Introduction to Domain Adaptation

guest lecturer: Ming-Wei Chang
CS 546

Spring, 2009

Before we start

Acknowledgment

- We use slides from (Jiang and Zhai 2007),(Daumé III 2007) and (Blitzer, McDonald, and Pereira 2006) extensively.
- Therefore, the pages number are sometimes off.

Before we start

Acknowledgment

- We use slides from (Jiang and Zhai 2007),(Daumé III 2007) and (Blitzer, McDonald, and Pereira 2006) extensively.
- Therefore, the pages number are sometimes off.

Please ask questions....

- Things get interesting only after we understand something.
- We do not need to go over all of the slides.
  - But I will try to cover the most important points.

The first step . . .

The Definition of Domain Adaptation

Natural language processing

- The Ultimate Goal:
  Build systems that can understand natural language.

Natural language processing

- The Ultimate Goal:
  Build systems that can understand natural language.
- Unfortunately, not easy.

Natural language processing

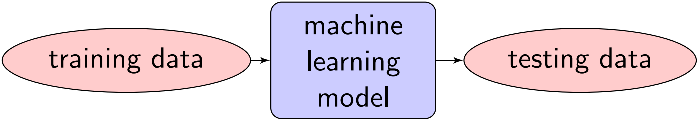
- The Ultimate Goal:
  Build systems that can understand natural language.
- Unfortunately, not easy.
- Intermediate goals:
  Build systems can do parsing, tagging, . . . well

Natural language processing

- The Ultimate Goal:
  Build systems that can understand natural language.
- Unfortunately, not easy.
- Intermediate goals:
  Build systems can do parsing, tagging, ... well
- The current tools: machine learning algorithms

Natural language processing

- The Ultimate Goal:
  Build systems that can understand natural language.
- Unfortunately, not easy.
- Intermediate goals:
  Build systems can do parsing, tagging, . . . well
- The current tools: machine learning algorithms



Example: named entity recognition (NER)

- Input: Jim bought 300 shares of Acme Corp. in 2006.

Example: named entity recognition (NER)

- Input: Jim bought 300 shares of Acme Corp. in 2006.
- Output: [PER Jim] bought 300 shares of [ORG Acme Corp.] in 2006.

Example: named entity recognition (NER)

- Input: Jim bought 300 shares of Acme Corp. in 2006.
- Output: [PER Jim] bought 300 shares of [ORG Acme Corp.] in 2006.
- It is often ambiguous... For example, Bush can be a person...

Example: named entity recognition (NER)

- Input: Jim bought 300 shares of Acme Corp. in 2006.
- Output: [PER Jim] bought 300 shares of [ORG Acme Corp.] in 2006.
- It is often ambiguous... For example, Bush can be a person...
- . . . or not. 

How to solve NER

Example

B-PER	O	O	O	O	B-ORG	I-ORG	O	O
Jim	bought	300	shares	of	Acme	Corp.	in	2006

How to solve NER

Example

B-PER	O	O	O	O	B-ORG	I-ORG	O	O
Jim	bought	300	shares	of	Acme	Corp.	in	2006

Feature Vectors

For the word "Acme", the feature vector

1. **Current word and its part of speech tag**: Acme, NNP
2. **Two words before the current word**
3. **Two words after the current word**

How to solve NER

Example

B-PER	O	O	O	O	B-ORG	I-ORG	O	O
Jim	bought	300	shares	of	Acme	Corp.	in	2006

Feature Vectors

For the word "Acme", the feature vector

1. **Current word and its part of speech tag**: Acme, NNP
2. **Two words before the current word**
3. **Two words after the current word**

Training and Testing

- A multi-class classification problem: Logistic Regression, SVM

The current status of NER

Quote from Wikipedia

"State-of-the-art NER systems produce near-human performance. For example, the best system entering MUC-7 scored 93.39% of f-measure while human annotators scored 97.60% and 96.95%"

The current status of NER

Quote from Wikipedia

"State-of-the-art NER systems produce near-human performance. For example, the best system entering MUC-7 scored 93.39% of f-measure while human annotators scored 97.60% and 96.95%"

Wow, that is so cool! At the end, we finally solved something!

The current status of NER

Quote from Wikipedia

"State-of-the-art NER systems produce near-human performance. For example, the best system entering MUC-7 scored 93.39% of f-measure while human annotators scored 97.60% and 96.95%"

Wow, that is so cool! At the end, we finally solved something!

Truth: The NER problem is still not solved. Why?

The problem: domain over-fitting

- The issues of supervised machine learning algorithms: Need Labeled Data
- What people have done: Labeled large amount of data on news corpus
- However, it is still not enough.....
- The Web contains all kind of data....
  - Blogs, Novels, Biomedical Documents, ...
  - Many domains!
- We might do a good job on news domain, but not on other domains...

Domain Adaptation

- Many NLP tasks are cast into classification problems
- Lack of training data in new domains
- Domain adaptation:
  - POS: WSJ → biomedical text
  - NER: news → blog, speech
  - Spam filtering: public email corpus → personal inboxes
- Domain overfitting

NER Task	Train → Test	F1
to find PER, LOC, ORG from news text	NYT → NYT	0.855
	Reuters → NYT	0.641
to find gene/protein from biomedical literature	mouse → mouse	0.541
	fly → mouse	0.281

Possible solutions?

- Don't care (The current solution)
  - Bad performance

Possible solutions?

- Don't care (The current solution)
  - Bad performance
- Annotate more data
  - Annotate data for the new domain?
  - Need create data for each new domain

Possible solutions?

- Don't care (The current solution)
  - Bad performance
- Annotate more data
  - Annotate data for the new domain?
  - Need create data for each new domain
- Build a generic corpus?
  - Wikipedia
  - Good, still not cover all possible solutions. For example, NER for a company.
  - Not sure about the performance

Possible solutions?

- Don't care (The current solution)
  - Bad performance
- Annotate more data
  - Annotate data for the new domain?
  - Need create data for each new domain
- Build a generic corpus?
  - Wikipedia
  - Good, still not cover all possible solutions. For example, NER for a company.
  - Not sure about the performance
- Special design algorithms (for each domain)
  - Good, but need to redesign for every domain

Possible solutions?

- Don't care (The current solution)
  - Bad performance
- Annotate more data
  - Annotate data for the new domain?
  - Need create data for each new domain
- Build a generic corpus?
  - Wikipedia
  - Good, still not cover all possible solutions. For example, NER for a company.
  - Not sure about the performance
- Special design algorithms (for each domain)
  - Good, but need to redesign for every domain
- Our Focus: General purpose adaptation algorithms

Why does the performance drop?

Different distributions

- $P(x)$: The distribution of training and testing data are different
- $P(y|x)$: With the same example, the label are different in different domains

Why does the performance drop?

Different distributions

- $P(x)$: The distribution of training and testing data are different
- $P(y|x)$: With the same example, the label are different in different domains

Unknown words

- There are many unseen words in the new domain

Why does the performance drop?

Different distributions

- $P(x)$: The distribution of training and testing data are different
- $P(y|x)$: With the same example, the label are different in different domains

Unknown words

- There are many unseen words in the new domain

New Types

- There are some new types in the new domain. For example, now predicting locations.
- We will not talk about this today.

