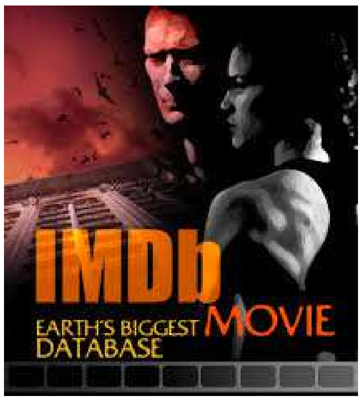
Instance-based Domain Adaptation

Rui Xia
School of Computer Science and Engineering
Nanjing University of Science and Technology

Problem Background

Training data



Movie Domain Sentiment Classifier

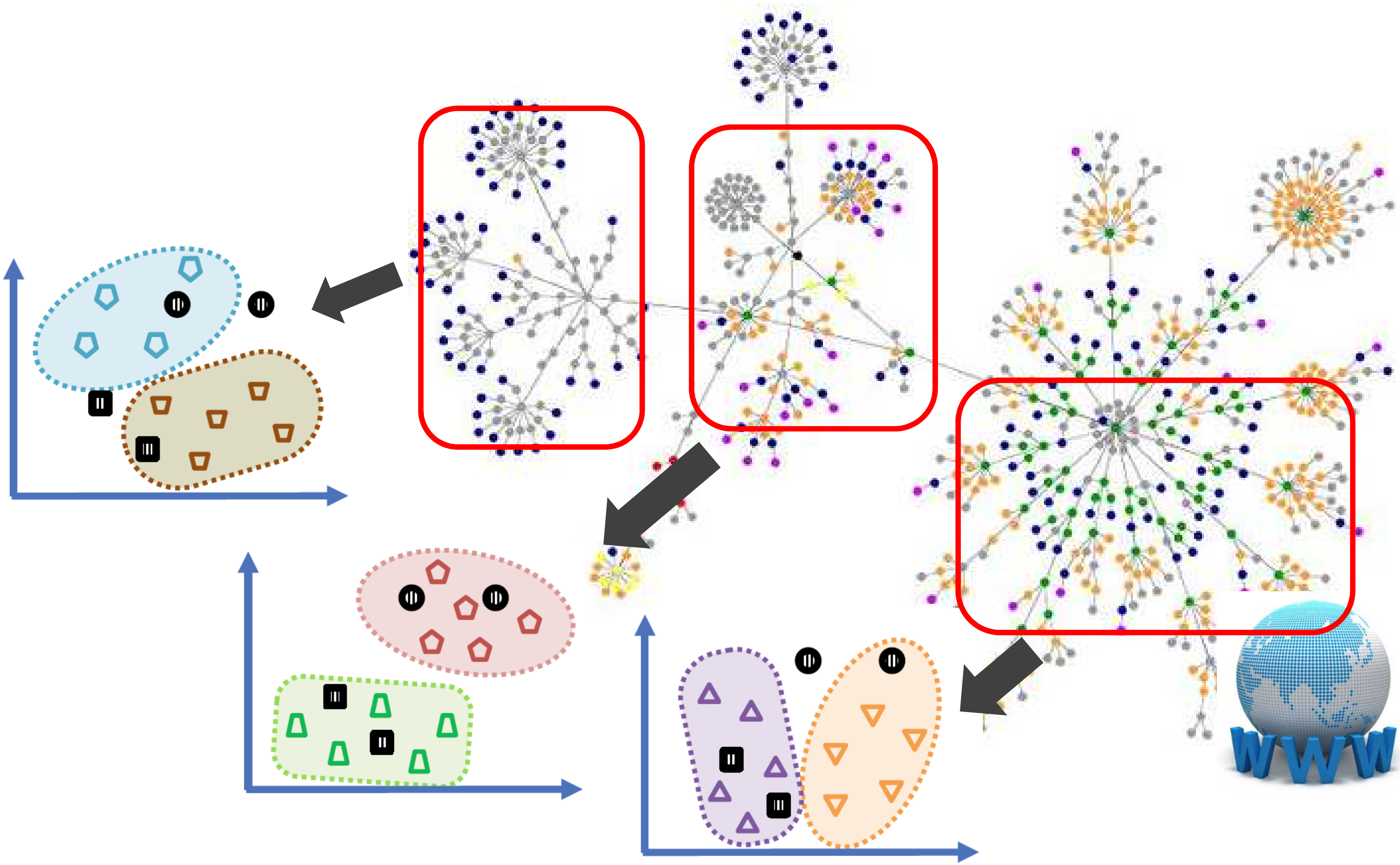
Test data

85%

65%

75%

How to find the “right” training data?



Picture from IJCAI-13 tutorial: Transfer Learning with Applications

A Summary of Domain Adaptation Methods in NLP

Methods	Description	References
Semi-supervised based	Build a semi-supervised classifier based on the source domain labeled data, and target domain unlabeled data	[Aue, 2005; Tan, 2007; Dai, 2007; Tan, 2009]
Parameter based	Assume that models of the source and target domain share the same prior parameter	[Chelba, 2006; Li, 2009; Xue, 2008; Finkel, 2009]
Feature based	Learn a new feature representation (or a new labeling function) for the target domain	[Daume III, 2007; Blitzer, 2007; Gao, 2008; Pan, 2009; Pan, 2010b; Ji, 2011; Xia, 2011; Samdani, 2011; Glorot, 2011; Duan, 2012]
Instance based	Learn the importance of labeled data in the source domain by instance selection and instance weighting	[Sugiyama, 2007; Bickel, 2009; Axelrod, 2011]

Our Work

- PUIS/PUIW (Instance Selection and Instance Weighting via PU learning)

Rui Xia, Xuelei Hu, Jianfeng Lu, Jian Yang, and Chengqing Zong. Instance Selection and Instance Weighting for Cross-domain Sentiment Classification via PU Learning. *IJCAI*-2013.

- ILA (Instance Adaptation via In-target-domain Logistic Approximation)

Rui Xia, Jianfei Yu, Feng Xu, and Shumei Wang. Instance-based Domain Adaptation in NLP via In-target-domain Logistic Approximation. *AAAI*-2014.

