Ph.D. Defense Presentation, CSE Dept, CUHK

Statistical Machine Learning for Data Mining and Collaborative Multimedia Retrieval

Presented by Steven C.H. Hoi

Supervisor: Prof. Michael R Lyu

The Chinese University of Hong Kong

Date: 28 Aug, 2006
Time: 4:00 – 6:00 p.m.

Outline

- Background
- Contributions
- Learning Unified Kernel Machines
- Batch Mode Active Learning
- Collaborative Multimedia Retrieval
- Conclusions
- Future Work

Background

- Statistical Machine Learning
  - Supervised Learning
  - Unsupervised Learning
  - Semi-Supervised Learning
  - Active Learning
  - Distance Metric Learning
  - Others (reinforcement learning, etc.)

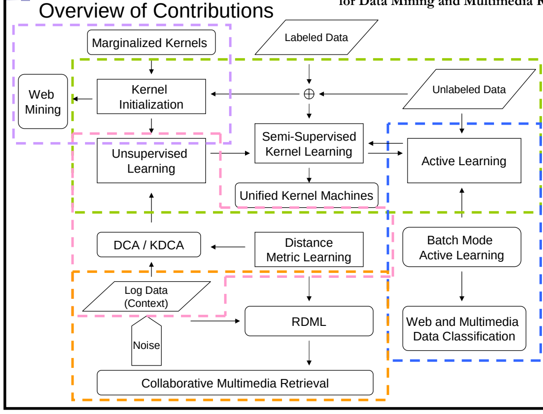
Background

- Challenging Issues
  - How to unify a variety of machine learning techniques in an effective fashion?
  - How to perform active learning efficiently and effectively?
  - How to learn distance metrics from context data?
  - How to develop appropriate metric learning techniques for real-world applications?

Contributions

- Learning Unified Kernel Machines
  - Spectral Kernel Learning
  - Unified Kernel Logistic Regression
  - Kernel Design via Marginalized Kernel
  - Publications: KDD 06, WWW 06
- Batch Mode Active Learning
  - BMAL for Text and Image Categorization
  - Publications: ICML 06, WWW 06
- Distance Metric Learning
  - Discriminative Component Analysis (DCA) and KDCA
  - Publication: CVPR 06
- Collaborative Multimedia Retrieval
  - Learning Log-Based Relevance Feedback
  - Learning Reliable Distance Metrics
  - Publications: MM04, EMMA 05, TKDE 06, MMSJ 06

Overview of Contributions



Part I: UKM

Part I: Learning Unified Kernel Machines

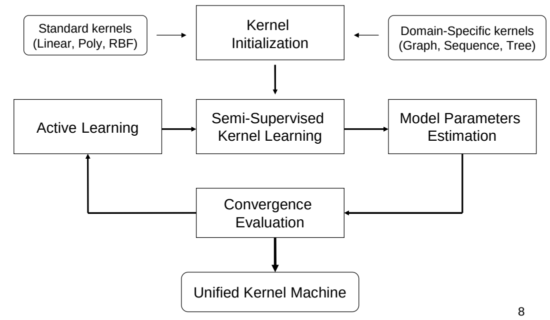
- Motivation of Our Framework
  - Kernel machines play an important role in the state-of-the-art machine-learning techniques for data mining.
  - Supervised Learning
    - Support Vector Machines (SVM)
    - Kernel Logistic Regressions (KLR)
    - Regularized Least-Square Classifiers (RLS)
  - Unsupervised Learning
    - Spectral Clustering, Kernel PCA, …
  - Active Learning
    - Margin-Based Active Learning with Kernel Machines, etc.
  - How to combine these kernel machine-learning techniques in a unified solution?

7

Part I: UKM

Learning Unified Kernel Machines

- A Unified Framework



8

Part I: UKM

Semi-Supervised Kernel Learning

- Goal
  - To learn an effective kernel (matrix) from both labeled and unlabeled data
- Theoretical Principles
  - Unsupervised Kernel Design
    - Learning Kernel from unlabeled data
  - Kernel Target Alignment
    - Learning Kernel from labeled data

9

Part I: UKM

Semi-Supervised Kernel Learning

- Overview of Kernel Machine Learning
  - Supervised Learning
    - Given l training examples $(\mathbf{x}_1, y_1), \ldots (\mathbf{x}_l, y_l)$, one can train a prediction function p in the RKHS by the following formula

$$\hat{p} = \arg\inf_{p \in \mathcal{H}} \Bigg(\underbrace{\frac{1}{l}\sum_{i=1}^{l} \mathcal{L}(p(\mathbf{x}_i), y_i)}_{\text{Empirical loss term}} + \underbrace{\lambda \|p\|^2_{\mathcal{H}}}_{\text{Regularization term}} \Bigg) \quad (1)$$

    - The solution of (1) can be represented as:

$$\hat{p}(\mathbf{x}) = \sum_{i=1}^{l} \hat{\alpha}_i k(\mathbf{x}_i, \mathbf{x})$$

$$\alpha = \arg\inf_{\alpha \in \mathbb{R}^l} \left(\frac{1}{l}\sum_{i=1}^{l} L(p(\mathbf{x}_i), y_i) + \lambda \sum_{i,j=1}^{l} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \right) \quad (2)$$

10

Part I: UKM

Semi-Supervised Kernel Learning

- Overview of Kernel Machine Learning
  - Semi-Supervised Learning
    - Given l training examples $(\mathbf{x}_1, y_1), \ldots (\mathbf{x}_l, y_l)$, and $(n-l)$ unlabeled data examples $(\mathbf{x}_{l+1}, \mathbf{x}_{l+2}, \ldots, \mathbf{x}_n)$, let f be n-dimensional real vector, which is learned by the following semi-supervised learning method:

$$\hat{f} = \arg\inf_{f \in \mathbb{R}^n} \left(\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(f_i, y_i) + \lambda f^\top K^{-1} f \right) \quad (3)$$

    - **Theorem** (Zhang et al., NIPS'05): The solution of (3) is equivalent to the solution of (1):

$$\hat{f}_j = \hat{p}(\mathbf{x}_j) \quad j = 1, \ldots, n.$$

11

Part I: UKM

Unsupervised Kernel Design

- The equivalence theorem shows that, in order to exploit the unlabeled data, we can consider the following supervised learning approach with unsupervised kernel design:
  - (1) Design a new kernel K' using unlabeled data
  - (2) Apply the new K' in the supervised learning formula
- Spectral Kernel Design

$$K = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^\top \quad \Longrightarrow \quad \bar{K} = \sum_{i=1}^{n} g(\lambda_i) \mathbf{v}_i \mathbf{v}_i^\top$$

- **Principle**: A kernel with **faster spectra decay** should be more preferred. (Zhang et al., NIPS'05)

12

Kernel Target Alignment

- **Kernel Alignment (Cristianini et al. 2002):** *The empirical alignment of two given kernels K1 and K2 with respect to a sample set is the following quantity:*

$$\hat{A}(K_1, K_2) = \frac{\langle K_1, K_2\rangle_F}{\sqrt{\langle K_1, K_1\rangle_F \langle K_2, K_2\rangle_F}}$$

where $\langle K_1, K_2\rangle_F = \sum_{i,j=1}^{n} k_1(\mathbf{x}_i, \mathbf{x}_j) k_2(\mathbf{x}_i, \mathbf{x}_j)$

- Target Kernel
  - Let **y**=[y_1,…,y_l]' be a label vector of training data, for binary classification, the target kernel can be defined as:

$$T = \mathbf{y}\mathbf{y}^\top \qquad \mathrm{y} = [-1 \;\; 1 \;\; -1]^\mathrm{T} \qquad T = \mathrm{y}\mathrm{y}^\mathrm{T} = \begin{pmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{pmatrix}$$

Kernel Target Alignment

- Let *K* the kernel matrix of all data, which can be represented as the following structure

$$K = \begin{pmatrix} K_{tr} & K_{trt} \\ K_{trt}^\top & K_t \end{pmatrix}$$

- **Principle**: A better kernel can be optimized by maximizing the following kernel target alignment:

$$\hat{A}(K_{tr}, T) = \frac{\langle K_{tr}, T\rangle_F}{\sqrt{\langle K_{tr}, K_{tr}\rangle_F \langle T, T\rangle_F}}$$

Spectral Kernel Learning

- Principles
  - Maximizing **<u>kernel target alignment</u>** meanwhile keeping **<u>fast spectra decay</u>**!
- Formulation of Algorithm

$$\begin{array}{ll} \max\limits_{\bar{K},\mu} & \hat{A}(K_{tr}, T) \\ \text{subject to} & \bar{K} = \sum_{i=1}^{d} \mu_i \mathbf{v}_i \mathbf{v}_i^\top \\ & trace(\bar{K}) = 1 \\ & \mu_i \geq 0, \\ & \mu_i \geq C\mu_{i+1}, i = 1, \ldots, d-1\,, \end{array}$$

top **d** eigenvectors of initial kernel

C is a decay factor to enforce a faster decay rate of spectra **(C>=1)**

Spectral Kernel Learning

- Formulation of Algorithm (cont')

$$\hat{A}(K_{tr}, T) = \frac{\langle K_{tr}, T\rangle_F}{\sqrt{\langle K_{tr}, K_{tr}\rangle_F \langle T, T\rangle_F}}$$

$$\frac{\langle \bar{K}_{tr}, T\rangle_F}{\sqrt{\langle \bar{K}_{tr}, \bar{K}_{tr}\rangle_F}}$$

fix numerator to1

$$\begin{array}{ll} \min\limits_{\mu} & \sqrt{\langle \bar{K}_{tr}, \bar{K}_{tr}\rangle_F} \\ \text{subject to} & \bar{K} = \sum_{i=1}^{d} \mu_i \mathbf{v}_i \mathbf{v}_i^\top \\ & \langle \bar{K}_{tr}, T\rangle_F = 1 \\ & \mu_i \geq 0, \\ & \mu_i \geq C\mu_{i+1}, i = 1 \ldots d-1\,. \end{array}$$

Let $D = [vec(V_{1,tr}) \ldots vec(V_{d,tr})] \quad V_i = \mathbf{v}_i \mathbf{v}_i^\top$

$$\begin{array}{ll} \min\limits_{\mu} & ||D\mu|| \\ \text{subject to} & vec(T)^\top D\mu = 1 \\ & \mu_i \geq 0 \\ & \mu_i \geq C\mu_{i+1}, i = 1 \ldots d-1\,. \end{array}$$

Spectral Kernel Learning

- Formulation of Algorithm (cont')

$$\begin{array}{ll} \min\limits_{\mu} & \mu^\top D^\top D\mu \\ \text{subject to} & vec(T)^\top D\mu = 1 \\ & \mu_i \geq 0 \\ & \mu_i \geq C\mu_{i+1}, i = 1 \ldots d-1\,. \end{array}$$

This is a standard Quadratic Programming (QP) problem.

