contrast, methods bounding the partition function from above[160] search over an enlarged outer approximation of the set of model distributions. Learning parameters then becomes a single maximization problem.

THIRD, approximation of the model class by a tractable model and exact learning on the tractable approximation. This is the main idea behind the *piecewise training* method proposed by Sutton and McCallum[161]. A graphical model is suitably decomposed into pieces, each of which is trained individually. A piece may be as small as a single pairwise potential.

This strategy has recently been successfully used to train large computer vision conditional random fields, see Shotton et al.[162]. Sutton and McCallum[163] further combined the idea of piecewise training with pseudolikelihood

[153] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Mateo, California, 1988. ISBN 0934613737

[154] Steffen L. Lauritzen and David J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, B 50(2): 157–224, 1988

[155] Brendan J. Frey and David J. C. MacKay. A revolution: Belief propagation in graphs with cycles. In *NIPS*, 1997

[156] Kevin Patrick Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, pages 467–475, July 1999

[157] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *NIPS*, pages 689–695. MIT Press, 2000; and Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Understanding belief propagation and its generalizations. Technical report, Mitsubishi Electric Research Laboratories, 2001

[158] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006; and David MacKay. *Information Theory, Inference, and Learning Algorithms*. September 2003. URL `http://www.inference.phy.cam.ac.uk/mackay/itila/book.html`

[159] Jakob J. Verbeek and Bill Triggs. Scene segmentation with CRFs learned from partially labeled images. In *NIPS*. MIT Press, 2007

[160] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1 (1-2):1–305, 2008

[161] Charles A. Sutton and Andrew McCallum. Piecewise training for undirected models. In *UAI*, pages 568–575, 2005

[162] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision*, 81(1), January 2007

[163] Charles A. Sutton and Andrew McCallum. Piecewise pseudolikelihood for efficient training of conditional random fields. In *ICML*, 2007

training.

Recently, more radical approximations have been proposed by Domke[164] and by Pletscher et al.[165]. Domke proposes to build a sequence of tractable conditional random field models, each conditioning on the previous layer. Inference in this model is very efficient but during training the layers have to be built greedily and possibly suboptimal. In each iteration, a layer is constructed to optimize the "*maximum posterior marginal*" accuracy that decomposes linearly over the nodes of the model. Pletscher et al. also approximate the intractable model but instead of using a sequence of models they use a mixture of randomly sampled spanning trees of the graphical model. They show that these mixtures perform well empirically but do not prove any theoretical properties such as consistency of the estimator or convexity of the training objective.

[164] Justin Domke. Crossover random fields. Technical report, University of Maryland, 2009

[165] Patrick Pletscher, Cheng Soon Ong, and Joachim M. Buhmann. Spanning tree approximations for conditional random fields. In *AISTATS*, 2009

FOURTH, sampling based methods which evaluate expectations of a function weighted by the current model distribution. The sampling approximation can be used to approximately evaluate both the partition function as well as its derivative. For a general introduction to sampling based methods and state-of-the-art Markov Chain Monte Carlo (MCMC) methods see Neal[166] and Bishop[167]. By evaluating the approximate gradient of the partition function one can obtain the maximum likelihood estimate of the model parameters using a gradient descent procedure. Although beautiful in theory, sampling can be slow in practice and tuning a sampling procedure to perform well on a task can be difficult.

Recently, Hinton proposed *contrastive divergence*[168] to overcome some of the disadvantages of naive sampling. Although too early to draw definite conclusions, it has been used successfully in computer vision conditional random fields by He et al.[169].

[166] Radford. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, September 1993

[167] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006

[168] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8): 1771–1800, 2002; and Miguel Á. Carreira-Perpiñán and Geoffrey E. Hinton. On contrastive divergence learning. In *AISTATS*, 2005

[169] Xuming He, Richard S. Zemel, and Miguel Á. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *CVPR*, pages 695–702, 2004

TAKING A STEP BACK, LeCun[170] proposes "energy-based models" as a unified framework for prediction, ranking, detection and density estimation. His general model encompasses neural networks, random field models and many other popular machine learning algorithms. This unified perspective is helpful in order to categorize and analyze classes of algorithms and their shared properties and will certainly influence future research in the direction of structured learning.

Having discussed the parameter learning problem we now focus on the problem to be solved at test time. There we want to solve the MAP-MRF problem, that is, to infer the most likely assignment of the latent unobserved states, given the observations.

[170] Yann LeCun, Sumit Chopra, Raia Hadsell, Marc A. Ranzato, and Fu Jie Huang. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006

Maximum a Posteriori Problem

In the previous section we have discussed the problem of finding a good model from a larger family when we are given fully observed training data. In this section we discuss the application of the model found to partially observed test samples: given an image $\boldsymbol{x}$, we want to infer a likely state $\boldsymbol{y}$. For example, when the task is image segmentation, $\boldsymbol{y}$ would be a per-pixel segmentation mask.

We discuss two methods particularly popular in the computer vision community: graphcut-based methods and linear programming relaxations. Whereas graphcut-based methods are popular for their outstanding efficiency, the linear programming relaxation is particularly amenable to theoretical analysis.

Graphcut MAP-MRF

The most popular method in computer vision to minimize the energy of the MAP-MRF problem is the *graphcut algorithm* of Boykov et al.[171].

[171] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(11):1222–1239, 2001

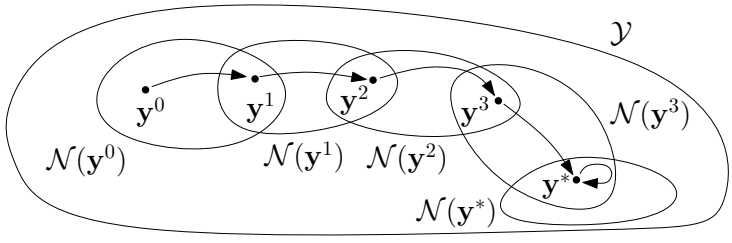


Figure 38: Illustration of the large scale neighborhood search used in graph-cut based optimization: each solution $\boldsymbol{y}$ has a neighborhood $\mathcal{N}(\boldsymbol{y})$ associated to it. The local search iteration improves from $\boldsymbol{y}^t$ to $\boldsymbol{y}^{t+1}$ by searching for the optimal solution within $\mathcal{N}(\boldsymbol{y}^t)$. When $\boldsymbol{y}^t$ and $\boldsymbol{y}^{t+1}$ coincide then $\boldsymbol{y}^* = \boldsymbol{y}^t$ is returned as optimal within its neighborhood.

The algorithm is illustrated in Figure 38. It is a *local search* algorithm, iteratively improving a candidate solution until no further improvement is possible. Two properties make the algorithm efficient. First, at each candidate solution $\boldsymbol{y}^t$, the neighborhood $\mathcal{N}(\boldsymbol{y}^t)$ of solutions considered is of exponential size. Second, the neighborhood is constructed in such a way that the solution $\boldsymbol{y}^{t+1} \in \mathcal{N}(\boldsymbol{y}^t)$ which decreases the objective function the most can be found efficiently by solving a minimum cut problem on an auxiliary graph.

This construction, "exponential size neighborhood" + "efficient minimization within the neighborhood" is a recent theme in combinatorial optimization, known as *very large scale neighborhood* search (VLSN), see Ahuja et al.[172]. The difficulty is in finding a suitable definition of the neighborhood $\mathcal{N} : \mathcal{Y} \to 2^{\mathcal{Y}}$, in which the neighborhood is both large and has a structure which can be efficiently optimized over. Empirically, the VLSN algorithms all have the desirable property that after only a few improvement steps a near-optimal solution has been constructed.

The general graphcut based energy minimization algorithm is shown in Algorithm GRAPHCUTMAPMRF.

[172] Ravindra K. Ahuja, Özlem Ergun, James B. Orlin, and Abraham P. Punnen. A survey of very large-scale neighborhood search techniques. In Endre Boros and Peter L. Hammer, editors, *Proceedings of the 1999 Workshop on Discrete Optimization (DO-99)*, volume 123, 1-3 of *Discrete Applied Mathematics*, pages 75–102, Amsterdam, July 25–30 2002. Elsevier Science B.V

Algorithm 6 Graphcut MAP-MRF

$\boldsymbol{y}^* = \textsc{GraphCutMAPMRF}(\boldsymbol{y}^0)$
Input:
  $\boldsymbol{y}^0 \in \mathcal{Y}$ initial solution
Output:
  $\boldsymbol{y}^* \in \mathcal{Y}$ optimal within $\mathcal{N}(\boldsymbol{y}^*)$
Algorithm:
$t \leftarrow 0$
for $t = 0, 1, \ldots$ **do**
  $\boldsymbol{y}^{t+1} \leftarrow \underset{\boldsymbol{y} \in \mathcal{N}(\boldsymbol{y}^t)}{\operatorname{argmin}} E(\boldsymbol{y})$ {Minimize within neighborhood}
  if $\boldsymbol{y}^{t+1} = \boldsymbol{y}^t$ **then**
    break {Local optima w.r.t. $\mathcal{N}(\boldsymbol{y}^t)$}
  end if
end for
$\boldsymbol{y}^* \leftarrow \boldsymbol{y}^t$

We now discuss the most important ingredient: the definition of $\mathcal{N}$. Boykov defines two parametrized neighborhoods, namely the "α-expansion" neighborhood $\mathcal{N}_\alpha : \mathcal{Y} \times \mathbb{N} \to 2^{\mathcal{Y}}$ and the "α-β-swap" neighborhood $\mathcal{N}_{\alpha,\beta} : \mathcal{Y} \times \mathbb{N} \times \mathbb{N} \to 2^{\mathcal{Y}}$. We will discuss both neighborhoods separately, starting with the simpler α-β-swap. First, let us define some notation.

Let $\mathcal{Y} = \mathcal{Y}_1 \times \mathcal{Y}_2 \times \cdots \times \mathcal{Y}_{|V|}$ be the set of all feasible labelings, where $\mathcal{Y}_i$ is the set of all possible states at node $i \in V$. Let the energy be defined as sum of unary and pairwise energies as follows.

$$
\begin{aligned}
&E : \mathcal{Y} \to \mathbb{R} \\
&E(\boldsymbol{y}) = \sum_{i \in V} E_i^{(1)}(y_i) + \sum_{(i,j) \in E} E_{i,j}^{(2)}(y_i, y_j).
\end{aligned}
$$

For many labeling tasks the pairwise energy function is the same for all edges, but we do not require this. What we do require for both the α-expansion and the α-β-swap neighborhoods is that the pairwise energy terms are a *semi-metric*, satisfying for all $(i,j) \in E$, $(y_i, y_j) \in \mathcal{Y}_i \times \mathcal{Y}_j$ the conditions

$$E_{i,j}^{(2)}(y_i, y_j) = 0 \Leftrightarrow y_i = y_j, \tag{44}$$

$$E_{i,j}^{(2)}(y_i, y_j) = E_{i,j}^{(2)}(y_j, y_i) \geq 0. \tag{45}$$

The first condition (44) is the *identity of indiscernibles*, the second condition (45) is *symmetry*. Moreover, the α-expansion further requires the pairwise energies to be a true metric, i.e. to satisfy (44), (45) and for all $(i,j) \in E$, for all $(y_i, y_j) \in \mathcal{Y}_i \times \mathcal{Y}_j$, for all $y_k \in \mathcal{Y}_i \cap \mathcal{Y}_j$ that

$$E_{i,j}^{(2)}(y_i, y_j) \leq E_{i,j}^{(2)}(y_i, y_k) + E_{i,j}^{(2)}(y_k, y_j), \tag{46}$$

which is the well known *triangle inequality*. We now consider the definition of the neighborhood.

THE α-β-SWAP NEIGHBORHOOD is defined as follows.

$$\begin{aligned}&\mathcal{N}_{\alpha,\beta} : \mathcal{Y} \times \mathbb{N} \times \mathbb{N} \to \mathcal{Y}^* \\ &\mathcal{N}_{\alpha,\beta}(\boldsymbol{y}, \alpha, \beta) := \{\boldsymbol{z} \in \mathcal{Y} : z_i = y_i \text{ if } y_i \notin \{\alpha, \beta\}\}.\end{aligned} \tag{47}$$

Therefore the neighborhood $\mathcal{N}_{\alpha,\beta}(\boldsymbol{y}, \alpha, \beta)$ contains the solution $\boldsymbol{y}$ itself as well as all variants in which the nodes labeled α or β are free to change their label to either β or α, respectively. Finding the minimizer becomes a binary labeling problem because the only two states of interest are α and β. We can decompose the following minimization problem.

$$\begin{aligned}\boldsymbol{y}^{t+1} &= \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_{\alpha,\beta}(\boldsymbol{y}^t, \alpha, \beta)} E(\boldsymbol{y}) \\ &= \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_{\alpha,\beta}(\boldsymbol{y}^t, \alpha, \beta)} \sum_{i \in V} E_i^{(1)}(y_i) + \sum_{(i,j) \in E} E_{i,j}^{(2)}(y_i, y_j) \\ &= \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_{\alpha,\beta}(\boldsymbol{y}^t, \alpha, \beta)} \Big[\underbrace{\sum_{\substack{i \in V, \\ y_i^t \notin \{\alpha,\beta\}}} E_i^{(1)}(y_i^t)}_{\text{constant}} + \underbrace{\sum_{\substack{i \in V, \\ y_i^t \in \{\alpha,\beta\}}} E_i^{(1)}(y_i)}_{\text{unary}} \\ &\quad + \underbrace{\sum_{\substack{(i,j) \in E, \\ y_i^t \notin \{\alpha,\beta\}, y_j^t \notin \{\alpha,\beta\}}} E_{i,j}^{(2)}(y_i^t, y_j^t)}_{\text{constant}} + \underbrace{\sum_{\substack{(i,j) \in E, \\ y_i^t \in \{\alpha,\beta\}, y_j^t \notin \{\alpha,\beta\}}} E_{i,j}^{(2)}(y_i, y_j^t)}_{\text{unary}} \\ &\quad + \underbrace{\sum_{\substack{(i,j) \in E, \\ y_i^t \notin \{\alpha,\beta\}, y_j^t \in \{\alpha,\beta\}}} E_{i,j}^{(2)}(y_i^t, y_j)}_{\text{unary}} + \underbrace{\sum_{\substack{(i,j) \in E, \\ y_i^t \in \{\alpha,\beta\}, y_j^t \in \{\alpha,\beta\}}} E_{i,j}^{(2)}(y_i, y_j)}_{\text{pairwise}} \Big].\end{aligned} \tag{48}$$

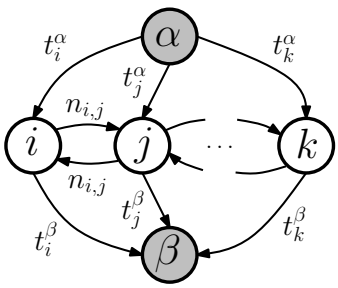


Figure 39: Directed edge-weighted auxiliary graph construction. The linear min-cut in this graph corresponds to the optimal energy configuration in $\mathcal{N}_{\alpha,\beta}(\boldsymbol{y}, \alpha, \beta)$.

[173] Dimitri P. Bertsekas. *Network Optimization*. 1998

When dropping the constant terms and combining the unary terms, problem (48) is simplified and can be solved by solving a network flow problem[173] on a specially constructed auxiliary graph, with structure as shown in Figure 39.

The directed graph $G' = (V', E')$ with non-negative edge weights t_i^α, t_i^β and $n_{i,j}$ is constructed as follows.

$$\begin{aligned}V' &= \{\alpha, \beta\} \cup \{i \in V : y_i \in \{\alpha, \beta\}\}, \\ E' &= \{(\alpha, i, t_i^\alpha) : \forall i \in V : y_i \in \{\alpha, \beta\}\} \cup \\ &\qquad \{(i, \beta, t_i^\beta) : \forall i \in V : y_i \in \{\alpha, \beta\}\} \cup \\ &\qquad \{(i, j, n_{i,j}) : \forall (i,j), (j,i) \in E : y_i, y_j \in \{\alpha, \beta\}\}.\end{aligned}$$

The edge weights are calculated as follows.

$$n_{i,j} = E^{(2)}_{i,j}(\alpha,\beta), \tag{49}$$

$$t^{\alpha}_i = E^{(1)}_i(\alpha) + \sum_{\substack{(i,j)\in E,\\ y_j\notin\{\alpha,\beta\}}} E^{(2)}_{i,j}(\alpha,y_j), \tag{50}$$

$$t^{\beta}_i = E^{(1)}_i(\beta) + \sum_{\substack{(i,j)\in E,\\ y_j\notin\{\alpha,\beta\}}} E^{(2)}_{i,j}(\beta,y_j). \tag{51}$$

Finding a directed minimum α-β-cut, that is, a cut which separates α and β in the graph G', solves (48). To see how this is possible, consider the cut shown in Figure 40. The value $f(\mathcal{C})$ of a cut $\mathcal{C}$ is the sum of the directed edge weights it cuts. For the example graph this would be

$$\begin{aligned} f(\mathcal{C}) &= t^{\alpha}_i + n_{i,j} + t^{\beta}_j + t^{\beta}_k \\ &= E^{(1)}_i(\alpha) + \sum_{\substack{(i,s)\in E,\\ y^t_s\notin\{\alpha,\beta\}}} E^{(2)}_{i,s}(\alpha,y^t_s) + E^{(2)}_{i,j}(\alpha,\beta) \\ &\quad + E^{(1)}_j(\beta) + \sum_{\substack{(i,s)\in E,\\ y^t_s\notin\{\alpha,\beta\}}} E^{(2)}_{i,s}(\beta,y^t_s) \\ &\quad + E^{(1)}_k(\beta) + \sum_{\substack{(k,s)\in E,\\ y^t_s\notin\{\alpha,\beta\}}} E^{(2)}_{k,s}(\beta,y^t_s), \end{aligned}$$

which corresponds exactly to (48) for $y_i = \alpha, y_j = \beta$ and $y_k = \beta$. This holds in general and Boykov[174] showed that the optimal labeling can be constructed from the α-β-mincut $\mathcal{C}$ as

$$y_i = \begin{cases} \alpha & \text{if } (\alpha,i)\in\mathcal{C} \\ \beta & \text{if } (i,\beta)\in\mathcal{C} \end{cases}.$$

Because exactly one of the edges must be cut for $\mathcal{C}$ to be an α-β-cut, the min-cut exactly minimizes (48).

Solving the min-cut problem on the auxiliary graph G' can be done efficiently by using max-flow algorithms. For graphs such as the one shown in G' where all nodes are connected to the source- and sink-node, specialized max-flow algorithms with superior *empirical* performance have been developed, see Boykov and Kolmogorov[175]. The best known algorithms for linear max-flow problems have a computational complexity of $O(|V|^3)$ and $O(|V||E|\log(|V|))$, see Bertsekas[176].

The α-β-swap neighborhood depends on two label parameters α and β. Each combination of α and β induces a different neighborhood. Thus, in Algorithm GraphCutMAPMRF, all pairwise combinations in $\mathcal{Y}_\ell = \bigcup_{i\in V}\mathcal{Y}_i$ are searched, i.e., in each loop iteration all neighborhoods $\mathcal{N}_{\alpha,\beta}(\boldsymbol{y}^{t,k},\alpha_k,\beta_k)$ are evaluated in some order $k = 0,1,\ldots,K$, where $(\alpha_k,\beta_k)\in\{(\alpha,\beta)\in\mathcal{Y}_\ell\times\mathcal{Y}_\ell :$

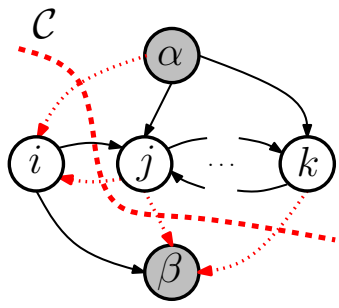


Figure 40: A minimum α-β-cut $\mathcal{C}$ and its directed edge cutset (shown dotted).

[174] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(11):1222–1239, 2001

[175] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26 (9):1124–1137, 2004

[176] Dimitri P. Bertsekas. *Network Optimization*. 1998

$\alpha < \beta\}$ and $\boldsymbol{y}^{t,0} \leftarrow \boldsymbol{y}^t$ and

$$\boldsymbol{y}^{t,k+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_{\alpha,\beta}(\boldsymbol{y}^{t,k}, \alpha_k, \beta_k)} E(\boldsymbol{y}),$$

with the final solution set to $\boldsymbol{y}^{t+1} \leftarrow \boldsymbol{y}^{t,K+1}$.

Because the min-cut problem is solvable efficiently only if all edge weights are non-negative, it is now clear why $E^{(2)}$ has to be a semi-metric: this property guarantees non-negativity of all edge weights in the auxiliary graph G'.

THE α-EXPANSION NEIGHBORHOOD is slightly different from the α-β-swap: the α-β-swap neighborhood was defined by choosing two labels, α and β, and allowing all nodes currently labeled α or β to change their state within the set $\{\alpha, \beta\}$.

The parametrized α-expansion neighborhood $\mathcal{N}_\alpha(\boldsymbol{y}, \alpha)$ is similar in that every node is allowed to remain in its current state *or* to change its state to α. Finding the optimal solution within the neighborhood of the current solution is again just a binary labeling problem. However, in order to work it requires $E^{(2)}_{i,j}$ to satisfy the triangle inequality for all $(i,j) \in E$ and is thus more limited, compared to the α-β-swap.

Formally, the α-expansion neighborhood is defined as follows.

$$\mathcal{N}_\alpha : \mathcal{Y} \times \mathbb{N} \to \mathcal{Y}^*,$$
$$\mathcal{N}_\alpha(\boldsymbol{y}, \alpha) := \{\boldsymbol{z} \in \mathcal{Y} : \forall i \in V : z_i \in \{y_i, \alpha\}\}.$$

[177] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(11):1222–1239, 2001

As for the α-β-swap neighborhood, Boykov et al.[177] showed that the minimizer within $\mathcal{N}_\alpha(\boldsymbol{y}, \alpha)$ can be found by solving a network flow problem on a auxiliary graph whose edge weights can be derived by decomposing the energy function within the neighborhood.

$$\begin{aligned}
\boldsymbol{y}^{t+1} &= \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_\alpha(\boldsymbol{y}^t, \alpha)} E(\boldsymbol{y}) \\
&= \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_\alpha(\boldsymbol{y}^t, \alpha)} \sum_{i \in V} E^{(1)}_i(y_i) + \sum_{(i,j) \in E} E^{(2)}_{i,j}(y_i, y_j) \\
&= \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_\alpha(\boldsymbol{y}^t, \alpha)} \Big[\sum_{\substack{i \in V, \\ y_i = \alpha}} E^{(1)}_i(\alpha) + \sum_{\substack{i \in V, \\ y_i \neq \alpha}} E^{(1)}_i(y^t_i) \\
&\qquad + \sum_{\substack{(i,j) \in E, \\ y_i = \alpha, y_j = \alpha}} E^{(2)}_{i,j}(\alpha, \alpha) + \sum_{\substack{(i,j) \in E, \\ y_i = \alpha, y_j \neq \alpha}} E^{(2)}_{i,j}(\alpha, y^t_j) \\
&\qquad + \sum_{\substack{(i,j) \in E, \\ y_i \neq \alpha, y_j = \alpha}} E^{(2)}_{i,j}(y^t_i, \alpha) + \sum_{\substack{(i,j) \in E, \\ y_i \neq \alpha, y_j \neq \alpha}} E^{(2)}_{i,j}(y^t_i, y^t_j) \Big].
\end{aligned} \tag{52}$$

The graph structure of the auxiliary graph depends on the current solution $\boldsymbol{y}^t$ and is illustrated in Figure 41.

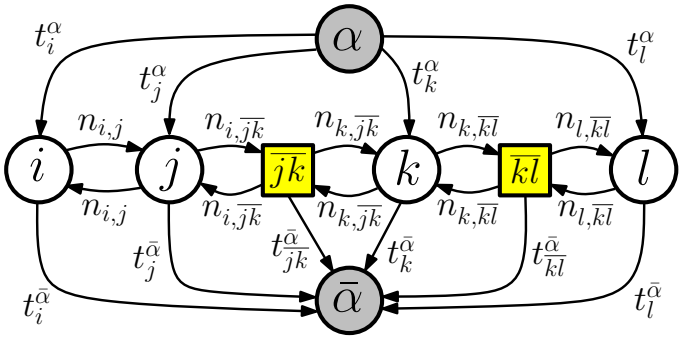


Figure 41: Alpha expansion graph construction: all pixels i,j,k and l are embedded into a graph and connected to a source node "α" and a sink node "$\bar{\alpha}$" (drawn in gray). For pairs of pixels $(i,j) \in E$ which are currently labeled with different labels, $y_i^t \neq y_j^t$ a new node "$\overline{ij}$" is introduced (drawn squared). The minimum directed α-$\bar{\alpha}$ cut on this graph is the minimum energy solution in $\mathcal{N}_\alpha(\boldsymbol{y}^t, \alpha)$.