What we are going to talk about today

- We will talk about several previous works
- Cover several issues that mention in the previous slides
- Reminder: It is still an open problem. There might exist better solutions.
- Terminology:
  - source domain, the domain we know a lot
  - target domain, the domain we do not know (or know very little)
  - we want to evaluate on target domain

Recap: training a standard logistic regression

Training a standard logistic regression model: a review

- Training data: L
- Training procedure: find the best w by maximizing $P(w|L)$

$$P(w|L) = \frac{P(L|w)P(w)}{P(L)} \propto P(L|w)\,\mathrm{P(w)}$$

- The fist term: training error, The second term: regularization term

$$w \leftarrow \arg\max \log P(L|w) + \log P(w)$$

Recap: training a standard logistic regression

Training a standard logistic regression model: a review

- Training data: L
- Training procedure: find the best w by maximizing $P(w|L)$

$$P(w|L) = \frac{P(L|w)P(w)}{P(L)} \propto P(L|w)\ \mathrm{P(w)}$$

- The fist term: training error, The second term: regularization term

$$w \leftarrow \arg\max \log P(L|w) + \log P(w)$$

Regularization

- Assume the prior is the Gaussian distribution
- $\log P(w) = \frac{1}{2\sigma^2}\|w\|^2 + C$
- Similar to SVM, want to find the maximum margin line.

Outline

Different distributions: $P(X)$

- Solution: Instance Weighting
- (Bickel, Brüeckner, and Scheffer 2007), (Jiang and Zhai 2007)

Outline

Different distributions: $P(X)$

- Solution: Instance Weighting
- (Bickel, Brüeckner, and Scheffer 2007), (Jiang and Zhai 2007)

Different distributions: $P(Y \mid X)$

- Assume $P_s(Y \mid X)$ is close to $P_t(Y \mid X)$
- Solution: Regularization
- (Evgeniou and Pontil 2004), (Daumé III 2007)

Outline

Different distributions: $P(X)$

- Solution: Instance Weighting
- (Bickel, Brüeckner, and Scheffer 2007), (Jiang and Zhai 2007)

Different distributions: $P(Y \mid X)$

- Assume $P_s(Y \mid X)$ is close to $P_t(Y \mid X)$
- Solution: Regularization
- (Evgeniou and Pontil 2004), (Daumé III 2007)

Unknown Words

- Solution: Find the relations between old words and unseen words through auxiliary tasks
- (Ando and Zhang 2005), (Blitzer, McDonald, and Pereira 2006)

Outline

Different distributions: $P(X)$

- Solution: Instance Weighting
- (Bickel, Brüeckner, and Scheffer 2007), (Jiang and Zhai 2007)

Different distributions: $P(Y \mid X)$

- Assume $P_s(Y \mid X)$ is close to $P_t(Y \mid X)$
- Solution: Regularization
- (Evgeniou and Pontil 2004), (Daumé III 2007)

Unknown Words

- Solution: Find the relations between old words and unseen words through auxiliary tasks
- (Ando and Zhang 2005), (Blitzer, McDonald, and Pereira 2006)

Why doing instance weighting?

What we really want

- Training and testing distributions can be different: λ, θ
- Minimize the expected loss on testing data
- Find a function f that minimize

$$E_{(x,y)\sim\theta}[loss(f(x), y)]$$

Why doing instance weighting?

What we really want

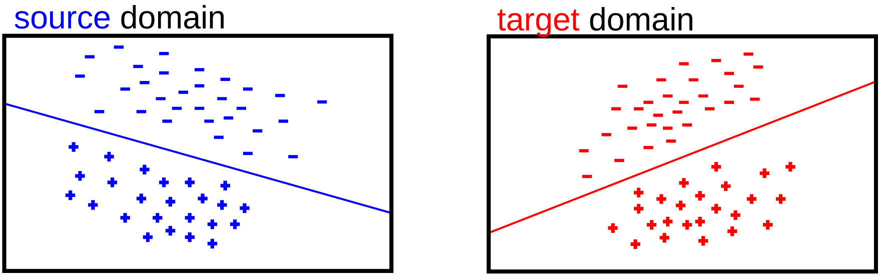
- Training and testing distributions can be different: λ, θ
- Minimize the expected loss on testing data
- Find a function f that minimize

$$E_{(x,y)\sim\theta}[loss(f(x), y)]$$

Intuitions: A good weighting algorithm should ...

- Put more weights on the training examples that are similar to testing examples
- Put less weights on the training examples that are not similar to testing examples