Spectral Kernel Learning

- Connections to Other Kernel Techniques
  - Spectral Kernel Learning (SKL)

$$\bar{K} = \sum_{i=1}^{n} g(\lambda_i) \mathbf{v}_i \mathbf{v}_i^\top$$

  - Cluster Kernel ([1, …,1, 0, …,0], Spectral Clustering)

$$\mu_i = \begin{cases} 1 & \text{for} \quad i \leq d \\ 0 & \text{for} \quad i > d \end{cases}$$

  - Truncated Kernel (top eigen components, Kernel PCA)

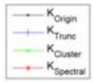
$$\mu_i = \begin{cases} \lambda_i & \text{for} \quad i \leq d \\ 0 & \text{for} \quad i > d \end{cases}$$

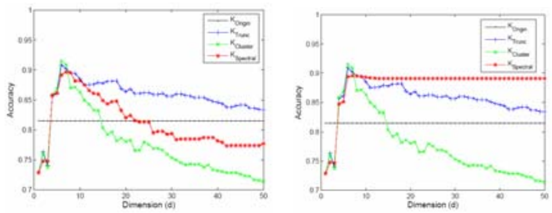
  - When setting C=1, d=n, and assuming the initial kernel K is constructed from graph laplacian L, our SKL method is equivalent to the order-constrained graph kernel (Jerry Zhu, NIPS'2005)

Part I: UKM

Spectral Kernel Learning

- Empirical Observations
  - On "Ionosphere" dataset, initial RBF Kernel



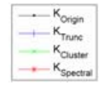


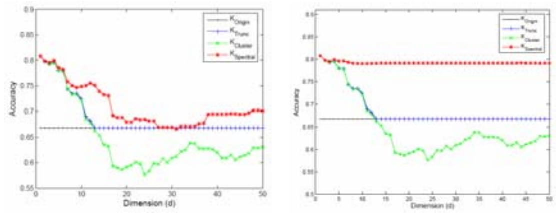
(a) C = 1 (b) C = 2

Part I: UKM

Spectral Kernel Learning

- Empirical Observations
  - On "Heart" dataset, initial linear kernel





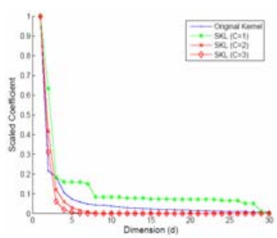
(a) C = 1 (b) C = 2

Part I: UKM

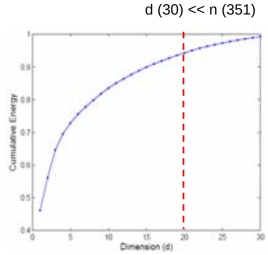
Spectral Kernel Learning

- Empirical Observations

d (30) << n (351)



Kernel Spectra



Cumulative Eigen Energy

Part I: UKM

Unified Kernel Logistic Regression

- Unified KLR Paradigm for Classification
  - 1) Calculate an initial kernel matrix K0
  - 2) Learn a new kernel by the SKL algorithm

$$K \leftarrow \text{Spectral\_Kernel}(K_0, L, U);$$

  - 3) Train a standard KLR classifier with new K

$$\min_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^{l} \ln(1 + e^{-y_i f(x_i)}) + \frac{\lambda}{2} \|f\|^2_{\mathcal{H}_K} \qquad f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$

  - 4) Active learning to seek informative data

$$H(\mathbf{x}; \alpha, K) = -\sum_{i=1}^{N_C} p(C_i \mid \mathbf{x}) \log(p(C_i \mid \mathbf{x}))$$

Part I: UKM

Unified Kernel Logistic Regression

- Remarks
  - It is an open issue to determine the convergence condition!
  - We simply repeat the learning procedure in a fixed step.
  - Active learning may be done more elegantly, e.g., to search a batch of informative examples.

Algorithm: Unified Kernel Logistic Regression
Input
- K_0: Initial normalized kernel
- L: Set of labeled data
- U: Set of unlabeled data

Repeat
- Spectral Kernel Learning
  $K \leftarrow \text{Spectral\_Kernel}(K_0, L, U)$;
- KLR Parameter Estimation
  $\alpha \leftarrow \text{KLR\_Solver}(L, K)$;
- Convergence Test
  If (*converged*), **Exit** Loop;
- Active Learning
  $\mathbf{x}^* \leftarrow max_{\mathbf{x} \in U}\ H(\mathbf{x}; \alpha, K)$
  $L \leftarrow L \cup \{\mathbf{x}^*\}$, $U \leftarrow U - \{\mathbf{x}^*\}$

Until converged.
Output
- $\text{UKLR} = \mathcal{M}(K, \alpha)$.

Part I: UKM

Experimental Results

- Experimental Testbed and Setups
  - Four UCI datasets

Table 1: List of UCI machine learning datasets.

Dataset	#Instances	#Features	#Classes
Heart	270	13	2
Ionosphere	351	34	2
Sonar	208	60	2
Wine	178	13	3

  - Two objectives of experimental evaluation
    - How effective is our SKL algorithm in learning semi-supervised kernels?
    - How effective is our UKLR scheme compared with traditional classification solutions?

Experimental Results

- Semi-Supervised Kernel Learning
  - Compared Kernels
    - 3 standard kernels
      - Linear, Quadratic, RBF
    - 5 semi-supervised kernels
      - 3 SKL methods with different initial kernels
      - 2 Order-constraint graph kernels
    - Standard KLR classifier for classification
  - Settings
    - Fix decay factor C (C>1)
    - Set dimension cut-off d = 20
    - 20 trials for each experimental comparison

Experimental Results

- Semi-Supervised Kernel Learning

Table 2. Classification performance of different kernels using KLR classifiers on UCI datasets. The mean accuracies and standard errors are shown in the table. Each cell in the table has two rows. The upper row shows the test set accuracy with standard error; the lower row gives the average time used in kernel learning.

Train Size	Standard Kernels: Linear	Quadratic	RBF	Semi-Supervised Kernels: Order	Imp-Order	SKL(Linear)	SKL(Quad)	SKL(RBF)
Heart								
10	67.19 ± 1.94 —	71.90 ± 1.23 —	70.04 ± 1.61 —	63.60 ± 1.94 (0.67)	63.60 ± 1.94 (0.81)	70.58 ± 1.63 (0.07)	72.33 ± 1.60 (0.06)	**73.37 ± 1.50** (0.06)
20	67.40 ± 1.87 —	70.36 ± 1.51 —	72.64 ± 1.37 —	65.88 ± 1.69 (0.71)	65.88 ± 1.69 (0.81)	76.26 ± 1.29 (0.06)	75.36 ± 1.30 (0.06)	**76.30 ± 1.33** (0.06)
30	75.42 ± 0.88 —	70.71 ± 0.83 —	74.40 ± 0.70 —	71.73 ± 1.14 (0.95)	71.73 ± 1.14 (0.97)	78.42 ± 0.59 (0.06)	78.65 ± 0.52 (0.06)	**79.23 ± 0.58** (0.06)
40	78.24 ± 0.89 —	71.28 ± 1.10 —	78.48 ± 0.77 —	75.48 ± 0.69 (1.35)	75.48 ± 0.69 (1.34)	80.61 ± 0.45 (0.07)	80.26 ± 0.45 (0.07)	**80.98 ± 0.51** (0.07)
Ionosphere								
10	73.71 ± 1.27 —	71.30 ± 1.70 —	73.56 ± 1.91 —	71.86 ± 2.79 (0.90)	71.86 ± 2.79 (0.87)	75.53 ± 1.75 (0.05)	69.25 ± 1.67 (0.05)	**83.36 ± 1.31** (0.05)
20	75.62 ± 1.24 —	76.00 ± 1.58 —	81.71 ± 1.74 —	83.04 ± 2.10 (0.87)	83.04 ± 2.10 (0.79)	78.78 ± 1.60 (0.05)	80.30 ± 1.77 (0.06)	**88.55 ± 1.32** (0.05)
30	76.59 ± 0.82 —	79.10 ± 1.46 —	86.21 ± 0.84 —	87.20 ± 1.16 (0.93)	87.20 ± 1.16 (0.97)	82.18 ± 0.56 (0.05)	83.08 ± 1.36 (0.05)	**90.39 ± 0.84** (0.05)
40	77.97 ± 0.79 —	82.93 ± 1.33 —	89.39 ± 0.65 —	90.56 ± 0.64 (1.34)	90.56 ± 0.64 (1.38)	83.26 ± 0.53 (0.05)	87.03 ± 1.02 (0.04)	**92.14 ± 0.46** (0.04)

Experimental Results

- Semi-Supervised Kernel Learning (cont')