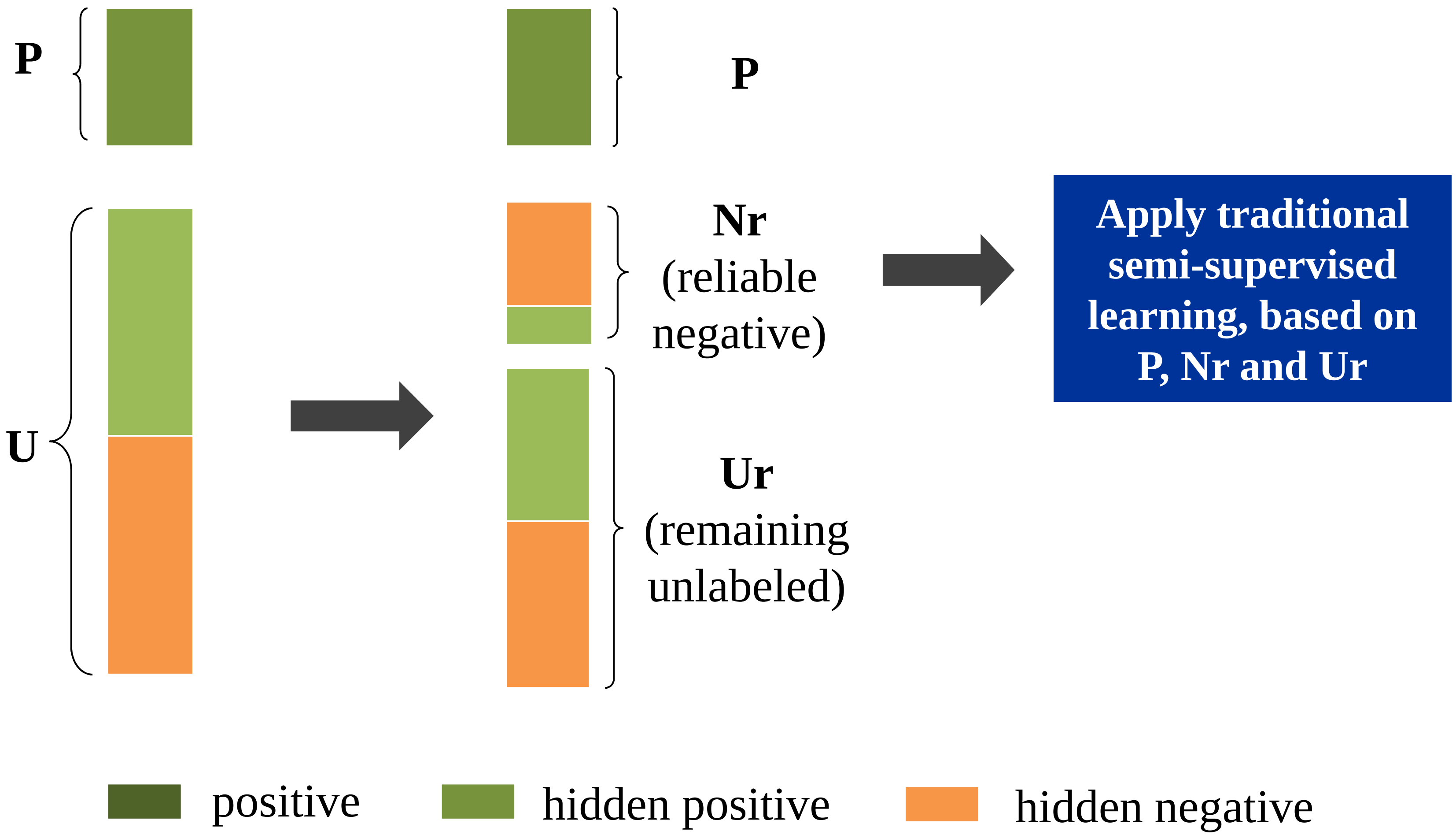
Part 1. PUIS/PUIW (Instance Selection and Instance Weighting via PU Learning)

Introduction to PU learning

- What is PU learning?
  - Give a set of positive (P) samples of a particular class, and a set of unlabeled (U) samples that contains hidden positive and negative ones
  - Build a binary-class classifier using P and U

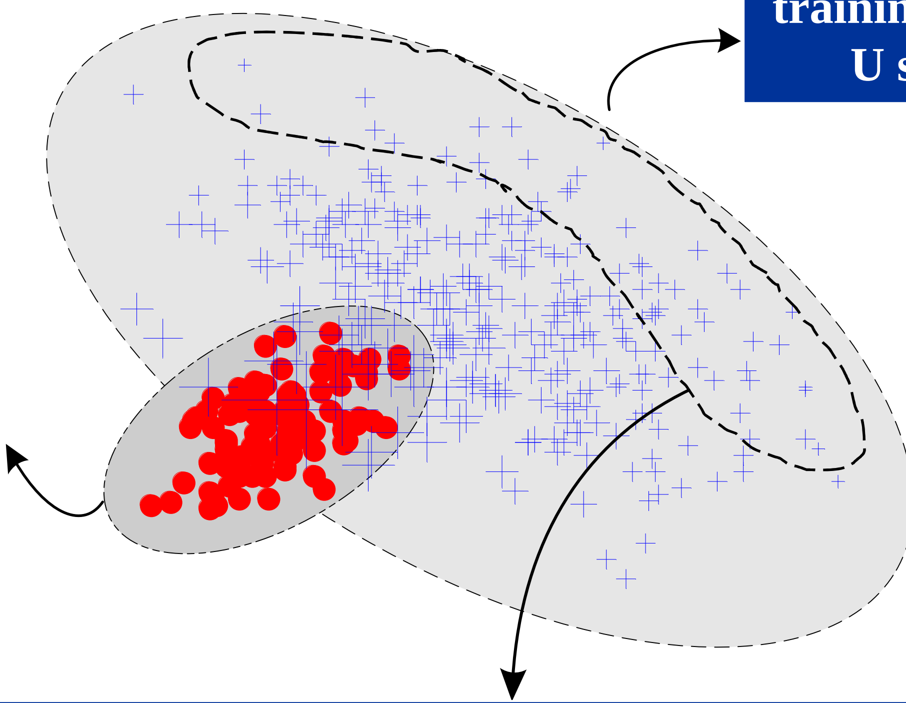
- Relation to traditional semi-supervised learning
  - Traditional semi-supervised learning : Learning with a small set of labeled data and a large set of unlabeled data (LU learning)
  - PU learning: labeled data only contain positive samples

Illustration of PU Learning



A Motivation Example of PUIS/PUIW

Large-size source domain training data are used as U set (blue cross)



