On Feature Combination for Multiclass Object Classification

Peter Gehler and Sebastian Nowozin

July 12, 2011

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

BIOLOGISCHE KYBERNETIK

Introduction

- Images may be described using a multitude of image features,
  - shape, texture, color, ...
- Each single feature alone may not be discriminative enough to yield good performance.
- Goal: classification system
  - capable of combining different image features.
  - handles multiclass problems

Introduction

- Images may be described using a multitude of image features,
  - shape, texture, color, ...
- Each single feature alone may not be discriminative enough to yield good performance.
- Goal: classification system
  - capable of combining different image features.
  - handles multiclass problems

Feature Combinations as Kernel Combination

- Kernel learning algorithms show good performance in image classification tasks.
- Question: How to enable feature combination for kernel learning algorithms?
- Idea: Associate a separate kernel with each feature.
  $\Rightarrow$ Feature combination problem becomes a kernel combination problem.

Feature Combinations as Kernel Combination

- Kernel learning algorithms show good performance in image classification tasks.
- Question: How to enable feature combination for kernel learning algorithms?
- Idea: Associate a separate kernel with each feature.
  $\Rightarrow$ Feature combination problem becomes a kernel combination problem.

Feature Combinations as Kernel Combination

- Kernel learning algorithms show good performance in image classification tasks.

- Question: How to enable feature combination for kernel learning algorithms?
- Idea: Associate a separate kernel with each feature.
  $\Rightarrow$ Feature combination problem becomes a kernel combination problem.

Learning With Multiple Kernels

- Support Vector Machines may use a single kernel function ...

$$k(x, x'), \quad x, x' \in \mathcal{X},$$

- ... a linear combination of different kernels ...

$$k(x, x') = \sum_{m=1}^{M} \beta_m k_m(x, x'), \quad \beta_m \in \mathbb{R}_+$$

- ... or a product of kernels.

$$k(x, x') = \prod_{m=1}^{M} k_m(x, x')$$

Learning With Multiple Kernels

- Support Vector Machines may use a single kernel function ...

$$k(x, x'), \quad x, x' \in \mathcal{X},$$

- ... a linear combination of different kernels ...

$$k(x, x') = \sum_{m=1}^{M} \beta_m k_m(x, x'), \quad \beta_m \in \mathbb{R}_+$$

- ... or a product of kernels.

$$k(x, x') = \prod_{m=1}^{M} k_m(x, x')$$

Learning With Multiple Kernels

- Support Vector Machines may use a single kernel function ...

$$k(x, x'), \quad x, x' \in \mathcal{X},$$

- ... a linear combination of different kernels ...

$$k(x, x') = \sum_{m=1}^{M} \beta_m k_m(x, x'), \quad \beta_m \in \mathbb{R}_+$$

- ... or a product of kernels.

$$k(x, x') = \prod_{m=1}^{M} k_m(x, x')$$

SVM → Multiple Kernel Learning (MKL)

- SVM: single kernel k
- MKL: set of kernels $\{k_1, \ldots, k_M\}$
  - learn classifier *and* combination weights β
  - can be cast as a convex optimization problem

$$f(x) = \sum_{m=1}^{M} \beta_m \sum_{i=1}^{N} \alpha_i k_m(x, x_i), \quad \sum_{m=1}^{M} \beta_m = 1$$

SVM $\rightarrow$ Multiple Kernel Learning (MKL)

- SVM: single kernel k
- MKL: set of kernels $\{k_1, \ldots, k_M\}$
  - learn classifier *and* combination weights β
  - can be cast as a convex optimization problem

$$f(x) = \sum_{m=1}^{M} \beta_m \sum_{i=1}^{N} \alpha_i k_m(x, x_i), \quad \sum_{m=1}^{M} \beta_m = 1$$

Remarks about MKL

- Special case: average ($\beta_m = \frac{1}{M}$) (no learning of β.)
- It is possible to use infinitely many kernels.
  Argyriou et.al. COLT05, Gehler&Nowozin, CVPR09
- Different MKL formulations have been proposed:
  1. Lankriet et.al. JMLR04
  2. Sonnenburg et.al JMLR06 (variant of regularization)
  3. Varma&Ray ICCV07 (extra regularization term $\sigma\|\beta\|$)
- All formulations are equivalent!
  - Zien&Ong ICML07, Kloft et.al. NIPS09