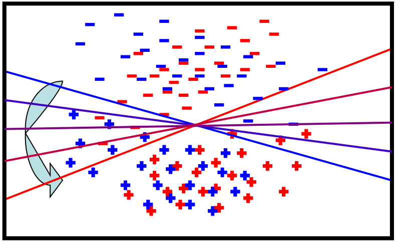
The Need for Domain Adaptation



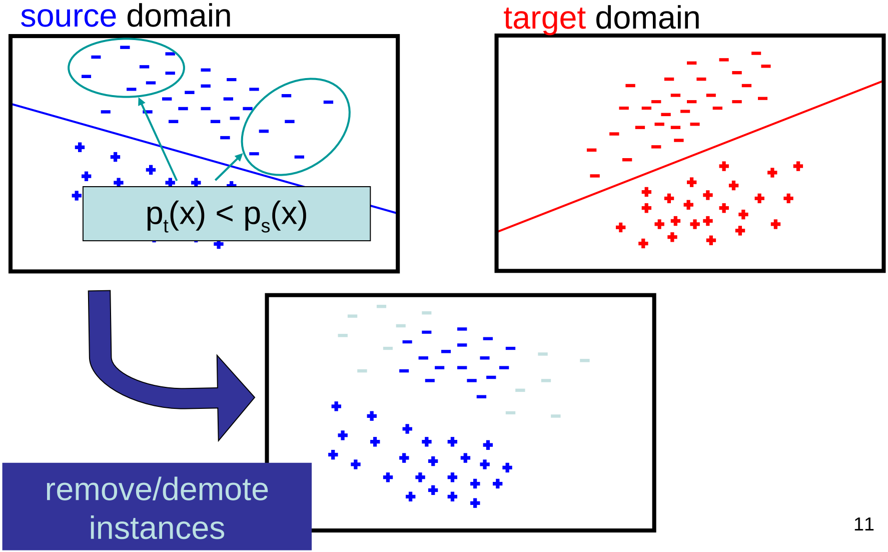
The Need for Domain Adaptation

source domain

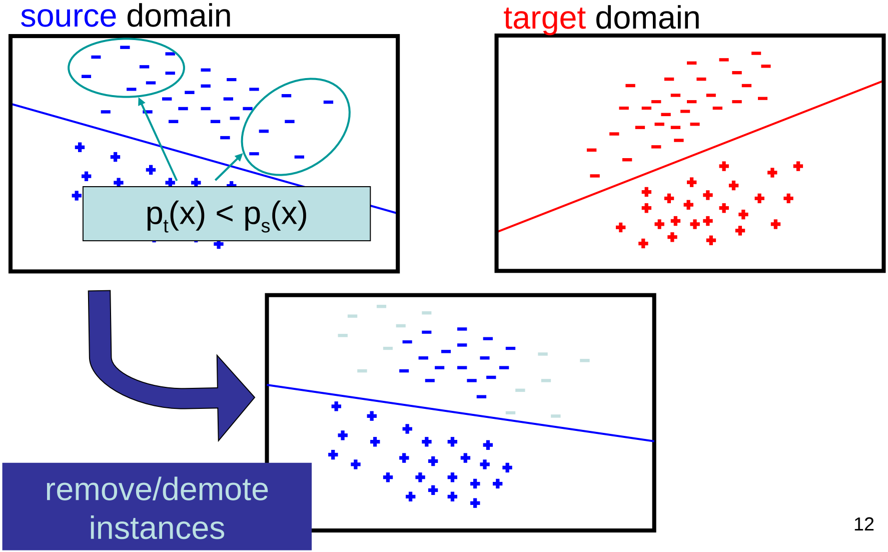
target domain



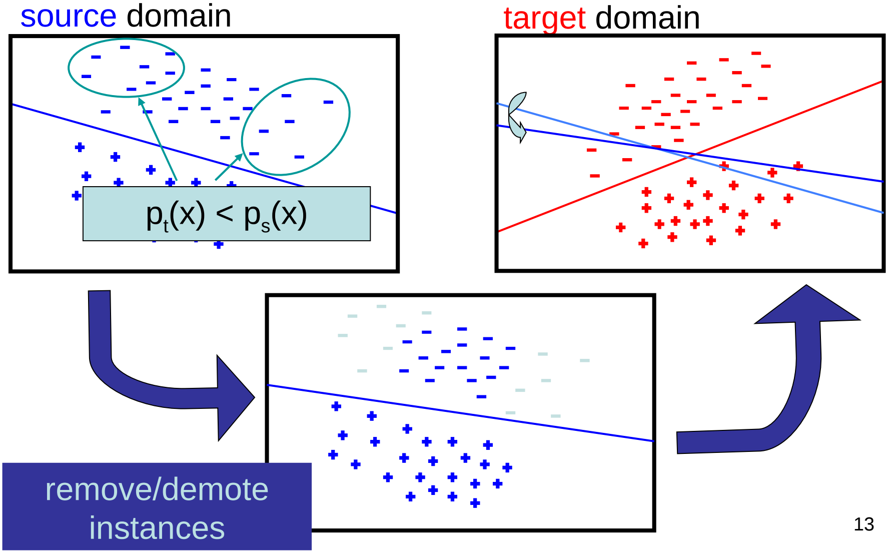
An Instance Weighting Solution (Instance Adaptation: $p_t(x) < p_s(x)$)



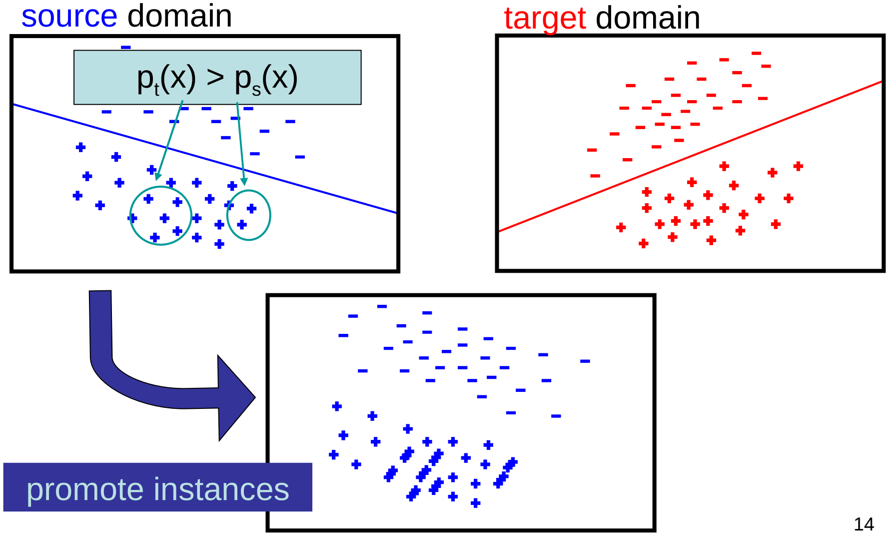
An Instance Weighting Solution
(Instance Adaptation: $p_t(x) < p_s(x)$)



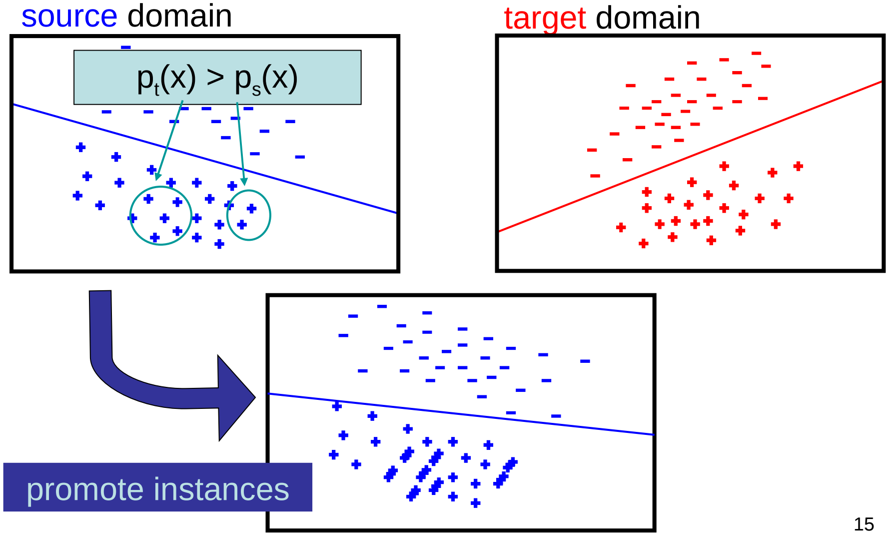
An Instance Weighting Solution
(Instance Adaptation: $p_t(x) < p_s(x)$)



An Instance Weighting Solution
(Instance Adaptation: $p_t(x) > p_s(x)$)

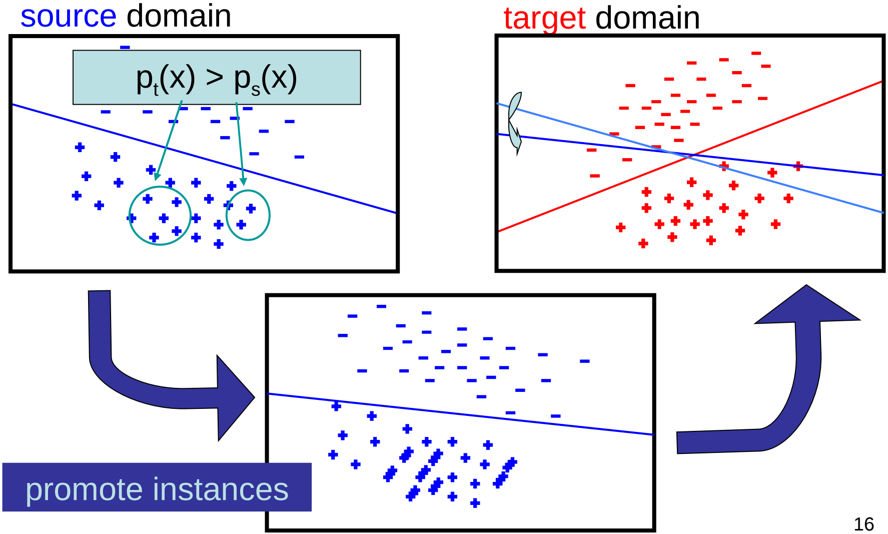


An Instance Weighting Solution
(Instance Adaptation: $p_t(x) > p_s(x)$)