Some target domain unlabeled data are used as P set (red dot)

PUL firstly identifies a reliable negative set Nr from U, and

secondly builds a semi-supervised classifier using P, Nr, and U-Nr

PU Learning for Instance Selection (PUIS)

Algorithm S-EM($\mathcal{P}$, $\mathcal{U}$, a, b)
$\mathcal{N}_r = \varnothing$;
$\tilde{\mathcal{P}}$ = Sample($\mathcal{P}$, $a\%$);
Assign each instance in $\mathcal{P} - \tilde{\mathcal{P}}$ the class label $d = 1$;
Assign each instance in $\mathcal{U} \cup \tilde{\mathcal{P}}$ the class label $d = 0$;
Build a NB classifier g using $\mathcal{P} - \tilde{\mathcal{P}}$ and $\mathcal{U} \cup \tilde{\mathcal{P}}$;
Classify each $\mathbf{x} \in \mathcal{U} \cup \tilde{\mathcal{P}}$ using g;
for each $u \in \mathcal{U}$
  if $p(+|u) < b$
    $\mathcal{N}_r \leftarrow \mathcal{N}_r \cup \{u\}$;
$\mathcal{U}_r = \mathcal{U} - \mathcal{N}_r$;
Assign each instance in $\mathcal{P}$ the class label $d = 1$;
Assign each instance in $\mathcal{N}_r$ the class label $d = 0$;
Learn an EM classifier f iteratively on $\mathcal{P}$, $\mathcal{N}_r$ and $\mathcal{U}_r$;
for each $\mathbf{x}_n \in \mathcal{U}$
  Predict $p(d|\mathbf{x}_n)$ using f;
  if $p(d = 1|\mathbf{x}_n) > 0.5$
    Output $\mathbf{x}_n$ as a positive (in-target-domain) sample;
  else
    Output $\mathbf{x}_n$ as a negative (not-in-target-domain) sample;

Step 1: Detect reliable negative samples by sampling some spy samples and building a supervised NB classifier

Step 2: Build a semi-supervised NB based on EM training (NBEM)

Step 3: Use the NBEM classifier as an in-target-domain selector

The Instance Weighting Framework

- Expected log-likelihood of the target domain

$$\mathcal{L}(\theta) = \int_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p_t(x, y) \log p(x, y|\theta) dx$$

$$\approx \int_{x \in \mathcal{X}} \boxed{\frac{p_t(x)}{p_s(x)} p_s(x)} \sum_{y \in \mathcal{Y}} p_s(y|x) \log p(x, y|\theta) dx$$

Instance Weighting: Density Ratio Estimation

$$= \int_{x \in \mathcal{X}} \boxed{r(x) p_s(x)} \sum_{y \in \mathcal{Y}} p_s(y|x) \log p(x, y|\theta) dx$$

- Instance-weighted maximum likelihood estimation

$$\theta^* = \arg\max_{\theta} \frac{1}{N_s} \sum_{n=1}^{N_s} r(x_n) \log p(x_n, y_n|\theta)$$

PU Learning for Instance Weighting (PUIW)

- In-target-domain Sampling Assumption

$$q_t(x) = r(x)p_s(x) \propto p(d = 1|x)p_s(x)$$

- Instance Weighting via PU learning and Probability Calibration

$$r(x) = \frac{p_s(x)}{p_t(x)} \propto p(d = 1|x) = \frac{1}{1 + \exp -\boxed{\alpha}\boxed{f(x)}}$$

The calibration parameter

The log-likelihood output of PU learning

Instance-weighted Classification Model

- Learning in traditional Naïve Bayes

$$p(c_j) = \frac{\sum_k I(y_k = c_j)}{\sum_k \sum_{j'} I(y_k = c'_j)} = \frac{N_j}{N} \qquad p(t_i|c_j) = \frac{\sum_k I(y_k = c_j) N(t_i, x_k)}{\sum_k I(y_k = c_j) \sum_{i'=1} N(t_{i'}, x_k)}$$

- Learning in instance-weighted Naïve Byes

$$p(c_j) = \frac{\sum_k I(y_k = c_j) \boxed{r(x_k)}}{\sum_k \boxed{r(x_k)}} \qquad p(t_i|c_j) = \frac{\sum_k I(y_k = c_j) \boxed{r(x_k)} N(t_i, x_k)}{\sum_k I(y_k = c_j) \boxed{r(x_k)} \sum_{i'=1}^{V} N(t_{i'}, x_k)}$$

Note: PUIW can also be applied to Discriminative model (e.g., weighted MaxEnt, weighted SVMs)

Experiments on Cross-domain Sentiment Classification

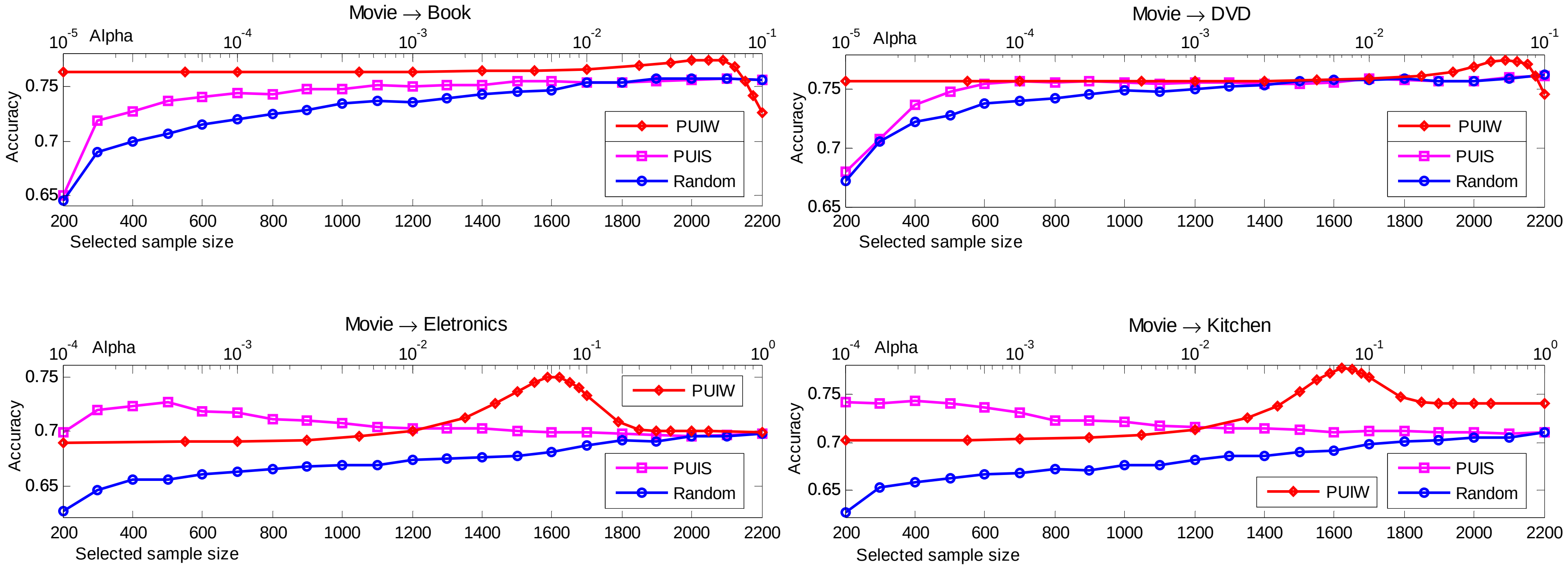
- Setting 1 (with a normal size of training set)
  - Source domain: Movie (training set size: 2,000)
  - Target domains: {Book, DVD, Electronics, Kitchen}