Train Size	Standard Kernels: Linear	Quadratic	RBF	Semi-Supervised Kernels: Order	Imp-Order	SKL(Linear)	SKL(Quad)	SKL(RBF)
Sonar								
10	63.01 ± 1.47 —	62.85 ± 1.53 —	60.76 ± 1.80 —	59.67 ± 0.89 (0.63)	59.67 ± 0.89 (0.63)	64.27 ± 1.91 (0.08)	64.37 ± 1.64 (0.07)	**65.30 ± 1.78** (0.07)
20	68.09 ± 1.11 —	69.55 ± 1.22 —	67.63 ± 1.15 —	64.68 ± 1.57 (0.68)	64.68 ± 1.57 (0.82)	70.61 ± 1.14 (0.07)	69.79 ± 1.30 (0.07)	**71.76 ± 1.07** (0.08)
30	66.40 ± 1.06 —	69.80 ± 0.93 —	68.23 ± 1.48 —	66.54 ± 0.79 (0.88)	66.54 ± 0.79 (1.02)	70.20 ± 1.48 (0.07)	68.48 ± 1.59 (0.07)	**71.69 ± 0.87** (0.07)
40	64.94 ± 0.74 —	71.37 ± 0.52 —	71.61 ± 0.89 —	69.82 ± 0.82 (1.14)	69.82 ± 0.82 (1.20)	72.35 ± 1.06 (0.07)	71.28 ± 0.96 (0.08)	**72.89 ± 0.68** (0.07)
Wine								
10	82.26 ± 2.18 —	85.89 ± 1.73 —	87.80 ± 1.63 —	87.44 ± 2.21 (1.02)	87.44 ± 2.21 (0.86)	86.49 ± 2.48 (0.09)	86.55 ± 2.40 (0.09)	**93.72 ± 0.65** (0.09)
20	86.39 ± 1.39 —	86.96 ± 1.30 —	93.77 ± 0.99 —	92.72 ± 1.32 (0.92)	92.72 ± 1.32 (0.91)	88.86 ± 3.31 (0.09)	93.39 ± 0.59 (0.09)	**95.63 ± 0.45** (0.09)
30	92.50 ± 0.76 —	87.43 ± 0.63 —	94.63 ± 0.50 —	93.99 ± 0.53 (1.28)	93.99 ± 0.53 (1.27)	93.99 ± 1.55 (0.09)	94.63 ± 0.50 (0.10)	**96.32 ± 0.33** (0.09)
40	94.96 ± 0.65 —	88.80 ± 0.93 —	96.38 ± 0.35 —	96.34 ± 0.33 (1.41)	96.34 ± 0.33 (1.39)	95.98 ± 0.41 (0.08)	95.25 ± 0.47 (0.08)	**96.74 ± 0.27** (0.10)

Experimental Results

- Unified Kernel Machines
  - Compared Schemes
    - KLR (initial classifier)
    - KLR + Rand (initial KLR classifier with additional labeled examples sampled randomly)
    - KLR + Active (initial KLR classifier with additional labeled examples by active learning)
    - UKLR (Unified Kernel Logistic Regression)

Experimental Results

- Unified Kernel Machines

Table 3: Classification performance of different classification schemes on four UCI datasets. The mean accuracies and standard errors are shown in the table. "KLR" represents the initial classifier with the initial train size; other three methods are trained with additional **10 random/active** examples.

Train Size	Linear Kernel: KLR	KLR+Rand	KLR+Active	UKLR	RBF Kernel: KLR	KLR+Rand	KLR+Active	UKLR
Heart								
10	67.19 ± 1.94	68.22 ± 2.16	69.22 ± 1.71	**77.24 ± 0.74**	70.04 ± 1.61	72.24 ± 1.23	75.36 ± 0.60	**78.44 ± 0.88**
20	67.40 ± 1.87	73.79 ± 1.29	73.77 ± 1.27	**79.27 ± 1.00**	72.64 ± 1.37	75.10 ± 0.74	76.23 ± 0.81	**79.88 ± 0.90**
30	75.42 ± 0.88	77.70 ± 0.92	78.65 ± 0.62	**81.13 ± 0.42**	74.40 ± 0.70	76.43 ± 0.68	76.61 ± 0.61	**81.48 ± 0.41**
40	78.24 ± 0.89	79.30 ± 0.75	80.18 ± 0.79	**82.55 ± 0.28**	78.48 ± 0.77	78.50 ± 0.53	79.95 ± 0.62	**82.66 ± 0.36**
Ionosphere								
10	73.71 ± 1.27	74.89 ± 0.95	75.91 ± 0.96	**77.31 ± 1.23**	73.56 ± 1.91	82.57 ± 1.78	82.76 ± 1.37	**90.48 ± 0.83**
20	75.62 ± 1.24	77.09 ± 0.67	77.51 ± 0.66	**81.42 ± 1.10**	81.71 ± 1.74	85.95 ± 1.30	88.22 ± 0.78	**91.28 ± 0.94**
30	76.59 ± 0.82	78.41 ± 0.79	77.91 ± 0.77	**84.49 ± 0.37**	86.21 ± 0.84	89.04 ± 0.66	90.32 ± 0.56	**92.35 ± 0.59**
40	77.97 ± 0.79	79.05 ± 0.49	80.30 ± 0.79	**84.49 ± 0.40**	89.39 ± 0.65	90.55 ± 0.59	91.83 ± 0.49	**93.89 ± 0.28**
Sonar								
10	61.19 ± 1.56	63.72 ± 1.65	65.51 ± 1.55	**66.12 ± 1.94**	57.40 ± 1.48	60.19 ± 1.32	59.49 ± 1.46	**67.13 ± 1.58**
20	67.31 ± 1.07	68.85 ± 0.84	69.38 ± 1.05	**71.60 ± 0.91**	62.93 ± 1.36	64.72 ± 1.24	64.52 ± 1.07	**72.30 ± 0.98**
30	66.10 ± 1.08	67.59 ± 1.14	69.79 ± 0.86	**71.40 ± 0.80**	63.03 ± 1.32	63.72 ± 1.51	66.67 ± 1.53	**72.26 ± 0.98**
40	66.34 ± 0.82	68.16 ± 0.81	70.19 ± 0.90	**73.04 ± 0.69**	66.70 ± 1.25	68.70 ± 1.19	67.56 ± 0.90	**73.16 ± 0.88**
Wine								
10	82.26 ± 2.18	87.31 ± 1.01	**89.05 ± 1.07**	87.31 ± 1.03	87.80 ± 1.63	92.75 ± 1.27	94.49 ± 0.54	**94.87 ± 0.49**
20	86.39 ± 1.39	93.99 ± 0.40	93.82 ± 0.71	**94.43 ± 0.54**	93.77 ± 0.99	95.57 ± 0.38	**97.13 ± 0.18**	96.76 ± 0.26
30	92.50 ± 0.76	95.25 ± 0.47	**96.96 ± 0.40**	96.12 ± 0.47	94.63 ± 0.50	96.27 ± 0.35	97.17 ± 0.38	**97.21 ± 0.26**
40	94.96 ± 0.65	96.21 ± 0.63	97.54 ± 0.37	**97.70 ± 0.34**	96.38 ± 0.35	96.33 ± 0.45	97.97 ± 0.23	**98.12 ± 0.21**

Summary of Part I

- We presented a framework of learning unified kernel machines (UKM) for classification.
- A new semi-supervised kernel learning algorithm was proposed, which is related to an equivalent quadratic programming (QP) problem.
- A classification paradigm was developed by applying our UKM framework on the KLR model.
- Empirical evaluations are conducted on several UCI datasets.

Part II: Batch Mode Active Learning for Text Categorization

- Motivation
  - Text Categorization
  - Logistic Regression and Active Learning
- Batch Mode Active Learning
  - Theoretical Foundation
  - Convex Optimization Formulation
  - Eigen Space Simplification
  - Bound Optimization Algorithm
- Experimental Results
- Summary

Motivation

- Text Categorization
  - Problem: assign documents to predefined topics
  - Significances
    - Core Web data mining technique
    - Applications: category browsing, vertical search, etc.
  - Challenges
    - To build efficient classifiers
    - To minimize human labeling effort

Motivation

- Logistic Regression
  - Efficiency for Training and Prediction
  - Natural Probability Output
  - State-of-the-art performance, etc…
  - Linear model

$$p(y|\mathbf{x}) = \frac{1}{1+\exp(-y(\mathbf{w}^T\mathbf{x}+b))}$$

where $y \in \{+1, -1\}$ is the class label.
Simplified notation:

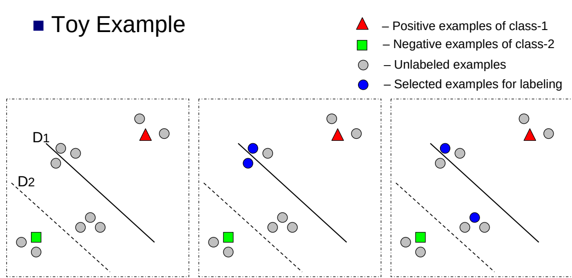
$$p(y|\mathbf{x}) = \frac{1}{1+\exp(-y\alpha^T\mathbf{x})}$$

Motivation

- Active Learning
  - Goal: to find most informative unlabeled data
  - Traditional Methodology
    - Choose one unlabeled example for labeling
    - Retrain the classifier with the additional example
    - Limitation: only one example, large retraining cost
  - Batch Mode Active Learning
    - To find a batch of most informative unlabeled examples

Batch Mode Active Learning

- Toy Example



(a) Binary classification example (b) Margin-based active learning (c) Batch mode active learning

Theoretical Foundation

- Main Idea:
  - Based on the theoretical framework of maximization of Fisher information
- Problem Setting

In a probabilistic classification framework, assume the classification model is a semi-parametric form

$$p(x, y|\alpha) = p(x)p(y|x, \alpha)$$

For example, the logistic regression model:

$$p(x, y|\alpha) = \frac{1}{1+\exp(-y\alpha^T x)}p(x)$$

Theoretical Foundation

- The problem of batch mode active learning can be regarded as a problem to seek a resample distribution **q(x)** of the unlabeled data.
- The examples with large resampling probabilities will be selected as the most informative ones for labeling.
- According to statistical estimation theory, active learning should consider a resample distribution ***q(x)*** that maximizes the following Fisher information

$$I_q(\alpha) = -\int q(\mathbf{x})d\mathbf{x} \int p(y|\mathbf{x},\alpha)\frac{\partial^2}{\partial\alpha^2}\log p(y|\mathbf{x},\alpha)dy$$

Theoretical Foundation

- The maximization of Fisher information is equivalent to find the resample distribution ***q(x)*** that minimizes the ratio of two Fisher information matrixces:

$$q^* = \arg\min_q \mathrm{tr}(I_q(\alpha)^{-1}I_p(\alpha))$$

- For the logistic regression model, the Fisher information matrix can be expressed as:

$$I_q(\alpha) = -\int q(\mathbf{x})\sum_{y=\pm1} p(y|\mathbf{x})\frac{\partial^2}{\partial\alpha^2}\log p(y|\mathbf{x})d\mathbf{x} = \int \frac{1}{1+\exp(\alpha^T\mathbf{x})}\frac{1}{1+\exp(-\alpha^T\mathbf{x})}\mathbf{x}\mathbf{x}^T q(\mathbf{x})d\mathbf{x}$$