Remarks about MKL

- Special case: average ($\beta_m = \frac{1}{M}$) (no learning of β.)
- It is possible to use infinitely many kernels.
  Argyriou et.al. COLT05, Gehler&Nowozin, CVPR09
- Different MKL formulations have been proposed:
  1. Lankriet et.al. JMLR04
  2. Sonnenburg et.al JMLR06 (variant of regularization)
  3. Varma&Ray ICCV07 (extra regularization term $\sigma\|\beta\|$)
- All formulations are equivalent!
  - Zien&Ong ICML07, Kloft et.al. NIPS09

Remarks about MKL

- Special case: average ($\beta_m = \frac{1}{M}$) (no learning of β.)
- It is possible to use infinitely many kernels.
  Argyriou et.al. COLT05, Gehler&Nowozin, CVPR09
- Different MKL formulations have been proposed:
  1. Lankriet et.al. JMLR04
  2. Sonnenburg et.al JMLR06 (variant of regularization)
  3. Varma&Ray ICCV07 (extra regularization term $\sigma\|\beta\|$)
- All formulations are equivalent!
  - Zien&Ong ICML07, Kloft et.al. NIPS09

MKL classification function

$$f(x) = \sum_{m=1}^{M} \beta_m \sum_{i=1}^{N} \alpha_i k_m(x, x_i), \quad \sum_{m=1}^{M} \beta_m = 1$$

- Convex combination of SVMs all of which share the same parameters.
- A support vector x_i must be representative w.r.t. *all* kernels
- Idea: combine separate SVMs

$$f(x) = \sum_{m=1}^{M} \beta_m f_m(x), \quad \sum_{m=1}^{M} \beta_m = 1$$

MKL classification function

$$f(x) = \sum_{m=1}^{M} \beta_m \sum_{i=1}^{N} \alpha_i k_m(x, x_i), \quad \sum_{m=1}^{M} \beta_m = 1$$

- Convex combination of SVMs all of which share the same parameters.
- A support vector x_i must be representative w.r.t. *all* kernels
- Idea: combine separate SVMs

$$f(x) = \sum_{m=1}^{M} \beta_m f_m(x), \quad \sum_{m=1}^{M} \beta_m = 1$$

Multiclass ν-LP-Boost: LP-β and LP-B

- Multiclass extension of Linear-Program-Boosting

  Demiriz et.al. ML02, Weston&Watkins,ESANN99
- LP-β : mixing weights for all classes *jointly* - $\beta \in [0,1]^M$
- LP-B: mixing weights for each class *separately* - $B \in [0,1]^{MC}$

$$\min_{\beta,\xi,\rho} \quad -\rho + \frac{1}{\nu n}\sum_{i=1}^{N} \xi_i$$

$$\text{sb.t.} \quad \sum_{m=1}^{M} \beta_m f_{m,y_i}(x_i) - \max_{y_j \neq y_i} \sum_{m=1}^{M} \beta_m f_{m,y_j}(x_i) + \xi_i \geq \rho, \forall i$$

$$\sum_{m=1}^{M} \beta_m = 1, \quad \beta_m \geq 0,\ \forall m$$

$$\xi_i \geq 0, \quad \forall i.$$

Multiclass ν-LP-Boost: LP-β and LP-B

- Multiclass extension of Linear-Program-Boosting

  Demiriz et.al. ML02, Weston&Watkins,ESANN99
- LP-β : mixing weights for all classes *jointly* - $\beta \in [0,1]^M$
- LP-B: mixing weights for each class *separately* - $B \in [0,1]^{MC}$

$$\min_{B,\xi,\rho} \quad -\rho + \frac{1}{\nu n} \sum_{i=1}^{N} \xi_i$$

$$\text{sb.t.} \quad \sum_{m=1}^{M} B_m^{y_i} f_{m,y_i}(x_i) - \max_{y_j \neq y_i} \sum_{m=1}^{M} B_m^{y_j} f_{m,y_j}(x_i) + \xi_i \geq \rho, \forall i$$

$$\sum_{m=1}^{M} B_m^c = 1, \quad B_m^c \geq 0, \ \forall m, c$$

$$\xi_i \geq 0, \quad \forall i.$$

LP-Boosting training

Ideally: train jointly - but limited data available.

- 2-stage training procedure:

1. Train each one-versus-rest SVM f_m separately.
2. Obtain Cross-Validation scores for all SVMs $f_1, \ldots, f_M$.
3. Train LP-β, LP-B on Cross-Validation scores.

- Less principled, but effective.
- Small number of parameters β allows for true multiclass learning

LP-Boosting training

Ideally: train jointly - but limited data available.