Formally, given $G = (V, E)$ and a current solution $\boldsymbol{y}^t \in \mathcal{Y}$, the auxiliary directed, edge-weighted graph $G' = (V', E')$ is constructed as follows.

$$
\begin{aligned}
V' &= \{\alpha, \bar{\alpha}\} \cup V \cup \{\overline{ij} : \forall (i,j) \in E : y_i^t \neq y_j^t\}, \\
E' &= \{(\alpha, i, t_i^\alpha) : \forall i \in V\} \cup \{(i, \bar{\alpha}, t_i^{\bar{\alpha}}) : \forall i \in V\} \\
&\quad \cup \{(i, j, n_{i,j}), (j, i, n_{i,j}) : \forall (i,j) \in E : y_i^t = y_j^t\} \\
&\quad \cup \{(\overline{ij}, \bar{\alpha}, t_{\overline{ij}}^{\bar{\alpha}}) : \forall (i,j) \in E : y_i^t \neq y_j^t\} \\
&\quad \cup \{(i, \overline{ij}, n_{i,\overline{ij}}), (\overline{ij}, i, n_{i,\overline{ij}}), (j, \overline{ij}, n_{j,\overline{ij}}), (\overline{ij}, j, n_{j,\overline{ij}}) : \forall (i,j) \in E : y_i^t \neq y_j^t\},
\end{aligned}
$$

with non-negative edge weights calculated from the current solution $\boldsymbol{y}^t$ as follows.

$$
\begin{aligned}
t_i^\alpha &= E_i^{(1)}(\alpha), \\
t_i^{\bar{\alpha}} &= \begin{cases} \infty & \text{if } y_i^t = \alpha, \\ E_i^{(1)}(y_i^t) & \text{otherwise} \end{cases}, \\
n_{i,j} &= E_{i,j}^{(2)}(y_i^t, \alpha) \quad \left(= E_{i,j}^{(2)}(\alpha, y_j^t)\right), \\
t_{\overline{ij}}^{\bar{\alpha}} &= E_{i,j}^{(2)}(y_i^t, y_j^t), \\
n_{i,\overline{ij}} &= E_{i,j}^{(2)}(y_i^t, \alpha).
\end{aligned}
$$

The min-cut on G' corresponds to the minimum in (52) by constructing $\boldsymbol{y}^{t+1}$ from the minimum weight edge cutset $\mathcal{C}$ of G' as

$$
y_i^{t+1} = \begin{cases} \alpha & \text{if } (\alpha, i) \in \mathcal{C} \\ y_i^t & \text{otherwise} \end{cases},
$$

for all $i \in V$. The analysis and proof can be found in Boykov et al.[178].

The requirement that $E^{(2)}$ must satisfy the triangle inequality is needed to show that cuts like the one shown in Figure 42 cannot be minimal. If the triangle inequality holds, then the cut cannot be minimal as cutting $(\overline{jk}, \bar{\alpha})$

[178] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(11):1222–1239, 2001

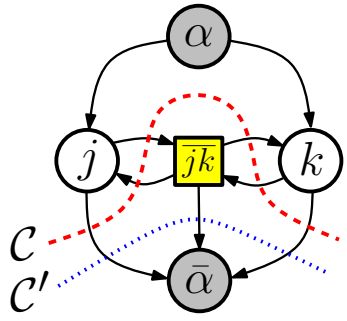


Figure 42: A cut $\mathcal{C}$ of the shown type (drawn dashed) can never be a minimal cut in G'. The cut $\mathcal{C}'$ (drawn dotted) always has an energy no greater than $\mathcal{C}$, due to the triangle inequality assumption on $E^{(2)}$.

directly gives a lower energy:

$$\begin{aligned} E(\mathcal{C}) &= n_{j,\overline{jk}} + n_{k,\overline{jk}} + t_j^{\bar{\alpha}} + t_k^{\bar{\alpha}} \\ &= E_{j,k}^{(2)}(y_j^t, \alpha) + E_{j,k}^{(2)}(y_k^t, \alpha) + t_j^{\bar{\alpha}} + t_k^{\bar{\alpha}} \\ &\geq E_{j,k}^{(2)}(y_j^t, y_k^t) + t_j^{\bar{\alpha}} + t_k^{\bar{\alpha}} \\ &= t_{\overline{jk}}^{\bar{\alpha}} + t_j^{\bar{\alpha}} + t_k^{\bar{\alpha}} \\ &= E(\mathcal{C}'). \end{aligned}$$

As already done for the α-β-swap, the parameter α in the α-expansion is iterated over as follows. We set $\boldsymbol{y}^{t,0} \leftarrow \boldsymbol{y}^t$, and iterate $k = 0, 1, \ldots, K$, where $K = |\bigcup_{i \in V} \mathcal{Y}_i| - 1$, so we have

$$\boldsymbol{y}^{t,k+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{y} \in \mathcal{N}_\alpha(y^t, k)} E(\boldsymbol{y}),$$

with the final result defining the next iterate as $\boldsymbol{y}^{t+1} \leftarrow y^{t,K+1}$.

In practice the α-expansion is often preferred over the α-β-swap because it converges faster and Boykov established a worst case bound on the energy with respect to the true optimal energy.[179]

[179] One advantage of the α-β-swap algorithm is that it can be easily parallelized by processing disjoint pairs (α_1, β_1), (α_2, β_2), $\alpha_2, \beta_2 \notin \{\alpha_1, \beta_1\}$ at the same time.

THE EFFICIENCY OF GRAPH-CUT BASED ENERGY MINIMIZATION ALGORITHMS has lead to a flurry of research into this direction. We give a brief overview of the main results and research directions.

The class of energy functions which can be minimized using graphcuts has first been characterized by Kolmogorov and Zabih[180] and Freedman and Drineas[181]. Their main result characterize general energy functions involving interactions between two and three variables with binary states by stating sufficient conditions such that the solution produced by α-expansion is the true optimal solution: an energy is exact graphcut-solvable if it is *regular*. For the case of pairwise energies and binary states the requirement is

$$E_{i,j}^{(2)}(0,0) + E_{i,j}^{(2)}(1,1) \leq E_{i,j}^{(2)}(0,1) + E_{i,j}^{(2)}(1,0),$$

which can be understood as requiring that adjacent nodes must have a lower energy if they are labeled with the same state than when they have different states. For interactions involving three nodes, this holds if each projection onto two variables satisfies the above condition.

Ishiwaka[182] extended the above to the case of multilabel states, i.e., where $|\mathcal{Y}_i| > 2$ for some $i \in V$. In general, to characterize solvable energies with high-order interactions is ongoing research. Kohli et al.[183] gave an example of an energy term with simple structure, called the $\mathcal{P}^n$ generalized Potts potential which can be optimized using graph cuts. See Ramalingam et al.[184] for an application of $\mathcal{P}^n$ to image segmentation.

For energies which do not satisfy regularity conditions, Kolmogorov and Rother[185] give a graphcut-based iterative algorithm using *probing* techniques

[180] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2): 147–159, 2004

[181] Daniel Freedman and Petros Drineas. Energy minimization via graph cuts: Settling what is possible. In *CVPR*, pages 939–946, 2005

[182] Hiroshi Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Trans. Pattern Anal. Mach. Intell*, 25(10):1333–1336, 2003

[183] Pushmeet Kohli, L'ubor Ladický, and Philip H. S. Torr. Robust higher order potentials for enforcing label consistency. In *CVPR*, 2008

[184] Srikumar Ramalingam, Pushmeet Kohli, Karteek Alahari, and Philip H. S. Torr. Exact inference in multi-label CRFs with higher order cliques. In *CVPR*, 2008

[185] Vladimir Kolmogorov and Carsten Rother. Minimizing nonsubmodular functions with graph cuts-A review. *IEEE Trans. Pattern Anal. Mach. Intell*, 29 (7):1274–1279, 2007

from combinatorial optimization, producing an approximate minimizer. In case the nodes have only binary states, the algorithm enjoys a favorable *partial optimality* property: all node states determined by the algorithm are either certain or uncertain with the guarantee that there exists an optimal solution which, when considering the certain nodes only, is identical to the solution provided by the algorithm.

Another research direction has been to improve the efficiency of graphcut based minimization algorithms. For planar graph structures common in computer vision progress has been made by using efficient network flow algorithms specific to planar graphs, see Schraudolph and Kamenetsky[186] and Schmidt et al.[187]. For general graphs with multilabel states, the most efficient current algorithms are due to Alahari et al.[188] and Komodakis et al.[189]. Both algorithms reuse computations from previous iterations.

[186] Nicol N. Schraudolph and Dmitry Kamenetsky. Efficient exact inference in planar ising models. In *NIPS*. MIT Press, 2008

[187] Frank R. Schmidt, Eno Töppe, and Daniel Cremers. Efficient planar graph cuts with applications in computer vision. In *CVPR*. IEEE Computer Society, 2009

[188] Karteek Alahari, Pushmeet Kohli, and Philip H. S. Torr. Reduce, reuse & recycle: Efficiently solving multi-label MRFs. In *CVPR*. IEEE Computer Society, 2008

[189] Nikos Komodakis, Georgios Tziritas, and Nikos Paragios. Fast, approximately optimal solutions for single and dynamic MRFs. In *CVPR*. IEEE Computer Society, 2007

Linear Programming Relaxation

We now discuss a method for solving the discrete MAP-MRF problem in which the problem is modeled as linear integer programming problem. A tractable *relaxation* can be obtained by replacing the integrality constraints with simple interval constraints. The resulting linear program is then the "*linear programming relaxation*" to the MAP-MRF problem.

The original linear programming formulation to the MAP-MRF problem is due to Schlesinger[190] in 1976. Recently it has been rediscovered[191]. It is used for solving for the MAP solution $\boldsymbol{y}^*$ when the underlying graph $G = (V, E)$ consists of only unary and pairwise potentials. Then, the MAP-MRF integer linear program formulation is exact.

Although the formulation is *exact* in case the variables are restricted to be binary, we can relax the integer requirement to obtain a corresponding linear program (LP) which can be solved in polynomial time. The solution of the relaxed problem might not correspond to an exact MAP state.

[190] M.I. Schlesinger. Sintaksicheskiy analiz dvumernykh zritelnikh signalov v usloviyakh pomekh (syntactic analysis of two-dimensional visual signals in noisy conditions). *Kibernetika*, 4:113–130, 1976. In Russian; and V.K. Koval and M.I. Schlesinger. Dvumernoe programmirovanie v zadachakh analiza izobrazheniy (two-dimensional programming in image analysis problems). *Automatics and Telemechanics*, 8:149–168, 1976. In Russian

[191] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, November 2005; Tomáš Werner. A linear programming approach to max-sum problem: A review. Research report, Center for Machine Perception, Czech Technical University, December 2005; and Chen Yanover, Talya Meltzer, and Yair Weiss. Linear programming relaxations and belief propagation - an empirical study. *JMLR*, 7:1887–1907, 2006

The outline for deriving the relaxation is the following: we first linearize the MAP objective in a new overcomplete parametrization and then consider the additional properties that must be satisfied for the new parameters in order to map to an original feasible configuration in $\mathcal{Y}$.

The energy function we want to minimize is of the form

$$E(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{w}) = \underbrace{\sum_{i \in V} \boldsymbol{w}_1^\top \phi_i^{(1)}(y_i, \boldsymbol{x})}_{(A)} + \underbrace{\sum_{(i,j) \in E} \boldsymbol{w}_2^\top \phi_{i,j}^{(2)}(y_i, y_j, \boldsymbol{x})}_{(B)}. \tag{53}$$

Both terms, (A) and (B) have a non-linear dependency on $\boldsymbol{y}$. Because the set of feasible labelings is finite we can introduce *new* variables in such a way that (A) can be rewritten equivalently in linear form in the new parametrization. For this, let us introduce for all $i \in V$, for all $s \in \mathcal{Y}_i$ a binary variable $\mu_i(s) \in \{0, 1\}$

which indicates whether $y_i = s$. By *linearizing*, that is, by instantiating the variable y_i for all its values in the non-linear expression, we rewrite (A) as

$$\sum_{i \in V} \boldsymbol{w}_1^\top \boldsymbol{\phi}_i^{(1)}(y_i, \boldsymbol{x}) = \sum_{i \in V} \sum_{s \in \mathcal{Y}_i} \mu_i(s) \underbrace{\left[\boldsymbol{w}_1^\top \boldsymbol{\phi}_i^{(1)}(s, \boldsymbol{x}) \right]}_{\text{constant}}.$$

Whereas on the left hand side the dependency on y_i is present, the right hand side depends only on $\mu_i(s)$. In order to ensure correctness of the above transformation we need to enforce that only one variable $\mu_i(s)$ is set to one over all configurations $s \in \mathcal{Y}_i$ of the node. We therefore restrict the configurations using the following two constraints.

$$\sum_{s \in \mathcal{Y}_i} \mu_i(s) = 1, \qquad \forall i \in V, \tag{54}$$

$$\mu_i(s) \in \{0, 1\}, \qquad \forall i \in V, \forall s \in \mathcal{Y}_i. \tag{55}$$

Likewise, the pairwise term (B) can be *linearized* by instantiating pairwise configurations and selecting exactly one of them. We introduce for all $(i, j) \in E$, for all $(s, t) \in \mathcal{Y}_i \times \mathcal{Y}_j$ a binary variable $\mu_{i,j}(s, t) \in \{0, 1\}$ which indicates whether $y_i = s$ *and* $y_j = t$. We can now rewrite (B) as

$$\sum_{(i,j) \in E} \boldsymbol{w}_2^\top \boldsymbol{\phi}_{i,j}^{(2)}(y_i, y_j, \boldsymbol{x}) = \sum_{(i,j) \in E} \sum_{(s,t) \in \mathcal{Y}_i \times \mathcal{Y}_j} \mu_{i,j}(s, t) \underbrace{\left[\boldsymbol{w}_2^\top \boldsymbol{\phi}_{i,j}^{(2)}(s, t, \boldsymbol{x}) \right]}_{\text{constant}}.$$

Again, the right hand side is a linear form in the new parametrization. In order to ensure consistency between the pairwise and unary variables we need to enforce by definition for all $(i, j) \in E$, for all $(s, t) \in \mathcal{Y}_i \times \mathcal{Y}_j$:

$$\mu_{i,j}(s, t) = I(y_i = s \wedge y_j = t) = I(y_i = s) \cdot I(y_j = t) = \mu_i(s) \mu_j(t), \tag{56}$$

which implicitly includes the constraints

$$\mu_{i,j}(s, t) \in \{0, 1\}, \qquad \forall (i, j) \in E : \forall (s, t) \in \mathcal{Y}_i \times \mathcal{Y}_j. \tag{57}$$

Unfortunately constraint (56) is a non-linear equality constraint and thus does not describe a convex set.[192]

[192] All equality constraints which describe convex sets must define an affine subspace.

Fortunately, a clever transformation can linearize (56). By summing over $t \in \mathcal{Y}_j$, we can obtain for all $(i, j) \in E$ and for all $s \in \mathcal{Y}_i$ the following set of constraints.

$$\begin{aligned} & \sum_{t \in \mathcal{Y}_j} \mu_{i,j}(s, t) = \sum_{t \in \mathcal{Y}_j} \mu_i(s) \mu_j(t) \\ \Leftrightarrow \quad & \sum_{t \in \mathcal{Y}_j} \mu_{i,j}(s, t) = \mu_i(s). \end{aligned} \tag{58}$$

The above transformation is *exact*: assume we are given a set of variables $\boldsymbol{\mu}$ such that (54), (55), (57) and (58) hold. Then (56) is also satisfied for all $(i, j) \in E$, for all $(s, t) \in \mathcal{Y}_i \times \mathcal{Y}_j$.

Proof. First, note that from (58) and (54) by summing over $s \in \mathcal{Y}_i$ we obtain that $\sum_{(s,t)\in\mathcal{Y}_i\times\mathcal{Y}_j} \mu_{i,j}(s,t) = 1$ holds for all $(i,j) \in E$. Then we have $\forall (i,j) \in E$: $\forall (s,t) \in \mathcal{Y}_i \times \mathcal{Y}_j$:

$$
\begin{aligned}
\mu_i(s)\mu_j(t) &= \left(\sum_{v\in\mathcal{Y}_j} \mu_{i,j}(s,v)\right)\left(\sum_{u\in\mathcal{Y}_i} \mu_{i,j}(u,t)\right) \\
&= \left(\sum_{v\in\mathcal{Y}_j\setminus\{t\}} \mu_{i,j}(s,v) + \mu_{i,j}(s,t)\right)\left(\sum_{u\in\mathcal{Y}_i\setminus\{s\}} \mu_{i,j}(u,t) + \mu_{i,j}(s,t)\right) \\
&= \underbrace{\sum_{(u,v)\in\mathcal{Y}_i\setminus\{s\}\times\mathcal{Y}_j\setminus\{t\}} \mu_{i,j}(u,t)\mu_{i,j}(s,v)}_{0} \\
&\quad + \underbrace{\mu_{i,j}(s,t)\sum_{u\in\mathcal{Y}_i\setminus\{s\}} \mu_{i,j}(u,t)}_{0} + \underbrace{\mu_{i,j}(s,t)\sum_{v\in\mathcal{Y}_j\setminus\{t\}} \mu_{i,j}(s,v)}_{0} \\
&\quad + \underbrace{\mu_{i,j}(s,t)\mu_{i,j}(s,t)}_{\mu_{i,j}(s,t)} \\
&= \mu_{i,j}(s,t),
\end{aligned}
$$

so that (56) holds. □

Putting the pieces together, we now state the complete integer linear program. In order to avoid confusion, in the following problem only $\boldsymbol{\mu}$ are variables, all remaining expressions are constants.

$$
\begin{aligned}
\min_{\boldsymbol{\mu}} \quad & \sum_{i\in V}\sum_{y_i\in\mathcal{Y}_i} \mu_i(y_i)\left(\boldsymbol{w}_1^\top \boldsymbol{\phi}_i^{(1)}(y_i,\boldsymbol{x})\right) && (59)\\
& \quad + \sum_{\substack{(i,j)\\ \in E}}\sum_{\substack{(y_i,y_j)\\ \in\mathcal{Y}_i\times\mathcal{Y}_j}} \mu_{i,j}(y_i,y_j)\left(\boldsymbol{w}_2^\top \boldsymbol{\phi}_{i,j}^{(2)}(y_i,y_j,\boldsymbol{x})\right) \\
\text{sb.t.} \quad & \sum_{y_i\in\mathcal{Y}_i} \mu_i(y_i) = 1, \qquad i \in V, && (60)\\
& \sum_{y_j\in\mathcal{Y}_j} \mu_{i,j}(y_i,y_j) = \mu_i(y_i),\ (i,j)\in E, y_i \in \mathcal{Y}_i, && (61)\\
& \mu_i(y_i) \in \{0,1\}, \quad i \in V, y_i \in \mathcal{Y}_i, && (62)\\
& \mu_{i,j}(y_i,y_j) \in \{0,1\},\ (i,j)\in E, (y_i,y_j) \in \mathcal{Y}_i \times \mathcal{Y}_j. && (63)
\end{aligned}
$$

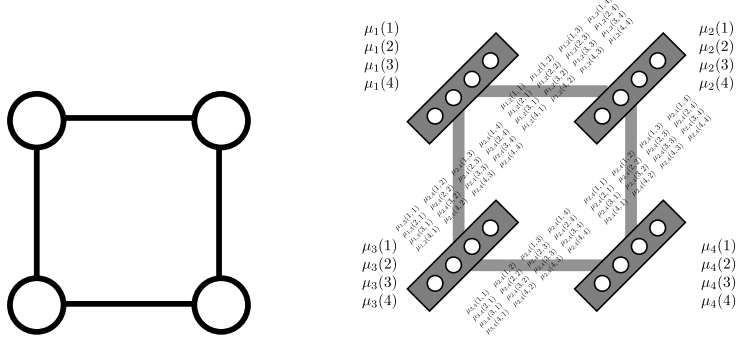
As discussed above, the first set of equality constraints (60) enforce that each node is assigned exactly one label. The second set of equality constraints (61) enforce proper consistency between node and edge states.

Given a solution vector $\boldsymbol{\mu}$ to the ILP (59) the labeling $\boldsymbol{y}^*$ is obtained by setting

$$y_i \leftarrow \operatorname{argmax}_{y_i\in\mathcal{Y}_i} \mu_i(y_i).$$

The integer program (59) is exact but NP-hard. The corresponding *linear programming relaxation* is obtained by relaxing (62) and (63) to the range $[0, 1]$. The resulting LP relaxation has been analyzed extensively [193].

Although linear programming is among the best developed numerical disciplines, the primal LP (59) is practically restricted to medium sized graphs with a few tens of thousands of nodes and tens of node labels, because on the order of $O(|V|^2(\max_{i \in V} |\mathcal{Y}_i|)^2)$ variables are used. This problem is illustrated in Figure 43(b) which shows the 80 introduced variables for the simple four-node four-state MRF shown in Figure 43(a). The scaling problem is further discussed in Yanover et al.[194].



(a) A small four-node MRF, each node has four states. (Only dependent variables are shown.)

(b) Variables introduced by the new parametrization: $4 \cdot 4 = 16$ node variables, $4 \cdot 4 \cdot 4 = 64$ edge variables, for a total of 80 variables.

Figure 43: Size of the LP relaxation.

The above remark shows that the linear program (59) does not scale when applied naively. Instead, the linear program has been used as a model to derive efficient algorithms. By considering the *dual* of (59), Globerson and Jaakkola[195], Kumar and Torr[196], and Sontag et al.[197] derived *message passing* algorithms directly from the linear program.

Komodakis et al.[198] derived a simple convergent version of the tree-reweighted message passing (TRW) scheme of Wainwright et al.[199] by decomposing the linear program (59) into a set of tree-structured models and introducing coupling constraints which are subsequently relaxed using Lagrangian relaxation. This *Lagrangian decomposition* technique is well known in the optimization literature and one advantage of treating the MAP-MRF problem in terms of its LP relaxation (59) is that it makes these techniques applicable.

IMPROVING THE QUALITY OF THE RELAXATION is another active research area. The convex hull of the feasible integer solutions of (59) is known as the *marginal polytope* $\mathcal{M}$. By relaxing the integrality constraints in (59) one has constructed an outer approximation to this set. This approximation is known as *local consistency polytope*. By analysis of the marginal polytope, Sontag and Jaakkola[200] derive additional inequalities valid for the marginal polytope

[193] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, November 2005; and Tomáš Werner. A linear programming approach to max-sum problem: A review. Research report, Center for Machine Perception, Czech Technical University, December 2005

[194] Chen Yanover, Talya Meltzer, and Yair Weiss. Linear programming relaxations and belief propagation - an empirical study. *JMLR*, 7:1887–1907, 2006

[195] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *NIPS*, 2007

[196] Mudigonda Pawan Kumar and Philip Torr. Efficiently solving convex relaxations for MAP estimation. In *ICML*, 2008

[197] David Sontag, Talya Meltzer, Amir Globerson, Tommi Jaakkola, and Yair Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, 2008

[198] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*. IEEE, 2007

[199] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, November 2005

[200] David Sontag and Tommi Jaakkola. New outer bounds on the marginal polytope. In *NIPS*, 2007

which, when added to the linear program *tighten* the LP relaxation. They derive the inequalities by identifying projections of the marginal polytope with the *cut polytope* and applying known *cycle inequalities* to the projection. By mapping the cycle inequalities back to the original space, valid inequalities for the marginal polytopes are derived. The resulting tightened relaxation is shown to be much tighter than the standard LP relaxation.[201]

The tightness of the linear programming relaxation versus other relaxations has been analyzed by Kumar et al.[202]. They showed that the LP relaxation dominates other known relaxations. Kohli et al.[203] consider the issue of deriving *partial-optimal* solutions for the MAP-MRF problem: a solution is said to be partial-optimal if, for a subset of nodes, the labeled node states are guaranteed to be the same in any optimal solution. These nodes can be removed from the problem entirely and a reduced sized problem consisting of only the unsure nodes has to be solved. Kohli et al. show that it is indeed possible to obtain partial-optimality for the multi-label case by considering a different relaxation based on *roof duality*. The tightness has also been analyzed by Komodakis and Paragios[204] and Werner[205].

PROBLEMS INVOLVING HIGH-ORDER INTERACTIONS are not directly solvable using the linear programming relaxation (59). Werner extends the *max-sum diffusion* algorithm to handle interactions involving more than two random variables. When the interactions are efficiently computable the algorithm yields a polynomial-time approximation to the MAP state.

In the next chapter we consider a particular type of global interaction which ensures that the output labeling forms a connected component.

[201] Sontag and Jaakkola also consider the problem of computing marginal probabilities which can be posed as maximizing the entropy of the distribution parametrized by μ over the marginal polytope. Because the exact entropy is also difficult to compute an upper bound is used instead. Interestingly, the results indicate that most of the remaining inaccuracy in estimating marginals comes from the entropy bound and the approximation of the marginal polytope is already sufficiently tight.