- We replace the integration in the above equation with the summation over the unlabeled data:

$$I_q(\hat\alpha) = \sum_{i=1}^n \pi_i(1-\pi_i)\mathbf{x}_i\mathbf{x}_i^T q_i + \delta I_d \quad I_p(\hat\alpha) = \frac{1}{n}\sum_{i=1}^n \pi_i(1-\pi_i)\mathbf{x}_i\mathbf{x}_i^T + \delta I_d$$

$$\pi_i = p(-|\mathbf{x}_i) = \frac{1}{1+\exp(\hat\alpha^T\mathbf{x}_i)} \quad \sum_{i=1}^n q_i = 1$$

Convex Optimization Formulation

- Rewrite the objective function $\mathbf{tr}(I_q^{-1}I_p)$ as

$$\mathbf{tr}(I_p^{1/2}I_q^{-1}I_p^{1/2})$$

- Introduce a slack matrix $M \in \mathbf{R}^{n\times n}$,then turn the original problem into the following optimization:

$$\begin{aligned} \min_{\mathbf{q},M} \quad & \mathbf{tr}(M) \\ \text{s. t.} \quad & M \succeq I_p^{1/2}I_q^{-1}I_p^{1/2} \\ & \sum_{i=1}^n q_i = 1, q_i \geq 0, i = 1,\dots,n \end{aligned}$$

- In the above, we use $\mathbf{tr}(A) \geq \mathbf{tr}(B)$ if $A \succeq B$

Convex Optimization Formulation

- By the Schur complementary theorem, i.e.,

$$D \succeq AB^{-1}A^T \Leftrightarrow \begin{pmatrix} B & A^T \\ A & D \end{pmatrix} \succeq 0 \quad \text{if } B \succeq 0.$$

- we turn it into the following optimization :

$$\begin{aligned} \min_{\mathbf{q},M} \quad & \mathbf{tr}(M) \\ \text{s. t.} \quad & \begin{pmatrix} I_q & I_p^{1/2} \\ I_p^{1/2} & M \end{pmatrix} \succeq 0 \\ & \sum_{i=1}^n q_i = 1, q_i \geq 0, i = 1,\dots,n \end{aligned}$$

Convex Optimization Formulation

- The final optimization problem can be expressed

$$\begin{aligned} \min_{\mathbf{q},M} \quad & \mathbf{tr}(M) \\ \text{s. t.} \quad & \sum_{i=1}^n q_i\pi_i(1-\pi_i)\begin{pmatrix} \mathbf{x}_i\mathbf{x}_i^T & I_p^{1/2} \\ I_p^{1/2} & M \end{pmatrix} \succeq 0 \\ & \sum_{i=1}^n q_i = 1, q_i \geq 0, i = 1,\dots,n \end{aligned}$$

- The above problem belongs to the family of Semi-definite programming (SDP) and can be solved by convex optimization techniques.

Eigen Space Simplification

- Directly solving the above optimization problem may be computationally expensive for the large-size slack matrix variable of **M.**
- In order to reduce the computational complexity, we propose an Eigen space simplification method to make the solution simpler and more effective.
- We assume that **M** is expanded in the Eigen space of the Fisher information matrix $\mathbf{I_p}$.

Part II: BMAL

Eigen Space Simplification

- Let $\{(\lambda_1, \mathbf{v}_1), \ldots, (\lambda_s, \mathbf{v}_s)\}$ be the top ***s*** eigen vectors of the Fisher information matrix $\mathbf{I_p}$, where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_s$, then we assume the matrix **M** has the following form:

$$M = \sum_{k=1}^{s} \gamma_k \mathbf{v}_k \mathbf{v}_k^T \qquad \gamma_k \geq 0,\ k = 1, \ldots, s.$$

- The inequality $M \succeq I_p^{1/2} I_q^{-1} I_p^{1/2}$ can be rewritten as:

$$I_q \succeq I_p^{1/2} M^{-1} I_p^{1/2}.$$

$$\begin{aligned} \min_{\mathbf{q}, M} \quad & \mathbf{tr}(M) \\ \text{s. t.} \quad & M \succeq I_p^{1/2} I_q^{-1} I_p^{1/2} \\ & \sum_{i=1}^{n} q_i = 1, q_i \geq 0, i = 1, \ldots, n \end{aligned}$$

Part II: BMAL

Eigen Space Simplification

- Using the eigen expression, we have

$$I_p^{1/2} M^{-1} I_p^{1/2} = \sum_{k=1}^{s} \gamma_k^{-1} \lambda_k \mathbf{v}_k \mathbf{v}_k^T$$

- Since the necessary condition for $I_q \succeq I_p^{1/2} M^{-1} I_p^{1/2}$

$$\mathbf{v}^T I_q \mathbf{v} \geq \mathbf{v}^T I_p^{1/2} M^{-1} I_p^{1/2} \mathbf{v},\ \forall \mathbf{v} \in \mathbf{R}^d,$$

- we then have the following result

$$\mathbf{v}_k^T I_q \mathbf{v}_k \geq \gamma_k^{-1} \lambda_k \text{ for } k = 1, \ldots, s.$$

$$\gamma_k \geq \frac{\lambda_k}{\mathbf{v}_k^T I_q \mathbf{v}_k}$$

Part II: BMAL

Eigen Space Simplification

- The previous necessary condition leads to following constraints:

$$\gamma_k \geq \frac{\lambda_k}{\mathbf{v}_k^T I_q \mathbf{v}_k} = \frac{\lambda_k}{\sum_{i=1}^{n} q_i \pi_i (1 - \pi_i)(\mathbf{x}_i^T \mathbf{v}_k)^2},\ k = 1, \ldots, s$$

- Meanwhile, the objective function of **tr(M)** can be expressed as

$$\mathbf{tr}(M) = \sum_{k=1}^{s} \gamma_k$$

Part II: BMAL

Eigen Space Simplification

- By putting the above two expressions together, we transform the SDP problem into the following approximate optimization problem:

$$\begin{aligned} \min_{\mathbf{q}, M} \quad & \mathbf{tr}(M) \\ \text{s. t.} \quad & M \succeq I_p^{1/2} I_q^{-1} I_p^{1/2} \\ & \sum_{i=1}^{n} q_i = 1, q_i \geq 0, i = 1, \ldots, n \end{aligned}$$

$$\begin{aligned} \min_{\mathbf{q} \in \mathbf{R}^n} \quad & \sum_{k=1}^{s} \frac{\lambda_k}{\sum_{i=1}^{n} q_i \pi_i (1 - \pi_i)(\mathbf{x}_i^T \mathbf{v}_k)^2} \\ \text{s.t.} \quad & \sum_{i=1}^{n} q_i = 1, q_i \geq 0, i = 1, \ldots, n \end{aligned}$$

- Note that the above optimization problem belongs to convex optimization since ***f(x)* = 1/*x*** is convex when $x \geqslant 0$.

Part II: BMAL

Bound Optimization Algorithm

- **Lemma 1**: Let $L(\mathbf{q})$ be the objective function,

$$\mathcal{L}(\mathbf{q}) = \sum_{k=1}^{s} \frac{\lambda_k}{\sum_{i=1}^{n} q_i \pi_i (1 - \pi_i)(\mathbf{x}_i^T \mathbf{v}_k)^2}$$

we have the following conclusion:

$$\mathcal{L}(\mathbf{q}) \leq \sum_{i=1}^{n} \frac{(q_i')^2}{q_i} \pi_i (1 - \pi_i) \sum_{k=1}^{s} \frac{(\mathbf{x}_i^T \mathbf{v}_k)^2 \lambda_k}{\left(\sum_{j=1}^{n} q_j' \pi_j (1 - \pi_j)(\mathbf{x}_j^T \mathbf{v}_k)^2\right)^2}$$

Proof in Appendix.

Part II: BMAL

Bound Optimization Algorithm

- Given the **lemma 1**, now instead of optimizing the original objective function L(**q**), we can optimize its upper bound using simple updating equations:,

$$q_i \longleftarrow q_i^2 \pi_i (1 - \pi_i) \sum_{k=1}^{s} \frac{(\mathbf{x}_i^T \mathbf{v}_k)^2 \lambda_k}{\left(\sum_{j=1}^{n} q_j \pi_j (1 - \pi_j)(\mathbf{x}_j^T \mathbf{v}_k)^2\right)^2}$$

$$q_i \longleftarrow \frac{q_i}{\sum_{j=1}^{n} q_j}$$

- This algorithm will guarantee to **converge to a local optimal**. Since the original problem is a **convex optimization** problem, the above updating procedure will guarantee to converge to a **global optimal**.

Part II: BMAL

Bound Optimization Algorithm

- The updating step:

$$q_i \longleftarrow q_i^2 \boxed{\pi_i(1-\pi_i)} \sum_{k=1}^{s} \frac{\boxed{(\mathbf{x}_i^T \mathbf{v}_k)^2 \lambda_k}}{\left(\sum_{j=1}^{n} q_j \pi_j (1-\pi_j)(\mathbf{x}_j^T \mathbf{v}_k)^2\right)^2}$$

- Some Observations
  - (i) The example with a large classification uncertainty will be assigned with a large probability.
  - (ii) The example that is similar to many unlabeled examples is more likely to be selected.

Part II: BMAL

Experimental Testbeds

- 3 standard text datasets
  - Reuters-21578 dataset (10788)
  - Two web-related datasets: WebKB (4518) and Newsgroup (10966)

Category	# of total samples
course	930
department	182
faculty	1124
project	504
staff	137
student	1641

Table 2: A list of 6 categories of the WebKB dataset in our experiments.

Category	# of total samples
earn	3964
acq	2369
money-fx	717
grain	582
crude	578
trade	485
interest	478
wheat	283
ship	286
corn	237

Table 1: A list of 10 major categories of the Reuters-21578 dataset in our experiments.

Category	# of total samples
0	1000
1	1000
2	1000
3	1000
4	1000
5	1000
6	999
7	1000
8	1000
9	1000
10	997

Table 3: A list of 11 categories of the Newsgroup dataset in our experiments.

Part II: BMAL

Experimental Settings

- A standard feature selection by Information Gain is conducted to remove uninformative features, in which 500 of the most informative features are selected.
- The *F*1 metric is adopted as our evaluation metric, which has been shown to be more reliable metric than other metrics such as the classification accuracy. More specifically, the F1 is defined as

$$F1 = \frac{2 * p * r}{p + r}$$

where *p* and *r* are *precision* and *recall*.

- Parameters of LogReg and SVM are determined by a standard cross validation method.