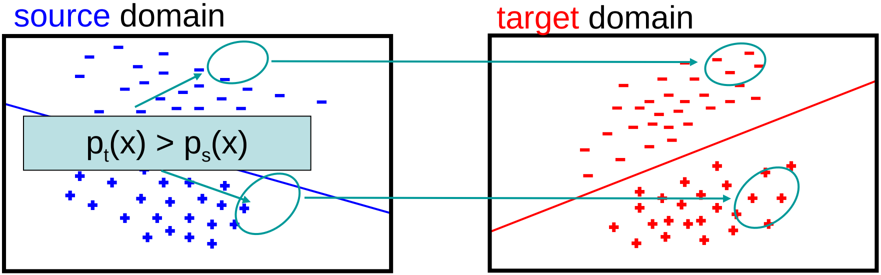


An Instance Weighting Solution

(Instance Adaptation: $p_t(x) > p_s(x)$)



An Instance Weighting Solution
(Instance Adaptation: $p_t(x) > p_s(x)$)



- Labeled target domain instances are useful
- Unlabeled target domain instances may also be useful

The problem now: how to figure out the weights

Assumptions: (Bickel, Brüeckner, and Scheffer 2007)

- Labeled source data
- Unlabeled target data
- No labeled target data is available

Some simple arguments

- The weight should be $\frac{P(x|\theta)}{P(x|\lambda)}$ (Shimodaira 2000)
- One can show $P(x, y|\theta) = P(x, y|\lambda)\frac{P(x|\theta)}{P(x|\lambda)}$

The problem now: how to figure out the weights

Assumptions: (Bickel, Brüeckner, and Scheffer 2007)

- Labeled source data
- Unlabeled target data
- No labeled target data is available

Some simple arguments

- The weight should be $\frac{P(x|\theta)}{P(x|\lambda)}$ (Shimodaira 2000)
- One can show $P(x, y|\theta) = P(x, y|\lambda)\frac{P(x|\theta)}{P(x|\lambda)}$
- The Idea: Learn another model to estimate $\frac{P(x|\theta)}{P(x|\lambda)}$

Figuring out the weights

An additional model

- Learn a model to predict if the example are coming from training data or testing data
- $P(\sigma|x, \lambda, \theta)$. Training: $\sigma = 1$, Testing: $\sigma = 0$.

Figuring out the weights

An additional model

- Learn a model to predict if the example are coming from training data or testing data
- $P(\sigma|x,\lambda,\theta)$. Training: $\sigma = 1$, Testing: $\sigma = 0$.
- Using Bayes rules, it is fairly easy to show

$$P(x|\lambda) = P(x|\sigma = 1, \theta, \lambda) = \frac{P(\sigma = 1|x, \lambda, \theta)P(x|\theta, \lambda)}{P(\sigma = 1|\lambda, \theta)}$$

Figuring out the weights

An additional model

- Learn a model to predict if the example are coming from training data or testing data
- $P(\sigma|x,\lambda,\theta)$. Training: $\sigma = 1$, Testing: $\sigma = 0$.
- Using Bayes rules, it is fairly easy to show

$$P(x|\lambda) = P(x|\sigma = 1, \theta, \lambda) = \frac{P(\sigma = 1|x,\lambda,\theta)P(x|\theta,\lambda)}{P(\sigma = 1|\lambda,\theta)}$$

- And,

$$\frac{P(x|\theta)}{P(x|\lambda)} = \frac{P(\sigma = 1|\lambda,\theta)P(\sigma = 0|x,\lambda,\theta)}{P(\sigma = 0|\lambda,\theta)P(\sigma = 1|x,\lambda,\theta)}$$

What does it mean?

The Equation Revisited

$$\frac{P(x|\theta)}{P(x|\lambda)} = \frac{P(\sigma{=}1|\lambda,\theta)}{P(\sigma{=}0|\lambda,\theta)} \quad \frac{P(\sigma{=}0|x,\lambda,\theta)}{P(\sigma{=}1|x,\lambda,\theta)}$$

What does it mean?

The Equation Revisited

$$\frac{P(x|\theta)}{P(x|\lambda)} = \frac{P(\sigma{=}1|\lambda,\theta)}{P(\sigma{=}0|\lambda,\theta)} \; \frac{P(\sigma{=}0|x,\lambda,\theta)}{P(\sigma{=}1|x,\lambda,\theta)}$$

- The First Term: Just calculate the frequency!
- The Second Term: The confidence from the additional classifier

What does it mean?

The Equation Revisited

$$\frac{P(x|\theta)}{P(x|\lambda)} = \boxed{\frac{P(\sigma{=}1|\lambda,\theta)}{P(\sigma{=}0|\lambda,\theta)}} \boxed{\frac{P(\sigma{=}0|x,\lambda,\theta)}{P(\sigma{=}1|x,\lambda,\theta)}}$$

- The First Term : Just calculate the frequency!
- The Second Term : The confidence from the additional classifier

Algorithm 1: Two stages approaches

- Train a classifier $P(\sigma|x,\lambda,\theta)$
- Apply the classifier on the training instances and get their weight s_i
- Minimize the loss function on the training data

$$\sum_i s_i \text{Loss}(f(x_i), y_i)$$

An alternative approach: joint learning

Training a standard logistic regression model: a review

- Training data: L
- Training procedure: find the best w by maximizing $P(w|L)$

$$P(w|L) = \frac{P(L|w)P(w)}{P(L)} \propto P(L|w)\ \mathrm{P(w)}$$

- The fist term: training error, The second term: regularization term

An alternative approach: joint learning

Training a standard logistic regression model: a review

- Training data: L
- Training procedure: find the best w by maximizing $P(w|L)$

$$P(w|L) = \frac{P(L|w)P(w)}{P(L)} \propto P(L|w)\ \mathrm{P(w)}$$

- The fist term: training error, The second term: regularization term

The Idea: Learn two models together

- We can learn the classification model (for real task) and the addition model (for figuring out the weight) together!

An alternative approach: joint learning (cond.)

Training a standard logistic regression model: a review

$$\max P(w|L) \propto P(L|w)\, \mathrm{P(w)}$$

An alternative approach: joint learning (cond.)

Training a standard logistic regression model: a review

$$\max P(w|L) \propto P(L|w)\ \mathrm{P(w)}$$

Algorithm 2: Joint approach

- Training data L, unlabeled testing data T.
- The weight vector for original task w. The weight vector for the "instance weight" v.

An alternative approach: joint learning (cond.)

Training a standard logistic regression model: a review

$$\max P(w|L) \propto P(L|w)\ \mathrm{P(w)}$$

Algorithm 2: Joint approach

- Training data L, unlabeled testing data T.
- The weight vector for original task w. The weight vector for the "instance weight" v.

$$P(w, v|L, T) = P(w|v, L, T)P(v|L, T) = P(w|v, L)P(v|L, T)$$
$$\propto P(L|v, w)\ P(L, T|v)\ \mathrm{P(w)\ P(v)}$$

An alternative approach: joint learning (cond.)