[202] Mudigonda Pawan Kumar, Vladimir Kolmogorov, and Philip Torr. An analysis of convex relaxations for MAP estimation. In *NIPS*, 2008

[203] Pushmeet Kohli, Alexander Shekhovtsov, Carsten Rother, Vladimir Kolmogorov, and Philip H. S. Torr. On partial optimality in multi-label MRFs. In *ICML*, volume 307, pages 480–487, 2008

[204] Nikos Komodakis and Nikos Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. In *ECCV*, 2008

[205] Tomáš Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (MAP-MRF). In *CVPR*, 2008

Image Segmentation under Connectivity-Constraints

The previous chapter summarized the state of the art in structured prediction. We have seen that one limitation of current graphical models is that they are forced to consider "small" interactions such that the model can be decomposed into tractable parts.

The key contribution of this chapter is a novel method to incorporate truly global interactions into random field models. The approach is general and extends the state of the art of random field models.

In particular, the interaction we consider is specified by a potential function that is not only global, but is in itself computationally intractable.

THE POTENTIAL FUNCTIONS we consider are defined on *all* nodes in the graph, denoted $\psi_V(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{w})$. We consider a "connectedness potential", which enforces connectedness of the output labelings with respect to a graph. We derive our algorithm in a principled way using results from polyhedral combinatorics.

Although in this chapter we only consider one global potential function, the overall approach by which we incorporate the function is general and applicable to other higher-order potential functions.

In the section that follows, we formalize the notion of connectedness by analyzing the set of all connected MRF labelings: the connected subgraph polytope. The discussion contains the main results on the structure of the problem and proposes a tractable relaxation. Continuing the analysis, we discuss in an extra section the properties of the approximate solution that our relaxation provides. In a third section we show how the tractable relaxation for connected subgraphs can be used to define global potential functions in conditional random fields.

The remaining part of the chapter provide the experimental evaluation of the proposed MRF/CRF with connectedness potentials on both a synthetic data set and on the challenging PASCAL VOC 2008 segmentation data set; we finish with an outlook on problems where our technique can be applied.

Connected Subgraph Polytope

The LP relaxation (59) has variables $\mu_i(y_i) \in \{0, 1\}$ encoding if a node i has label y_i. In this section we derive a polyhedral set which can be intersected with the feasible set of LP (59) such that for all remaining feasible solutions

all nodes labeled with the same label form a connected subgraph. This set is the *connected subgraph polytope*, the convex hull of all possible labeling that are connected. We first define this set and then analyze its properties.

Definition 18 (Connected Subgraph Polytope) *Given a simple, connected, undirected graph $G = (V, E)$, consider indicator variables $y_i \in \{0, 1\}$, $i \in V$. Let $C = \{\boldsymbol{y} : G' = (V', E') \text{ connected, with } V' = \{i : y_i = 1\}, E' = (V' \times V') \cap E\}$ denote the finite set of connected subgraphs of G. Then we call the convex hull $Z = conv(C)$ the* connected subgraph polytope.

The convex hull of a finite set of points is the tightest possible convex relaxation of the set. Furthermore, for the case of minimizing a linear function over the convex hull, it is known from classic linear programming theory[206] that at least one optimal solution exists at a vertex of the polytope. By construction, this solution is then also in C and the relaxation is exact. Unfortunately, optimizing over this polytope is NP-hard, as the following theorem shows. The theorem is identical to Theorem 1 in[207]; we state it here for the reference to the earlier work of Karp[208].

Theorem 4 (Karp, 2002) *It is NP-hard to optimize a linear function over $Z = conv(C)$.*

The proof can be found in[209], where the problem appears under the name "Maximum-Weight Connected Subgraph Problem".

Therefore, if we plan to intersect $\text{conv}(C)$ with the feasible set of (59), we are planning to optimize a linear function over this polytope. Unfortunately, from Theorem 4 it follows that optimizing a linear function over $\text{conv}(C)$ is NP-hard, and it is unlikely that $\text{conv}(C)$ has a "simple" description, a description in terms of linear inequalities which is polynomial-time separable[210]. To overcome this difficulty we will derive a tight relaxation to $\text{conv}(C)$ which is still polynomially solvable.

To do this, we focus on the properties of C and its convex hull Z. We first show that Z has full dimension, i.e., does not live in a proper subspace. Second, we show that $y_i \geq 0$ and $y_i \leq 1$ are facet-defining inequalities for all graphs. Figure 44 shows what this means: $d_1^\top y \leq 1$ and $d_2^\top y \leq 1$ are both *valid*, but only $d_3^\top y \leq 1$ is facet-defining[211].

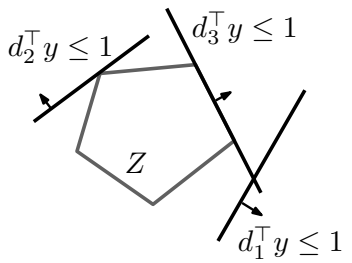


Figure 44: Three valid inequalities, only one of which is facet-defining for the polytope Z.

Lemma 6 *$dim(Z) = |V|$.*

Lemma 7 *For all $i \in V$, the inequalities $y_i \geq 0$ and $y_i \leq 1$ are facet-defining for Z.*

The proofs can be found in the appendix.

Definition 19 (Vertex-Separator Set) *Given a simple, connected, undirected graph $G = (V, E)$, for any pair of vertices $i, j \in V$, $i \neq j$, $(i, j) \notin E$, the set $S \subseteq V \setminus \{i, j\}$ is said to be a* vertex-separator set *with respect to $\{i, j\}$ if the removal of S from G disconnects i and j.*

[206] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. 1997; and Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998

[207] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Graph cut based image segmentation with connectivity priors. In *CVPR*, 2008

[208] Richard M. Karp. Maximum-weight connected subgraph problem, 2002. `http://www.cytoscape.org/ISMB2002/`

[209] Trey Ideker, Owen Ozier, Benno Schwikowski, and Andrew F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. In *ISMB*, 2002; and Richard M. Karp. Maximum-weight connected subgraph problem, 2002. `http://www.cytoscape.org/ISMB2002/`

[210] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998

[211] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998

If the removal of S from G disconnects i and j, then there exists no path between i and j in $G' = (V \setminus S, E \setminus (S \times S))$. As an additional definition, a set $\bar{S}$ is said to be an *essential vertex-separator set* if it is a vertex-separator set with respect to $\{i, j\}$ and any strict subset $T \subset \bar{S}$ is not. Let $\mathcal{S}(i, j) = \{S \subset V : S \text{ is a vertex-separator set with respect to } \{i, j\}\}$ denote the collection of all vertex-separator sets, and $\bar{\mathcal{S}}(i, j) \subset \mathcal{S}(i, j)$ be the subset of essential vertex-separator sets.

Theorem 5 *C, the set of all connected subgraphs, can be described exactly by the following constraint set.*

$$y_i + y_j - \sum_{k \in S} y_k \leq 1, \forall (i, j) \notin E : \forall S \in \mathcal{S}(i, j), \tag{64}$$

$$y_i \in \{0, 1\}, \qquad i = 1, \ldots, |V|. \tag{65}$$

The proof can be found in the appendix.

Theorem 5 has a simple intuitive interpretation, shown in Figure 45. If two vertices i and j are selected ($y_i = y_j = 1$, shown in black), then any set S of vertices separating them must contain at least one selected vertex. Otherwise i and j cannot be connected because any path from i to j must pass through at least one vertex in S.

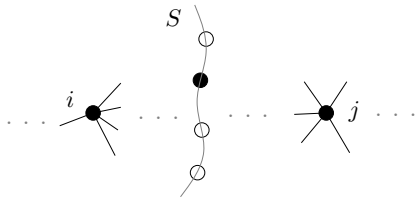


Figure 45: Vertex i and j and one vertex separator set $S \in \bar{\mathcal{S}}(i, j)$.

Having characterized the set of all connected subgraphs exactly by means of (64) and (65) it is natural to look at the linear relaxation, replacing (65) by $y_i \in [0; 1], \forall i$. Such a relaxation yields a polytope $P \supseteq Z = \text{conv}(C) \supset C$, which can be a tight (good) or loose (bad) approximation to $\text{conv}(C)$. The quality of the approximation improves if *facets* of the polytope P are true facets of $\text{conv}(C)$. The following theorem states that in our relaxation a large subset of the constraints (64) — exactly those associated to *essential* vertex-separator sets — are indeed facets of $\text{conv}(C)$.

Theorem 6 *The following linear inequalities are* facet-defining *for $Z = conv(C)$.*

$$y_i + y_j - \sum_{k \in S} y_k \leq 1, \quad \forall (i, j) \notin E : \forall S \in \bar{\mathcal{S}}(i, j). \tag{66}$$

The proof can be found in the appendix.

Let us summarize our progress so far. We have described the set of connected subgraphs and the associated connected subgraph polytope. Furthermore we have shown that a relaxation of the connected subgraph polytope is locally exact in that the set of linear inequalities (66) are true facets of $\text{conv}(C)$. However, in general the number of linear inequalities (66) used in our relaxation is exponential in $|V|$.

We now show that optimization over the set defined by (66) is still tractable because finding violated inequalities — the so called *separation problem* — can be solved efficiently using max-flow algorithms.

Theorem 7 (Polynomial-time separation) *For a given point $\boldsymbol{y} \in [0; 1]^{|V|}$ to find the most violated inequality (66) or prove that no violated inequality exists requires only time polynomial in $|V|$.*

Proof. We give a constructive separation algorithm based on solving a linear max-flow problem on an auxiliary directed graph. For a given point $\boldsymbol{y} \in [0;1]^{|V|}$, consider all $(i,j) \in V \times V$ with $i \neq j$, $(i,j) \notin E$ and $y_i > 0$, $y_j > 0$. For any such (i,j) consider the statement

$$y_i + y_j - \sum_{k \in S} y_k - 1 \leq 0, \qquad \forall S \in \bar{\mathcal{S}}(i,j).$$

Note that in the above statement, the individual variables y are not necessarily binary. We can rewrite the set of inequalities above in equivalent variational form,

$$\max_{S \in \bar{\mathcal{S}}(i,j)} \left(y_i + y_j - \sum_{k \in S} y_k - 1 \right) \leq 0. \tag{67}$$

If we prove that (67) is satisfied, we know that no violated inequalities exists for (i,j). If, however, a violation exists, then the essential vertex-separator set producing the highest violation is given as

$$S^*(i,j) = \operatorname{argmin}_{S \in \bar{\mathcal{S}}(i,j)} \sum_{k \in S} y_k. \tag{68}$$

In order to find this separator set, we transform G into a directed graph G' with edge capacities. In the directed graph each original edge is split into two directed edges with infinite capacity. Additionally each vertex k in the original graph is duplicated and an edge of finite capacity equal to y_k is introduced between the two copies.

Formally, we construct $G' = (V', E')$, $E' \subseteq V' \times V' \times \mathbb{R}$ as follows. Let $V' = V \cup \{k' : k \in V \setminus \{i,j\}\}$. Further let $E' = \{(i,k,\infty) : k \in V, (i,k) \in E\} \cup \{(k',j,\infty) : k \in V, (j,k) \in E\} \cup \{(s',t,\infty), (t',s,\infty) : (s,t) \in E \setminus (\{i,j\} \times \{i,j\})\} \cup \{(k,k',y_k) : k \in V \setminus \{i,j\}\}$. The construction is illustrated for an example graph in Figures 46 and 47.

Finding an (i,j)-cut of finite capacity in G' is equivalent to finding an essential (i,j) vertex separator set in G. This can be seen by recognizing that the only edges that can be cut — hence saturated in a max-flow problem — are the edges (k,k') with finite capacity, which correspond to vertices in the original graph. Solving the max-flow problem in the auxiliary directed graph solves (68). After finding $S^*(i,j)$, we simply check whether (67) is satisfied.

Solving a linear maximum network flow problem of this type is very efficient[212]. The best algorithms known have a computational complexity of $O(|V|^3)$ and $O(|V||E|\log(|V|))$. We need to solve one max-flow problem per (i,j) pair with $y_i > 0$, $y_j > 0$, so the overall separation problem of checking feasibility with respect to (66) can be solved in time $O(|V|^5)$. □

In practice we do not have to check all (i,j) node pairs. Instead, we decompose the graph into connected components such that for all vertices in a connected component there exists an *all*-1-*path* to each other vertex in the component. These connected components can be found in practically linear time using a disjoint set union-rank data structure[213]. Only one representative

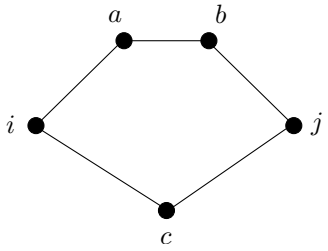


Figure 46: Example graph G. There are three vertex-separator sets in $\mathcal{S}(i,j) = \{\{a,c\},\{b,c\},\{a,b,c\}\}$, of which only $\{a,c\}$ and $\{b,c\}$ are *essential*.

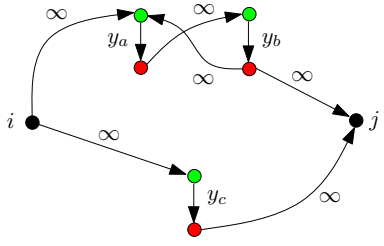


Figure 47: Directed auxiliary graph G' for finding the minimum essential vertex-separator set in G among all sets in $\bar{\mathcal{S}}(i,j)$.

[212] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26 (9):1124–1137, 2004

[213] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. 1990

node is chosen at random from each component and the separation is carried out only for the representative vertices. This procedure is exact.

The above procedure works and has guaranteed polynomial-time complexity. It requires the solution of $O(|V|^2)$ max-flow problems in order to obtain the minimum cut over all pairs of vertices.

Solution Integrality

THE INTEGRALITY OF THE SOLUTION in the intersection of two polytopes is of particular interest. Here, both the polytope defined by the MRF LP relaxation and our relaxation of the connected subgraph polytope are not exact: a relaxation is a superset of the true feasible set. This property allows tractable optimization of otherwise NP-hard problems. If the optimal solution over the relaxed feasible set is integral, that is, the solution is $0,1$-valued, then the relaxation is locally exact and the solution is globally optimal also over the true feasible set.

On the other hand, if the solution has fractional elements $0 < v < 1$, then the solution is outside the true feasible set and the achieved objective of the relaxation provides a lower bound on the true optimal objective. In this case, a popular method to deal with fractional solutions is to use rounding to construct a feasible solution from said fractional solution.

Our construction to enforce high-order potentials by intersecting a polytope with the MRF LP relaxation is exact if restricted to the set of integral solutions. But in order to obtain a tractable optimization problem, we do not enforce integrality but solve the relaxed LP instead. Then our approach provides only the solution to the relaxation, which may have fractional elements.

Because we started with two relaxations it seems natural that when intersecting their feasible sets we also obtain a relaxation. In general, however, even if we had started with the exact marginal polytope with only integral vertices, and another integral polytope, their intersection could have fractional vertices and therefore only provide a relaxation[214]. We now elaborate further on this important point by means of a simple example. For the following discussion, the property we are interested in is the preservation of tightness of the relaxation: if we have two polytopes describing tight relaxations and we construct the intersection, do we still obtain a tight relaxation?

[214] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998

IN GENERAL, THE ANSWER IS NO. By means of constructing a simple counterexample, we show that even if both the marginal polytope relaxation and the relaxation of the restricted feasible set in the node-label dimensions are tight, the intersection of both polytopes need not be. That is, it can contain new fractional vertices, even if both original polytopes contain only integral $\{0,1\}$-vertices.

To see this, consider the simple two node Markov random field shown as

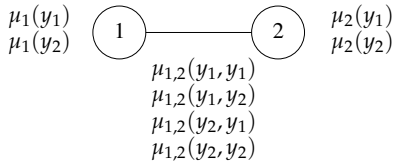


Figure 48: Simple two-node Markov Random Field. The representation used in the LP relaxation defines four variables for the node states, and four variables for the pairwise node states associated to the edge.

a graphical model in Figure 48. In the parametrization used by the linear programming relaxation (59), there are eight variables, four for the node states ($\mu_1(y_1)$, $\mu_1(y_2)$, $\mu_2(y_1)$, $\mu_2(y_2)$) and four for the pairwise node states at the edge ($\mu_{1,2}(y_1, y_1)$, $\mu_{1,2}(y_1, y_2)$, $\mu_{1,2}(y_2, y_1)$, $\mu_{1,2}(y_2, y_2)$).

The feasible set described by the constraints of the LP relaxation is given by the following set of constraints.

$$\begin{aligned} M = \{\mu : \mu_1(y_1) + \mu_1(y_2) = 1, \\ \mu_2(y_1) + \mu_2(y_2) = 1, \\ \mu_{1,2}(y_1, y_1) + \mu_{1,2}(y_1, y_2) = \mu_1(y_1), \\ \mu_{1,2}(y_2, y_1) + \mu_{1,2}(y_2, y_2) = \mu_1(y_2), \\ \mu_{1,2}(y_1, y_1) + \mu_{1,2}(y_2, y_1) = \mu_2(y_1), \\ \mu_{1,2}(y_1, y_2) + \mu_{1,2}(y_2, y_2) = \mu_2(y_2), \\ \mu_1(y_1), \mu_1(y_2), \mu_2(y_1), \mu_2(y_2) \geq 0, \\ \mu_{1,2}(y_1, y_1), \mu_{1,2}(y_1, y_2), \mu_{1,2}(y_2, y_1), \mu_{1,2}(y_2, y_2) \geq 0\}. \end{aligned} \tag{69}$$

The constraints above define the feasible set as a three-dimensional polytope embedded in eight dimensions. We can visualize the polytope partially by *projecting* it onto subspaces. For this, let us define the projection of a polytope.

Definition 20 (Projection of a Polytope) *For a given polytope $Q \subseteq (\mathbb{R}^n \times \mathbb{R}^p)$, the* projection *of Q onto the subspace $\mathbb{R}^n$, denoted $proj_x Q$ is defined as*

$$proj_x Q = \{x \in \mathbb{R}^n : (x, w) \in Q \text{ for some } w \in \mathbb{R}^p\}.$$

Therefore, a point is in the projected set if there is at least one point in the higher dimensional polytope which has identical coefficients in the projection dimensions. For additional properties of projected polytopes, see[215].

[215] Egon Balas. Projection, lifting and extended formulation in integer and combinatorial optimization. *Annals of Operations Research*, (140):125–161, 2005; Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998; and Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998

Figure 49(a) shows the projection $\text{proj}_{\mu_1(y_1), \mu_2(y_1), \mu_{1,2}(y_1, y_1)} M$ of the feasible set of the MRF shown in Figure 48. The full set of vertices of the polytope M is given as follows.

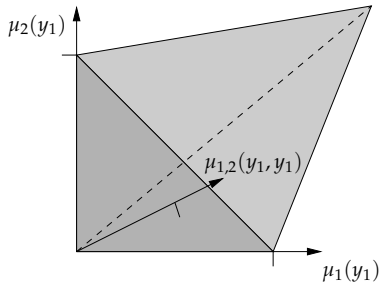
$$\begin{aligned} &\{(\mu_1(y_1), \mu_1(y_2), \mu_2(y_1), \mu_2(y_2), \\ &\qquad \mu_{1,2}(y_1, y_1), \mu_{1,2}(y_1, y_2), \mu_{1,2}(y_2, y_1), \mu_{1,2}(y_2, y_2))\} \\ &= \{(1,0,1,0,\ 1,0,0,0),\ (1,0,0,1,\ 0,1,0,0), \\ &\qquad (0,1,1,0,\ 0,0,1,0),\ (0,1,0,1,\ 0,0,0,1)\}. \end{aligned}$$

[216] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, November 2005

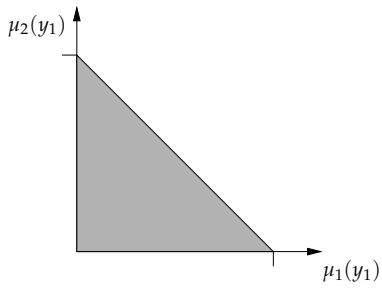
Therefore, all vertices are integral and for this particular MRF the LP relaxation is tight. The feasible set defined by the LP relaxation is therefore identical to the true set, the *marginal polytope*[216].

Now suppose we want to restrict the labelings such that not both nodes are labeled y_1. Then, the only allowed combinations for $(\mu_1(y_1), \mu_2(y_1))$ are from the set $L = \{(0,0), (0,1), (1,0)\}$. The convex hull $\text{conv}(L)$ is shown in Figure 49(b). The facet-defining constraints of the convex hull are simply

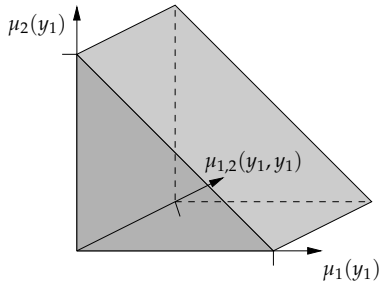
Figure 49: Three dimensional projection of the extended feasible set.



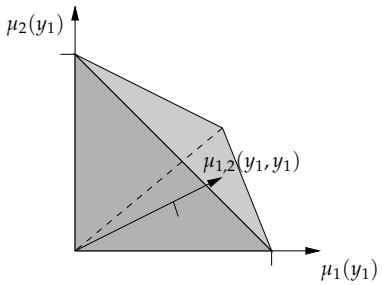
(a) Projection of the marginal polytope M onto the $\mu_1(y_1)$, $\mu_2(y_1)$ and $\mu_{1,2}(y_1,y_1)$ dimensions, i.e., $\text{proj}_{\mu_1(y_1),\mu_2(y_1),\mu_{1,2}(y_1,y_1)}M$.



(b) Desired feasible set with respect to $\mu_1(y_1)$, $\mu_2(y_1)$. The non-trivial facet-defining inequality is $\mu_1(y_1) + \mu_2(y_1) \leq 1$.



(c) Projected view of the extension to the full space of the desired feasible set with respect to $\mu_1(y_1)$, $\mu_2(y_1)$. Note that this polytope has only integral vertices.



(d) Projected view of the resulting intersection with new fractional vertex $(\mu_1(y_1), \mu_2(y_1), \mu_{1,2}(y_1,y_1)) = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$.

$\mu_1(y_1) \geq 0$, $\mu_2(y_1) \geq 0$ and $\mu_1(y_1) + \mu_2(y_1) \leq 1$. We plan to add this new constraints to the feasible set of the MRF, defined by (69). Because the first two non-negativity constraints are already in the constraint set, we only have to consider the new inequality $\mu_1(y_1) + \mu_2(y_1) \leq 1$.

Adding a constraint in the subspace of $\mu_1(y_1)$ and $\mu_2(y_1)$ is the same as first extending the set shown in Figure 49(b) to the full dimensional space and then intersecting it with the marginal polytope. We show a three-dimensional projection of the extended feasible set in Figure 49(c).

The intersection of polytopes shown in Figure 49(c) and Figure 49(a) is shown in Figure 49(d). The new polytope contains only points which satisfy $\mu_1(y_1) + \mu_2(y_1) \leq 1$ and (69). The polytope has the following set of vertices.

$$\begin{aligned}
&\{(\mu_1(y_1), \mu_1(y_2), \mu_2(y_1), \mu_2(y_2), \\
&\qquad \mu_{1,2}(y_1,y_1), \mu_{1,2}(y_1,y_2), \mu_{1,2}(y_2,y_1), \mu_{1,2}(y_2,y_2))\} \\
&= \{(1,0,0,1,\ 0,1,0,0),\ (0,1,1,0,\ 0,0,1,0), \\
&\qquad (0,1,0,1,\ 0,0,0,1),\ (\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2},\ \frac{1}{2},0,0,\frac{1}{2})\}.
\end{aligned}$$

Therefore, although both polytopes have only integral vertices, their intersection has fractional ones. Note that the restriction of the intersection to the set of integral vertices still remains the exact set we are interested in: the

subset of vertices of the marginal polytope satisfying $\mu_1(y_1) + \mu_2(y_1) \leq 1$.

In the above example, the simplified construction is qualitatively the same as the intersection of the connected subgraph polytope with the LP MAP-MRF relaxation *local polytope*[217]. Therefore, it is insightful in a number of ways.

First, having tight relaxations for both the connected subgraph polytope and the marginal polytope *does not guarantee* a tight relaxation for the convex hull of the integral vertices of their intersection.

Second, restricted to the set of integral solutions, the construction is exact. However, optimizing over only the integral solutions of the intersection is intractable, whereas optimizing over the intersection of two polytopes remains tractable if optimizing over the individual polytopes is tractable. To intersect polytopes can therefore be thought of as tractable relaxation to the intersection of their individual integral vertices: the new vertex set is a superset of the intersection of the individual polytopes' vertex sets.

In summary, intersecting polytopes weakens the overall relaxation. But in order to put this result into perspective, note the following three points.

First, we never had a tight relaxation to start with. For general pairwise potentials optimizing over the exact marginal polytope is NP-hard[218], so the LP relaxation is used. Optimizing over the exact subgraph polytope is NP-hard, so a relaxation is used. In order to remain tractable, both sets are relaxations and individually have fractional vertices. Whether the additional fractional vertices caused by intersection are an issue that has to be settled empirically, as shown in Figure 51(f).