Part II: BMAL

Comparison Schemes

- Two popular active learning methods:
  - **SVM-AL:** the classification uncertainty of an example **x** is determined by its distance to the decision boundary

$$d(\mathbf{x}; \mathbf{w}, b) = \frac{|\mathbf{w}^T \mathbf{x} + b|}{\|\mathbf{w}\|_2}$$

The smaller the distance *d*(**x**;**w**, *b*) is, the more the classification uncertainty will be.

  - **LogReg-AL:** the logistic regression active learning algorithm that measures the classification uncertainty based on the entropy of the distribution p(y|x).

$$H(p) = -p(-|\mathbf{x}) \log p(-|\mathbf{x}) - p(+|\mathbf{x}) \log p(+|\mathbf{x})$$

The larger the entropy of **x** is, the more uncertain we are about the class labels of **x**.

  - Our Batch Mode Active Learning algorithm with logistic regression, i.e., **LogReg-BMAL** in short.

Part II: BMAL

Empirical Evaluation

- *Experimental Results with Reuters-21578*
  - average results over 40 executions
  - 100 training examples and 100 active examples

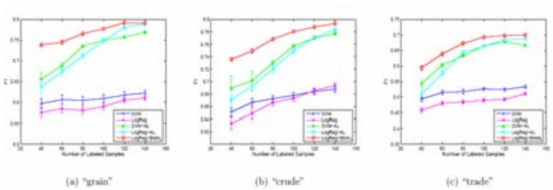
Category	SVM	LogReg	SVM-AL	LogReg-AL	LogReg-BMAL
earn	92.12 ± 0.22	92.47 ± 0.13	93.30 ± 0.28	93.40 ± 0.14	**94.00 ± 0.09**
acq	83.56 ± 0.26	83.35 ± 0.26	85.96 ± 0.34	86.57 ± 0.32	**88.07 ± 0.17**
money-fx	64.06 ± 0.60	63.71 ± 0.63	73.32 ± 0.38	71.21 ± 0.61	**75.54 ± 0.26**
grain	60.87 ± 1.04	58.97 ± 0.91	74.95 ± 0.42	74.82 ± 0.53	**77.77 ± 0.27**
crude	67.78 ± 0.39	67.32 ± 0.48	75.72 ± 0.24	74.97 ± 0.44	**78.04 ± 0.14**
trade	52.64 ± 0.46	48.93 ± 0.55	66.41 ± 0.33	66.31 ± 0.33	**69.29 ± 0.34**
interest	56.80 ± 0.60	53.59 ± 0.60	67.20 ± 0.39	66.15 ± 0.49	**68.71 ± 0.37**
wheat	62.71 ± 0.72	57.38 ± 0.79	86.01 ± 1.04	86.49 ± 0.27	**88.15 ± 0.21**
ship	67.11 ± 1.59	64.91 ± 1.75	75.86 ± 0.53	72.82 ± 0.46	**76.82 ± 0.34**
corn	44.39 ± 0.84	41.15 ± 0.69	71.27 ± 0.62	71.61 ± 0.60	**74.35 ± 0.47**

Table 4: Experimental results of F1 performance on the Reuters-21578 dataset using 100 training samples (%).

Part II: BMAL

Empirical Evaluation

- *Experimental Results with Reuters-21578*



(a) "grain" (b) "crude" (c) "trade"

Figure 2: Experimental results of F1 performance on the "grain", "crude" and "trade" categories

Empirical Evaluation

- *Experimental Results with Web-KB Dataset*

Category	SVM	LogReg	SVM-AL	LogReg-AL	LogReg-BMAL
course	87.11 ± 0.51	89.16 ± 0.45	88.55 ± 0.48	89.37 ± 0.65	**90.99 ± 0.39**
department	67.45 ± 1.36	68.92 ± 1.39	**82.02 ± 0.47**	79.22 ± 1.14	81.52 ± 0.46
faculty	70.84 ± 0.76	71.50 ± 0.59	75.59 ± 0.65	73.66 ± 1.23	**76.81 ± 0.51**
project	54.06 ± 0.82	56.74 ± 0.57	57.67 ± 0.98	56.90 ± 1.01	**59.71 ± 0.82**
staff	12.73 ± 0.44	12.73 ± 0.28	19.48 ± 1.07	**24.84 ± 0.58**	21.08 ± 0.73
student	74.05 ± 0.51	76.04 ± 0.49	77.03 ± 0.95	80.40 ± 1.16	**81.50 ± 0.44**

Table 5: Experimental results of F1 performance on the WebKB dataset using 40 training samples (%).

Empirical Evaluation

- *Experimental Results with Newsgroup Dataset*

Category	SVM	LogReg	SVM-AL	LogReg-AL	LogReg-BMAL
0	96.44 ± 0.35	95.02 ± 0.45	97.37 ± 0.52	95.66 ± 1.01	**98.73 ± 0.11**
1	83.38 ± 1.01	83.12 ± 0.96	**91.61 ± 0.57**	85.07 ± 1.51	91.12 ± 0.36
2	61.03 ± 1.51	59.01 ± 1.39	61.15 ± 2.08	64.91 ± 2.52	**66.13 ± 1.32**
3	72.36 ± 1.90	71.96 ± 1.67	73.15 ± 2.71	75.88 ± 3.13	**78.47 ± 1.95**
4	55.61 ± 1.06	56.09 ±1.21	56.05 ±2.18	61.87 ±2.25	**61.91 ± 1.03**
5	70.58 ± 0.51	72.47 ±0.40	71.69 ±1.11	72.99 ±1.46	**76.54 ± 0.43**
6	85.25 ± 0.45	86.30 ±0.45	89.54 ±1.09	89.14 ±0.89	**92.07 ± 0.26**
7	39.07 ± 0.90	40.22 ±0.90	42.19 ±1.13	46.72 ±1.61	**47.58 ± 0.76**
8	58.67 ± 1.21	59.14 ±1.25	63.77 ±2.05	66.57 ±1.24	**67.07 ± 1.34**
9	69.35 ± 0.82	70.82 ±0.92	74.34 ±1.79	77.17 ±1.06	**77.48 ± 1.20**
10	99.76 ± 0.10	99.40 ±0.21	**99.95 ± 0.02**	99.85 ±0.06	99.90 ± 0.06

Table 6: Experimental results of F1 performance on the Newsgroup dataset using 40 training samples (%).

Summary of Part II

- A new active learning scheme is suggested for text categorization to overcome the limitation of traditional active learning;
- A batch mode active learning solution is formulated by convex optimization techniques;
- An effective bound optimization algorithm is proposed to solve the batch mode active learning problem.
- Extensive experiments are conducted for empirical evaluations in comparisons with state-of-the-art active learning approaches for text categorization

Collaborative Multimedia Retrieval via Regularized Distance Metric Learning

- Problem Definition
  - Collaborative Multimedia Retrieval (CMR) is a Multimedia Information Retrieval (MIR) problem which involves human interactions, either with online relevance feedback explicitly or with historical log data of users' relevance feedback implicitly.

Motivation

- Relevance Feedback
  - A powerful tool for multimedia information retrieval
  - Popular methods: SVM Based solutions
- Log-based Relevance Feedback (LRF)
  - Combining log data for online relevance feedback
  - Our contribution: Soft Label SVM for LRF (MM 04, TKDE 06)
- Learning Distance Metrics with Log Data
  - Our contribution: Regularized Distance Metric Learning for learning robust and scalable metrics (ACM MM Journal 06)

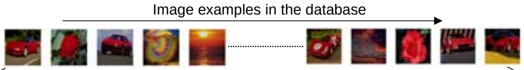
Regularized Distance Metric Learning

- Overview
  - The basic idea of this work is to learn a desired distance metric in the space of low-level image features that effectively bridges the semantic gap.
  - It is learned from the log data of user relevance feedback based on the Min/Max principle, i.e., minimize/maximize the distance between similar/dissimilar images.

Regularized Distance Metric Learning

- Formulation
  - The log data are given in terms of log sessions.
  - Each log session: each image was marked either relevant (+1), irrelevant (-1), or unknown (0).

Image examples in the database



Log Session (Q)

$$
\begin{pmatrix}
1 & -1 & 1 & -1 & -1 \cdots & 0 & \cdots 1 & -1 & -1 & 1 & 1 \\
-1 & 1 & -1 & -1 & -1 \cdots & 0 & \cdots -1 & -1 & 1 & -1 & -1
\end{pmatrix}
$$

Formulation

- We first exploit a metric learning algorithm for log data

$$
\begin{aligned}
&\min_{\mathbf{A}} \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}_q} \|\mathbf{x}_i-\mathbf{x}_j\|_{\mathbf{A}}^2 \\
&\text{s. t.} \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \|\mathbf{x}_i-\mathbf{x}_j\|_{\mathbf{A}}^2 \geq 1 \\
&\quad \mathbf{A} \succeq 0 \qquad (4)
\end{aligned}
$$

Where Q stands for number of log sessions in the log data.

This formulation tells us:

- When two images are judged as relevant in the same log session, they **could** be similar to each other;
- When one image is judged as relevant and another is judged as irrelevant in the same log session, they **must** be dissimilar to each other.