Training a standard logistic regression model: a review

$$\max P(w|L) \propto P(L|w)\ \mathrm{P(w)}$$

Algorithm 2: Joint approach

- Training data L, unlabeled testing data T.
- The weight vector for original task w. The weight vector for the "instance weight" v.

$$P(w, v|L, T) = P(w|v, L, T)P(v|L, T) = P(w|v, L)P(v|L, T)$$
$$\propto P(L|v, w)\ P(L, T|v)\ \mathrm{P(w)\ P(v)}$$

- Weighted Training Error for w , Training Error for v
- Training: use newton method to optimize this function with w and v

An alternative approach: joint learning (cond.)

Training a standard logistic regression model: a review

$$\max P(w|L) \propto P(L|w)\ \mathrm{P(w)}$$

Algorithm 2: Joint approach

- Training data L, unlabeled testing data T.
- The weight vector for original task w. The weight vector for the "instance weight" v.

$$P(w,v|L,T) = P(w|v,L,T)P(v|L,T) = P(w|v,L)P(v|L,T)$$
$$\propto P(L|v,w)\ P(L,T|v)\ \mathrm{P(w)}\ \mathrm{P(v)}$$

- Weighted Training Error for w, Training Error for v
- Training: use newton method to optimize this function with w and v
- Much better than Algorithm 1

Using heuristics to find weights

- The work (Jiang and Zhai 2007) assumes
  - Source labeled instances
  - small amount labeled target instances,
  - large amount labeled target instances

Using heuristics to find weights

- The work (Jiang and Zhai 2007) assumes
  - Source labeled instances
  - small amount labeled target instances,
  - large amount labeled target instances
- They do not train an additional model for figuring out weights of instances

Using heuristics to find weights

- The work (Jiang and Zhai 2007) assumes
  - Source labeled instances
  - small amount labeled target instances, $P_t(Y|x)$
  - large amount labeled target instances
- They do not train an additional model for figuring out weights of instances
- Instead, they use the target weight vector to select the examples

$$s_i = \begin{cases} 1, & \text{if } P_t(y_i|x_i) > t \\ 0, & \text{otherwise} \end{cases}$$

Using heuristics to find weights

- The work (Jiang and Zhai 2007) assumes
  - Source labeled instances
  - small amount labeled target instances, $P_t(Y|x)$
  - large amount labeled target instances
- They do not train an additional model for figuring out weights of instances
- Instead, they use the target weight vector to select the examples

$$s_i = \begin{cases} 1, & \text{if } P_t(y_i|x_i) > t \\ 0, & \text{otherwise} \end{cases}$$

- They also found that when training everything together, we should put more weights on the target labeled data.

Summary: Instance Weighting

What we have discussed

- Putting weights on instances is useful in many adaptation tasks
- We learn some algorithms and some heuristics about how to use unlabeled target examples to change the instance weight
- Two possible solutions:
    1. Train an additional model to tell if x comes from training or testing
    2. Using some heuristics to guess the training weights

Outline

Different distributions: $P(X)$

- Solution: Instance Weighting
- (Bickel, Brüeckner, and Scheffer 2007), (Jiang and Zhai 2007)

Different distributions: $P(Y \mid X)$

- Assume $P_s(Y \mid X)$ is close to $P_t(Y \mid X)$
- Solution: Regularization
- (Evgeniou and Pontil 2004), (Daumé III 2007)

Unknown Words

- Solution: Find the relations between known words and unseen words through auxiliary tasks
- (Ando and Zhang 2005), (Blitzer, McDonald, and Pereira 2006)

What is the best training strategy?

Assumption

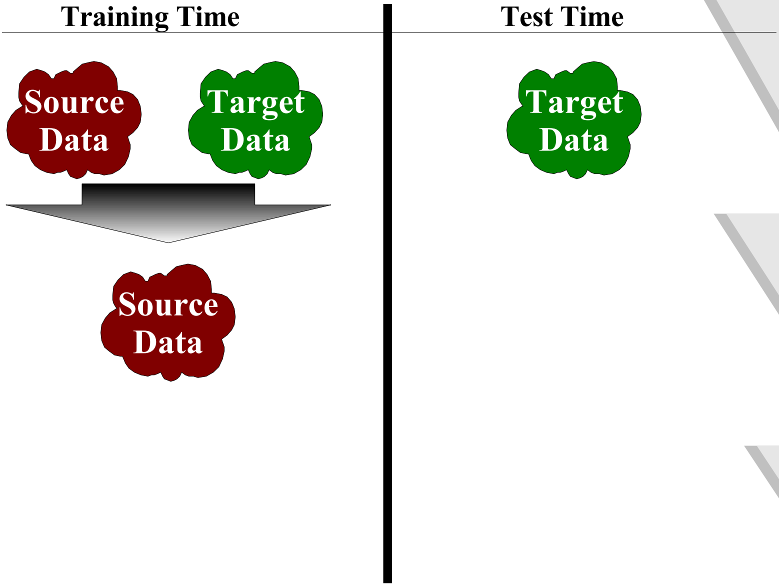
- We have both labeled source instances and labeled target instances
- The source label distribution is similar to the target distribution

$$P_s(y|x) \sim P_t(y|x)$$

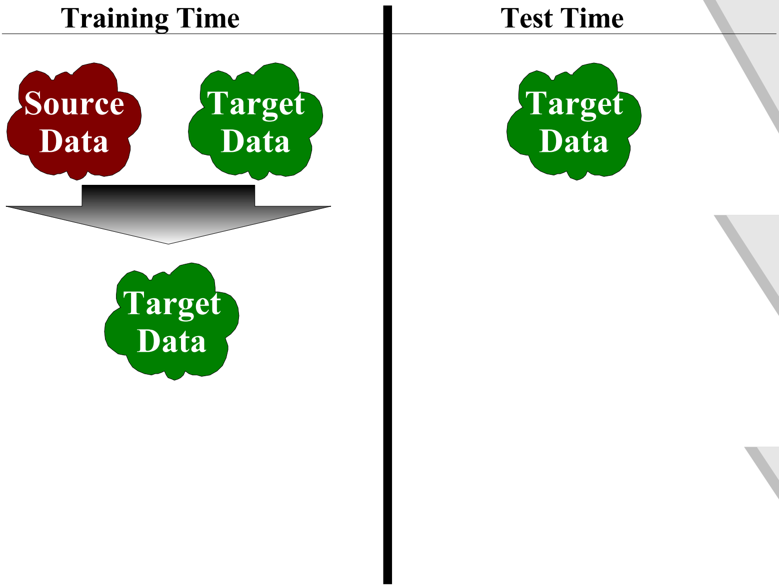
Possible Solutions

- Source only?
- Target only?
- Can you think of anything else?