- Setting 2 (with a large size of training set)
  - Source domain: Video (training set size: 10,000)
  - Target domains: 12 domains of reviews extracted from the Multi-domain dataset

Classification Accuracy @ Normal-size Training Set

Task	KLD	ALL	PUIS	PUIW
Movie → Book	43.81	0.7568	0.7572	**0.7747**
Movie → DVD	33.54	0.7622	0.7622	**0.7818**
Movie → Electronics	104.52	0.6975	0.7265	**0.7500**
Movie → Kitchen	119.70	0.7097	0.7435	**0.7775**
Average	–	0.7316	0.7474	**0.7710**

Accuracy Curve
@ Normal-size Training Set

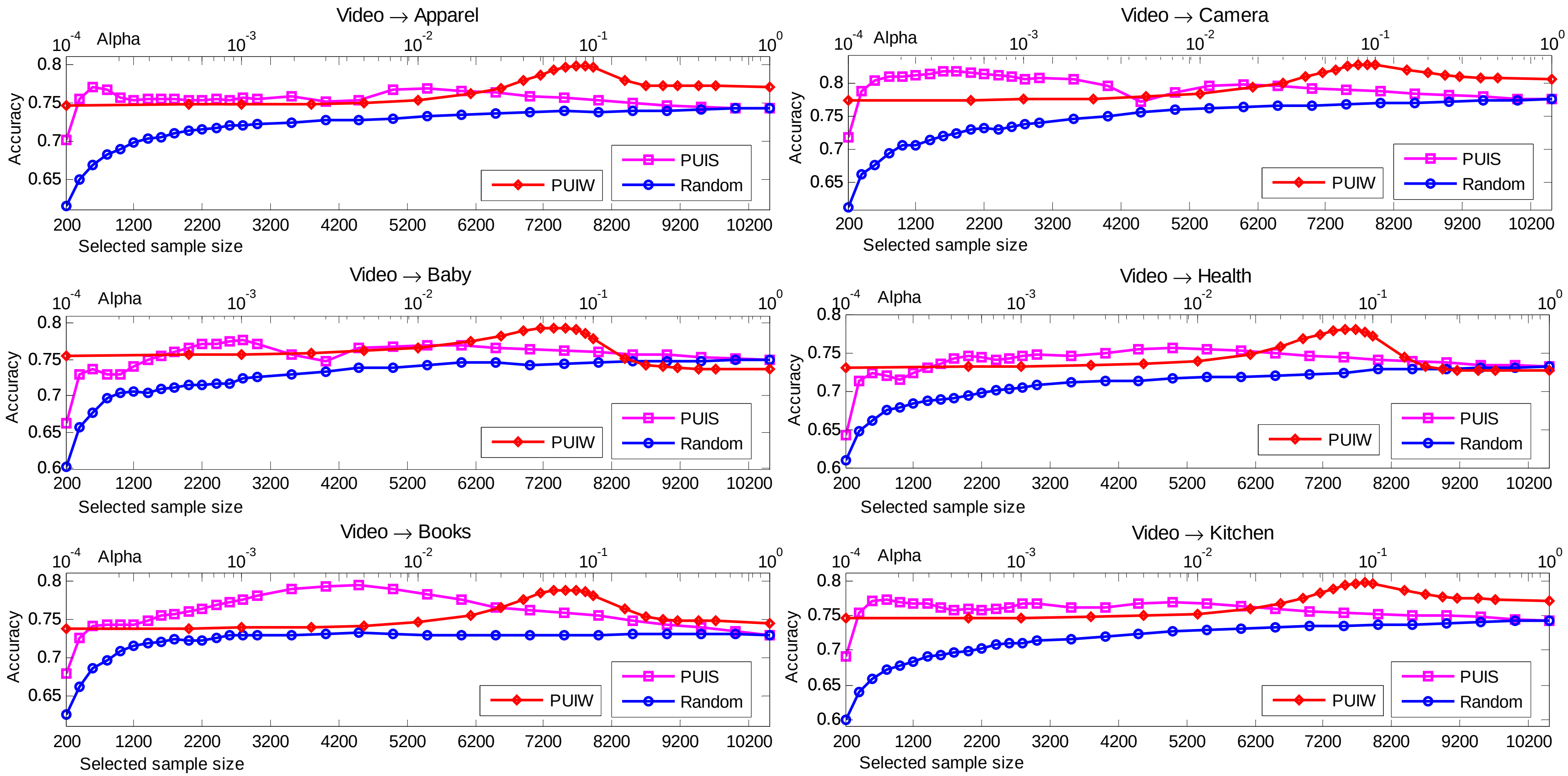


For Random and PUIS, the bottom x-axis (number of selected samples) is used; For PUIW, the top x-axis (calibration parameter alpha) is used.