Second, in general, finding inequalities which cut off fractional vertices of the intersection of two polytopes is hard, see Balas and also Wolsey[219].

Third, as observed by Finley and Joachims[220], structured learning of parameters in linear relaxations can "learn to avoid fractional solutions", as these always have a strictly positive loss.

[217] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, November 2005

[218] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, November 2005

[219] Egon Balas. Projection, lifting and extended formulation in integer and combinatorial optimization. *Annals of Operations Research*, (140):125–161, 2005; and Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998

[220] Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In *ICML*, 2008

From Polytopes to Potentials

We now transform the connected subgraph polytope into a potential function of a random field. Let $\mu^j(\boldsymbol{y}) = [\mu_1(y_j), \ldots, \mu_{|V|}(y_j)]^\top \in \mathbb{R}^{|V|}$ be the set of variables in the LP relaxation (59) indicating assignment to class j over all vertices. One way to enforce connectivity in the LP solution for the vertices assigned to the j'th class is to define the following *hard connectivity potential* function.

$$\psi_V^{\text{hard(j)}}(\boldsymbol{y}) = \begin{cases} 0 & \mu^j(\boldsymbol{y}) \in Z \\ \infty & \text{otherwise} \end{cases} \tag{70}$$

This potential function can be incorporated by adding the respective constraints (66) to the LP relaxation (59). Alternatively we can define a *soft connectivity potential* by defining a feature function measuring the violation of connectivity. We define $\psi_V^{\text{soft(j)}}(\boldsymbol{y}; \boldsymbol{w}) = w_{\text{soft(j)}} \phi^{\text{conn(j)}}(\boldsymbol{y})$ where $\phi^{\text{conn(j)}} \geq 0$

Algorithm 7 MAP-MRF LP Cutting Plane Method

$(\boldsymbol{y}, B) = \text{LPCUTTINGPLANE}(\boldsymbol{x}, \boldsymbol{w})$
Input:
  Sample $\boldsymbol{x} \in \mathcal{X}$, weight vector $\boldsymbol{w} \in \mathbb{R}^d$
Output:
  Approximate MAP-MRF labeling $\boldsymbol{y}^* \in \mathcal{Y}$
  Lower bound on MAP energy $B \in \mathbb{R}$
Algorithm:
$C \leftarrow \mathbb{R}^{\dim(\mathcal{Y})}$, $B \leftarrow -\infty$ {Initially: no cutting planes}
loop
  $\boldsymbol{y}^* \leftarrow \operatorname{argmin}_{\boldsymbol{y} \in \mathcal{Y}, \boldsymbol{y} \in C} E(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{w})$
  $\boldsymbol{c} \leftarrow$ most violating constraint (66) with $\boldsymbol{c}^\top \mu^j(\boldsymbol{y}^*) > 1$
  if no $\boldsymbol{c}^\top \mu^j(\boldsymbol{y}) > 1$ can be found **then**
    break
  end if
  $C \leftarrow C \cap \{\boldsymbol{y} : \boldsymbol{c}^\top \mu^j(\boldsymbol{y}) \leq 1\}$
end loop
$B \leftarrow E(\boldsymbol{y}^*; \boldsymbol{x}, \boldsymbol{w})$

measures the violation of connectivity:

$$\phi^{\text{conn(j)}}(\boldsymbol{y}) = \begin{cases} 0 & \mu^j \in Z \\ \max_{\boldsymbol{d} \in D}\{\boldsymbol{d}^\top \mu^j(\boldsymbol{y}) - 1\} & \text{otherwise} \end{cases},$$

where D is the set of coefficient vectors of the inequalities (66). We can calculate $\max_{\boldsymbol{d} \in D}\{\boldsymbol{d}^\top \mu^j(\boldsymbol{y}) - 1\}$ efficiently by means of Theorem 7. This potential function can be realized by introducing constraints into the LP relaxation as for $\psi^{\text{hard(j)}}$ but also adding one global non-negative slack variable lower bounded by $\phi^{\text{conn(j)}}$ for all $\boldsymbol{y} \in \mathcal{Y}$ and having an objective coefficient of $w_{\text{soft(j)}}$.

LP MAP-MRF with ψ_V

Algorithm 1 iteratively solves the MAP-MRF LP relaxation (59). After each iteration (70) is checked and if the labeling is connected, the algorithm terminates. In the case of an unconnected segmentation, a violated constraint is found and added to the master LP (59).

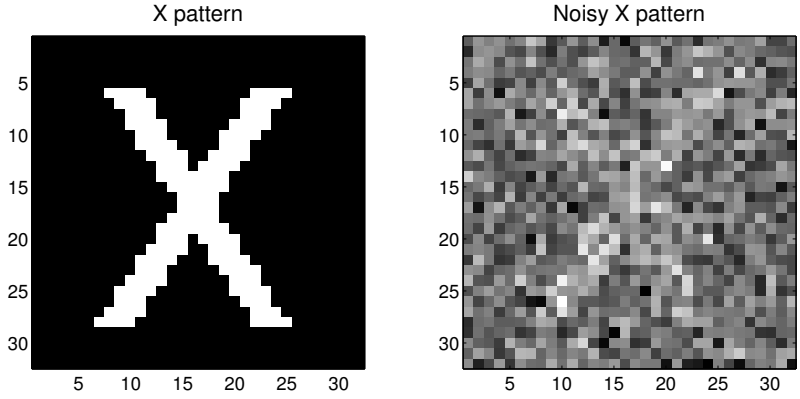
We now validate our connectedness potential on two tasks, i) a MRF denoising problem, and ii) object segmentation by learned CRFs.

Experiment: Denoising

We consider a standard denoising problem[221]. The 32x32 pixel pattern shown in Figure 50(a) is corrupted with additive Gaussian noise, as shown in Figure 50(b). The pattern should be recovered by means of solving a binary MRF.

[221] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2): 147–159, 2004

Figure 50: Denoising experiment.



(a) Pattern "X" to be recognized. (b) Noisy node potential, $\sigma = 0.9$.

We use a 4-neighborhood graph defined on the pixels, and the node potentials are derived from ground truth labeling as

$$\psi_i(\text{"FG"}) = \begin{cases} -1 + \mathcal{N}(0, \sigma) & \text{if } i \text{ is true foreground} \\ 0 & \text{otherwise} \end{cases}$$

$$\psi_i(\text{"BG"}) = \begin{cases} -1 + \mathcal{N}(0, \sigma) & \text{if } i \text{ is true background} \\ 0 & \text{otherwise} \end{cases}$$

[222] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2): 147–159, 2004

The edge potentials are regular[222] and chosen as Potts

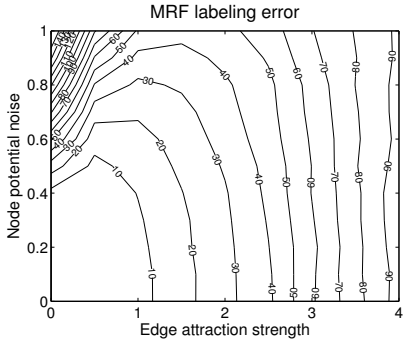
$$\psi_{i,j}(y_i, y_j) = |\mathcal{N}(0, k/\sqrt{d})| I(y_i \neq y_j),$$

where $d = 4$ is the average degree of our vertices. The parameters are varied over $\sigma \in \{0, 0.1, \ldots, 1.0\}$, $k \in \{0, 0.5, \ldots, 4\}$ and each run is repeated 30 times. For each of the 30 runs, the potentials are sampled once and we derive three solutions, i) "MRF", the solution to standard binary MRF, ii) "MRFcomp", the largest connected component of the MRF, iii) "CMRF", a binary MRF with additional hard-connectivity potential (70) on the foreground plane.

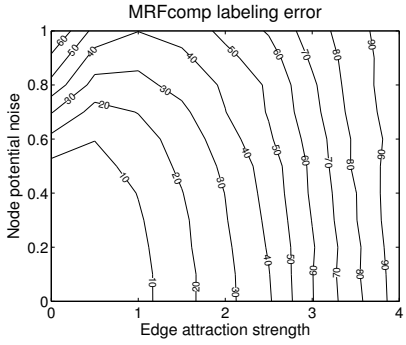
The results are shown in Figures 51(a) to (f). They show the connected MRF averaged absolute error over the parameter plane and the relative errors to the standard MRF and component heuristic.

The advantage of the connectedness constraint over a standard MRF can be seen by looking at the relative errors in Figure 51(d). For almost all parameter regimes the error of the MRF is higher (positive values in the plot). Also, from Figure 51(e) it can be seen that the connectedness constraint outperforms the largest-connected-component heuristic except when very weak edge potentials are used (upper left corner). Typical examples are shown in Figure 52 and 53.

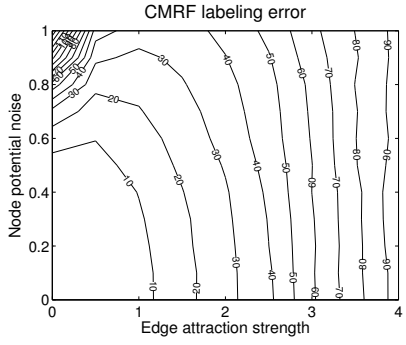
Regarding solution integrality, because we use relaxations for both the marginal polytope (the LP relaxation), and the connected subgraph polytope



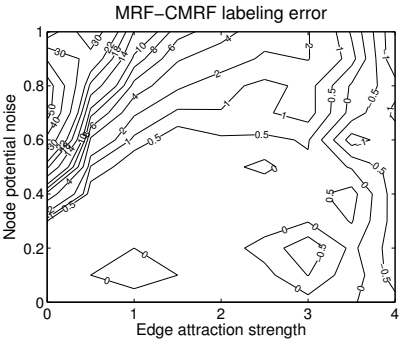
(a) MRF labeling error.



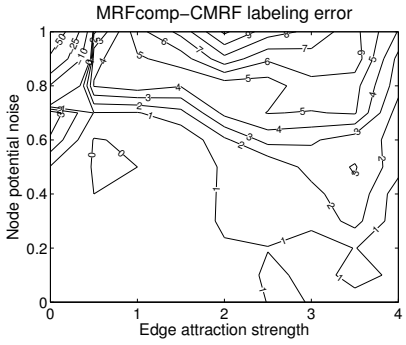
(b) MRFcomp labeling error.



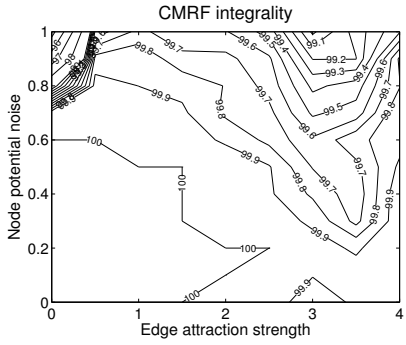
(c) Connected MRF labeling error.



(d) Error difference MRF-CMRF.



(e) Error diff. MRFcomp-CMRF.



(f) Mean solution integrality of the MRF with hard connectivity potential.

Figure 51: Denoising experiment results.

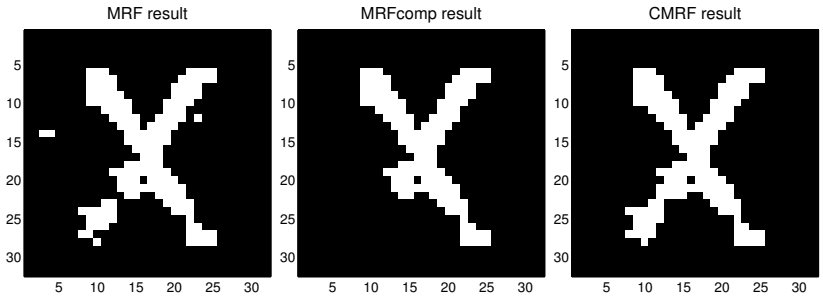


Figure 52: MRF/MRFcomp/CMRF results, with energies $E = -985.61$, $E = -974.16$, $E = -984.21$, and errors 36, 46, 28, respectively. The connectivity constraint solution CMRF is a substantial improvement over the solutions of MRF and MRFcomp.

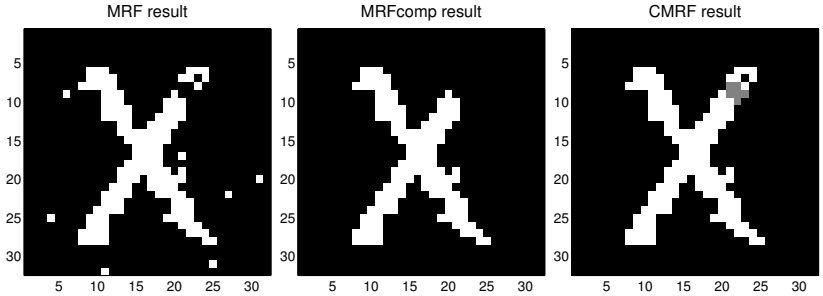


Figure 53: MRF/MRFcomp/CMRF results, with energies $E = -980.13$, $E = -974.03$, $E = -976.83$, and errors 34, 34, 24, respectively. Note although the CMRF solution becomes fractional, it is a substantial improvement over the MRF and MRFcomp results.

(the relaxation described by (66)), it is not a priori clear that the solution obtained will be integral. Only if it is, we have a solution to the true, unrelaxed

problem. If it is fractional, the solution is still optimal in the relaxation, but outside the true feasible set.

In Figure 51(f) we show the integrality, i.e., the fraction of variables which are integral.

We see that our approach is very effective: for medium noise and edge interactions, the solution is always integral, whereas even when there is more noise and edge interaction, very few variables — less than 0.5% for most configurations — become fractional.

The problems defined by the marginal polytope and the connected subgraph polytope are both NP-hard. Hence, it is likely that no polynomial-time approach can provide the guaranteed optimum. In theory, a logical step within our approach would be to prove properties about the fractional solutions, for example that they satisfy half-integrality or can be rounded with optimality guarantee to obtain a polynomial-time approximation algorithm. In practice, the approach already works very well.

Experiment: Learning Object Segmentation

Connectivity is a strong global prior for object segmentation. In this experiment we use the connectivity assumption to segment out objects from the background in the PASCAL VOC 2008 data set[223]. The data set is known to be particularly challenging as the images contain objects of 20 different classes with a lot of variability in lighting, viewpoint, size and positioning of the objects.

[223] Mark Everingham, Luc Van Gool, Christopher K.I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2008 Results. http://www.pascal-network.org/challenges/VOC/voc2008/

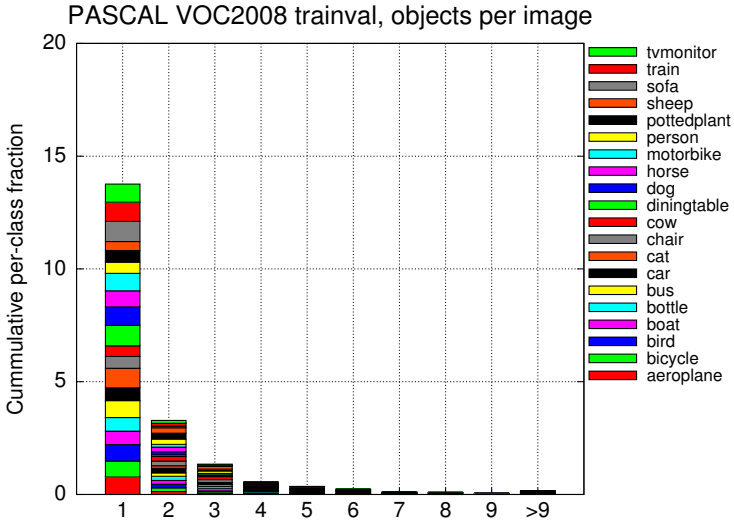


Figure 54: Number of objects of individual classes per image in the PASCAL VOC 2008 trainval data set for the object detection task.

We first look at a simple statistic of the training and validation set for the *detection* task: How many objects of each individual class are present on an image? Figure 54 shows the number of objects of individual classes per image

in the PASCAL VOC 2008 trainval data set.

The statistics confirm that if an object is present on an image, in 70% of the cases there is no other object of the same class on the image. For some classes, like `aeroplane`, `cat`, and `diningtable` this is more often the case than for classes like `bottle`, `chair`, `person` and `sheep`.

THE EXPERIMENTAL SETUP is as follows. In our setting, we let $\boldsymbol{x} = (V, E)$ be the graph resulting from a superpixel segmentation[224] of an image, where each $i \in V$ is a superpixel. The superpixel segmentation is obtained using the method[225] of Mori[226], where we use ≈ 100 superpixels. Example segmentations are shown on the left side of Figures 55 to 59.

[224] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, 2003

[225] `http://cs.sfu.ca/~mori/research/superpixels/`

[226] Greg Mori. Guiding model search using segmentation. In *ICCV*, 2005

USING SUPERPIXELS HAS THREE ADVANTAGES, i) the information in each superpixel is more discriminative because all image information in the region can be used to describe it, ii) the complexity of the inference is drastically reduced with only a negligible approximation error, and iii) the notion of connectivity becomes more meaningful if larger, equal-sized parts are considered.

Each superpixel becomes a vertex in the graph. An edge joins two vertices if the superpixels are adjacent in the image. Therefore connectivity in the graph implies connectivity of the segmentation.

For each image, we extract per image an average of 38,000 SURF features[227] at random positions in scale space as well as at interest operator responses and assign each feature to the superpixel which contains the center pixel of the feature. For each vertex, a bag-of-words histogram $x_i \in \mathbb{R}^H$ is created by nearest-neighbor quantizing the features associated to the superpixel in a codebook of 500 words ($H = 500$), created by k-means clustering[228] on a large random sample of features from the training set.

[227] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc J. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008

[228] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*, volume November. John Wily & Sons, Inc., New York, second edition, 2000. ISBN 0471056693

We treat each of the twenty classes separately as a binary problem. That is, for each image showing an object of the class, a class-vs-background labeling is sought. Hence each vertex i in the graph has a label vector $y_i \in \{0,1\} \times \{0,1\}$. We report the average intersection-union metric, defined as $\frac{TP}{TP+FP+FN}$ ratio, where TP, FP, FN are true positives, false positives and false negatives, respectively, per pixel labeling for the object class[229]. Because the VOC2008 segmentation `trainval` set includes only 1023 images for which ground truth is available, with some classes having as few as 44 positive images (only 19 for `train` alone), we use a three-fold cross validation estimate on the `trainval` set.

[229] Mark Everingham, Luc Van Gool, Christopher K.I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2008 Results. http://www.pascal-network.org/challenges/VOC/voc2008/

For all CRF variants we will describe later, we use the following feature functions.

- Node features, $\phi_i^{(1)}(y_i, \boldsymbol{x}) = \mathrm{vec}(x_i y_i^\top)$.

  Thus the output of $\phi_i^{(1)}(y_i, \boldsymbol{x})$ is a $(H, 2)$-matrix of two weighted replications of the node histogram x_i. The matrix is stacked columnwise.

- Edge features $\phi_{i,j}^{(2)}(y_i, y_j, \boldsymbol{x}) = \mathrm{vec}_\Delta(y_i y_j^\top)$.

This is the upper-triangular part including diagonal of the outer product $y_i y_j^\top$. By making this feature available, the CRF can learn the weights for the inter-class and intra-class Potts potentials separately.

We test three CRFs, i) a CRF with these feature functions, ii) the same CRF with $\psi_V^{\text{hard(class)}}$, and iii) the same CRF with $\psi_V^{\text{soft(class)}}$.

Learning the parameters

For learning the parameters $\boldsymbol{w}$ of the model, we use the structured output support vector machine framework[230], recently also used in computer vision[231]. As discussed in the previous chapter, it minimizes the following regularized risk function.

$$\min_{\boldsymbol{w}} \quad \|\boldsymbol{w}\|^2 + \frac{C}{\ell} \sum_{n=1}^{\ell} \max_{\boldsymbol{y} \in \mathcal{Y}} \left(\Delta(\boldsymbol{y}_n, \boldsymbol{y}) + E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) - E(\boldsymbol{y}; \boldsymbol{x}_n, \boldsymbol{w}) \right), \qquad (71)$$

where $(\boldsymbol{x}_n, \boldsymbol{y}_n)_{n=1,\ldots,\ell}$ are the given training samples and $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a compatibility function which has a high value if two segmentations are different and a low value if they are very similar. More precisely, we define

$$\Delta(\boldsymbol{y}^1, \boldsymbol{y}^2) = \sum_{i \in V} \frac{r_i}{\sum_{j \in V} r_j} \left(y_i^1 + y_i^2 - 2 y_i^1 y_i^2 \right),$$

where r_i is the size in pixels of the region i in the superpixel segmentation. Note that this definition is, i) symmetric, $\Delta(\boldsymbol{y}^1, \boldsymbol{y}^2) = \Delta(\boldsymbol{y}^2, \boldsymbol{y}^1)$, ii) zero-based, $\Delta(\boldsymbol{y}, \boldsymbol{y}) = 0$, and non-negative, iii) corresponds to the Hamming loss if all elements are binary, and iv) decomposes linearly over the individual elements if one of $\boldsymbol{y}^1$, $\boldsymbol{y}^2$ is constant.

Because of the last point it is easy to incorporate into the MRF inference procedure by means of a bias on the node potentials[232]. We train with $C \in \{.00001, .0001, \ldots, 10, 100\}$ and report the highest achieved performance of each model.

The objective (71) is convex, but non-differentiable. We use the StructuredSVM algorithm discussed in the last chapter, iteratively solving a quadratic program[233].

For solving the separation problem one is given a current parameter vector $\boldsymbol{w}$. Then for each sample $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ one needs to determine whether there exists a violated constraint of the form (39). To answer this question, for a given n, we rewrite the set of constraints as

$$\xi_n \geq \Delta(\boldsymbol{y}_n, \boldsymbol{y}) + E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) - E(\boldsymbol{y}; \boldsymbol{x}_n, \boldsymbol{w}), \quad \boldsymbol{y} \in \mathcal{Y}. \qquad (72)$$

By maximizing the right hand side of (72) over all possible $\boldsymbol{y} \in \mathcal{Y}$ we can find the most violating constraint. Therefore, we attempt to solve

$$\max_{\boldsymbol{y} \in \mathcal{Y}} \left(\Delta(\boldsymbol{y}_n, \boldsymbol{y}) + E(\boldsymbol{y}_n; \boldsymbol{x}_n, \boldsymbol{w}) - E(\boldsymbol{y}; \boldsymbol{x}_n, \boldsymbol{w}) \right).$$

[230] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, September 2005

[231] Matthew B. Blaschko and Christoph H. Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008; Yunpeng Li and Daniel Huttenlocher. Learning for stereo vision using the structured support vector machine. In *CVPR*, 2008; and Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning CRFs using graph cuts. In *ECCV*, 2008

[232] Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In *ICML*, 2008; and Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning CRFs using graph cuts. In *ECCV*, 2008

[233] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, September 2005

Figure 55: Image/CRF/CRF+conn. Case where connectedness helps: the local evidence is scattered, enforcing connectedness (right) helps.

The last term is constant and $\Delta(\boldsymbol{y}_n, \boldsymbol{y})$ can be incorporated into $E(\boldsymbol{y}; \boldsymbol{x}_n, \boldsymbol{w})$ by adjusting the node potentials. Finding the most violated constraint has been converted to a problem of the same form as the original MAP-inference problem. Therefore Algorithm LPCuttingPlane can be used to find the maximizer $\boldsymbol{y}_n^*$. It defines a new constraint and by iterating between generating constraints and solving the QP we can obtain successively better parameter vectors $\boldsymbol{w}$.

Finley and Joachims[234] have shown that if the inference in the learning problem is hard, then *approximately solving* this hard problem can lead to classification functions which do not generalize well. Instead, it is preferable to *solve exactly* a relaxation to the original inference problem. This is precisely what we are doing, because the intersection of (66) with the MAP-MRF LP local polytope defines an exactly solvable relaxation.

[234] Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In *ICML*, 2008

Results

Table 8 shows for each class the averaged intersection-union scores of the three different methods.

Method	aerop.	bicyc.	bird	boat	bottle	bus	car	cat	chair	cow
CRF	0.355	0.087	0.189	0.261	0.138	0.383	0.194	0.278	0.084	0.225
hard	**0.380**	0.091	0.202	0.275	0.115	0.391	0.185	**0.311**	**0.121**	0.236
soft	0.341	0.090	0.176	**0.288**	0.130	**0.406**	0.165	0.283	0.101	**0.270**
	dtable	dog	horse	mbike	person	plant	sheep	sofa	train	tv
CRF	0.279	0.245	**0.232**	0.239	0.188	0.088	**0.298**	0.214	0.419	0.158
hard	0.269	0.244	0.209	0.268	0.194	0.075	0.249	0.200	0.393	0.152
soft	0.294	0.220	0.194	**0.273**	0.184	0.074	0.277	0.209	0.419	0.151

Table 8: Results of the VOC2008 segmentation experiment. Marked bold are the cases where a method outperforms the others.