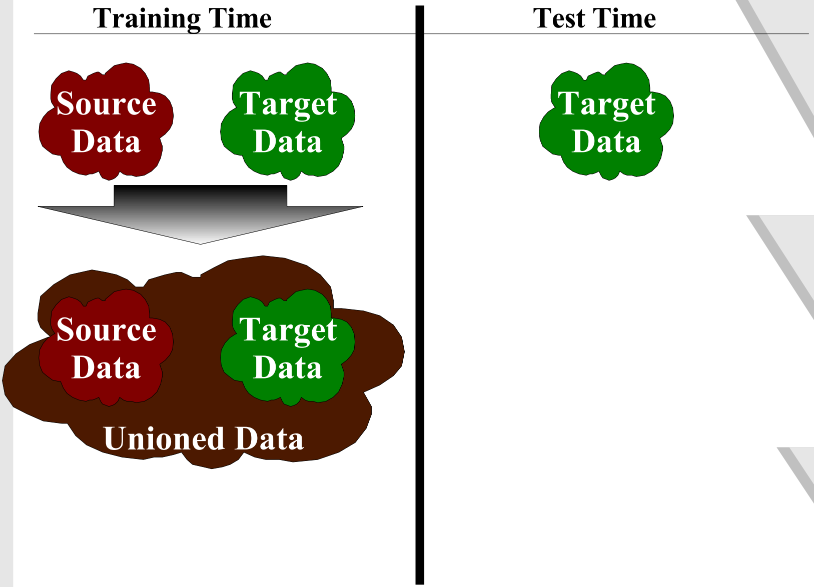
Obvious Approach 1: SrcOnly



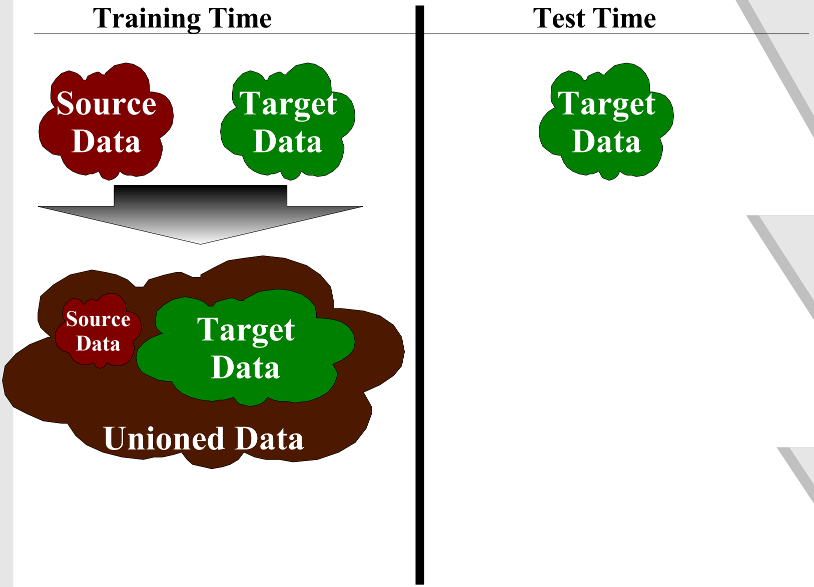
Obvious Approach 2: TgtOnly



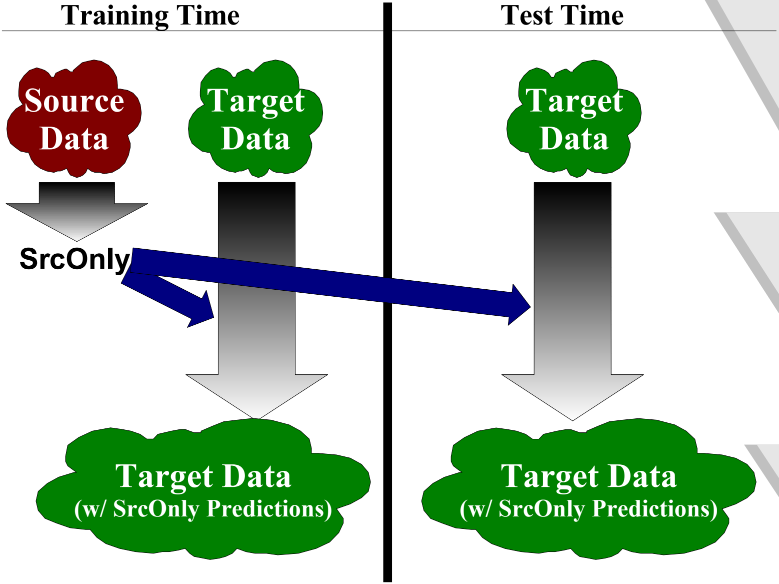
Obvious Approach 3: All



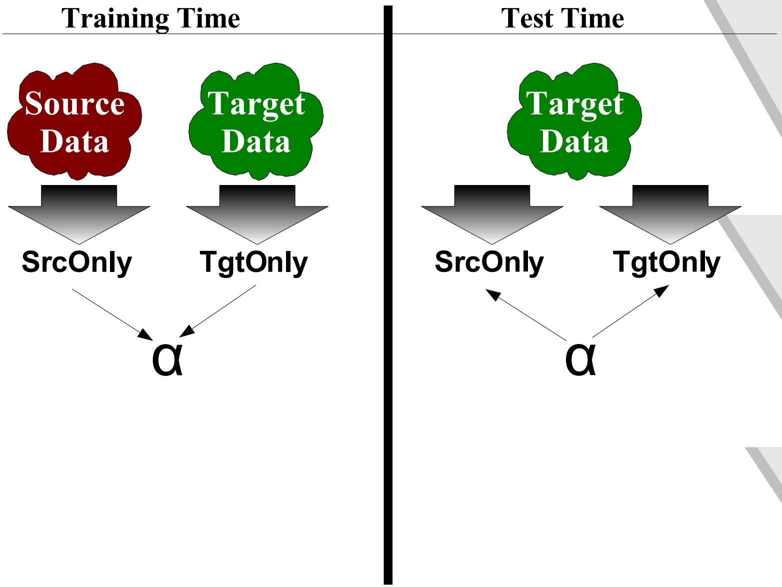
Obvious Approach 4: Weighted



Obvious Approach 5: Pred



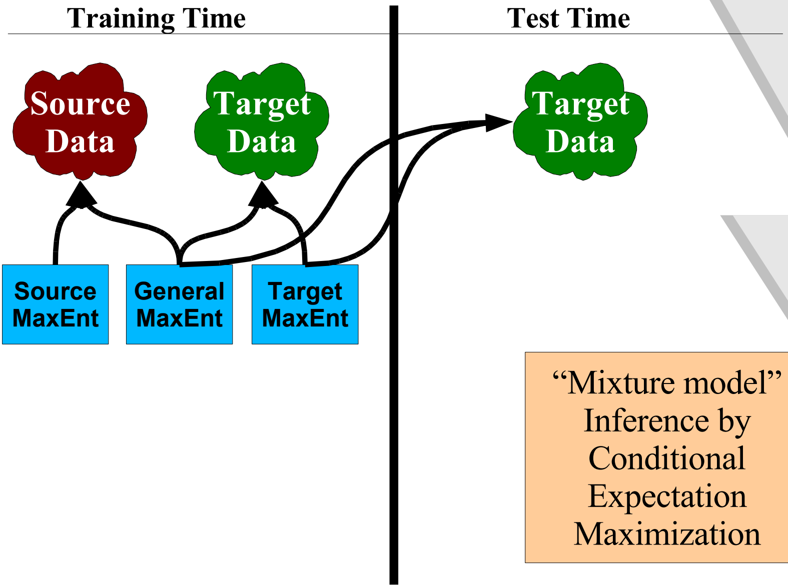
Obvious Approach 6: LinInt



Any other strategies?