Results @ Large-size Training Set

Task	KLD	ALL	PUIS	PUIW
Video → Apparel	166.69	0.7440	0.7704	**0.7995**
Video → Baby	160.50	0.7494	0.7759	**0.7932**
Video → Books	85.61	0.7328	**0.7952**	0.7893
Video → Camera	146.61	0.7747	0.8164	**0.8278**
Video → DVD	66.71	0.7877	0.8169	**0.8180**
Video → Electronics	143.87	0.7213	0.7603	**0.7712**
Video → Health	159.73	0.7331	0.7576	**0.7826**
Video → Kitchen	155.72	0.7424	0.7736	**0.7980**
Video → Magazines	122.53	0.8030	0.8344	**0.8484**
Video → Music	99.49	0.7562	0.7581	**0.7734**
Video → Software	136.48	0.7411	**0.8078**	0.7830
Video → Toys	134.84	0.7679	0.7858	**0.8066**
Average	–	0.7545	0.7877	**0.7993**

Accuracy Curve @ A Large-size Training Set



For Random and PUIS, the bottom x-axis is used; For PUIW, the top x-axis is used.

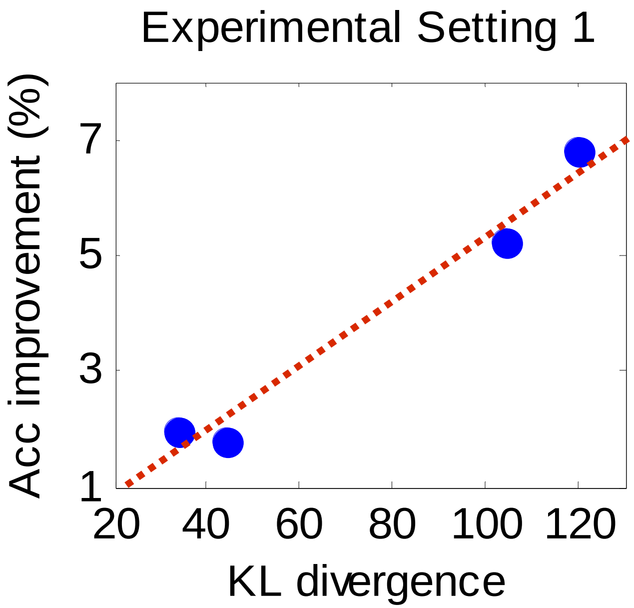
Conclusions from the Results

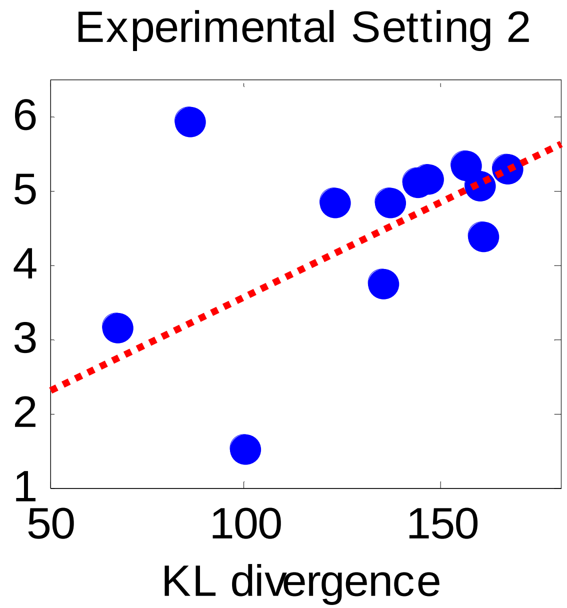
- The effect of adding more training samples
  - When the size of training data is small (<2000)
  - When the size of training data becomes larger (>3000)

- The necessity of PUIS/PUIW
  - The improvements of both PUIS/PUIW are significant
  - PUIW is overall better than PUIS

- The stability of PUIS/PUIW
  - The number of selected samples in PUIS is hard to determine
  - The curve of PUIW is unimodal (best \alpha≈0.1)

The Relation of K-L Distance and Accuracy Improvement





KLD and Accuracy Improvement are in a roughly linear relation.

Part 2. Instance Adaptation via In-target-domain Logistic Approximation (ILA)

In-target-domain Sampling Model

- PUIW (PU learning for Instance Weighting)

$$q_t(x) = r(x)p_s(x)$$
$$\propto \boxed{\frac{1}{1+\exp -\alpha f(x)}} p_s(x)$$

Shortcomings in PUIW: PU Learning and Probability Calibration are conducted separately

- ILA (In-target-domain logistic approximation)

$$q_t(x) \propto p(d=1|x)p_s(x)$$
$$= \boxed{\frac{1}{1+e^{-\omega^T x}}} p_s(x)$$