- 2-stage training procedure:

1. Train each one-versus-rest SVM f_m separately.
2. Obtain Cross-Validation scores for all SVMs $f_1, \dots, f_M$.
3. Train LP-β, LP-B on Cross-Validation scores.

- Less principled, but effective.
- Small number of parameters β allows for true multiclass learning

Flower Classification: Dataset

- 17 types of flowers - 80 images per class
- 7 different precomputed kernels
- Data from Nilsback&Zissermann CVPR06

Flower Classification: Results

Single feature			Combinations		
Kernel	Accuracy	Time(s)	Method	Accuracy	Time(s)
Colour	60.9 ± 2.1	3	product	85.5 ± 1.2	2
Shape	70.2 ± 1.3	4	averaging	84.9 ± 1.9	10
Texture	63.7 ± 2.7	3	MKL	85.2 ± 1.5	97
HOG	58.5 ± 4.5	4	LP-β	85.5 ± 3.0	80
HSV	61.3 ± 0.7	3	LP-B	85.4 ± 2.4	98
siftint	70.6 ± 1.6	4			
siftbdy	59.4 ± 3.3	5			

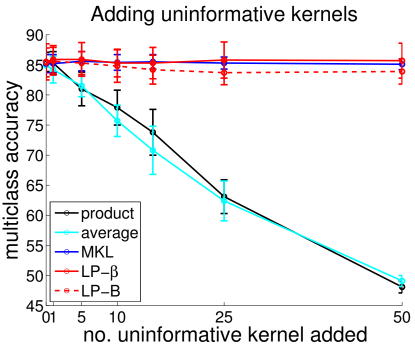
- Combination of features improves performance.
- All combination methods perform equally well.
- Time - combined time for model selection, training and testing

Flower Classification: Results

Single feature			Combinations		
Kernel	Accuracy	Time(s)	Method	Accuracy	Time(s)
Colour	60.9 ± 2.1	3	product	85.5 ± 1.2	2
Shape	70.2 ± 1.3	4	averaging	84.9 ± 1.9	10
Texture	63.7 ± 2.7	3	MKL	85.2 ± 1.5	97
HOG	58.5 ± 4.5	4	LP-β	85.5 ± 3.0	80
HSV	61.3 ± 0.7	3	LP-B	85.4 ± 2.4	98
siftint	70.6 ± 1.6	4			
siftbdy	59.4 ± 3.3	5			

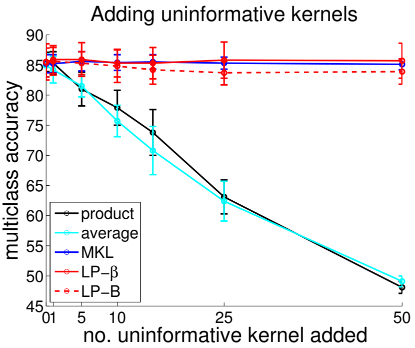
- Combination of features improves performance.
- All combination methods perform equally well.
- Time - combined time for model selection, training and testing

Flower Classification: Adding uninformative kernels



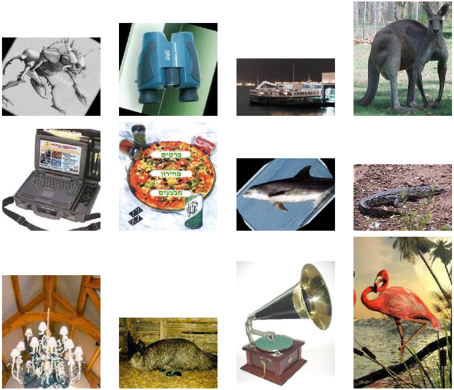
- Adding more and more kernels computed on pure noise
- In this scenario sparse kernel selection is useful.

Flower Classification: Adding uninformative kernels



- Adding more and more kernels computed on pure noise
- In this scenario sparse kernel selection is useful.