For most classes the connected CRF models outperform the baseline CRF. This is especially true for classes such as aeroplane and cat, whose images usually contain only one large object. In contrast, classes such as bottle and sheep often have more than one object in an image. This is a violation of our connectedness assumption and in this case the CRF model outperforms the connected ones. We also see that in some cases the extra flexibility of the soft connectedness over the hard connectedness prior pays off: for the boat, bus, cow and motorbike classes, the ability to weight the connectivity strength

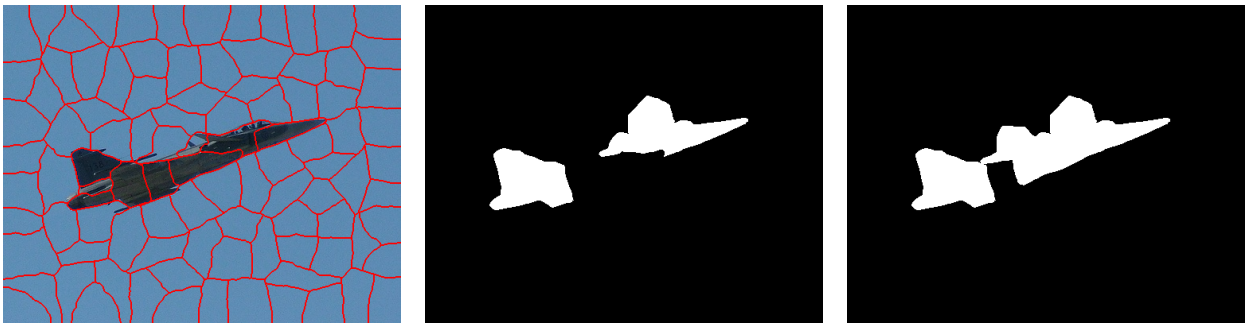


Figure 56: Image/CRF/CRF+conn. Another case where connectedness helps.

Figure 57: Image/CRF/CRF+conn. Connectedness can remove clutter: local evidence (edges on the runway) is overridden.

Figure 58: Image/CRF/CRF+conn. Another case where an erroneous detection is removed due to the connectivity constraint.

versus the other potentials is useful in improving over both the baseline CRF and the hard connected CRF.

The typical behavior of the hard-connectedness CRF on test images is shown in Figures 55 to 59 for the aeroplane class. In the first two segmentations, connectedness helps by completing a discontinuous segmentation and by removing clutter. Figure 59 shows a hopeless case: if the CRF segmentation is that wrong, connectedness cannot help.

Figure 59: Image/CRF/CRF+conn. Failure case: the CRF segmentation is bad (middle) connectedness does not help (right).

Figure 60: Image/CRF/CRF+conn. Failure case due to locally non-tight relaxation: there are two connected components in the CRF+conn solution. This is because the node variable associated to the foreground layer which corresponds to the connecting superpixel has a fractional value $\frac{1}{2}$. For the binary visualization image we round down fractional values.

Conclusions and Outlook

We have shown how the limitation of only considering local interactions in discrete random field models can be overcome in a principled way. We considered a hard global potential encoding whether a labeling is connected or not. We derived an efficient relaxation that can naturally be used with MAP-MRF LP relaxations.

Experimentally, we demonstrated that a connectedness potential reduces the segmentation error on both a synthetic denoising and real object segmentation task.

Clearly, other meaningful global potential functions could be devised by the method introduced in this paper. The principled use of polyhedral combinatorics opens a way to better model high-level vision tasks with random field models. Another direction of future work is to see if the addition of complicated primal constraints like (66) can be accommodated into recent efficient dual LP MAP solvers [235].

In a wider sense, most computer vision research into Markov random field models has focused only on low-order interactions in sparsely connected graphs. Although even for this setting the general case is already NP-hard,

[235] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *NIPS*, 2007; Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*. IEEE, 2007; Mudigonda Pawan Kumar and Philip Torr. Efficiently solving convex relaxations for MAP estimation. In *ICML*, 2008; and David Sontag, Talya Meltzer, Amir Globerson, Tommi Jaakkola, and Yair Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, 2008

the conditional independence embodied in the Markov properties allowed the development of tractable inference procedures.

But there is additional structure possible which does not fit well in this standard setting: the global potential function we considered in this paper does not have a factorizable structure. Still, efficient approximate inference is possible by exploiting the *combinatorial structure*. In this work we have achieved this by combining the LP MAP-MRF relaxation with a suitable polytope derived from the global potential function. Whether there are more efficient ways to achieve the same effect is an open question.

The software used in this chapter is made available as open-source at `http://www.kyb.mpg.de/bs/people/nowozin/cmrf/`.

Solution Stability in Linear Programming Relaxations

> Far better an approximate answer to the right question, which is often vague, than an exact answer to the wrong question, which can always be made precise.
>
> John Wilder Tukey

In the previous two chapters we have discussed inference and learning problems. Two problems, the MAP-MRF problem and the optimization over the connected subgraph polytope have led to hard combinatorial optimization problems. For both problems we have used the technique of linear programming relaxations to construct a tractable approximation to the true problem.

In this chapter we take a broader view at combinatorial optimization problems and their linear programming relaxations. In particular, we are interested in *solution stability*, that is, the behavior of the optimal solution when the input data is perturbed. We believe this is an important direction for the part of structured output learning research that abandoned probabilistic models in order to gain tractable learning procedures. The original probabilistic models offered natural concepts to analyze the prediction in form of a posterior distribution or statistics thereof, such as marginal probabilities, higher-order moments or generated samples. In modern non-probabilistic structured prediction models a posterior might no longer be available and other *efficiently computable* properties of the prediction become relevant. The restricted concept of per-instance solution stability in this chapter is a first step in this direction.

The main result brought forth in this chapter is a new method to quantify the per-instance solution stability of a large class of combinatorial optimization problems arising in machine learning. As a practical example we apply the method to a family of clustering problems. Although not directly related to computer vision, the insights gained from analyzing the stability of these problems are of general form and thus applicable in many of the combinatorial problems of interest to the computer vision community.

The proposed method is not only general but comes with rigorous theoretical guarantees. To this end we prove that when a *relaxation* is used to solve the original optimization problem, then the solution stability calculated by our method is conservative, that is, it never overestimates the solution stability of

the true, unrelaxed problem.

General Problem

Several fundamental problems in machine learning can be expressed as the combinatorial optimization task

$$z^* := \operatorname*{argmin}_{z \in \mathcal{B}} \quad w^\top z, \tag{73}$$

where $\mathcal{B} \subseteq \{0,1\}^n$ is a specific set of indicator vectors of length n.

For example, when posed as integer linear program, the MAP-MRF inference problem discussed in the previous chapters naturally falls in this category. Another example are clustering problems, which can be posed in the form of (73) by means of binary variables indicating whether two samples are in the same cluster.

The formulation (73) is general and powerful. However, depending on the problem parameter w, an optimal solution z^* might not be unique, or it might be *unstable*, i.e., a small perturbation to w will make another $z \neq z^*$ optimal.

To ensure a reliable and principled use of (73) it is important to analyze the *stability* of z^*, especially because the lack of stability can indicate serious modeling problems.

In machine learning, the value of w usually depends on the data, and possibly on a modeling parameter. Both these dependencies often introduce uncertainty. Real data commonly originates from noisy measurements or is assumed to be sampled from an underlying distribution. In these cases, data values correspond to estimates that indicate a small range of numerical values rather than fixed, certain numbers.

The data induces one w and thus one optimal solution, e.g., clustering, z_1^*. If a slight perturbation to the data completely changes the solution to z_2^*, then z_1^* must be treated with care. The preference of z_1^* over z_2^* could merely be due to noise. To account for uncertainty in the data, one commonly strives for stable solutions with respect to perturbations or re-sampling.

Modeling parameters are another source of uncertainty, for their "correct" value is usually unknown, and thus estimated or heuristically set. A stability analysis gives insight into how the parameter influences the solution on the given instance of data. Here too stability can indicate reliability.

In addition, a stability analysis can reveal characteristics of the data itself, as we illustrate in two examples. We can compute the path of all solutions as the perturbation increases systematically. Depending on the perturbation, this path may indicate structural information or help to analyze a modeling parameter.

If the perturbation is set accordingly, the comparison of these solutions may indicate structural information beyond a single solution z^*. Similarly, with

an appropriate perturbation, the solution path helps to analyze a modeling parameter.

The fact that a small perturbation changes the solution a lot suggests that the data has more structure than shown by one solution. We can compute the path of all solutions as the perturbation increases systematically. The change of solutions indicates structure in the data, information beyond a single solution z^*.

Another example where stability is important is when w originates from a parametric model, such as transforming some measured data X by means of a parametrized function $w = f(X; \tau)$, where τ are some parameters. In this case, the solution z^* obtained for a particular w and τ depends on τ in a non-trivial way and analyzing the stability of z^* can give insight into how it is influenced by τ.

We present a *new general* method to quantify the solution stability of Problem (73) and compute the solution path along a parametric perturbation. In particular, we overcome the inability of existing approaches to handle a basic characteristic of linear programming relaxations to (73), namely, that only few constraints are known at a time. Owing to our formulation, two close variants of the same algorithm will suffice to solve both the nominal Problem (73) and the stability analysis.

A running example for (73) makes the general discussion concrete: the Graph Partitioning Problem (GPP), which unifies a number of popular clustering tasks. Our stability analysis for GPP hence yields a new method for a more thoughtful analysis of these clusterings.

Graph Partitioning Problem and Relaxation

In many unsupervised learning problems, we only have information about pairwise relations of objects, and not about features of individuals. Examples include co-authorship and citations, or protein interactions. In this case, exemplar- or *centroid*-based approaches are inapplicable, and we directly use the graph of relations or similarities. Clustering corresponds to finding an appropriate partitioning of this graph.

A natural formalization of clustering with only pairwise information is the graph partitioning problem, defined as follows.

Problem 4 (Graph Partitioning Problem (GPP)) *Given an undirected, connected, simple graph $G = (V, E)$, and edge weights $w : E \to \mathbb{R}$, partition the vertex set into nonempty subsets so that the total weight of the edges with end points in different subsets is minimized.*

Note that, in contrast to common graph cut problems such as min-cut or normalized cut, GPP does *not* pre-specify the number of clusters. To describe a partitioning of G, we will use indicator variables $z_{i,j} \in \{0, 1\}$ for each edge

$(i,j) \in E$, where $z_{i,j} = 1$ if i and j are in different partitions, and $z_{i,j} = 0$ otherwise. Figure 61 shows an example. Let $\mathcal{Z}(G) = \{\boldsymbol{z} \in \{0,1\}^{|E|} \mid \exists \pi : V \to \mathbb{N} : \forall (i,j) \in E : z_{i,j} = [\![\pi(i) \neq \pi(j)]\!]\}$ be the set of all possible partitionings, where $[\![\cdot]\!]$ is the indicator function.

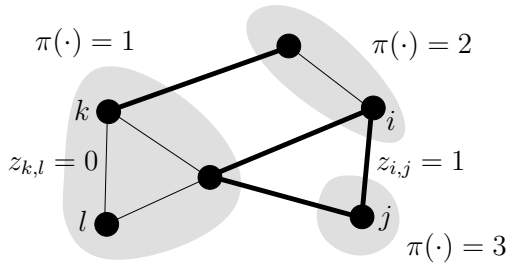


Figure 61: An example partitioning $\boldsymbol{z}$. Bold edges have $z_{i,j} = 1$, while others have $z_{k,l} = 0$.

Using this notation, we can formalize GPP as a special case of (73) with $\mathcal{B} = \mathcal{Z}(G)$, minimizing a linear function:

$$\begin{aligned} \min_{\boldsymbol{z}} \quad & \sum_{(i,j)\in E} w(i,j)\, z_{i,j} \\ \text{sb.t.} \quad & \boldsymbol{z} \in \mathcal{Z}(G). \end{aligned} \tag{74}$$

Problem (74) encompasses a wide range of clustering problems if we set the weights $\boldsymbol{w}$ accordingly. Table 9 summarizes the form of the coefficients $\boldsymbol{w}$ for a number of popular clustering problems, and also for two biases: one favoring clusters of equal sizes, and one penalizing large clusters.

The information contained in a single weight $w(i,j)$ is often enough to make local decisions about i and j being in the same cluster. Global agreement of these local decisions is enforced by $\boldsymbol{z}$ being a valid partitioning. Exactly this global constraint $\boldsymbol{z} \in \mathcal{Z}(G)$ makes GPP difficult to solve.

In general, Problem (73) is an integer linear program (ILP) and NP-hard. A common approach to solving (73) is to use a linear *relaxation* of the constraint $\boldsymbol{z} \in \mathcal{B}$.

Linear Relaxations

In general, the point set $\mathcal{B} \subseteq \{0,1\}^n$ is finite but exponentially large in n and usually intractable.

It is known from combinatorial optimization[236] that relaxing the set $\mathcal{B}$ to its convex hull $\operatorname{conv}(\mathcal{B})$ will not change the minimizer $\boldsymbol{z}^*$ of (73). The set $\operatorname{conv}(\mathcal{B})$ is by construction a bounded polyhedron — a so-called *polytope* — and at least one minimizer of a linear function over a polytope is a vertex. Therefore, at least one optimal solution of the relaxation will be integral, that means it is in $\mathcal{B}$ and thus an optimal solution of the exact problem. Thus the objective of problem (73) can equivalently be solved over $\boldsymbol{z} \in \operatorname{conv}(\mathcal{B})$. For GPP, the convex hull $\operatorname{conv}(\mathcal{Z}(G))$ is the *multicut polytope*.

[236] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998

THE CONVEX HULL IS DEFINED in terms of vertices $z \in \{0,1\}^n$. We can alternatively describe it in terms of intersecting halfspaces[237], i.e., linear inequalities. The minimal set of such inequalities to characterize the polytope exactly is the set of all *facet-defining inequalities*. Knowing these inequalities, we can derive a linear program *equivalent* to (74).

But often only a subset of the facet-defining inequalities is known, some are difficult to check and all are too many to handle efficiently. Therefore, one commonly replaces $\text{conv}(\mathcal{B})$ by an approximation $\widehat{\mathcal{B}} \supseteq \text{conv}(\mathcal{B}) \supset \mathcal{B}$ represented by a tractable subset of the facet-defining inequalities.

We will use such relaxations to derive a method for quantifying the stability of the optimal solution z^* with respect to perturbations in w. In the next section we first introduce our notion of stability analysis and then show how to overcome the difficulties of existing approaches. In the subsequent section we provide details about solving the formulated problems. We continue by describing the general cutting-plane algorithm for both Problem (73) and the stability analysis problem. Finally, in the following section we provide algorithmic details for the graph partitioning problems by describing a relaxation of the multicut polytope that is tighter than previous approximations for the problems in Table 9. Finally, the experiments section demonstrate the applications and properties of our method.

[237] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998

Stability Analysis

We first detail our notion of stability and then develop our approach. The method is based on local polyhedral approximations to the feasible set of the combinatorial problem and efficiently identifies solution break points for parametric perturbations of w.

We perturb the weight vector $w \in \mathbb{R}^n$ by a vector $d \in \mathbb{R}^n$. The resulting weights are then $w'(\theta) = w + \theta d$ for a perturbation level θ. *Stability analysis* asks for the range of θ for which the optimal solution does not change, i.e., the *stability range*.

Definition 21 (Stability Range) *Let the feasible set $\mathcal{B} \subseteq \{0,1\}^n$, a weight vector $w \in \mathbb{R}^n$ and the optimal solution $z^* := \text{argmin}_{z \in \mathcal{B}} w^\top z$ be given. For a* perturbation vector *$d \in \mathbb{R}^n$ and modified weights $w'(\theta) = w + \theta d$, the* stability range *is the interval $[\rho_{d,-}, \rho_{d,+}] \in (\{-\infty, \infty\} \cup \mathbb{R})^2$ of θ values for which z^* is optimal for the perturbed problem $\min_{z \in \mathcal{B}} w'(\theta)^\top z$.*

The geometry of stability ranges in the polytope $\text{conv}(\mathcal{B})$ is illustrated in Figure 62.

Problem	Description	Weights
Correlation Clustering	Given pairwise positive and negative similarity ratings $v(i,j) \in \mathbb{R}$ for samples i, j, find a partitioning that agrees as much as possible with these ratings	$w(i,j) = v(i,j),\ \forall (i,j) \in E$
Clustering Aggregation, Consensus Clustering	Also known as *clustering ensemble* and *clustering combination*. Find a single clustering that agrees as much as possible with a given set of m clusterings	$w(i,j) = \frac{1}{m}\sum_{k=1}^{m}\left(1 - 2r_{i,j}^{k}\right),\ \forall (i,j) \in V \times V$, where $\mathbf{r}^k$ represents clustering k analogous to $\boldsymbol{z}$.
Modularity Clustering	Maximize *modularity*, i.e., the difference between the achieved and expected fraction of intra-cluster edges. Originally for unweighted graphs it is straightforward to extend to weighted graphs and so are the weights on the right.	$w(i,j) = \frac{1}{2\lvert E\rvert}\left(\eta_{i,j} - \frac{\deg(i)\deg(j)}{2\lvert E\rvert}\right)$, $\forall (i,j) \in V \times V$, with $\eta_{i,j} = [(i,j) \in E]$, and deg denoting the degree of a node.
Relative Performance Significance Clustering	Maximize the achieved versus expected performance, i.e., fraction of edges within clusters and of missing edges between clusters	$w(i,j) = \frac{1}{n(n-1)}\left(2\eta_{i,j} - \frac{\deg(i)\deg(j)}{\lvert E\rvert}\right)$, $\forall (i,j) \in V \times V$
Bias: Squared Differences of Cluster Sizes	The criterion $\lambda\sum_{k,l=1}^{K}(\lvert C_k\rvert - \lvert C_l\rvert)^2$ favors clusters of equal sizes.	$\Delta w(i,j) = -2\lambda,\ \forall (i,j) \in V \times V$
Bias: Squared Cluster Sizes	A penalty for large clusters is $\lambda\sum_{k=1}^{K}\lvert C_k\rvert^2 = \lambda\sum_{k=1}^{K}\sum_{i,j\in V} 2\lambda\lvert V\rvert^2 - \lambda\sum_{i,j\in V} z_{i,j}$.	$\Delta w(i,j) = -\lambda,\ \forall (i,j) \in V \times V$

Table 9: Graph partitioning formulations of clustering problems for a set of objects V or graph $G = (V, E)$, and $\lambda > 0$.

The polytope is lightly shaded and bounded by lines representing the inequalities that define $\text{conv}(\mathcal{B})$. We know that z^* is optimal for $w'(\theta) = w + \theta d$ for $\theta = 0$. The point z^* is a vertex of the polytope. Two of the inequalities are binding (satisfied with "="), indicated by two boundary lines touching z^*. The negative normal vectors of the inequalities span a cone (shaded dark). As long as $w'(\theta)$ lies in this cone, z^* is optimal. If $w'(\theta)$ leaves the cone, say for a large enough $\theta > 0$, then we can improve over z^* by sliding along an edge of the polytope to another vertex $z' \in \mathcal{B}$ whose associated cone now contains the new vector $w + \theta d$. Formally, if $w'(\theta)$ is outside the cone, then a descent direction at an obtuse angle to w will be in $\mathcal{B}$. Moving z along this direction improves the value $w'(\theta)^\top z$.

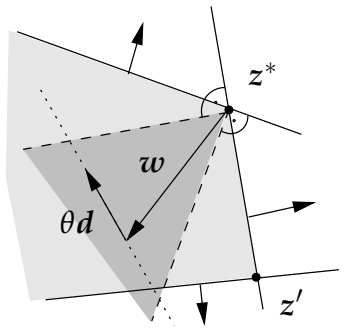


Figure 62: Geometry of Stability Analysis in a Polytope

We aim to find the value of θ where $w'(\theta)$ leaves the cone. If we know all inequalities defining the polytope, then we have an explicit description of the cone. Common approaches to compute stability ranges[238] rely on this knowledge and use the simplex *basis matrix*[239]. But the inequalities for the multicut polytope (and $\text{conv}(\mathcal{B})$ in general) are not explicitly known, since the polytope is defined as the convex hull of a complicated set. Even for relaxations $\widehat{\mathcal{B}}$, the set of constraints is too large to be handled as a whole, and just a few local constraints are known to the solver at a time. With such a small subset, the normal cone is only partially known and the basis matrix approach grossly underestimates the stability range, making it useless for anything but trivial instances.

In an online setting, Kılınc-Karzan et el.[240] use axis-aligned perturbations for the cost vector to obtain both an inner and outer polyhedral approximation to the *stability region*, the region where changes to w remain without effect. In contrast, we aim for an exact stability range for a given perturbation direction.

We will now present a method to compute stability ranges even without explicit knowledge of all constraints at all times. Owing to the formulation, two close variants of the same algorithm will suffice to solve both the original problem and the stability analysis. We will also relate the stability range obtained from relaxations to the stability range of the exact problem.

[238] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998; and Benjamin Jansen, J. J. de Jong, Cornelius Roos, and Tamás Terlaky. Sensitivity analysis in linear programming: Just be careful! *European Journal of Operational Research*, 101: 15–28, 1997

[239] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. 1997

[240] Fatma Kılınc-Karzan, Alejandro Toriello, Shabbir Ahmed, George Nemhauser, and Martin Savelsbergh. Approximating the stability region for binary mixed-integer programs. Technical report, Gatech, 2007

Linear Programming Stability Analysis using Separation Oracles

To avoid use of the basis matrix, we adopt a lesser known idea of Jansen et al.[241]: at optimality, the primal and dual optimal values are equal. Hence, z^* is optimal (and $w'(\theta)$ in the cone) as long as the optimal value of the perturbed dual equals $w'(\theta)^\top z^*$. Jansen et al. implement this idea in an LP derived from the dual of the original problem. With our implicit constraints, a dual-based approach is inapplicable. Therefore, we revert to the primal to construct a pair of *auxiliary linear programs* that search within the cone of *all possible* constraints defining $\text{conv}(\mathcal{B})$ around z^*.

The resulting formulation is similar to the original Problem (73), so we can

[241] Benjamin Jansen, J. J. de Jong, Cornelius Roos, and Tamás Terlaky. Sensitivity analysis in linear programming: Just be careful! *European Journal of Operational Research*, 101:15–28, 1997

use a similar solution procedure to take into account *all* implicit constraints — a point we elaborate in the next section. The following program yields the stability range for a given optimal solution $\boldsymbol{z}^*$ and perturbation direction $\boldsymbol{d}$.

$$\min_{\substack{\alpha\in\mathbb{R},\\ \boldsymbol{z}\in\mathbb{R}^n}} \quad \boldsymbol{w}^\top \boldsymbol{z} + \alpha \boldsymbol{w}^\top \boldsymbol{z}^* \tag{75}$$

$$\text{sb.t.} \quad (\frac{1}{\alpha}\boldsymbol{z}) \in \text{conv}(\mathcal{B}), \tag{76}$$

$$(\boldsymbol{d}^\top \boldsymbol{z}^*)\alpha - \boldsymbol{d}^\top \boldsymbol{z} = t : \gamma, \tag{77}$$

$$0 \leq \boldsymbol{z}_i \leq \alpha, \quad i = 1, \ldots, n. \tag{78}$$

where γ is the Lagrange multiplier of constraint (77). Constraint (76) is still linear, because it corresponds to $A(\frac{1}{\alpha}\boldsymbol{z}) \leq b$, or $A\boldsymbol{z} - \alpha b \leq 0$. From the variable upper bound constraints (78) it follows that $\alpha \geq 0$. Moreover, as $\text{conv}(\mathcal{B})$ is bounded, $\alpha > 0$.

The constant $t \in \{-1, 1\}$ in (77) determines whether we search for the left interval boundary $\rho_{\boldsymbol{d},-}$ or right interval boundary $\rho_{\boldsymbol{d},+}$ of the stability range $[\rho_{\boldsymbol{d},-}; \rho_{\boldsymbol{d},+}]$. At the optimum, the Lagrange multiplier γ of constraint (77) equals the boundary $\rho_{\boldsymbol{d},-}$ or $\rho_{\boldsymbol{d},+}$, depending on t.

Problem (75) is primal infeasible if and only if $\rho_{\boldsymbol{d},-} = -\infty$ for the left boundary ($t = -1$) or $\rho_{\boldsymbol{d},+} = \infty$ for the right boundary ($t = 1$).

The stability range could also be found approximately by probing various values of θ, similar to a *line search* in continuous optimization. In contrast, our method finds the breakpoint exactly by solving one optimization per search direction. It is guaranteed not to miss any breakpoints, a property that is hard to ensure for an iterative point-wise testing procedure.

The hardness of (75), like that of the nominal problem (73), depends on the tractability of $\text{conv}(\mathcal{B})$. That means we are forced to replace $\text{conv}(\mathcal{B})$ by a tractable approximation $\widehat{\mathcal{B}}$ to solve (75) efficiently. We will outline the relaxation for GPP in the next section.