In ILA: We will learn the instance weight in one single model

Illustration of instance-weighted sampling

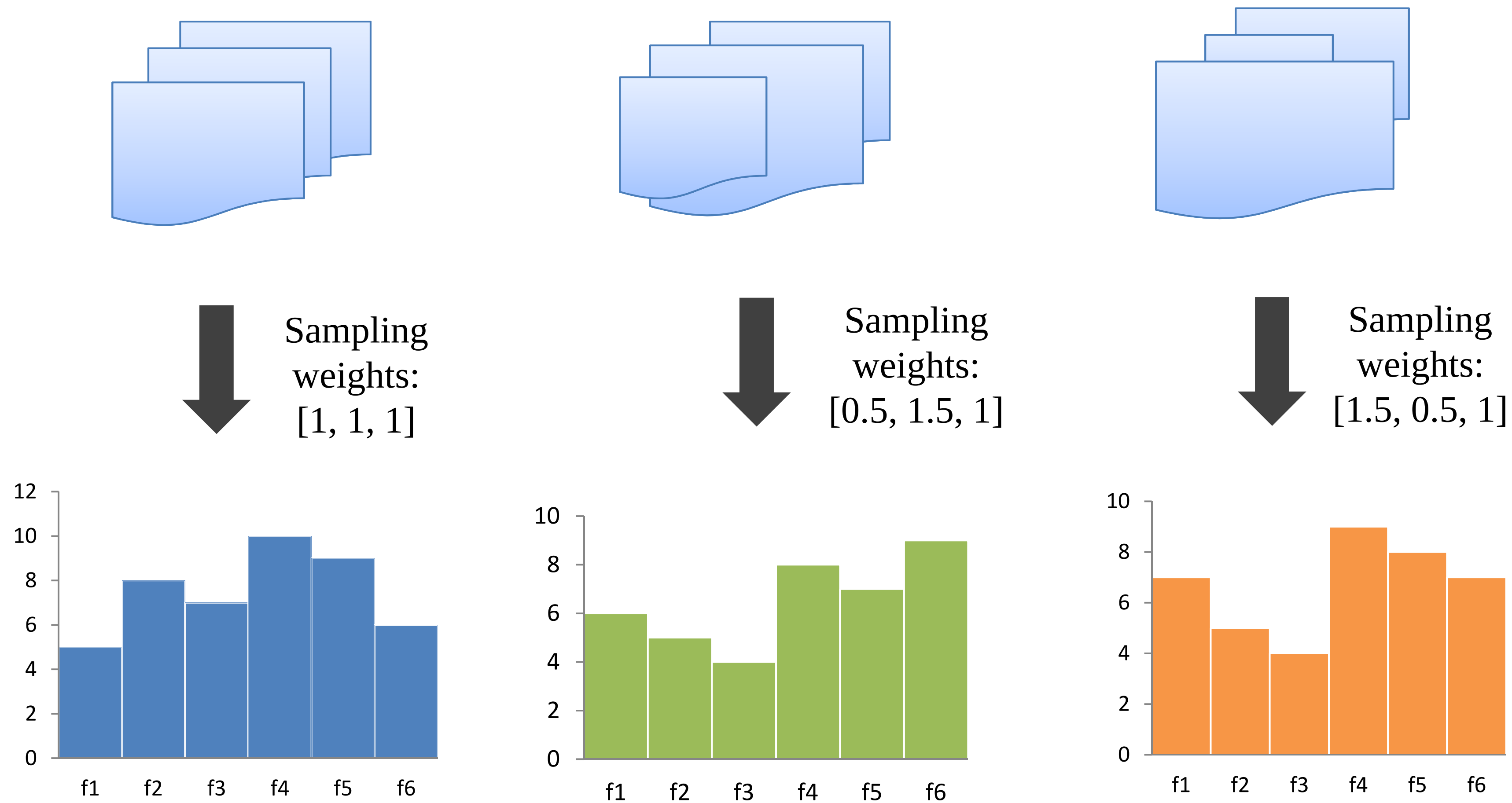
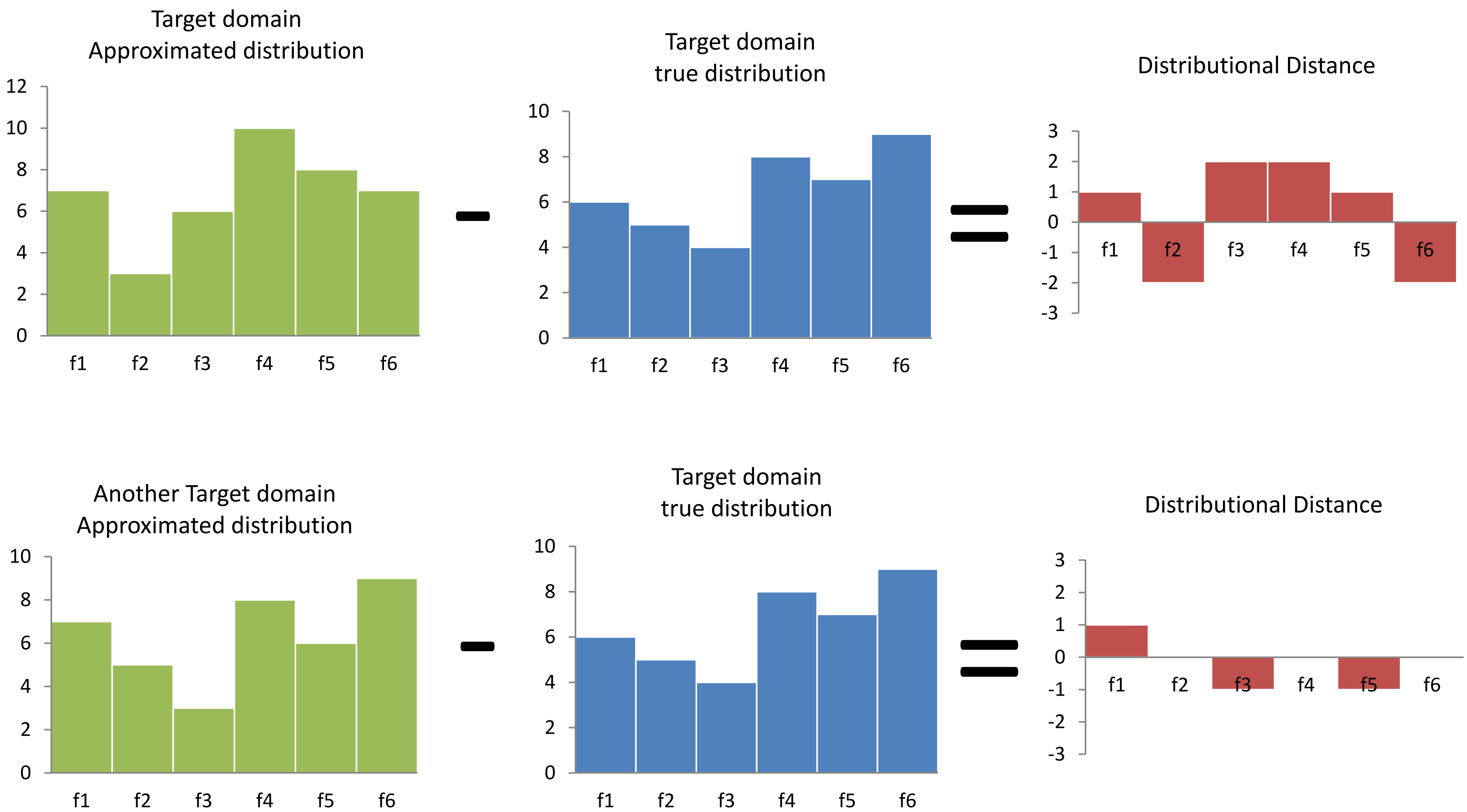


Illustration of parameter learning criterion



Parameter Learning in ILA

- K-L Distance (KLD) of the True and Approximated Target-domain Probability

$$
\begin{aligned}
KL(p_t(x)||q_t(x)) &= \int_{x\in\mathcal{X}} p_t(x)\log\frac{p_t(x)}{q_t(x)}dx \\
&= \int_{x\in\mathcal{X}} p_t(x)\log\frac{p_t(x)}{\frac{\beta}{1+e^{-\omega^T x}}p_s(x)}dx
\end{aligned}
$$