- Next, feature augmentation
- (Daumé III 2007)

Prior Work – Daumé III and Marcu



"Mixture model"
Inference by
Conditional
Expectation
Maximization

“MONITOR” versus “THE”

News domain:
- “MONITOR” is a **verb**
- “THE” is a **determiner**

Technical domain:
- “MONITOR” is a **noun**
- “THE” is a **determiner**

Key Idea:
Share some features (“the”)
Don't share others (“monitor”)

(and let the *learner* decide which are which)

Feature Augmentation

We	monitor	the	traffic	The	monitor	is	heavy
N	V	D	N	D	N	V	R

Original Features

monitor	the	The	monitor
W:monitor	W:the	W:the	W:monitor
P:we	P:monitor	P:<s>	P:the
N:the	N:traffic	N:monitor	N:is
C:a+	C:a+	C:Aa+	C:a+

Augmented Features

monitor	the	The	monitor
SW:monitor	SW:the	TW:the	TW:monitor
SP:we	SP:monitor	TP:<s>	TP:the
SN:the	SN:traffic		
SC:a+	SC:a+		

Why should this work?

In feature-vector lingo:

$\Phi(x) \rightarrow \langle \Phi(x), \Phi(x), 0 \rangle$ (for source domain)

$\Phi(x) \rightarrow \langle \Phi(x), 0, \Phi(x) \rangle$ (for target domain)

Results – Error Rates

Task	Dom	SrcOnly	TgtOnly	Baseline	Prior	Augment
ACE-NER	bn	4.98	2.37	2.11 (pred)	2.06	**1.98**
	bc	4.54	4.07	3.53 (weight)	**3.47**	**3.47**
	nw	4.78	3.71	3.56 (pred)	3.68	**3.39**
	wl	2.45	2.45	**2.12** (all)	2.41	**2.12**
	un	3.67	2.46	2.10 (linint)	2.03	**1.91**
	cts	2.08	0.46	0.40 (all)	**0.34**	**0.32**
CoNLL	tgt	2.49	2.95	**1.75** (wgt/li)	1.89	**1.76**
PubMed	tgt	12.02	4.15	3.95 (linint)	3.99	**3.61**
CNN	tgt	10.29	3.82	3.44 (linint)	**3.35**	**3.37**
Tree bank-Chunk	wsj	6.63	4.35	4.30 (weight)	4.27	**4.11**
	swbd3	15.90	4.15	4.09 (linint)	3.60	**3.51**
	br-cf	5.16	6.27	**4.72** (linint)	5.22	5.15
	br-cg	4.32	5.36	**4.15** (all)	4.25	4.90
	br-ck	5.05	6.32	**5.01** (prd/li)	5.27	5.41
	br-cl	5.66	6.60	**5.39** (wgt/prd)	5.99	5.73
	br-cm	3.57	6.59	**3.11** (all)	4.08	4.89
	br-cn	4.60	5.56	**4.19** (prd/li)	4.48	4.42
	br-cp	4.82	5.62	**4.55** (wgt/prd/li)	4.87	4.78
	br-cr	5.78	9.13	**5.15** (linint)	6.71	6.30
Treebank-	brown	6.35	5.75	4.72 (linint)	4.72	**4.65**

Some analysis on feature extension: (Chang and Roth 2008)

- This has already been done before!
- The feature augmentation equals to a form of regularization

Some analysis on feature extension: (Chang and Roth 2008)

- This has already been done before!
- The feature augmentation equals to a form of regularization

Multi-task Learning: (Evgeniou and Pontil 2004)

- The function for task i: $w_i + v$
- The regularization becomes: $\frac{1}{2}\|v\|^2 + \frac{1}{2}\sum_{i=1}^{k}\|w_i\|^2$

Some analysis on feature extension: (Chang and Roth 2008)

- This has already been done before!
- The feature augmentation equals to a form of regularization

Multi-task Learning: (Evgeniou and Pontil 2004)

- The function for task i: $w_i + v$
- The regularization becomes: $\frac{1}{2}\|v\|^2 + \frac{1}{2}\sum_{i=1}^{k}\|w_i\|^2$

- The question: When does this work?

Why does feature augmentation work? (Chang and Roth 2008)

Intuition

- Assumption: only two tasks.
- The regularization becomes: $\|v\|^2 + \|w_s\|^2 + \|w_t\|^2$
- function for source: $v + w_s$, function for source: $v + w_t$

Why does feature augmentation work? (Chang and Roth 2008)

Intuition

- Assumption: only two tasks.
- The regularization becomes: $\|v\|^2 + \|w_s\|^2 + \|w_t\|^2$
- function for source: $v + w_s$, function for source: $v + w_t$
- Intuition 1: v is shared across the tasks, so we can use some examples better

Why does feature augmentation work? (Chang and Roth 2008)

Intuition

- Assumption: only two tasks.
- The regularization becomes: $\|v\|^2 + \|w_s\|^2 + \|w_t\|^2$
- function for source: $v + w_s$, function for source: $v + w_t$
- Intuition 1: v is shared across the tasks, so we can use some examples better
- Intuition 2: v is shared across the tasks, so if two tasks are more "similar", AUG works better!

Why does feature augmentation work? (Chang and Roth 2008)

Intuition

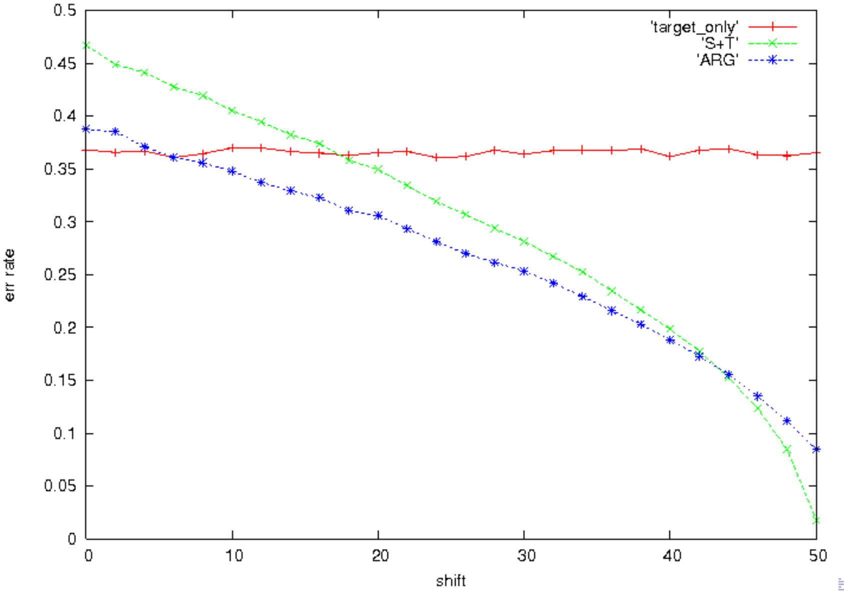
- Assumption: only two tasks.
- The regularization becomes: $\|v\|^2 + \|w_s\|^2 + \|w_t\|^2$
- function for source: $v + w_s$, function for source: $v + w_t$
- Intuition 1: v is shared across the tasks, so we can use some examples better
- Intuition 2: v is shared across the tasks, so if two tasks are more "similar", AUG works better!
- Is it?