Formulation

- The formulation in (4) may not be robust for noise, we form a new objective function for distance metric learning that takes into account both the **discriminative** issue and the **robustness** issue as:

$$
\begin{aligned}
&\min_{\mathbf{A}} \|\mathbf{A}\|_F + c_S \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}_q} \|\mathbf{x}_i-\mathbf{x}_j\|_{\mathbf{A}}^2 - c_D \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \|\mathbf{x}_i-\mathbf{x}_j\|_{\mathbf{A}}^2 \\
&\text{s. t. } \mathbf{A} \succeq 0 \qquad (5)
\end{aligned}
$$

where $\|\mathbf{A}\|_F$ stands for the Frobenius norm. If $\mathbf{A} = [a_{i,j}]_{m\times m}$, its Frobenius norm is define as:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i,j=1}^{m} a_{i,j}^2} \qquad (6)$$

Formulation

- Using distance expressions, both the second and the third items of objective function in (5) can be expanded into the following forms:

$$
\begin{aligned}
c_S \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}_q} \|\mathbf{x}_i-\mathbf{x}_j\|_{\mathbf{A}}^2 &= c_S \operatorname{tr}\left(\mathbf{A}\cdot \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}_q} (\mathbf{x}_i-\mathbf{x}_j)(\mathbf{x}_i-\mathbf{x}_j)^T\right) \\
&= c_S \sum_{i,j=1}^{m} a_{i,j}s_{i,j} \qquad (7) \\
c_D \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}_q} \|\mathbf{x}_i-\mathbf{x}_j\|_{\mathbf{A}}^2 &= c_D \operatorname{tr}\left(\mathbf{A}\cdot \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}_q} (\mathbf{x}_i-\mathbf{x}_j)(\mathbf{x}_i-\mathbf{x}_j)^T\right) \\
&= c_D \sum_{i,j=1}^{m} a_{i,j}d_{i,j} \qquad (8)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{S} &= [s_{i,j}]_{m\times m} = \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}_q} (\mathbf{x}_i-\mathbf{x}_j)(\mathbf{x}_i-\mathbf{x}_j)^T \\
\mathbf{D} &= [d_{i,j}]_{m\times m} = \sum_{q=1}^{Q} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}_q} (\mathbf{x}_i-\mathbf{x}_j)(\mathbf{x}_i-\mathbf{x}_j)^T
\end{aligned}
$$

Formulation

- Putting Eqn. (6), (7), (8) together, we have the final formulation for the regularized metric learning:

$$
\begin{aligned}
&\min_{\mathbf{A}} \left(\sum_{i,j=1}^{m} a_{i,j}^2\right)^{1/2} + c_S \sum_{i,j=1}^{m} a_{i,j}s_{i,j} - c_D \sum_{i,j=1}^{m} a_{i,j}d_{i,j} \\
&\text{s. t. } \mathbf{A} \succeq 0 \qquad (9)
\end{aligned}
$$

Formulation

- To convert the above problem into the standard form, we introduce a slack variable *t* that upper bounds the Frobenius norm of matrix A, which leads to an equivalent form of (9), i.e.,

$$
\begin{aligned}
&\min_{\mathbf{A},t}\ t + c_S \sum_{i,j=1}^{m} a_{i,j}s_{i,j} - c_D \sum_{i,j=1}^{m} a_{i,j}d_{i,j} \qquad (10) \\
&\text{s. t. } \left(\sum_{i,j=1}^{m} a_{i,j}^2\right)^{1/2} \leq t \\
&\quad \mathbf{A} \succeq 0, \qquad (11)
\end{aligned}
$$

The first constraint is called a second order cone constraint
The second constraint is a positive semi-definite constraint.
A special form of Convex optimization problems!
There exists efficient solutions to solve it in a polynomial time

Experimental Results

- Datasets
  - 20-Category
  - 50-Category
- Image Representation
  - 9-dimensional Color Histogram
  - 18-dimensional Edge Histogram
  - 9-dimension texture

Experimental Results

- Collection of Users' Log Data

Table 1. The characteristics of users' log data on the 20-category and the 50-category testbeds.

Testbeds	Small Noise		Large Noise	
	# Log Sessions	Noise Degree	# Log Sessions	Noise Degree
20-Category	100	7.8%	100	16.2%
50-Caregory	150	7.7%	150	17.1%

Experimental Results

- Compared Schemes:
  - 1) "**Euclidean":** Euclidean metric without log data.
  - 2) "**IML**": based on the semantic representation learned from the manifold learning algorithm.
  - 3) "**DML**": based on the metric learned by a typical distance metric learning algorithm.
  - 4) "**RDML**": based on the metric by proposed regularized metric learning algorithm.

Experimental Results

Table 2: Average precision (%) of top-ranked images on the 20-Category testbed over 2,000 queries. The relative improvement of algorithm **IML**, **DML**, and **RDML** over the baseline **Euclidean** is included in the parenthesis following the average accuracy.

Top Images	20	40	60	80	100
Euclidean	39.91	32.72	28.83	26.47	24.47
IML	42.66(6.9%)	34.32(4.9%)	30.00(4.1%)	26.47(0.3%)	23.80(-2.7%)
DML	41.45(3.9%)	34.89(6.6%)	31.21(8.2%)	28.63(8.5%)	26.44(8.0%)
RDML	44.55(11.6%)	37.39(14.3%)	33.11(14.8%)	30.13(14.1%)	27.82(13.7%)

Table 3: Average precision (%) of top-ranked images on the 50-Category testbed over 5,000 queries.

Top Images	20	40	60	80	100
Euclidean	36.39	28.96	24.96	22.21	20.18
IML	35.64(-2.1%)	29.16(0.7%)	24.75(-0.8%)	21.68(-2.4%)	19.32(-4.3%)
DML	33.52(-7.9%)	27.15(-6.3%)	23.77(-4.8%)	21.48(-3.3%)	19.74(-2.2%)
RDML	40.36(10.9%)	32.62(12.6%)	28.24(13.1%)	25.17(13.4%)	22.86(13.3%)

Robustness Evaluation

Table 4: Average precision (%) of top-ranked images on the 20-Category testbed for **IML**, **DML**, and **RMDL** using **noisy log data**. The relative improvement over the baseline **Euclidean** is included in the parenthesis following the average accuracy.

Top Images	20	40	60	80	100
Euclidean	39.91	32.72	28.83	26.47	24.47
IML (Large Noise)	37.94(-4.9%)	30.14(-7.9%)	25.93(-10.1%)	23.56 (-11.0%)	21.97(-10.2%)
DML (Large Noise)	38.62(-3.2%)	32.32(-1.2%)	28.95(0.4%)	26.61 (0.8%)	24.62(0.6%)
RDML (Large Noise)	41.19(3.2%)	34.15(4.4%)	30.40(5.4%)	27.92(5.8%)	25.89(5.8%)

Table 5: Average precision (%) of top-ranked images on the 50-Category testbed for **IML**, **DML**, and **RMDL** using **noisy log data**.

Top Images	20	40	60	80	100
Euclidean	36.39	28.96	24.96	22.21	20.18
IML (Large Noise)	33.80(-7.1%)	27.30(-5.8%)	23.56(-5.0%)	20.65(-6.7%)	18.36 (-8.1%)
DML (Large Noise)	32.85(-9.7%)	26.95 (-7.0%)	23.55(-5.7%)	21.22(-4.5%)	19.49(-3.4%)
RDML (Large Noise)	37.45(2.9%)	29.97(3.5%)	25.84(3.5%)	22.99(3.5%)	20.87(3.4%)

Efficiency and Scalability

Table 6: The training time cost (CPU seconds) of three algorithms on 20-Category (100 log sessions) and 50-Category (150 log sessions) testbeds.

Algorithm	IML	DML	RDML
20-Category	82.5	3,227	19.2
50-Category	2,864	12,341	20.5

Summary of Part III

- We proposed a novel algorithm for distance metric learning, which boosts the retrieval accuracy of CBIR by taking advantage of the log data of users' relevance judgments.
- A regularization mechanism is used in the proposed algorithm to improve the robustness of solutions, when the log data is small and noisy.
- It is formulated as a positive semi-definite programming problem, which can be solved efficiently.
- Experiment results have shown that the proposed algorithm for regularized distance metric learning substantially improves the retrieval accuracy of the baseline CBIR system.

Summary of Other Contributions

- Distance Metric Learning for Clustering
  - Discriminative Component Analysis (DCA)
  - Kernel DCA for learning nonlinear metrics
  - Details in Appendix A
- Marginalized Kernels for Web Mining
  - Time-dependent similarity measure scheme
  - Marginalized kernels to exploit both explicit similarity and implicit cluster semantic for similarity measure
  - Details in Appendix B

Conclusions

- We proposed a framework of statistical machine learning for data mining and collaborative multimedia retrieval.
- We suggested a unified framework to learn the unified kernel machines, in which a new semi-supervised kernel learning algorithm was proposed.
- We explored the batch mode active learning problem and proposed a novel algorithm to search a batch of informative examples.
- We studied a real-world application, collaborative multimedia retrieval, and proposed a regularized distance metric learning algorithm for learning robust and scalable metrics for multimedia retrieval.

Future Work

- Theoretical Analysis on UKM …
- More effective algorithms and extensions to UKM …
- Employing UKM to solve real-world problems, classification, regressions, information retrieval, …

Selected Publications (Regular Papers)

1. "Learning the Unified Kernel Machines for Classification," **Steven C.H. Hoi**, Michael R. Lyu, Edward Y Chang, In ACM SIGKDD (KDD2006), Philadelphia, USA, August 20 - 23, 2006.
2. "*Large-Scale Text Categorization by Batch Mode Active Learning*," **Steven C.H. Hoi**, R. Jin and M.R. Lyu, In WWW 2006, Edinburgh, England, UK, 2006.
3. "*Time-Dependent Semantic Similarity Measure of Queries Using Historical Click-Through Data*", Q. Zhao, **Steven C. H. Hoi**, T.-Y. Liu, et al, In WWW 2006, May 2006.
4. "Batch Mode Active Learning and Its application to Medical Image Classification", **Steven C.H. Hoi**, R. Jin, J. Zhu and M.R. Lyu, In ICML 2006, Pittsburgh, US, June 25-29, 2006.
5. "Learning Distance Functions with Contextual Constraints for Image Retrieval", **Steven C.H. Hoi**, W. Liu, Michael R Lyu, W-M. Ma, in IEEE CVPR 2006, New York, June, 2006
6. "A Unified Log-based Relevance Feedback Scheme for Image Retrieval," **Steven C. H. Hoi**, Michael R. Lyu and Rong Jin, In IEEE Transactions on KDE (TKDE), vol. 18, no. 4, 2006
7. "Collaborative Image Retrieval via Regularized Metric Learning", Luo Si, Rong Jin and **Steven C. H. Hoi** and Michael R. Lyu, ACM Multimedia Systems Journal (MMSJ), Special issue on Machine Learning Approaches to Multimedia Information Retrieval, 2006.
8. "A Semi-Supervised Active Learning Framework for Image Retrieval," **Steven C. H. Hoi** and Michael R. Lyu, in IEEE CVPR 2005, San Diego, CA, USA June 20-25, 2005
9. "A Unified Machine Learning Paradigm for Large-Scale Personalized Information Management,", Edward Y. Chang, **Steven C. H. Hoi**, Xinjing Wang, Wei-Ying Ma and Michael R. Lyu, EIT 2005, NTU Taipei, August 2005.
10. "A Novel Log-based Relevance Feedback Technique in Content-based Image Retrieval," **Steven C.H. Hoi** and Michael R. Lyu, ACM Multimedia, New York, pp. 24-31, 2004

Thanks!