But if we use $\widehat{\mathcal{B}}$, then the stability range only refers to the relaxation, i.e., for $\theta \notin [\rho_{\boldsymbol{d},-}, \rho_{\boldsymbol{d},+}]$, the optimal solution *of the relaxation* is guaranteed to change. Theorem 8 relates this stability range of the relaxation to the stability range of the exact problem.

Theorem 8 (Stability Inclusion) *Let $\boldsymbol{z}^*$ be the optimal solution of P1 for a given $\mathcal{B} \subseteq \{0,1\}^n$ and weights $\boldsymbol{w} \in \mathbb{R}^n$. For a perturbation $\boldsymbol{d} \in \mathbb{R}^n$, let $[\xi_{\boldsymbol{d},-}, \xi_{\boldsymbol{d},+}]$ be the true stability range for θ on conv$(\mathcal{B})$. If $\widehat{\mathcal{B}} \supseteq$ conv$(\mathcal{B})$ is a polyhedral relaxation of $\mathcal{B}$ using only facet-defining inequalities and if $\boldsymbol{z}^*$ is a vertex of $\widehat{\mathcal{B}}$, then the stability range $[\rho_{\boldsymbol{d},-}, \rho_{\boldsymbol{d},+}]$ on $\widehat{\mathcal{B}}$, i.e., for the relaxation $\min_{\boldsymbol{z}\in\widehat{\mathcal{B}}} \boldsymbol{w}^\top \boldsymbol{z}$, is included in the true range: $[\rho_{\boldsymbol{d},-}, \rho_{\boldsymbol{d},+}] \subseteq [\xi_{\boldsymbol{d},-}, \xi_{\boldsymbol{d},+}]$.*

Proof. Let $S_{\mathcal{B}}$ be the set of all constraints defining $\text{conv}(\mathcal{B})$ at $\boldsymbol{z}^*$ and $S_{\widehat{\mathcal{B}}}$ the set of all facet-defining constraints for $\widehat{\mathcal{B}}$ at $\boldsymbol{z}^*$. As $S_{\widehat{\mathcal{B}}}$ contains only facet-defining constraints, we have $S_{\widehat{\mathcal{B}}} \subseteq S_{\mathcal{B}}$. As a result, the cone spanned by the

negative constraint normals in $S_{\mathcal{B}}$ contains the cone spanned by the negative constraint normals in $S_{\widehat{\mathcal{B}}}$, and thus $[\rho_{\boldsymbol{d},-}, \rho_{\boldsymbol{d},+}] \subseteq [\xi_{\boldsymbol{d},-}, \xi_{\boldsymbol{d},+}]$ (recall Figure 62). □

Theorem 8 and problem (75) suggest that with a tight enough relaxation $\widehat{\mathcal{B}}$, we can efficiently compute a good approximation of the stability range by essentially the same algorithm that we apply to P1. Besides quantifying the robustness of a solution with respect to parametric perturbations, stability ranges help to recover an entire path of solutions, as we will show next.

Efficiently Tracing the Solution Path

As we increase the perturbation level θ, the optimal solution changes at certain breakpoints, the boundary points of the current stability range. That means we can trace the path of all optimal solutions along the weight path $w + \theta\boldsymbol{d}$ for $\theta \in [-\infty, \infty]$ by repeatedly jumping to the solution at the breakpoint and computing the stability range to find the next breakpoint.

The interpretation of the path of solutions depends on the choice of weights and the perturbation. For GPP, we will use weights derived from similarity matrices and obtain all clustering solutions on a path defined by shifting a linear bias term. This amounts to computing all clusterings between the extremes "one big cluster" and "each sample is its own cluster".

Implementation

In the previous sections, we formalized the nominal problem (73) and the stability analysis (75). Now we describe how to actually solve them. We first present a general algorithm and then specify details for GPP, mainly a suitable relaxation of the multicut polytope.

Cutting Plane Algorithm

The cutting plane method[242] shown in Algorithm 8 applies to both problem (73) and problem (75). Cutting plane algorithms provide a polynomial-time method to solve (appropriate) relaxations of ILPs.

[242] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998

The algorithm works with a small set of constraints that defines a loose relaxation $\mathcal{S}$ to the feasible set $\mathcal{B}$. It iteratively tightens $\mathcal{S}$ by means of *violated inequalities*. In Line 11, we solve the current LP relaxation. Having identified a minimizer z, we search for a violated inequality in the set of *all* constraints (Line 12). If we find a violated inequality, we add it to the current constraint set to reduce $\mathcal{S}$ (Line 16) and re-solve with the tightened relaxation. Otherwise, $z^* = z$ is optimal with all constraints.

The search for a violated inequality is the *separation oracle*. It depends on the particular set $\mathcal{B}$ of the combinatorial problem at hand and the description of the relaxation $\widehat{\mathcal{B}}$. The separation oracle is decisive for the runtime. If it runs in polynomial time, then the entire algorithm runs in polynomial time[243]. Hence,

[243] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998

Algorithm 8 Cutting Plane Algorithm

$(\boldsymbol{z}^*, f, \text{optimal}) = \textsc{CuttingPlane}(\mathcal{B}, \boldsymbol{w})$
Input:
  Set $\mathcal{B} \subseteq \{0,1\}^n$, weights $\boldsymbol{w} \in \mathbb{R}^n$
Output:
  Optimal solution $\boldsymbol{z}^* \in [0,1]^n$,
  Lower bound on the objective $f \in \mathbb{R}$,
  Optimality flag optimal $\in \{\text{true}, \text{false}\}$.
Algorithm:
$S \leftarrow [0,1]^n$ {Initial feasible set}
loop
  $\boldsymbol{z} \leftarrow \text{argmin}_{\boldsymbol{z} \in S}\, \boldsymbol{w}^\top \boldsymbol{z}$ {Solve LP relaxation}
  $S_{\text{violated}} \leftarrow \textsc{SeparateInequalities}(\mathcal{B}, \boldsymbol{z})$
  if no violated inequality found **then**
    break
  end if
  $S \leftarrow S \cap S_{\text{violated}}$ {Cut $\boldsymbol{z}$ from feasible set}
end loop
optimal $\leftarrow (\boldsymbol{z} \in \{0,1\}^n)$ {Integrality check}
$(f, \boldsymbol{z}^*) \leftarrow (\boldsymbol{w}^\top \boldsymbol{z}, \boldsymbol{z})$

polynomial-time separability is an important criterion for the relaxation $\widehat{\mathcal{B}}$. The next section addresses such a relaxation for GPP.

Relaxations of the Multicut Polytope

Solving GPP over $\mathcal{Z}(G)$ or $\text{conv}(\mathcal{Z}(G))$, the multicut polytope, is NP-hard[244]. To relax $\text{conv}(\mathcal{Z}(G))$ for an efficient optimization, we need facet-defining inequalities that describe an approximation to $\text{conv}(\mathcal{Z}(G))$ and are separable in polynomial time. In addition, the tighter the relaxation is, i.e., the more inequalities we use, the more accurate the stability analysis becomes.

The multicut polytope $\text{conv}(\mathcal{Z}(G))$ and variations have been researched in the late eighties and early nineties[245] and more recently[246]. We now discuss two subsets of the set of facet-defining inequalities for the multicut polytope that we use, *cycle inequalities* and *odd-wheel inequalities*. Both are polynomial-time separable, so we can tell efficiently whether a point satisfies all inequalities and if it does not, we can find a violated inequality.

Cycle inequalities are generalizations of the triangle inequality. Any valid graph partitioning $\boldsymbol{z}$ satisfies a *transitivity* relation: there is no all-zero path between any two adjacent vertices i, j that are in different subsets of the partition, i.e., for which $z_{i,j} = 1$. Formally, this property is described by the *cycle inequalities*[247] that are facet-defining for chord-free cycles $((i,j),p)$,

[244] Michel Marie Deza and Monique Laurent. *Geometry of cuts and metrics*, volume 15 of *Algorithms and Combinatorics*. 1997; and Sunil Chopra and M. R. Rao. The partition problem. *Math. Program*, 59:87–115, 1993

[245] Martin Grötschel and Yoshiko Wakabayashi. A cutting plane algorithm for a clustering problem. *Math. Prog.*, 45, 1989; Martin Grötschel and Yoshiko Wakabayashi. Facets of the clique partitioning polytope. *Math. Prog.*, 47: 367–387, 1990; Sunil Chopra and M. R. Rao. The partition problem. *Math. Program*, 59:87–115, 1993; Michel Marie Deza, Martin Grötschel, and Monique Laurent. Clique-web facets for multicut polytopes. *Mathematics of Operations Research*, 17(4):981–1000, 1992; and Michel Marie Deza and Monique Laurent. *Geometry of cuts and metrics*, volume 15 of *Algorithms and Combinatorics*. 1997

[246] Aykut Özsoy and Martine Labbé. Size constrained graph partitioning polytope. Technical Report 577, ULB, 2007

[247] Sunil Chopra and M. R. Rao. The partition problem. *Math. Program*, 59: 87–115, 1993

$p \in \text{Path}(i,j)$, where $\text{Path}(i,j)$ is the set of paths between i and j.

$$z_{i,j} \leq \sum_{(s,t)\in p} z_{s,t}, \qquad (i,j) \in E, \quad p \in \text{Path}(i,j). \tag{79}$$

In complete graphs, all cycles longer than three edges contain chords. Hence, for complete graphs we can simplify the cycle inequalities to a polynomial number of triangle inequalities, as done in Grötschel and Wakabayashi[248]; Chopra and Rao[249]; and Brandes et al.[250] The separation procedure for (79) is a simple series of shortest path problems, one for each edge and has been described by Chopra and Rao.

In the separation problem, for a given point z we can check whether all inequalities are satisfied as follows. Consider the original graph $G = (V, E)$ with an edge weighting $W'_z : E \to \mathbb{R}^+$ defined by $W'_z(e) = z_e$. For each edge $m \in E$, consider the adjacent vertices $(v_i, v_j) = \text{adj}(m)$. Clearly, the length of the shortest path between v_i and v_j in G with weights W'_z is upper bounded by z_m. Iff there exists a shorter path p, this corresponds to a violated constraint $z_m \leq \sum_{z_s \in p} z_s$. If there is no shorter path for all $m \in E$, then all inequalities are satisfied.

Previous LP relaxations for correlation and modularity clustering[251] limit their approximation of the multicut polytope to cycle inequalities only. We call these equivalent relaxations LP-C relaxation. Our experiments will show that the LP-C relaxation is not very tight, and additional *odd-wheel inequalities*[252] improve the approximation.

Odd-Wheel inequalities are another class of known facet-defining inequalities for the multicut polytope. Let a *q-wheel* be a connected subgraph $S = (V_s, E_s)$ with a central vertex $j \in V_s$ and a cycle of the q vertices in $C = V_s \setminus \{j\}$. For each $i \in C$ there exists an edge $(i,j) \in E_s$. An example 3-wheel is shown in Figure 63.

For every q-wheel, a valid partitioning z satisfies the inequality

$$\sum_{(s,t)\in E(C)} z_{s,t} - \sum_{i \in C} z_{i,j} \leq \lfloor \frac{1}{2} q \rfloor, \tag{80}$$

where $E(C)$ denotes the set of all edges in the outer cycle C. Deza et al.[253] prove that the odd-wheel inequalities (80) are facet-defining for every odd $q \geq 3$. These inequalities are polynomially separable. The odd-wheel inequalities are a special case of *clique-web inequalities* which are also facet-defining for the multicut polytope. Because the general clique-web inequalities are NP-hard to separate, we do not use them.

We now describe the separation procedure, as in Deza and Laurent[254]. Given a graph $G = (V, E)$, a solution z satisfying all cycle inequalities (79), the odd-wheel inequalities can be separated efficiently as follows:

1. For each vertex $v_j \in V$, perform the following:

[248] Martin Grötschel and Yoshiko Wakabayashi. A cutting plane algorithm for a clustering problem. *Math. Prog.*, 45, 1989

[249] Sunil Chopra and M. R. Rao. The partition problem. *Math. Program*, 59: 87–115, 1993

[250] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE TKDE*, 20(2):172–188, 2008

[251] Isabelle Warnesson. Applied linguistics: Optimization of semantic relations by data aggregation techniques. *Applied Stochastic Models and Data Analysis*, 1:121–141, 1985; D. Emanuel and A. Fiat. Correlation clustering – minimizing disagreements on arbitrary weighted graphs. In *Proceedings of the ESA*, 2003; Thomas Finley and Thorsten Joachims. Supervised clustering with support vector machines. In *ICML*, pages 217–224, 2005; Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theor. Comput. Sci*, 361(2-3):172–187, 2006; and Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE TKDE*, 20(2):172–188, 2008

[252] Michel Marie Deza, Martin Grötschel, and Monique Laurent. Clique-web facets for multicut polytopes. *Mathematics of Operations Research*, 17(4):981–1000, 1992; and Sunil Chopra and M. R. Rao. The partition problem. *Math. Program*, 59:87–115, 1993

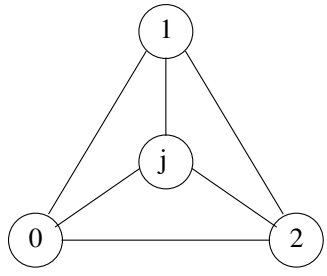


Figure 63: 3-wheel graph

[253] Michel Marie Deza, Martin Grötschel, and Monique Laurent. Clique-web facets for multicut polytopes. *Mathematics of Operations Research*, 17(4):981–1000, 1992

[254] Michel Marie Deza and Monique Laurent. *Geometry of cuts and metrics*, volume 15 of *Algorithms and Combinatorics*. 1997

(a) Let $N(v_j) \subseteq V$ be the set of adjacent neighbors to v_j.

(b) Let $E_{N(v_j)} = \{(v_s, v_t) : v_s \in N(v_j), v_t \in N(v_j)\}$ be the subset of E which lies completely in $N(v_j)$.

(c) Form a new graph $G_j = (N(v_j), E_{N(v_j)})$.

(d) For each edge in G_j, define a weight $W^j_{s,t} = \frac{1}{2} - z_{s,t} + \frac{1}{2}(z_{v_j,s} + z_{v_j,t})$. As z satisfies the cycle inequalities, we have $W^j_{s,t} \geq 0$.

(e) Find an *odd-cycle* $C = (V(C), E(C))$ in G_j such that

$$\begin{aligned} \sum_{(s,t)\in E(C)} W^j_{s,t} &= \sum_{(s,t)\in E(C)} \left(\frac{1}{2} - z_{s,t} + \frac{1}{2}(z_{v_j,s} + z_{v_j,t})\right) \\ &= \frac{|C|}{2} - \sum_{(s,t)\in E(C)} z_{s,t} + \sum_{v_i \in V(C)} z_{i,j} \\ &\leq \frac{1}{2}. \end{aligned}$$

If and only if such odd-cycle exists, C corresponds to a violated odd-wheel inequality in the original graph. If no odd-cycle satisfying the above inequality exists, then no odd-wheel inequality with v_j in the center is violated.

Finding the minimum weight odd-cycle in $G_j = (N(v_j), E_{N(v_j)})$ is polynomially solvable as follows.

i. Construct a new graph G'_j containing for each $v_i \in N(v_j)$ two copies v'_i, v''_i. For each edge $(v_s, v_t) \in E_{N(v_j)}$ add two edges (v'_s, v''_t) and (v''_s, v'_t) to the graph. Assign to both these edges the weight $W^j_{s,t}$.

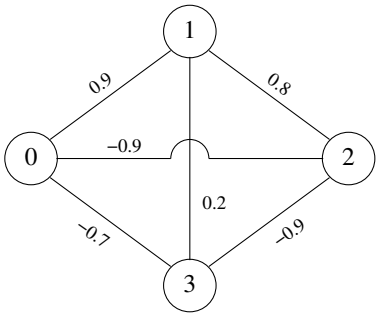
ii. For each $v_i \in N(v_j)$, solve a shortest path problem in the new graph between v'_i and v''_i). By construction, the path, if one exists, must be a cycle as v'_i and v''_i correspond to the same vertex in the original graph. Further, the path must be of odd length as the newly constructed graph is bipartite.

The odd-wheel inequalities are especially useful for graphs which contain dense subgraphs. Consider the graph shown in Figure 64(a), where the signed edge weights are shown. Using only the cycle inequalities leads to the fractional relaxed solution shown in Figure 64(b). Upon addition of a 3-wheel inequality, the solution becomes integral and optimal, and the relaxation becomes tight, shown in Figure 64(c).

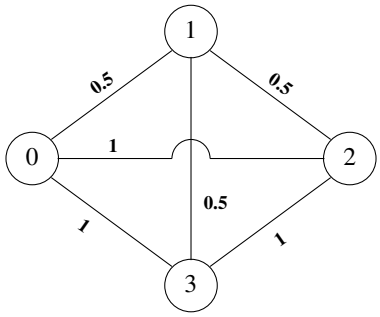
Although not used in our implementation, we want to point out that another subset of clique-web inequalities, known as *bicycle inequalities*[255] can be separated in polynomial time.

[255] Sunil Chopra and M. R. Rao. The partition problem. *Math. Program*, 59: 87–115, 1993

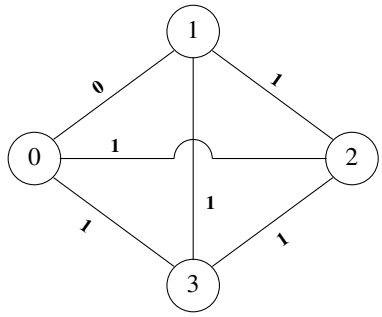
TOGETHER THE INEQUALITIES (79) AND (80) describe a tight polynomial-time solvable relaxation to $\text{conv}(\mathcal{Z}(G))$ that we will call LP-CO relaxation.



(a) Example input graph with four vertices and edge weights as shown.



(b) Fractional solution with $f(z^*) = -1.55$, obtained by the simple LP relaxation (without odd wheel inequalities).



(c) Integer solution with $f(z^*) = -1.5$, obtained by adding the odd wheel inequality $z_{0,2} + z_{0,3} + z_{2,3} - z_{0,1} - z_{1,2} - z_{1,3} \leq 1$.

Figure 64: Example of tightening by the odd-wheel inequality.

Sensitivity Analysis Details: Basis Matrix Approach and its Problems

In this section we discuss why the basis matrix approach cannot work well for linear programming relaxations. To illustrate this, we compare the stability ranges computed using the basis matrix approach with our exact approach on a small example graph. The basis matrix approach is shown to be very weak, even on this simple example.

Using the additional information provided by the simplex solver, namely the basis matrix and the dual variables for active constraints, we can compute partial stability ranges towards $\theta < 0$, $\rho^E_{\boldsymbol{d},-} : E \to \mathbb{R} \cup \{-\infty, \infty\}$ and towards $\theta > 0$, $\rho^E_{\boldsymbol{d},+} : E \to \mathbb{R} \cup \{-\infty, \infty\}$ for each $W(e)$ individually. Each partial stability range quantifies the allowed θ perturbations along a 1D subspace associated with a single edge variable; that is, it gives us the θ interval for which $W(e) + \theta \boldsymbol{d}(e)$ lies within the cone spanned by the active constraints.

The global stability range with respect to the known constraints is then given as respective maxima and minima over all edge stabilities; that is, as soon as one edge looses optimality, the entire solution does as well. We have the global stability range

$$\rho_{\boldsymbol{d},-} = \max_{e \in E} \rho^E_{\boldsymbol{d},-}(e), \qquad \rho_{\boldsymbol{d},+} = \min_{e \in E} \rho^E_{\boldsymbol{d},+}(e).$$

Lets see how the sensitivity functions $\rho_{\boldsymbol{d},-} : E \to \mathbb{R} \cup \{-\infty, \infty\}$ and $\rho_{\boldsymbol{d},+} : E \to \mathbb{R} \cup \{-\infty, \infty\}$ can be derived. (For an excellent introduction into sensitivity analysis, see chapter 5 in Bertsimas and Tsitsiklis[256].)

To be able to access basic results from linear programming, we first transform our problem into the so called *standard form* $\min_{\boldsymbol{z}} \boldsymbol{w}^\top \boldsymbol{z}$, sb.t. $A\boldsymbol{z} = b, \boldsymbol{z} \geq$

[256] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. 1997

$\mathbf{0}$. Our problem can be written as

$$\begin{aligned} \min_{\boldsymbol{z}} \quad & \boldsymbol{w}^\top \boldsymbol{z} \\ \text{sb.t.} \quad & A\boldsymbol{z} \leq b, \qquad \text{(cycle and odd-wheel inequalities)} \\ & \boldsymbol{z} \leq \mathbf{1}, \\ & \boldsymbol{z} \geq \mathbf{0}. \end{aligned}$$

Equivalently, adding non-negative slack variables $\boldsymbol{s}$ and $\boldsymbol{t}$, we write it as

$$\begin{aligned} \min_{\boldsymbol{z},\boldsymbol{s},\boldsymbol{t}} \quad & \boldsymbol{w}^\top \boldsymbol{z} \\ \text{sb.t.} \quad & A\boldsymbol{z} + \boldsymbol{s} = b, \qquad \text{(cycle and odd-wheel inequalities)} \\ & \boldsymbol{z} + \boldsymbol{t} = \mathbf{1}, \\ & \boldsymbol{z} \geq \mathbf{0}, \\ & \boldsymbol{s} \geq \mathbf{0}, \\ & \boldsymbol{t} \geq \mathbf{0}. \end{aligned}$$

For a given cost vector $\boldsymbol{w}$, we can obtain an optimal solution $\boldsymbol{z}^*$ to this linear program. Associated with this optimal solution are dual variables and an invertible *basis matrix* $\boldsymbol{B}$ and an index set of non-zero basic variables B. Together these satisfy

$$\begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{s} \\ \boldsymbol{t} \end{bmatrix}_B = \boldsymbol{B}^{-1} \begin{bmatrix} b \\ \mathbf{1} \end{bmatrix},$$

where $[\cdot]_B$ selects the subvector of variables in B; all other variables are zero [257]. The linear programming optimality conditions for the standard form linear program are

$$\begin{bmatrix} \bar{\boldsymbol{w}} \\ \bar{\boldsymbol{c}}_s \\ \bar{\boldsymbol{c}}_t \end{bmatrix}^\top = \begin{bmatrix} \boldsymbol{w} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^\top - \begin{bmatrix} \boldsymbol{w} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}_B^\top \boldsymbol{B}^{-1} \begin{bmatrix} A & I & \mathbf{0} \\ I & \mathbf{0} & I \end{bmatrix} \geq \mathbf{0}^\top,$$

and the lefthand vector is denoted *reduced cost*. At an optimal solution, all reduced costs are non-negative.

If for a given basis matrix $\boldsymbol{B}$ and a perturbation $\boldsymbol{w}' = \boldsymbol{w} + \theta\boldsymbol{d}$, the reduced costs remain non-negative, the basis and hence the solution remains optimal. However, as we will see below, the converse is not necessarily true: even with negative reduced cost, the solution might not change. The optimality condition with respect to the perturbed $\boldsymbol{w}'$ vector is given as

$$\begin{bmatrix} \bar{\boldsymbol{d}} \\ \bar{\boldsymbol{c}'}_s \\ \bar{\boldsymbol{c}'}_t \end{bmatrix}^\top = \begin{bmatrix} \boldsymbol{w} + \theta\boldsymbol{d} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^\top - \begin{bmatrix} \boldsymbol{w} + \theta\boldsymbol{d} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}_B^\top \boldsymbol{B}^{-1} \begin{bmatrix} A & I & \mathbf{0} \\ I & \mathbf{0} & I \end{bmatrix} \geq \mathbf{0}^\top,$$

[257] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. 1997; and Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998

which can be transformed using the linearity to yield the condition

$$\theta \left(\begin{bmatrix} \boldsymbol{d} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^\top - \begin{bmatrix} \boldsymbol{d} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}_B^\top \boldsymbol{B}^{-1} \begin{bmatrix} A & I & \mathbf{0} \\ I & \mathbf{0} & I \end{bmatrix} \right) = \theta \begin{bmatrix} \bar{\boldsymbol{d}} \\ \bar{c'}_s \\ \bar{c'}_t \end{bmatrix}^\top \geq - \begin{bmatrix} \bar{\boldsymbol{w}} \\ \bar{\boldsymbol{c}}_s \\ \bar{\boldsymbol{c}}_t \end{bmatrix}^\top .$$

Further, as $\bar{c'}_s = \bar{c}_s$ and $\bar{c'}_t = \bar{c}_t$, the basis remains optimal for $\boldsymbol{w}'$ if $\theta \bar{\boldsymbol{d}} \geq -\bar{\boldsymbol{w}}$ is fulfilled.[258] Obviously, for $\theta = 0$ this is the case, because our current solution is optimal for $\boldsymbol{w}$.

[258] All modern simplex-based linear programming solvers allow the calculation of reduced costs for arbitrary cost vectors, so $\bar{\boldsymbol{d}}$ is easily obtained.