- Optimization with normalization constraint

$$
\min_{\omega,\beta} KL = \max_{\omega,\beta}\int_{x\in\mathcal{X}} p_t(x)\log\frac{\beta}{1+e^{-\omega^T x}}dx
$$

$$
s.t. \int_{x\in\mathcal{X}} q_t(x)dx = \int_{x\in\mathcal{D}_t}\frac{\beta}{1+e^{-\omega^T}}p_s(x)dx = 1
$$

Empirical Form of KLD Minimization

- Empirical K-L distance minimization

$$\max_{w,b,c} \frac{1}{N_t} \sum_{i=1}^{N_t} \log \frac{\beta}{1+e^{-w^T x_i}}$$

$$s.t. \frac{1}{N_s} \sum_{j=1}^{N_s} \frac{\beta}{1+e^{-w^T x_j}} = 1 \iff \beta = \frac{N_s}{\sum_{j=1}^{N_s} \frac{1}{1+e^{-w^T x_j}}}$$

- Final optimization formula (loss function)

$$w^* = \arg\max_w \frac{1}{N_t} \sum_{i=1}^{N_t} \log \frac{\frac{N_s}{\sum_{j=1}^{N_s} \frac{1}{1+e^{-w^T x_j}}}}{1+e^{-w^T x_i}}$$

$$= \arg\min_w \sum_{i=1}^{N_t} \log \sum_{j=1}^{N_s} \frac{\frac{1}{1+e^{-w^T x_j}}}{\frac{1}{1+e^{-w^T x_i}}}$$

A standard unconstraint optimization problem!

Gradient-based Optimization

- Gradient of the loss function

$$\frac{\partial J}{\partial \omega_k} = \frac{1}{\sum_{j=1}^{N_s} s(\omega^T x_j)} \sum_{j=1}^{N_s} s(\omega^T x_j)(1 - s(\omega^T x_j)) x_{j,k} - \frac{1}{N_t} \sum_{i=1}^{N_t} (1 - s(\omega^T x_i)) x_{i,k}$$

where $s(\omega^T x_j) = \frac{1}{1+e^{-w^T x_j}}$

- Optimization method
  - Gradient Descent

$$\omega_k^{(t+1)} = \omega_k^{(t)} - \eta \frac{\partial J}{\partial \omega_k^{(t)}}$$

  - Newton, Quasi-Newton, L-BFGS can also work

Instance-weighted Classification

- In-target-domain sampling weights for each source-domain labeled data

$$r(x_n) = \frac{q_t(x_n)}{p_s(x_n)} = \frac{\beta}{1 + e^{-\omega^T x_n}} \text{ where } \beta = \frac{N_s}{\sum_{j=1}^{N_s} \frac{1}{1+e^{-w^T x_j}}}$$

- Instance-weighted classification model
  – Generative: weighted naïve Bayes
  – Discriminative: weighted logistic regression, weighted SVMs

Instance Adaptation Feature Selection

- Domain-sensitive Information Gain

$$
\begin{aligned}
IG(t_k) = &- \sum_{l \in \{0,1\}} p(d=l) \log p(d=l) \\
&+ p(t_k) \sum_{l \in \{0,1\}} p(d=l|t_k) \log p(d=l|t_k) \\
&+ p(\bar{t}_k) \sum_{l \in \{0,1\}} p(d=l|\bar{t}_k) \log p(d=l|\bar{t}_k)
\end{aligned}
$$

- Feature Space
  - ILA: from original feature space to dimension-reduced feature space $\mathbf{x} \rightarrow \hat{\mathbf{x}}$
  - KLIEP: from original feature space to kernel space $\mathbf{x} \rightarrow \phi(\mathbf{x})$

Experimental Settings

- Setting 1: Cross-domain Document Categorization
  - 20 Newsgroups dataset
  - Top categories are used as class labels, and subcategories are used to generate source and target domains [Dai, 2007]
- Setting 2: Cross-domain Sentiment Classification
  - Source domain: Movie
  - Target domains: {Book, DVD, Electronics, Kitchen}

Experimental Results

- Cross-domain Document Categorization

Dataset	K-L divergence	No Adaptation	KLIEP		PUIS	PUIW	ILA
			Linear	Gaussian			
sci vs com	28.3	0.602	0.504	0.624	0.602	0.619	**0.630**
talk vs com	18.5	0.908	0.910	0.922	0.909	0.907	**0.959**
sci vs talk	29.3	0.852	0.851	0.855	0.880	0.863	**0.921**
rec vs sci	28.3	0.651	0.593	0.652	0.689	**0.742**	**0.742**
rec vs com	20.6	0.900	0.693	0.910	0.901	0.911	**0.922**
talk vs rec	35.3	0.820	0.821	0.821	0.821	0.834	**0.837**
Avg.	26.7	0.788	0.729	0.797	0.800	0.813	**0.835**

"A vs B" means that the top category A and B are used as class labels, and subcategories under the top categories are used to generate the source and target domain datasets.