Visual Object Classification: Caltech 101/256



- 102/256 categories of visual object categories

Visual Object classification: Image Features

- Histogram of SIFTs
- PHOG Bosch et.al. CIVR07
- LBP Ojala et.al. PAMI02
- Region Covariance Tuzel et.al. CPVR07
- V1S+ Pinto et.al. PLOS08

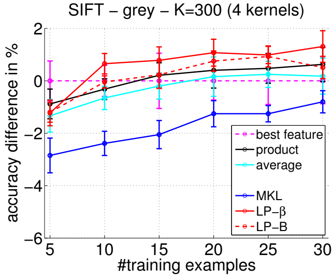
- ... and spatial pyramid representation (4 levels)

Visual Object classification: Results on Caltech 101

Two scenarios:

1. Combining similar features
2. Combining diverse features

Performance with respect to best single feature



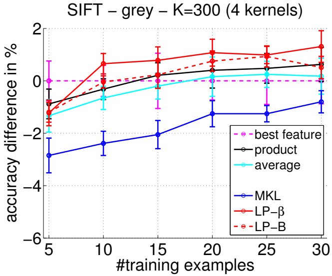
similar: almost no gain diverse:combination helps

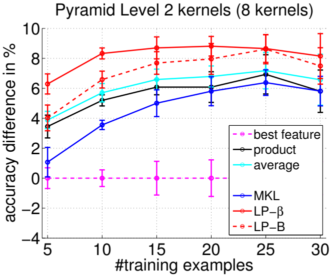
Visual Object classification: Results on Caltech 101

Two scenarios:

1. Combining similar features
2. Combining diverse features

Performance with respect to best single feature

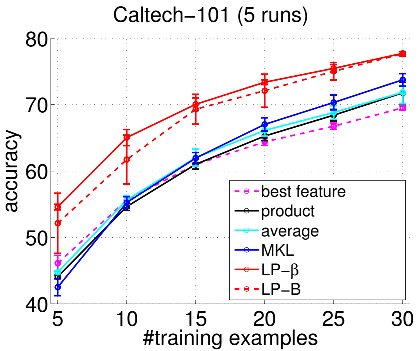




similar: almost no gain

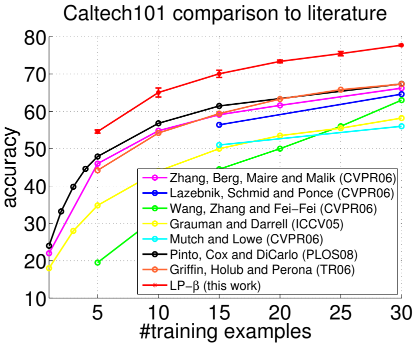
diverse:combination helps

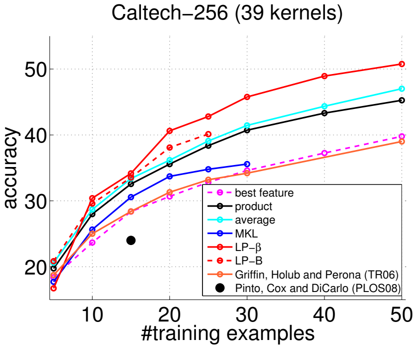
Caltech 101 - combining 39 kernels



- No significant improvement of MKL over baselines
- LP-β yields sparse mixing weights for *all* classes (7 out of 39)

Caltech 101/256 comparison





- Over 10% improvement using LP-β
- Latest LP-β results $\approx +5\%$ after adding more features

  Vedaldi&Fulkerson `www.vlfeat.org`

Conclusion

- Kernel combinations can improve performance, thanks to strong features!
  - Expect performance gain if combining diverse features.
  - If in doubt: average strong features - simple and efficient.
  - In presence of uninformative kernels use selection techniques.
- MKL not as effective as may have been thought,
  $\Rightarrow$ use proper model selection instead!
- For example LP-β : multiclass, sparse, easily expandable and simple.

- Code and Data available at `www.ee.ethz.ch/~pgehler`
- Thanks to C. Lampert and N. Pinto

Conclusion

- Kernel combinations can improve performance, thanks to strong features!
  - Expect performance gain if combining diverse features.
  - If in doubt: average strong features - simple and efficient.
  - In presence of uninformative kernels use selection techniques.
- MKL not as effective as may have been thought,
  $\Rightarrow$ use proper model selection instead!
- For example LP-β : multiclass, sparse, easily expandable and simple.
- Code and Data available at `www.ee.ethz.ch/~pgehler`
- Thanks to C. Lampert and N. Pinto

Conclusion

- Kernel combinations can improve performance, thanks to strong features!
  - Expect performance gain if combining diverse features.
  - If in doubt: average strong features - simple and efficient.
  - In presence of uninformative kernels use selection techniques.
- MKL not as effective as may have been thought,
  $\Rightarrow$ use proper model selection instead!
- For example LP-β : multiclass, sparse, easily expandable and simple.
- Code and Data available at `www.ee.ethz.ch/~pgehler`
- Thanks to C. Lampert and N. Pinto