For $\theta \neq 0$, we consider for each $m \in E$ and $(\bar{d}_m, \bar{w}_m)$ and the following cases

1. If $\bar{w}_m \neq 0$ and $\bar{d}_m \neq 0$, let $a_m = -\frac{\bar{w}_m}{\bar{d}_m}$, then if $a_m < 0$ we have $\rho^E_{\boldsymbol{d},-}(m) = a_m$, $\rho^E_{\boldsymbol{d},+}(m) = \infty$ and if $a_m > 0$ we have $\rho^E_{\boldsymbol{d},-}(m) = -\infty$, $\rho^E_{\boldsymbol{d},+} = a_m$.

2. If $\bar{w}_m = 0$ and $\bar{d}_m \neq 0$, there are multiple optimal solution for the current cost and any perturbation $\theta \boldsymbol{d}$ might lose optimality for the current basis, hence $\rho^E_{\boldsymbol{d},-}(m) = \rho^E_{\boldsymbol{d},+}(m) = 0$.

3. If $\bar{w}_m \neq 0$ and $\bar{d}_m = 0$, then with regard to the edge m, no perturbation $\theta \boldsymbol{d}$ can change the reduced cost and $\rho^E_{\boldsymbol{d},-}(m) = -\infty$, $\rho^E_{\boldsymbol{d},+}(m) = \infty$.

4. If $\bar{w}_m = 0$ and $\bar{d}_m = 0$, then similar to the previous case we have $\rho^E_{\boldsymbol{d},-}(m) = -\infty$, $\rho^E_{\boldsymbol{d},+}(m) = \infty$ and with regard to the edge m, the solution is stable for all $\theta \in \mathbb{R}$.

Problems with the Basis Matrix Approach

Our linear program is solved by iteratively adding cutting planes. Therefore, at the global optima the linear program consists only of a small subset of all constraints. This is a problem for the basis matrix approach if around the optimal solution we have *degeneracy*, as shown in Figure 65.

Due to the two-dimensional drawing, the figure is somewhat misleading in how degeneracy occurs: it is the rule rather than the exception. Even in case only facet-defining inequalities are used, in high dimensions there typically exists a large number of *binding inequalities* at the optimal solution. All these inequalities are necessary to describe the polytope, yet only a small subset is known to the linear programming solver. In Figure 65 one inequality is redundant.

Because this additional constraint has never been generated it is not active and therefore the basis matrix approach will underestimate the stable range. If on the other hand the constraint would be active, then the enlarged cone (dotted vertical line in Figure 65) would permit larger absolute values for negative θ.

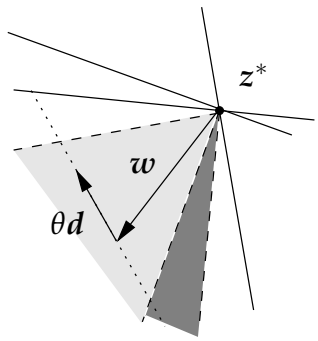


Figure 65: Degeneracy causes problems for the basis matrix approach to sensitivity analysis: an additional constraint which is unknown to the restricted problem enlarges the cone spanned by the constraints at the optima (enlarged part shown in dark).

Example: Stability Ranges

In the main paper we have briefly discussed per-edge sensitivities and stability ranges. Here we give a small toy example, shown in Figure 66. The optimal graph partitioning has three components, encircled in color in Figure 66.

We perform a stability range analysis using both the basis matrix method and the exact auxiliary linear program method for all $\boldsymbol{d}_i = \boldsymbol{e}_i$, the vector of all zeros with a single one at element i, as described in the main paper. The result is an interval for each edge, as shown in Figure 67 (basis matrix method) and Figure 68 (exact auxiliary LP method). If a single edge weight is modified by adding any number from within its respective stability range interval, the current graph partitioning shown in Figure 66 is guaranteed to remain optimal. However, the basis matrix method is too pessimistic compared to the exact auxiliary LP method and most stability ranges estimated by the basis matrix method (Figure 67) are strict subintervals of the true intervals (Figure 68). For the true intervals, if any constant outside this interval is added to the respective edge weight in the input graph, we are guaranteed the new optimal solution will be different to the one shown in Figure 66.

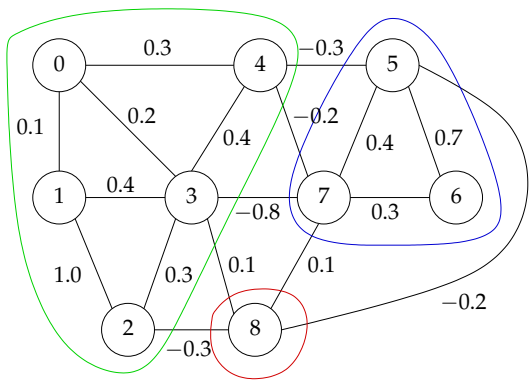


Figure 66: Toy example input graph with signed edge weights shown. The optimal graph partitioning has an objective of -1.6 and produces the three sets as shown.

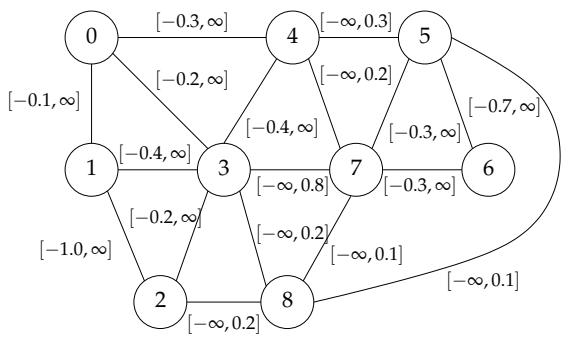


Figure 67: Per-edge weight sensitivities at the optimal solution, estimated by the basis matrix method.

All cut edges have a stability range of the form $[-\infty, a]$ with $a \geq 0$ and all intra-cluster edges have stability ranges of the form $[b, \infty]$ with $b \leq 0$.

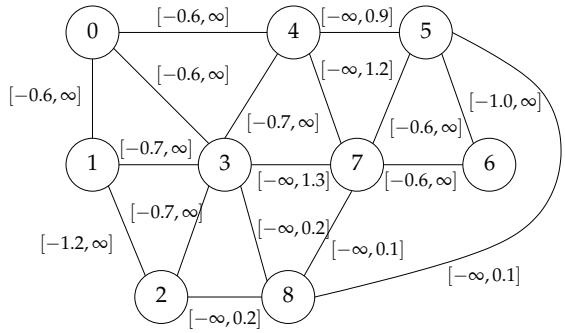


Figure 68: Per-edge weight sensitivities at the optimal solution, exact by the auxiliary linear programming method.

Experiments and Results

The first part of the experiments addresses properties of our algorithm and relaxation. We compare our solution method to a popular heuristic and demonstrate the gain of tightening the relaxation to LP-CO. This experiment relates optimality and runtime to properties of the data. The second part illustrates example applications: critical edges for modularity clustering and an analysis of the solution path for similarity data.

Tightness and comparison to a heuristic

In the introduction section we have shown how to solve modularity clustering via GPP. Here we examine solution qualities of our LP relaxation and the Kernighan-Lin (KL) heuristic[259]. The KL heuristic is a very large-scale neighborhood search method performing greedy steps to iteratively improve a given partitioning. Due to the way the next step is found, the method can make large changes to the current partitioning in each iteration and generally converges fast. However, as with all local methods no guarantee on the solution obtained can be given, in contrast with the LP relaxations, where integrality indicates optimality.

We compare KL to two variants of relaxation: LP-C, which is limited to cycle-inequalities, and the tightened LP-CO, which also includes odd-wheel inequalities. Note that all previous LP relaxations of correlation and modularity clustering[260] correspond to LP-C.

The solution produced by the KL heuristic is always feasible but possibly suboptimal, and LP-C and LP-CO are weak and tight relaxations, respectively. Hence the *maximized* modularity always satisfies KL $\leq$ **OPT** $\leq$ LP-CO $\leq$ LP-C, where **OPT** is the true optimum.

We evaluate solutions on five networks described in Brandes et al.[261]; and Newman and Girvan[262]: `dolphins`, `karate`, `polbooks`, `lesmis` and `att180` (62, 34, 105, 77 and 180 nodes, respectively). These small-scale networks datasets are available at `http://www-personal.umich.edu/~mejn/netdata/`.

Table 10 shows the achieved modularity and the runtime. For all data sets, the LP-CO solutions are optimal (OPT=LP-CO) and all modularity scores

[259] Brian W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, pages 291–307, February 1970

[260] Thomas Finley and Thorsten Joachims. Supervised clustering with support vector machines. In *ICML*, pages 217–224, 2005; Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE TKDE*, 20(2):172–188, 2008; Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theor. Comput. Sci*, 361(2-3):172–187, 2006; D. Emanuel and A. Fiat. Correlation clustering – minimizing disagreements on arbitrary weighted graphs. In *Proceedings of the ESA*, 2003; and Isabelle Warnesson. Applied linguistics: Optimization of semantic relations by data aggregation techniques. *Applied Stochastic Models and Data Analysis*, 1:121–141, 1985

[261] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE TKDE*, 20(2):172–188, 2008

[262] Mark E. J. Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(026113), 2004

agree with the best modularity in the literature.[263]

The Kernighan-Lin heuristic is always the fastest method and its solutions are close to optimal, as the upper bound provided by LP-C and LP-CO shows. KL itself does not give hints about closeness to optimality. Because it is a heuristic it cannot provide a guarantee on the solution quality and we are only able to state that it is close to optimal because we do know an upper bound on the solution value. The LP-C relaxation is in general very weak and obtains the optimal solution only on the smallest data set (karate). All it yields otherwise is an upper bound on the optimal modularity. So the effort of a tighter approximation (LP-CO) does improve the quality of the solution already on small examples.

[263] Except for the karate data set which differs from the optimal modularity of 0.431 reported in . We contacted the authors who discovered a corruption in their data set and confirmed our value of 0.4198.

Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE TKDE*, 20(2):172–188, 2008

Table 10: Modularity and runtimes on standard small network datasets. Fractional solutions are bracketed, optimal solutions are in boldface.

	Kernighan-Lin		LP-C		LP-CO	
	obj	time	obj	time	obj	time
dolphins	0.5268	0.4s	(0.5315)	4.2s	**0.5285**	9.1s
karate	**0.4198**	0.1s	**0.4198**	0.2s	**0.4198**	0.2s
polbooks	0.5226	7.0s	(0.5276)	147.4s	**0.5272**	148.5s
lesmis	0.5491	1.5s	(0.5609)	6.9s	**0.5600**	11.7s
att180	0.6559	14.5s	(0.6633)	302.3s	**0.6595**	1119.6s

LP-CO Scaling Behavior

After investigating the gain of the tighter relaxation, we now examine the scaling behavior of LP-CO with respect to edge density, problem difficulty and noise.

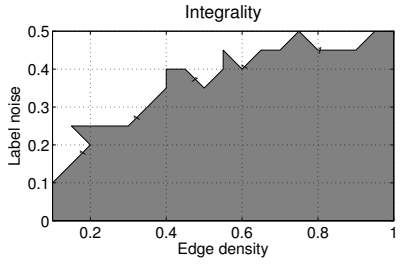
We sample a total of 100 vertices and uniformly assign one out of three "latent" class labels to each vertex. For a given edge density $d \in \{0.1, 0.15, \ldots, 1.0\}$ we sample a set E of $\frac{100 \cdot 99}{2} d$ non-duplicate edges from the complete graph. To each edge $e \in E$ we assign with probability $n \in \{0, 0.05, \ldots, 0.5\}$ a "noisy" weight uniformly at random from the interval $[-1, 1]$. To all other edges we assign a "true" weight from either $[-1, 0]$ if the latent class label of the adjacent vertices are different, or from $[0, 1]$ if the latent class labels are equal. For each pair (d, n) we create ten graphs with the above properties and solve GPP on each instance.

Figures 69(a) to (c) show where integrality was achieved, the average runtime and Rand index to the underlying labels. The index is 1 if the partitioning is identical to the latent classes. The expected Rand index of a random partitioning[264] is $\frac{2}{3}$.

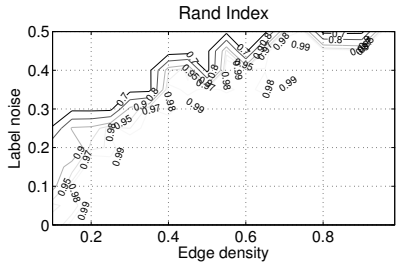
The figures suggest two relations between properties of the data and the algorithm. First, integrality of the LP-CO solution (gray region in Figure 69(a)) mostly coincides with the optimal solution being close to the "latent" labels, i.e., cases where the Rand index in Figure 69(b) is 1. Second, the runtime

[264] William M. Rand. Objective criteria for the evaluation of clustering methods. *American Statistical Association Journal*, 66 (336):846–850, 1971

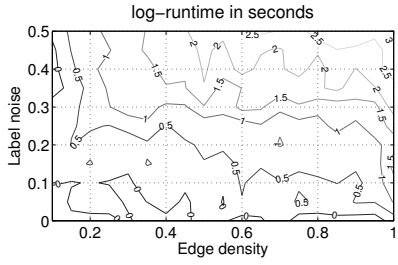
depends more on the noise level than on edge density. We do not illustrate corresponding results for the weaker LP-C relaxation. It generates 12% fewer integral solutions and smaller corresponding Rand indices, but runs faster when there is lots of noise.



(a) Parameters for which solutions were integral (gray).



(b) Mean Rand index of the partitioning vs. latent classes.



(c) Log_{10}-runtime in seconds, averaged over ten runs.

Figure 69: Experimental results for the synthetic data.

Example applications of stability

We now apply stability analysis to investigate the properties of clustering solutions in two applications.

''CRITICAL EDGES'' IN MODULARITY CLUSTERING. Modularity clustering is a popular tool to analyze networks. But which edges are *critical* for the partition at hand, i.e., their removal will change the optimal solution?

To test whether an edge e is critical, we compute the stability range for the perturbation $\boldsymbol{d} = w_M(V, E \setminus \{e\}) - w_M(V, E)$, where w_M computes the modularity edge weights from the original undirected, unweighted graph. For $\theta = 1$, the GPP weights will correspond to $E \setminus \{e\}$, so e is critical if and only if $1 \notin [\rho_{\boldsymbol{d},-}, \rho_{\boldsymbol{d},+}]$. Figure 70 illustrates the critical edges on top of the partitioning of the `karate` network, an example for a social network.

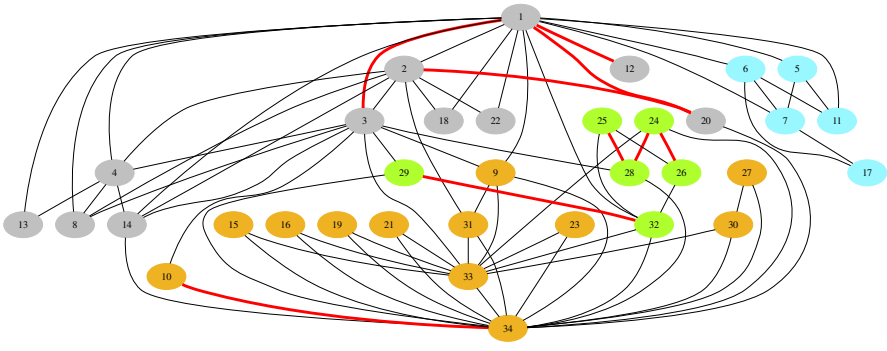


Figure 70: Critical edges in Zarachy's karate club network with four groups. A removal of any critical edge (drawn thick/red) would change the current (best) partitioning. All other edges can be removed individually without changing the solution.

THE SOLUTION PATH can reveal more information about a data set than one partition alone. Our data, courtesy of Frank Jäkel, contains pairwise

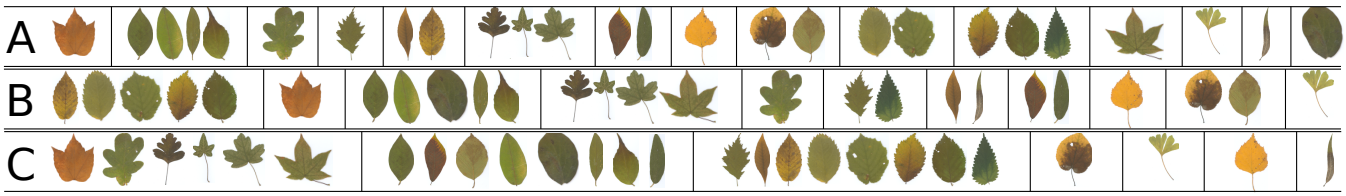


Figure 72: Stable solution (A) for $[-0.315, -0.259]$ (15 clusters), (B) for $[-0.228, -0.189]$ (11 clusters), (C) for $[-0.112, -0.087]$ (7 clusters). Grouped leaves are in the same cluster.

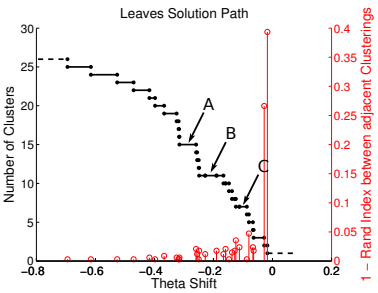


Figure 71: Clustering solution path for the leaves dataset. The stems show the difference of adjacent clusterings.

[265] William M. Rand. Objective criteria for the evaluation of clustering methods. *American Statistical Association Journal*, 66 (336):846–850, 1971

similarities of 26 types of leaves in the form of human confusion rates. To investigate groups of leaves induced by those similarities, we solve GPP on a similarity graph with edge weights equal to the symmetrized confusion rates. This corresponds to weighted correlation clustering, where negative weights indicate dissimilarity.

We make low similarities negative by adding a threshold $\theta < 0$ from each edge ($\boldsymbol{d} = \mathbf{1}$). It is not obvious how to set θ; a higher θ will result in few clusters. Hence, we trace the solution path for $\theta = 0$ to the point when each node is a cluster.

Figure 71 illustrates how the stability ranges of the solutions vary along the path. Figure 72 shows some stable solutions.

At change points of the path, the optimal solution often changes only little, as indicated by the Rand index[265]. This means that many solutions are very similar and might represent the same underlying clustering. Indeed, the path reveals structural characteristics of the data: low-density areas in the graph will be cut first, whereas some leaves remain together throughout almost the entire path and form dense sub-communities.

Thus, stable solutions at different levels of θ can indicate sub-structures of communities. Leaves that are fluctuating between groups are not clearly categorized and likely to be at the boundary between two clusters.

In general, the solution path provides richer information than one single clustering and permits a more careful analysis of the data, in particular if the value of a decisive model parameter is uncertain.

Conclusions

We have shown a new general method to compute stability ranges for combinatorial problems. Applied to a unifying formulation, GPP, this method opens up new ways to carefully analyze graph partitioning problems. The experiments illustrate examples for GPP and an analysis of the method.

A useful extension will be to find the perturbation to which the solution is most sensitive, rather than specifying the direction beforehand.

Given the generality of the method developed in this work, where else could the analysis of solution stability lead to further insights? Examples

may be other learning settings, algorithms that make use of combinatorial optimization, or theoretical analysis.

Discussion

> Approach each new problem not with a view of finding what you hope will be there, but to get the truth, the realities that must be grappled with. You may not like what you find. In that case you are entitled to try to change it. But do not deceive yourself as to what you do find to be the facts of the situation.
>
> Bernard M. Baruch

In this thesis we have studied machine learning methods for structured input and structured output data together with their applications to high-level computer vision problems.

Structured learning methods are a recent trend in machine learning but their application to computer vision problems has largely remained unexplored. We believe this is not due to missing applicability — in fact the rich structure present in the input and output domain of many computer vision problems lends itself almost ideally to such methods — but rather due to three reasons. *First*, it can be difficult to adequately formalize and model the structure. *Second*, there is no established consensus on best practices, standard models and learning methods. *Third*, many models result in hard to solve inference problems, often of combinatorial flavor.

We have seen the latter point in the graph-based recognition approach, that required the solution of NP-hard graph isomorphism problems, and in the image segmentation under connectivity constraints, that also yielded an NP-hard MAP estimation problem.

We have shown how this issue of computational tractability in structured models can be addressed. For the case of structured input learning we proposed the substructure poset framework where *efficient enumeration methods* from the data mining community allow us to learn discriminative classifiers using large substructure-induced feature spaces. For structured prediction we argued for the principled construction of *relaxations* to the original problem using polyhedral combinatorics. For structured output problems with a finite output domain our construction is universal. We believe both contributions have broad applicability beyond computer vision.

The ability to learn prediction functions with highly structured

output spaces is often achieved at the cost of *giving up* the probabilistic interpretation of the model.

In our image segmentation application we have seen that by giving up the probabilistic interpretation we can enforce even highly combinatorial constraints on the prediction outputs, such as the connectivity constraint. However, by giving up the probabilistic interpretation, basic natural operations such as maximum likelihood learning and computing marginal probabilities become inapplicable.

We address this issue partially by considering *solution stability* as an alternative to quantify certainty in a structured prediction. As a result of our proposed method we have shown that the solution stability can always be computed if we can compute the structured prediction itself; it is thus always tractable under our computational assumptions.

In general we believe that alternative, non-probabilistic measures of prediction uncertainty could be a viable addition to structured prediction models in order to compensate for the non-probabilistic nature of many of these models. Yet, our contribution can only be seen as a first step in this direction.

THROUGHOUT THE THESIS WE HAVE EXTENSIVELY EVALUATED the proposed approaches experimentally on high-level computer vision problems. In some cases, such as for the graph-based recognition approach, the results did not show a clear general improvement in prediction accuracy of our proposed approach over existing baseline models. We have discussed possible reasons specific to our computer vision applications earlier, but would like to briefly point out a more *fundamental issue* raised by our research in structured models.

Structured models are more complex to *build*, more complex to *train* and more complex to *understand*. While current research including this thesis focuses on the issues of training and interpreting the model output, there is a lack of effort into examining problems of *model building* outside the probabilistic regime in a principled way.

We believe that in order to fully benefit from the capabilities of structured machine learning models further research into model building is necessary.

Appendix: Proofs

Proof to Lemma 6

Every single node k constitutes a connected subgraph. By setting $y_k = 1$, $y_h = 0$ for $h \neq k$ a feasible solution is obtained. All these solutions are affinely independent. Furthermore the empty graph is also a feasible subgraph. It follows that $\dim(Z) = |V|$, i.e., the connected subgraph polytope has full dimension. □

Proof to Lemma 7

First, $y_i \geq 0$. For each i, we construct $|V|$ affinely independent points in C with $y_i = 0$. Fix i, then one solution is obviously $\boldsymbol{x} = \mathbf{0}$, the empty subgraph. Next, for all $p \neq i$, obtain one solution by setting only $y_p = 1$, and for all $j \neq p$ set $y_j = 0$. Clearly, $y_j = 0$ and the $|V| - 1$ solutions thus obtained are affinely independent. In total we have $|V|$ solutions with $y_i = 0$, thus $y_i \geq 0$ is facet-defining.

Second, $y_i \leq 1$. Again let i be arbitrary. We construct $|V|$ affinely independent points in C with $y_i = 1$. For this, set $y_i = 1$ and $y_j = 0$ for all $j \neq i$. This is obviously one solution. Now root a spanning tree in i and set one node k at a time to $y_k = 1$, respecting the order of the spanning tree, i.e., the subgraph selected all nodes j with $y_j = 1$ always remains a connected subgraph of the spanning tree. This constructs $|V| - 1$ solutions, all affinely independent. Adding the first solution yields $|V|$ solutions in total, completing the proof. □

Proof to Theorem 5

First, the direction "is feasible" implying "is connected". Assume any given feasible $\boldsymbol{y}$ given, hence any $y_i \in \{0, 1\}$. If $\sum_i y_i \leq 1$, the resulting subgraph is trivially connected, hence assume $\sum_i y_i \geq 2$. For arbitrary $y_i = 1$, $y_j = 1$, $i \neq j$, assume i and j are not connected, that is $(i, j) \notin E$ and moreover there exists no path on G with all vertex variables being one. Trivially, we construct a vertex-separator set $S = \{k \in V : y_k = 0\}$ with $S \in \mathcal{S}(i, j)$. The removal of S from V must disconnect i and j, as $(i, j) \notin E$. However, by (64) we must have $y_i + y_j - \sum_{k \in S} y_k - 1 = 2 - 0 - 1 = 1 \leq 0$, which is clearly violated. Thus, feasibility implies connectedness. Second, the direction "is connected" implying "is feasible". Take any $y_i = 1$, $y_j = 1$, $i \neq j$, and i, j connected in G by a path starting at i and ending at j such that all intermediate nodes k

satisfy $y_k = 1$. For all separators $S \in \mathcal{S}(i,j)$, at least one node t of this path must satisfy $t \in S$. Therefore $y_i + y_j - \sum_{k \in S} y_k - 1 \leq y_i + y_j - y_t - 1 = 0 \leq 0$ is satisfied. Thus any connected subgraph is feasible. □

Proof to Theorem 6

We will prove this for any $i,j \in V$ by constructing $|V|$ affinely independent points in C which satisfy the inequality as equality. By section 9.2.3 in [266] this shows that the inequality is facet-defining.