Q & A

Appendix

- A: Distance Metric Learning for Clustering
- B: Marginalized Kernels for Web Mining
- C: Proof of Lemma 1 in BMAL
- D: Definition of Semi-Definite Programming

79

Appendix A: Distance Metric Learning for Clustering

- Motivation
  - We address important limitations of existing metric learning methods, Relevant Component Analysis (RCA)
  - It lacks of considering negative constraints
  - It cannot capture nonlinear relationship of data instances via linear transformation
- Solution:
  - Discriminative Component Analysis (DCA)
  - Kernel DCA to learn nonlinear metrics

80

Discriminative Component Analysis

- Formulation
  - Given a set of data points $X = \{x_i\}_{i=1}^{N}$ and a set of contextual constraints
  - Form n chunklets using the positive constraints: $C_j = \{x_{ji}\}_{i=1}^{n_j}$
  - Form a discriminative set D_j to indicate which chunklets can be discriminated each other by the negative constraints.

81

Discriminative Component Analysis

- Two covariance matrixes are computed:

$$\hat{C}_b = \frac{1}{N_b}\sum_{j=1}^{n}\sum_{i\in D_j}(m_j - m_i)(m_j - m_i)^T \quad (1)$$

$$\hat{C}_w = \frac{1}{N_w}\sum_{j=1}^{n}\sum_{i=1}^{n_j}(x_{ji} - m_j)(x_{ji} - m_j)^T \quad (2)$$

- where $N_b = \sum_{j=1}^{n}|D_j|$, m_i is the mean of the i-th chunklet, i.e., $m_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_{ij}$
- Finding the optimal transformation is equivalent to solve the following optimization:

$$J(\mathrm{A}) = \arg\max_{\mathrm{A}} \frac{|\mathrm{A}^T \mathrm{C}_b \mathrm{A}|}{|\mathrm{A}^T \mathrm{C}_w \mathrm{A}|}$$

82

Discriminative Component Analysis

- Algorithm for solving DCA
  - Idea: Based on the Fisher's criterion, the DCA problem can be solved by diagonalizing Cb and Cw simultaneously

$$J(\mathrm{A}) = \arg\max_{\mathrm{A}} \frac{|\mathrm{A}^T \mathrm{C}_b \mathrm{A}|}{|\mathrm{A}^T \mathrm{C}_w \mathrm{A}|}$$

  - Steps:
    - 1) Compute the covariance matrices C_b and C_w by Eq.(1),(2)
    - 2) Diagonalize Cb by eigenanalysis
    - 3) Project and diagonalize Cw by eigenanalysis
    - 4) Output transformation matrix A

83

Kernel DCA

- The kernel techniques first map the input data into a feature space ***F***.
- The data can be then analyzed in the projected feature space.
- The linear transformation in the feature space corresponds the nonlinear analysis in the input space.
- For example: Kernel PCA, Kernel ICA, Kernel LDA, etc.

84

Kernel DCA

- **Formulation**
  - We implicitly map the original data $X = \{x_i\}_{i=1}^{N}$ in the input space *I* to a high-dimensional feature space *F* via some defined basis function.

$$\phi : x \to \phi(x) \in F$$

  - The similarity of two instances is measured:

$$K(x_i, x_j) = <\phi(x_i), \phi(x_j)>.$$

  - In general, we want to find the optimal M:

$$d_\phi(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j))^T \mathrm{M} (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j))}$$

$$\mathrm{M} = \mathrm{W}^T \mathrm{W}$$

85

Kernel DCA

- The transformation matrix W can be represented as

$$\mathrm{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_m]^T$$

in which each of the column vector is a span of all the training samples in the feature space, such that

$$\mathbf{w}_i = \sum_j \alpha_{ij} \phi_j$$

where α_{ij} are the coefficients for the samples in the feature space.

86

Kernel DCA

- For each given data instance **x**, we can compute its projection onto the *i*-th direction $\mathbf{w}_i$ in the feature space as

$$(\mathbf{w}_i \bullet \phi(\mathbf{x})) = \sum_j \alpha_{ij} K(\mathbf{x}_{ij}, \mathbf{x})$$

- Hence the original distance can be turned into

$$d_\phi(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\tau_i - \tau_j)^T \mathrm{A}^T \mathrm{A} (\tau_i - \tau_j)}$$

where

$$\tau_i = [K(\mathbf{x}_1, \mathbf{x}_i), K(\mathbf{x}_2, \mathbf{x}_i), \ldots, K(\mathbf{x}_l, \mathbf{x}_i)]^T$$

$$\mathrm{A} = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_m] \qquad \boldsymbol{\alpha}_i = [\alpha_{i1}, \alpha_{i1}, \ldots, \alpha_{il}]^T$$

87

Kernel DCA

- We can compute the two corresponding covariance matrixes:

$$\widehat{\mathrm{K}}_b = \frac{1}{N_b} \sum_{j=1}^{n} \sum_{i \in D_j} (u_j - u_i)(u_j - u_i)^T \quad (5)$$

$$\widehat{\mathrm{K}}_w = \frac{1}{N_w} \sum_{j=1}^{n} \sum_{i=1}^{n_j} (\tau_{ji} - u_j)(\tau_{ji} - u_j)^T \quad (6)$$

where

$$u_i = [\frac{1}{n_i} \sum_{j=1}^{n_i} K(\mathbf{x}_1, \mathbf{x}_j), \frac{1}{n_i} \sum_{j=1}^{n_i} K(\mathbf{x}_2, \mathbf{x}_j), \ldots, \frac{1}{n_i} \sum_{j=1}^{n_i} K(\mathbf{x}_l, \mathbf{x}_j)]^T.$$

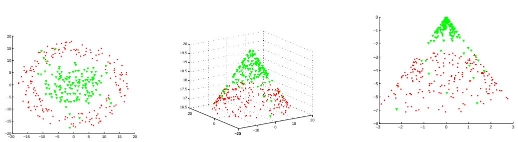
88

Kernel DCA

- The optimization problem for Kernel DCA can therefore be given as follows

$$J(\mathrm{A}) = \arg\max_{\mathbf{A}} \frac{|\mathrm{A}^T \mathrm{K}_b \mathrm{A}|}{|\mathrm{A}^T \mathrm{K}_w \mathrm{A}|}$$

- The algorithm to solve the Kernel DCA is similar to the linear DCA.



(a) Original Input Space (b) Projected Space via Kernel (c) Embedding Space by KDCA

89

Experimental Results

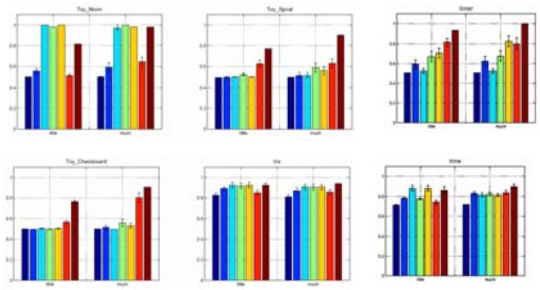
- Datasets

Dataset	#Classes	#Instances	#Features
Norm	2	100	2
Chessboard	2	100	2
Double-Spiral	2	100	3
Iris	3	150	4
Sonar	2	208	60
Wine	3	178	12

- Compared Schemes
  - (1) K-means-EU: the baseline method, i.e., typical k-means clustering
  - based on the original Euclidean distance;
  - (2) CK-means-EU: the constrained k-means clustering method based
  - on the original Euclidean distance [146];
  - (3) CKmeans-RCA: the constrained k-means clustering method based
  - on the distance metrics learned by RCA [8];
  - (4) CKmeans-Xing: the constrained k-means clustering method based
  - on the distance metrics learned by Xing et al. [153];
  - (5) CKmeans-DCA: the constrained k-means clustering method based
  - on the distance metrics learned by our DCA algorithm;
  - (6) CKmeans-RBF: the constrained k-means clustering method based
  - on the RBF kernel metrics;
  - (7) CKmeans-KDCA: the constrained

90

Experimental Results



91

Summary

- We studied the problem of learning distance metrics and data transformation using the contextual information for data clustering.
- we proposed the Discriminative Component Analysis (DCA), which can exploit both positive and negative constraints in an efficient learning scheme.
- We proposed KDCA to learn nonlinear metrics for data clustering.

92

Appendix B: Marginalized Kernels for Time-Dependent Similarity Measures

- Motivation
- Our Approach
- Time-Dependent Concepts
- Marginalized Kernels for Similarity Measure
- Empirical Results

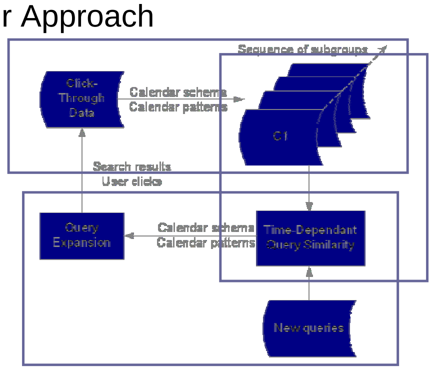
93

Motivations

- Exploit the click-through data for semantic similarity of queries by incorporating temporal information
- To combine explicit content similarity and implicit semantic similarity via marginalized kernel techniques

94

Our Approach



95

Time-Dependent Concepts

- Calendar schema and pattern

DEFINITION 1. **Calendar Schema:** *A calendar schema, $S = (R, C)$, is a relational schema R with a constraint C, where $R = (f_n : D_n, f_{n-1} : D_{n-1}, \cdots, f_1 : D_1)$, C is a Boolean valid constraint on $D_n \times D_{n-1} \times \cdots \times D_1$ that specifies which combinations of the values in $D_n \times D_{n-1} \times \cdots \times D_1$ are valid.* □

DEFINITION 2. **Calendar Pattern:** *Given a calendar schema $S = (R, C)$, a calendar pattern, denoted as CAP, is a tuple on R of the form $< d_n, d_{n-1}, \cdots, d_1 >$ where $d_i \in D_i \cup \{*\}$.* □

- Example
  - Calendar schema <day, month, year>
  - Calendar pattern <15, *,*>
  - <15, 1, 2002> is contained in the pattern <15, *,*>

96

Time-Dependent Concepts

- Click-Through Subgroup

DEFINITION 6. Click-Through Subgroup (CTS): *Given a calendar schema S and a set of calendar patterns $\{CAP_1, CAP_2, \cdots, CAP_m\}$, the click-through data can be segmented into a sequence of click-through subgroups (CTSs) $< CTS_1, CTS_2, \cdots, CTS_m >$, where all query-page pairs $< \vec{q}, p_{i_1}, \vec{t}_i > \in CTS_l, \vec{t}_i \prec CAP_l, 1 \le l \le m$.* □

- Example
  - Based on the schema <day, week>, and the pattern <1,*>, <2,*>,…,<7,*>, we can partition the data into 7 groups, which correspond to Sun, Mon, Tue, …, Sat.