Experimental Results

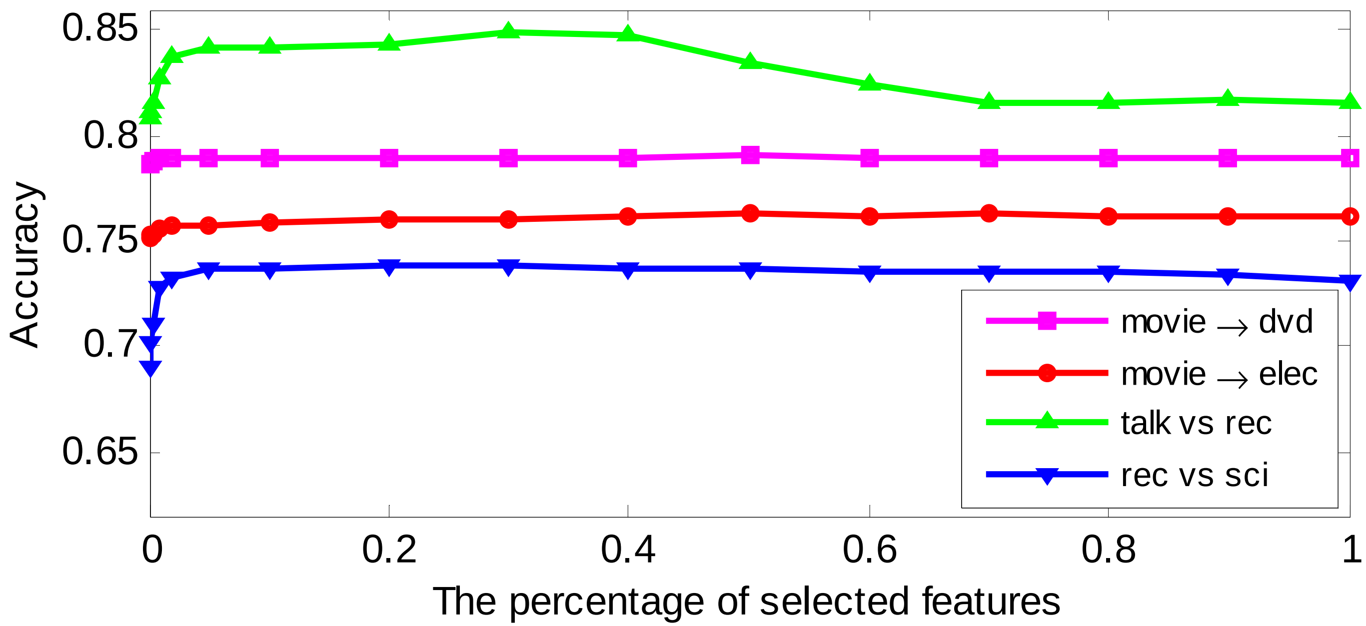
- Cross-domain Sentiment Classification

Dataset	K-L divergence	No Adaptation	KLIEP		PUIS	PUIW	ILA
			Linear	Gaussian			
movie → book	4.06	0.756	0.737	0.768	0.757	0.774	**0.780**
movie → dvd	2.12	0.762	0.738	0.783	0.762	0.782	**0.796**
movie → elec	13.4	0.697	0.673	0.741	0.726	0.750	**0.768**
movie → kitchen	13.4	0.709	0.626	0.759	0.743	0.777	**0.785**
Average	8.25	0.731	0.694	0.763	0.747	0.771	**0.783**

"A → B" denote that we use dataset A as the source domain, and B as the target domain.

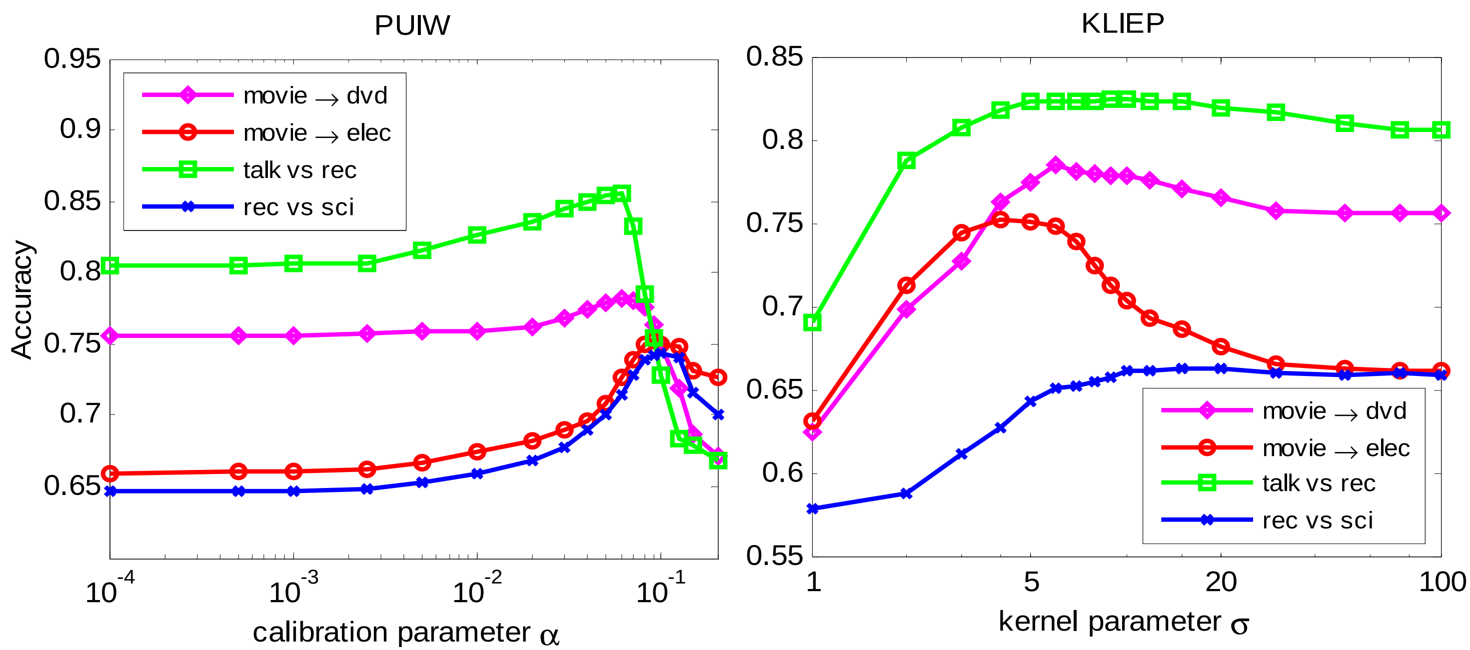
Parameter Sensitivity

- Effect of Instance Adaptation Feature Selection



ILA can work efficiently in a dimensionality-reduced linear feature

Comparison of Parameter Stability



Parameter stability of PUIW and KLIEP-Gaussian. The x-axis denotes the value of the calibration parameter \alpha in PUIW (left), and the kernel parameter \delta in KLIEP-Gaussian (right).

ILA is more convenient at parameter tuning in comparison with PUIW and KLIEP

Comparison of Computational Efficiency

- Computational Time Cost

Task	KLIEP-Gaussian		PUIW	ILA
	#1000	#100		
Text categorization	19346s	2121s	241s	161s
Sentiment classification	5219s	482s	184s	97s

- Overall Summary
  - Joint model rather than separate model
  - Parameters are learnt rather then tuned
  - Better performance in Classification Accuracy, Parameter Stability, and Computational Efficiency

Feasibility to apply to the other NLP tasks

- Classification Task
  - Cross-domain Text / Sentiment Classification
  - Cross-domain NER?
  - Cross-domain WSD?
- Sequential Learning Task
  - Cross-domain Chinese word segmentation?
  - Cross-domain parsing?
- Bilingual Alignment Learning Task
  - Cross-domain MT?

Any Questions?