For $i,j \in V$ arbitrarily chosen, for any $S \in \bar{\mathcal{S}}(i,j)$, let $S = \{s_1, \ldots, s_{|S|}\}$ be the set of nodes in the essential vertex-separator set.

Further let S induce a partitioning of the graph into the set S, the connected subgraphs P_i, P_j, containing i and j, respectively, and the connected subgraphs P_s connected to exactly one $s \in S$ (if it is connected to more than one $s \in S$, remove all but one edge arbitrarily). This is shown in Figure 73.
First, we construct $|P_i| + |P_j|$ affinely independent solutions in C which satisfy the equality.

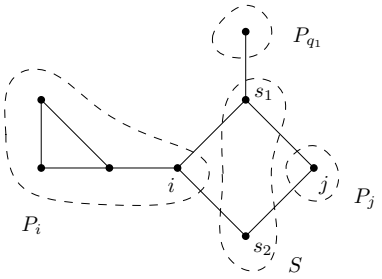


Figure 73: The separator set S induces a graph partitioning.

1. For the connected subgraph P_i, root a spanning tree in i. Set $y_i = 1$, $y_k = 0$, $\forall k \in P_i, k \neq i$. For each such $k \in P_i$, enlarge the subgraph incrementally by one node in an arbitrary ordering respecting the spanning tree, i.e., set $y_k = 1$. Each enlarged solution is a connected subgraph of P_i and G, and affinely independent to all previous ones and satisfied the equality.

2. Likewise, do this for P_j, starting with just $y_j = 1$.

Next, for each $s \in S$, we construct $|P_s| + 1$ affinely independent solutions satisfying the equality as follows.

1. Set $y_k = 1$, $\forall k \in P_i \cup P_j$, and $y_s = 1$. This solution is in C because S is essential and thus s connects P_i and P_j. Construct $|P_s|$ more solutions by building a spanning tree for P_s, rooted in the node connected to s. By incrementally setting $y_k = 1$ in an order respecting the spanning tree, $|P_s|$ affinely independent solutions in C are obtained.

We now consider the total number of solutions constructed.

$$|P_i| + |P_j| + \sum_{s \in S} (|P_s| + 1) = |V|.$$

We have constructed $|V|$ affinely independent solutions in C satisfying the equality. Therefore, by section 9.2.3 in [267], the inequality defines a facet of $\text{conv}(C)$. □

[266] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998

[267] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998

Bibliography

[1] Ankur Agarwal and Bill Triggs. Learning to track 3D human motion from silhouettes. In *ICML*. ACM, 2004.

[2] Shivani Agarwal, Aatif Awan, and Dan Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. Pattern Anal. Mach. Intell*, 26(11):1475–1490, 2004.

[3] Ravindra K. Ahuja, Özlem Ergun, James B. Orlin, and Abraham P. Punnen. A survey of very large-scale neighborhood search techniques. In Endre Boros and Peter L. Hammer, editors, *Proceedings of the 1999 Workshop on Discrete Optimization (DO-99)*, volume 123, 1-3 of *Discrete Applied Mathematics*, pages 75–102, Amsterdam, July 25–30 2002. Elsevier Science B.V.

[4] Karteek Alahari, Pushmeet Kohli, and Philip H. S. Torr. Reduce, reuse & recycle: Efficiently solving multi-label MRFs. In *CVPR*. IEEE Computer Society, 2008.

[5] Nachman Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

[6] David Avis and Komei Fukuda. Reverse search for enumeration. *Discrete Appl. Math.*, 65:21–46, 1996.

[7] Egon Balas. Projection, lifting and extended formulation in integer and combinatorial optimization. *Annals of Operations Research*, (140):125–161, 2005.

[8] Herbert Bay, Tinne Tuytelaars, and Luc J. Van Gool. SURF: Speeded up robust features. In *ECCV*, pages 404–417, 2006.

[9] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc J. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[10] Dimitri P. Bertsekas. *Network Optimization*. 1998.

[11] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. 1997.

[12] Julian Besag. Statistical analysis of non-lattice data. *The Statistician*, 24 (3):179–195, 1975.

[13] Julian Besag. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrica*, (64):616–618, 1977.

[14] Irving Biederman. Recognition by components - a theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[15] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[16] Matthew B. Blaschko and Christoph H. Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008.

[17] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(3):257–267, 2001.

[18] Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *NIPS*, 2007.

[19] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[20] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124–1137, 2004.

[21] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell*, 23 (11):1222–1239, 2001.

[22] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE TKDE*, 20(2):172–188, 2008.

[23] Leo Breiman. Prediction games and arcing algorithms. Technical report, December 1997. Technical Report 504, University of California, Berkeley.

[24] Michael C. Burl, Markus Weber, and Pietro Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *ECCV*, pages 628–641, 1998.

[25] Miguel Á. Carreira-Perpiñán and Geoffrey E. Hinton. On contrastive divergence learning. In *AISTATS*, 2005.

[26] Bryan Catanzaro, Narayanan Sundaram, Bor-Yiing Su, Yunsup Lee, Mark Murphy, and Kurt Keutzer. Damascene: Highly parallel image contour detection, March 2009. URL `http://www.gigascale.org/pubs/1510.html`.

[27] Sunil Chopra and M. R. Rao. The partition problem. *Math. Program*, 59: 87–115, 1993.

[28] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14: 462–467, 1968.

[29] Michael Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms, July 2002.

[30] Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *ICCV*, pages 1197–1203, 1999.

[31] Francis Comets. On consistency of a class of estimators for exponential families of Markov random fields on the lattice. *The Annals of Statistics*, 20(1):455–468, 1992.

[32] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. 1990.

[33] David J. Crandall, Pedro F. Felzenszwalb, and Daniel P. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *CVPR*, 2005.

[34] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.

[35] Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theor. Comput. Sci*, 361(2-3):172–187, 2006.

[36] Ayhan Demiriz, Kristin P. Bennett, and John Shawe-Taylor. Linear programming boosting via column generation. *Journal of Machine Learning*, 46:225–254, 2002.

[37] Michel Marie Deza and Monique Laurent. *Geometry of cuts and metrics*, volume 15 of *Algorithms and Combinatorics*. 1997.

[38] Michel Marie Deza, Martin Grötschel, and Monique Laurent. Clique-web facets for multicut polytopes. *Mathematics of Operations Research*, 17 (4):981–1000, 1992.

[39] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005.

[40] Justin Domke. Crossover random fields. Technical report, University of Maryland, 2009.

[41] Justin Domke, Alap Karapurkar, and Yiannis Aloimonos. Who killed the directed model? In *CVPR*. IEEE Computer Society, 2008.

[42] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*, volume November. John Wily & Sons, Inc., New York, second edition, 2000. ISBN 0471056693.

[43] Alexei A. Efros, Alexander C. Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *ICCV*, pages 726–733, 2003.

[44] D. Emanuel and A. Fiat. Correlation clustering – minimizing disagreements on arbitrary weighted graphs. In *Proceedings of the ESA*, 2003.

[45] Mark Everingham, Luc Van Gool, Christopher K.I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2008 Results. http://www.pascal-network.org/challenges/VOC/voc2008/.

[46] Mark Everingham, Andrew Zisserman, Chris Williams, and Luc Van Gool. The pascal visual object classes challenge 2006 (VOC2006) results. Technical report, 2006.

[47] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59 (2):167–181, 2004.

[48] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1): 55–79, 2005.

[49] Pedro F. Felzenszwalb, David A. McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[50] Robert Fergus, Pietro Perona, and Andrew Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages 264–271, 2003.

[51] Thomas Finley and Thorsten Joachims. Supervised clustering with support vector machines. In *ICML*, pages 217–224, 2005.

[52] Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In *ICML*, 2008.

[53] Martin A. Fischler and Robert A. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Computer*, 22(1):67–92, January 1973.

[54] Daniel Freedman and Petros Drineas. Energy minimization via graph cuts: Settling what is possible. In *CVPR*, pages 939–946, 2005.

[55] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EUROCOLT*, 1994.

[56] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proc. 13th International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996.

[57] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[58] Brendan J. Frey and David J. C. MacKay. A revolution: Belief propagation in graphs with cycles. In *NIPS*, 1997.

[59] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2): 337–374, 2000.

[60] Thomas Gärtner. A survey of kernels for structured data. *SIGKDD Explorations*, 5(1):49–58, 2003.

[61] Arthur M. Geoffrion. Elements of large-scale mathematical programming: Part i: Concepts. *Management Science*, 16(11):652–675, 1970.

[62] Arthur M. Geoffrion. Elements of large-scale mathematical programming: Part ii: Synthesis of algorithms and bibliography. *Management Science*, 16(11):676–691, 1970.

[63] Basilis Gidas. Consistency of maximum likelihood and pseudo-likelihood estimators for Gibbs distributions. In *Stochastic Differential Systems, Stochastic Control Theory and Applications*. Springer, 1988.

[64] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *NIPS*, 2007.

[65] Irwin R. Goodman and Samuel Kotz. Multivariate θ-generalized normal distributions. *Journal of Multivariate Analysis*, 3(2):204–219, June 1973.

[66] Martin Grötschel and Yoshiko Wakabayashi. A cutting plane algorithm for a clustering problem. *Math. Prog.*, 45, 1989.

[67] Martin Grötschel and Yoshiko Wakabayashi. Facets of the clique partitioning polytope. *Math. Prog.*, 47:367–387, 1990.

[68] David Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, University of California at Santa Cruz, Santa Cruz, CA, USA, July 1999.

[69] Xuming He, Richard S. Zemel, and Miguel Á. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *CVPR*, pages 695–702, 2004.

[70] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[71] Derek Hoiem, Carsten Rother, and John M. Winn. 3D layoutCRF for multi-view object class recognition and segmentation. In *CVPR*, 2007.

[72] Huixiao Hong, Hong Fang, Qian Xie, Roger Perkins, Daniel M. Sheehan, and Weida Tong. Comparative molecular field analysis (comfa) model using a large diverse set of natural, synthetic and environmental chemicals for binding to the androgen receptor. *SAR QSAR Environmental Research*, 14(5-6):373–388, 2003.

[73] Aapo Hyvärinen. Consistency of pseudolikelihood estimation of fully visible boltzmann machines. *Neural Computation*, 18(10):2283–2292, 2006.

[74] Trey Ideker, Owen Ozier, Benno Schwikowski, and Andrew F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. In *ISMB*, 2002.

[75] Hiroshi Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Trans. Pattern Anal. Mach. Intell*, 25(10):1333–1336, 2003.

[76] Tommi S. Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *NIPS*. 1999.

[77] Benjamin Jansen, J. J. de Jong, Cornelius Roos, and Tamás Terlaky. Sensitivity analysis in linear programming: Just be careful! *European Journal of Operational Research*, 101:15–28, 1997.

[78] Thorsten Joachims. *Learning to Classify Text using Support Vector Machines*. Kluwer Academic Publishers, 2002.

[79] Richard M. Karp. Maximum-weight connected subgraph problem, 2002. `http://www.cytoscape.org/ISMB2002/`.

[80] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Marginalized kernels between labeled graphs. In *ICML*, 2003.

[81] Yan Ke, Rahul Sukthankar, and Martial Hebert. Efficient visual event detection using volumetric features. In *ICCV*, pages 166–173, 2005.

[82] Michael Kearns. Thoughts on hypothesis boosting. (Unpublished), December 1988. URL `http://www.cis.upenn.edu/~mkearns/papers/boostnote.pdf`.

[83] Brian W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, pages 291–307, February 1970.

[84] Fatma Kılınc-Karzan, Alejandro Toriello, Shabbir Ahmed, George Nemhauser, and Martin Savelsbergh. Approximating the stability region for binary mixed-integer programs. Technical report, Gatech, 2007.

[85] Pushmeet Kohli, L'ubor Ladický, and Philip H. S. Torr. Robust higher order potentials for enforcing label consistency. In *CVPR*, 2008.

[86] Pushmeet Kohli, Alexander Shekhovtsov, Carsten Rother, Vladimir Kolmogorov, and Philip H. S. Torr. On partial optimality in multi-label MRFs. In *ICML*, volume 307, pages 480–487, 2008.

[87] Vladimir Kolmogorov and Carsten Rother. Minimizing nonsubmodular functions with graph cuts-A review. *IEEE Trans. Pattern Anal. Mach. Intell*, 29(7):1274–1279, 2007.

[88] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2):147–159, 2004.

[89] Nikos Komodakis and Nikos Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. In *ECCV*, 2008.

[90] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*. IEEE, 2007.

[91] Nikos Komodakis, Georgios Tziritas, and Nikos Paragios. Fast, approximately optimal solutions for single and dynamic MRFs. In *CVPR*. IEEE Computer Society, 2007.

[92] V.K. Koval and M.I. Schlesinger. Dvumernoe programmirovanie v zadachakh analiza izobrazheniy (two-dimensional programming in image analysis problems). *Automatics and Telemechanics*, 8:149–168, 1976. In Russian.

[93] Stefan Kramer, Nada Lavrac, and Peter Flach. Propositionalization approaches to relational data mining. In Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*, pages 262–291. Springer, September 2001. ISBN 3-540-42289-7.

[94] Samuel Krempp, Donald Geman, and Yali Amit. Sequential learning of reusable parts for object detection. Technical report, 2002.

[95] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

[96] Taku Kudo, Eisaku Maeda, and Yuji Matsumoto. An application of boosting to graph classification. In *NIPS*, 2004.

[97] Mudigonda Pawan Kumar and Philip Torr. Efficiently solving convex relaxations for MAP estimation. In *ICML*, 2008.

[98] Mudigonda Pawan Kumar, Vladimir Kolmogorov, and Philip Torr. An analysis of convex relaxations for MAP estimation. In *NIPS*, 2008.

[99] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[100] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.

[101] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *PAMI*, 2009.

[102] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[103] Julia A. Lasserre, Christopher M. Bishop, and Thomas P. Minka. Principled hybrids of generative and discriminative models. In *CVPR*, pages 87–94. IEEE Computer Society, 2006.

[104] Steffen L. Lauritzen. *Graphical Models*. Clarendon Press, Oxford, 1996. ISBN 0-19-852219-3.

[105] Steffen L. Lauritzen and David J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, B 50(2):157–224, 1988.

[106] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. A maximum entropy framework for part-based texture and object recognition. In *ICCV*, 2005.

[107] Yann LeCun, Sumit Chopra, Raia Hadsell, Marc A. Ranzato, and Fu Jie Huang. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006.

[108] Fei-Fei Li, Robert Fergus, and Pietro Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *ICCV*, 2003.

[109] Yunpeng Li and Daniel Huttenlocher. Learning for stereo vision using the structured support vector machine. In *CVPR*, 2008.

[110] David MacKay. *Information Theory, Inference, and Learning Algorithms*. September 2003. URL `http://www.inference.phy.cam.ac.uk/mackay/itila/book.html`.

[111] Llew Mason, Jonathan Baxter, Peter L. Bartlett, and Marcus R. Frean. Boosting algorithms as gradient descent. In *NIPS*, pages 512–518. The MIT Press, 1999.

[112] David Mease and Abraham Wyner. Evidence contrary to the statistical view of boosting. *Journal of Machine Learning Research*, 9:131–156, February 2008.

[113] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging. In *Advanced Lectures on Machine Learning*, pages 119–184. Springer, 2003.

[114] Tom Minka. Discriminative models, not discriminative training. Technical Report MSR-TR-2005-144, Microsoft Research (MSR), October 2005.

[115] Alastair P. Moore, Simon Prince, Jonathan Warrell, Umar Mohammed, and Graham Jones. Superpixel lattices. In *CVPR*, 2008.

[116] Greg Mori. Guiding model search using segmentation. In *ICCV*, 2005.

[117] Shinichi Morishita. Computing optimal hypotheses efficiently for boosting. In *Progress in Discovery Science*, volume 2281, pages 471–481. Springer, 2002. URL `http://citeseer.ist.psu.edu/492998.html`.

[118] Kevin Patrick Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, pages 467–475, July 1999.

[119] Radford. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, September 1993.

[120] Mark E. J. Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(026113), 2004.

[121] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *British Machine Vision Conference*, page III:1249, 2006.

[122] Nils J. Nilsson. *Artificial Intelligence: A New Synthesis*. Morgan Kaufmann Publishers, San Francisco, 1998. ISBN 1558604677.

[123] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer, second edition, 2006. ISBN 0-387-30303-0.

[124] Sebastian Nowozin and Koji Tsuda. Frequent subgraph retrieval in geometric graph databases. In *ICDM*, 12 2008.

[125] Sebastian Nowozin, Gökhan Bakır, and Koji Tsuda. Discriminative subsequence mining for action classification. In *ICCV 2007: Proceedings of the 2007 IEEE Computer Society International Conference on Computer Vision*, 2007.

[126] Aykut Özsoy and Martine Labbé. Size constrained graph partitioning polytope. Technical Report 577, ULB, 2007.

[127] Constantine Papageorgiou and Tomaso Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[128] Sridevi Parise and Max Welling. Learning in Markov random fields: An empirical study. In *Joint Statistical Meeting JSM2005*, 2005.

[129] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Mateo, California, 1988. ISBN 0934613737.

[130] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Jianyong Wang, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. Mining sequential patterns by pattern-growth: The prefixspan approach. *IEEE Trans. Knowl. Data Eng*, 16(11):1424–1440, 2004.

[131] John C. Platt, Nello Cristianini, and John Shawe-Taylor. Large margin DAGs for multiclass classification. In *NIPS*, pages 547–553. The MIT Press, 1999.

[132] Patrick Pletscher, Cheng Soon Ong, and Joachim M. Buhmann. Spanning tree approximations for conditional random fields. In *AISTATS*, 2009.

[133] Ariadna Quattoni, Michael Collins, and Trevor Darrell. Conditional random fields for object recognition. In *NIPS*, 2004.

[134] Srikumar Ramalingam, Pushmeet Kohli, Karteek Alahari, and Philip H. S. Torr. Exact inference in multi-label CRFs with higher order cliques. In *CVPR*, 2008.

[135] Deva Ramanan and David A. Forsyth. Automatic annotation of everyday movements. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *NIPS*. MIT Press, 2003. ISBN 0-262-20152-6.

[136] Jan Ramon and Thomas Gärtner. Expressivity versus efficiency of graph kernels. In *First International Workshop on Mining Graphs, Trees and Sequences (MGTS-2003)*, pages 65–74, September 2003.

[137] William M. Rand. Objective criteria for the evaluation of clustering methods. *American Statistical Association Journal*, 66(336):846–850, 1971.

[138] Gunnar Rätsch, Ayhan Demiriz, and Kristin P. Bennett. Sparse regression ensembles in infinite and finite hypothesis spaces. *Machine Learning*, 48(1-3):189–218, 2002.

[139] Gunnar Rätsch, Sebastian Mika, Bernhard Schölkopf, and Klaus-Robert Müller. Constructing boosting algorithms from SVMs: An application to one-class classification. *IEEE Trans. Pattern Anal. Mach. Intell*, 24(9): 1184–1199, 2002.

[140] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, 2003.

[141] Hiroto Saigo, Sebastian Nowozin, Tadashi Kadowaki, Taku Kudo, and Koji Tsuda. gboost: A mathematical programming approach to graph classification and regression. *Machine Learning*, 75(1):69–89, 2009.

[142] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

[143] M.I. Schlesinger. Sintaksicheskiy analiz dvumernykh zritelnikh signalov v usloviyakh pomekh (syntactic analysis of two-dimensional visual signals in noisy conditions). *Kibernetika*, 4:113–130, 1976. In Russian.

[144] Frank R. Schmidt, Eno Töppe, and Daniel Cremers. Efficient planar graph cuts with applications in computer vision. In *CVPR*. IEEE Computer Society, 2009.

[145] Henry Schneiderman and Takeo Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *CVPR*, pages 45–51, 1998.

[146] Bernhard Schölkopf and Alexander J. Smola. *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[147] Nicol N. Schraudolph and Dmitry Kamenetsky. Efficient exact inference in planar ising models. In *NIPS*. MIT Press, 2008.

[148] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 1998.

[149] Christian Schüldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local SVM approach. In *ICPR (3)*, pages 32–36, 2004.

[150] Robert Sedgewick. *Algorithms in C: Part 5: Graph algorithms*. Addison-Wesley, 3rd edition, 2002. ISBN 0-201-31663-3.

[151] Chunhua Shen and Hanxi Li. A duality view of boosting algorithms. *CoRR*, abs/0901.3590, 2009.

[152] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision*, 81(1), January 2007.

[153] Fabian Sinz, Sebastian Gerwinn, and Matthias Bethge. Characterization of the p-generalized normal distribution. *Journal of Multivariate Analysis*, 100(5):817–820, May 2009.

[154] Yang Song, Luis Goncalves, and Pietro Perona. Unsupervised learning of human motion. *IEEE Trans. Pattern Anal. Mach. Intell*, 25(7):814–827, 2003.

[155] David Sontag and Tommi Jaakkola. New outer bounds on the marginal polytope. In *NIPS*, 2007.

[156] David Sontag, Talya Meltzer, Amir Globerson, Tommi Jaakkola, and Yair Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, 2008.

[157] Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In *Introduction to Statistical Relational Learning*, chapter 4. 2007.

[158] Charles A. Sutton and Andrew McCallum. Piecewise training for undirected models. In *UAI*, pages 568–575, 2005.

[159] Charles A. Sutton and Andrew McCallum. Piecewise pseudolikelihood for efficient training of conditional random fields. In *ICML*, 2007.

[160] Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning CRFs using graph cuts. In *ECCV*, 2008.

[161] Constantino Tsallis. Possible generalization of boltzmann-gibbs statistics. *Journal of Statistical Physics*, 52(1–2):479–487, 1988.

[162] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, September 2005.

[163] Koji Tsuda, Taishin Kin, and Kiyoshi Asai. Marginalized kernels for biological sequences. In *ISMB*, pages 268–275, 2002.

[164] Takeaki Uno, Masashi Kiyomi, and Hiroki Arimura. LCM ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *FIMI*, volume 126 of *CEUR Workshop Proceedings*, 2004.

[165] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[166] Vladimir N. Vapnik and Alexey Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

[167] Jakob J. Verbeek and Bill Triggs. Scene segmentation with CRFs learned from partially labeled images. In *NIPS*. MIT Press, 2007.

[168] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Graph cut based image segmentation with connectivity priors. In *CVPR*, 2008.

[169] Paul A. Viola and Michael Jones. Robust Real-Time face detection. In *ICCV*, pages 747–747, 2001.

[170] Paul A. Viola and Michael J. Jones. Robust real time object detection. In *Workshop on Statistical and Computational Theories of Vision*, 2001.

[171] Paul A. Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[172] SVN Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *ICML*, 2006.

[173] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

[174] Martin J. Wainwright, Tommi Jaakkola, and Alan S. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, 2005.

[175] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, November 2005.

[176] Isabelle Warnesson. Applied linguistics: Optimization of semantic relations by data aggregation techniques. *Applied Stochastic Models and Data Analysis*, 1:121–141, 1985.

[177] Markus Weber, Max Welling, and Pietro Perona. Unsupervised learning of models for recognition. In *ECCV*, 2000.

[178] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2-3):249–257, 2006.

[179] Tomáš Werner. A linear programming approach to max-sum problem: A review. Research report, Center for Machine Perception, Czech Technical University, December 2005.

[180] Tomáš Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (MAP-MRF). In *CVPR*, 2008.

[181] Gerhard Winkler. *Image Analysis, Random Fields, and Dynamic Monte Carlo Methods: A Mathematical Introduction*. Springer, 1995.

[182] John M. Winn and Jamie Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *CVPR*, pages 37–44, 2006.

[183] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998.

[184] Yaser Yacoob and Michael J. Black. Parameterized modeling and recognition of activities. *Computer Vision and Image Understanding*, 73(2):232–247, 1999.

[185] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *ICDM*, 2002.

[186] Chen Yanover, Talya Meltzer, and Yair Weiss. Linear programming relaxations and belief propagation - an empirical study. *JMLR*, 7:1887–1907, 2006.

[187] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *NIPS*, pages 689–695. MIT Press, 2000.

[188] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Understanding belief propagation and its generalizations. Technical report, Mitsubishi Electric Research Laboratories, 2001.

[189] Alper Yilmaz and Mubarak Shah. Actions sketch: A novel action representation. In *CVPR*, pages 984–989. IEEE Computer Society, 2005. ISBN 0-7695-2372-2.

[190] Stella X. Yu and Jianbo Shi. Multiclass spectral clustering. In *ICCV*, pages 313–319, 2003.

[191] Lihi Zelnik-Manor and Michal Irani. Event-based analysis of video. In *CVPR*, pages 123–130. IEEE Computer Society, 2001. ISBN 0-7695-1272-0.

[192] Tong Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Transactions on Information Theory*, 49(3): 682–691, 2003.

Index