Similarity Measure

- For efficiency and simplicity, we measure the query similarity in a certain time slot only based on the click-through data.
  - Vector representation of queries with respect to clicked documents.

$$\vec{q} = < w_1, w_2, \cdots, w_n >$$

  - w_i is defined by Page Frequency (PF) and Inverted Query Frequency (IQF)

$$w_i = PF(\vec{q}, p_i) \times IQF(p_i).$$

$$PF(\vec{q}, p_i) = \frac{f(\vec{q}, p_i)}{\sum_j f(\vec{q}, p_j)}, \; IQF(p_i) = \log \frac{|\vec{q}|}{| < \vec{q}, p_i > |}$$

Similarity Measure

- Query similarity measures
  - Cosine function $K_{\cos}(\vec{q}_1, \vec{q}_2) = \frac{\vec{q}_1^T \vec{q}_2}{\|\vec{q}_1\| \cdot \|\vec{q}_2\|}$
  - Marginalized kernel
    - By introducing query clusters, one can model the query similarity in a more semantic way.
    - **DEFINITION 8. Marginalized Kernel:** *Assume that a visible variable x is described as $x \in X$, where the domain X is a finite set. Suppose a hidden variable h is described as $h \in H$, where H is a finite set. A joint kernel $K_Z(z, z')$ is defined between the two combined variables $z = (x, h)$ and $z' = (x', h')$. The marginalized kernel in X is defined by taking the expectation with respect to the hidden variables as follows:* $K(x, x') = \sum_{h \in H} \sum_{h' \in H} p(h|x) p(h'|x') K_Z(z, z')$

Time-Dependent Similarity Measure

$$
\begin{aligned}
& K_T(\vec{q}, \vec{q}' | \vec{t}) \\
&= \sum_{\forall c} \sum_{\forall c'} K_Q\left(Q_{c|t}, Q'_{c'|t}\right) p(c|q,t) p(c'|q',t) \\
&= K_{\cos}\left(q, q'|t\right) \left(\sum_{\forall c} \sum_{\forall c'} \varphi(c, c'|t) p(c|q,t) p(c'|q',t) \right) \\
&= K_{\cos}\left(q, q'|t\right) \left(\sum_{c \in \Omega(t)} p(c|q,t) p(c|q',t) \right) \\
&= \frac{q_t \circ q'_t}{\|q_t\| \times \|q'_t\|} \left(\sum_{c \in \Omega(t)} p(c|q,t) p(c|q',t) \right)
\end{aligned}
$$

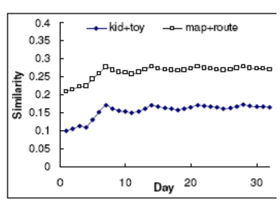
where c and c' are the guessed clusters given the queries, $Q_{c|t} = (q, c|t)$ and $Q'_{c'|t} = (q', c'|t)$. K_Q is a joint kernel, $\varphi(c, c'|t)$ is a function whose value is equivalent to 1 if c and c' are the same and 0 otherwise, and q_t and q'_t are time-dependent query vectors. □

Empirical Evaluation

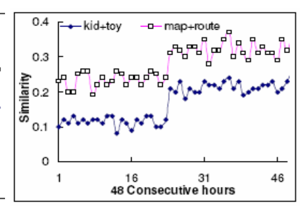
- Dataset
  - Click-through log of a commercial search engine:
    - June 16, 2005 to July 17,2005
    - Total size of 22GB
    - Only queries from US
  - Calendar schema and pattern
    - <hour, day, month>, <1, *, *>, <2, *, *>, …
    - Divide the data into 24 subgroups
    - Average subgroup size: 59,400,000 query-page pairs

Empirical Examples

- Kids+toy, map+route



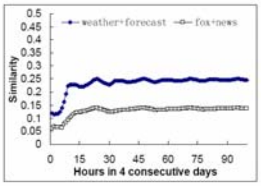
Incremented daily similarity



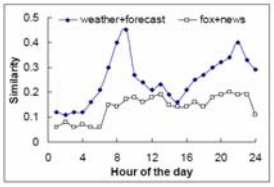
Time-dependent daily similarity

Empirical Examples

- weather + forecast, fox + news



Incremented daily similarity



Time-dependent daily similarity

Summary

- Presented a preliminary study of the dynamic nature of query similarity using click-through data
- Using marginalized kernels for building an time-dependent model
- Conducted empirical evaluations from real-world web search data

Appendix C: Proof of Lemma1

- **Lemma 1**: Let $L(\mathbf{q})$ be the objective function in (15), we have the following conclusion

$$\mathcal{L}(\mathbf{q}) \leq \sum_{i=1}^{n} \frac{(q_i')^2}{q_i}\pi_i(1-\pi_i)\sum_{k=1}^{s}\frac{(\mathbf{x}_i^T\mathbf{v}_k)^2\lambda_k}{\left(\sum_{j=1}^{n} q_j'\pi_j(1-\pi_j)(\mathbf{x}_j^T\mathbf{v}_k)^2\right)^2}$$

- **Proof.**

$$\begin{aligned}\mathcal{L}(\mathbf{q}) &= \sum_{k=1}^{s}\frac{\lambda_k}{\sum_{i=1}^{n} q_i\pi_i(1-\pi_i)(\mathbf{x}_i^\top\mathbf{v}_k)^2} \\ &= \sum_{k=1}^{s}\frac{\lambda_k}{\sum_{i=1}^{n} q_i'\pi_i(1-\pi_i)(\mathbf{x}_i^\top\mathbf{v}_k)^2} \times \frac{\sum_{i=1}^{n} q_i'\pi_i(1-\pi_i)(\mathbf{x}_i^\top\mathbf{v}_k)^2}{\sum_{i=1}^{n} q_i'\pi_i(1-\pi_i)(\mathbf{x}_i^\top\mathbf{v}_k)^2\frac{q_i}{q_i'}}\end{aligned}$$

- **Proof** (cont.):
  Using the convexity property of reciprocal function, namely

$$1/\sum_{i=1}^{n} p_i x \leq \sum_{i=1}^{n}\frac{p_i}{x}$$

for $x \geq 0$ and p.d.f. $\{p_i\}_{i=1}^{n}$

We can arrive the following deduction

$$\begin{aligned}&\frac{\sum_{i=1}^{n} q_i'\pi_i(1-\pi_i)(\mathbf{x}_i^T\mathbf{v}_k)^2}{\sum_{i=1}^{n} q_i'\pi_i(1-\pi_i)(\mathbf{x}_i^T\mathbf{v}_k)^2\frac{q_i}{q_i'}} \\ &\leq \sum_{i=1}^{n}\frac{q_i'\pi_i(1-\pi_i)(\mathbf{x}_i^T\mathbf{v}_k)^2}{\sum_{j=1}^{n} q_j'\pi_j(1-\pi_j)(\mathbf{x}_j^T\mathbf{v}_k)^2}\frac{1}{\frac{q_i}{q_i'}} = \sum_{i=1}^{n}\frac{(q_i')^2\pi_i(1-\pi_i)(\mathbf{x}_i^T\mathbf{v}_k)^2}{q_i\sum_{j=1}^{n} q_j'\pi_j(1-\pi_j)(\mathbf{x}_j^T\mathbf{v}_k)^2}\end{aligned}$$

- **Proof** (cont.):
  Substituting the above inequation back to L(q), we can attain the following inequality:

$$\begin{aligned}\mathcal{L}(\mathbf{q}) &\leq \sum_{k=1}^{s}\frac{\lambda_k}{\sum_{i=1}^{n} q_i'\pi_i(1-\pi_i)(\mathbf{x}_i^T\mathbf{v}_k)^2} \times \left(\sum_{i=1}^{n}\frac{(q_i')^2\pi_i(1-\pi_i)(\mathbf{x}_i^T\mathbf{v}_k)^2}{q_i\sum_{j=1}^{n} q_j'\pi_j(1-\pi_j)(\mathbf{x}_j^T\mathbf{v}_k)^2}\right) \\ &= \sum_{k=1}^{s}\frac{\lambda_k}{\left(\sum_{j=1}^{n} q_j'\pi_j(1-\pi_j)(\mathbf{x}_j^T\mathbf{v}_k)^2\right)^2} \times \sum_{i=1}^{n}\frac{(q_i')^2(\mathbf{x}_i^T\mathbf{v}_k)^2\pi_i(1-\pi_i)}{q_i} \\ &= \sum_{i=1}^{n}\frac{(q_i'^2)}{q_i}\pi_i(1-\pi_i)\sum_{k=1}^{s}\frac{(\mathbf{x}_i\mathbf{v}_k)^2\lambda_k}{(\sum_{j=1}^{n} q_j'\pi_j(1-\pi_j)(\mathbf{x}_j^T\mathbf{v}_k)^2)^2}\,.\end{aligned}$$

This finishes the proof of the inequality lemma. □

Back

Appendix D – Semi-Definite Programming (SDP)

$$\begin{array}{ll}\text{minimize} & c^T x \\ \text{subject to} & x_1F_1 + x_2F_2 + \cdots + x_nF_n + G \preceq 0 \\ & Ax = b\end{array}$$

with $F_i,\ G \in \mathbf{S}^k$

- inequality constraint is called linear matrix inequality (LMI)
- includes problems with multiple LMI constraints: for example,

$$x_1\hat{F}_1 + \cdots + x_n\hat{F}_n + \hat{G} \preceq 0, \qquad x_1\tilde{F}_1 + \cdots + x_n\tilde{F}_n + \tilde{G} \preceq 0$$

is equivalent to single LMI

$$x_1\begin{bmatrix}\hat{F}_1 & 0\\ 0 & \tilde{F}_1\end{bmatrix} + x_2\begin{bmatrix}\hat{F}_2 & 0\\ 0 & \tilde{F}_2\end{bmatrix} + \cdots + x_n\begin{bmatrix}\hat{F}_n & 0\\ 0 & \tilde{F}_n\end{bmatrix} + \begin{bmatrix}\hat{G} & 0\\ 0 & \tilde{G}\end{bmatrix} \preceq 0$$