Why does feature augmentation work? (Chang and Roth 2008)

Simple Analysis

- Assume u_s and u_t are the real separating lines
- $\cos(u_s, u_t)$ is small, AUG does not work (nothing to be shared)
- $\cos(u_s, u_t)$ close to 1, AUG does not work (single model is better)
- AUG only works in "good" range

Artificial experiments



Summary: Dealing with $P(y|x)$

- When you have labeled target instances, there are many training algorithms
  - Different ways to combine source labeled data and target labeled data
- We can apply multitask learning algorithms in this case
- Certain algorithms only work for limited situations

Outline

Different distributions: $P(X)$

- Solution: Instance Weighting
- (Bickel, Brüeckner, and Scheffer 2007), (Jiang and Zhai 2007)

Different distributions: $P(Y \mid X)$

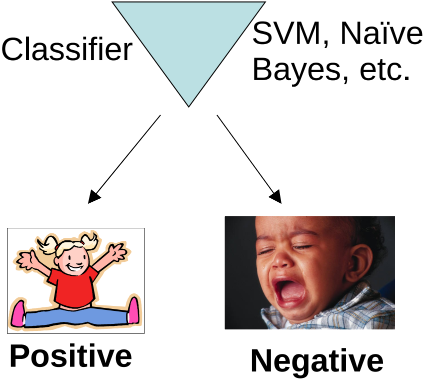
- Assume $P_s(Y \mid X)$ is close to $P_t(Y \mid X)$
- Solution: Regularization
- (Evgeniou and Pontil 2004), (Daumé III 2007)

Unknown Words

- Solution: Find the relations between old words and unseen words through auxiliary tasks
- (Ando and Zhang 2005), (Blitzer, McDonald, and Pereira 2006)

Sentiment Classification for Product Reviews

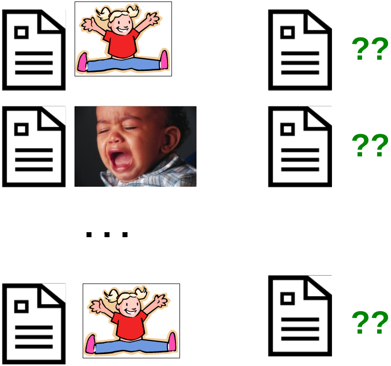
Product Review



Multiple Domains

books

kitchen appliances



books & kitchen appliances

Running with Scissors: A Memoir

Title: Horrible book, horrible.

This book was horrible. I **read half** of it, **suffering from a headache** the entire time, and eventually **i lit it on fire**. One less copy in the world...don't waste your money. I wish i had the time spent reading this book back so i could use it for better purposes. This book wasted my life

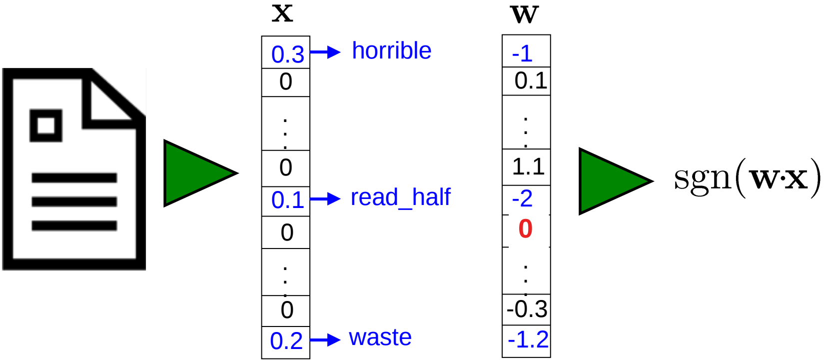
Avante Deep Fryer, Chrome & Black

Title: lid does not work well...

I love the way the Tefal deep fryer cooks, however, I am **returning** my second one due to a **defective** lid closure. The lid may close initially, but after a few uses it no longer stays closed. I **will not be purchasing** this one again.

Error increase: 13% => 26%

Features & Linear Models



Problem: If we've only trained on book reviews, then
w(defective) = 0

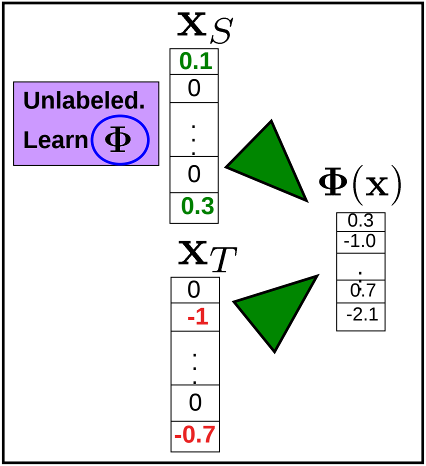
Structural Correspondence Learning (SCL)

- **Cut adaptation error by more than 40%**
- Use **unlabeled** data from the target domain
- Induce correspondences among different features
- **read-half, headache** ⟷ **defective, returned**
- Labeled data for **source** domain will help us build a good classifier for **target** domain

Maximum likelihood linear regression (MLLR) for speaker adaptation (Leggetter & Woodland, 1995)

SCL: 2-Step Learning Process

Step 1: Unlabeled – Learn correspondence mapping



Step 2: Labeled – Learn weight vector

Labeled. Learn $\mathbf{v}$

$$\Phi(\mathbf{x}) \blacktriangleright \operatorname{sgn}\left(\mathbf{v} \cdot \Phi(\mathbf{x})\right)$$

- **Φ should make the domains look as similar as possible**
- **But Φ should also allow us to classify well**

SCL: Making Domains Look Similar

Incorrect classification of kitchen review **defective** lid

Unlabeled **kitchen** contexts

- Do **not buy** the Shark portable steamer …. Trigger mechanism is **defective**.
- the very nice lady assured me that I must have a **defective** set …. What a **disappointment**!
- Maybe mine was **defective** …. The directions were **unclear**

Unlabeled **books** contexts

- The book is so **repetitive** that I found myself yelling …. I will definitely **not buy** another.
- A **disappointment** …. Ender was talked about for **<#> pages** altogether.
- it's **unclear** …. It's repetitive and **boring**

SCL: Pivot Features

Pivot Features

- Occur frequently in both domains
- Characterize the task we want to do
- Number in the hundreds or thousands
- Choose using labeled **source**, unlabeled **source** & **target** data

SCL: words & bigrams that occur frequently in both domains

book **one** **<num>** **so** **all** **very** **about** **they** **like** **good** **when**

SCL-MI: SCL but also based on mutual information with labels

a_must **a_wonderful** **loved_it** **weak** **don't_waste** **awful** **highly_recommended** **and_easy**

SCL Unlabeled Step: Pivot Predictors

Use **pivot features** to align other features

(1) The book is so **repetitive** that I found myself yelling …. I will definitely another.	**(2)** Do the Shark portable steamer …. Trigger mechanism is **defective**.

Binary problem: Does "**not buy**" appear here?

- **Mask** and predict pivot features using other features
- Train N **linear predictors**, one for each binary problem
- Each pivot predictor implicitly aligns non-pivot features from **source** & **target** domains

SCL: Dimensionality Reduction

$$\begin{bmatrix} | & & | & & | \\ \mathbf{w_1} & \dots & \mathbf{w_i} & \dots & \mathbf{w_N} \\ | & & | & & | \end{bmatrix}$$

- $\mathbf{W}^T\mathbf{x}$es N new features
- value of ith feature is the propensity to see **“not buy”** in the same document

- **We still want fewer new features (1000 is too many)**
- **Many pivot predictors give similar information**
  - **“horrible”, “terrible”, “awful”**
- **Compute SVD & use top left singular vectors** Φ

Latent Semantic Indexing (LSI), (Deerwester et al. 1990)

Latent Dirichlet Allocation (LDA), (Blei et al. 2003)

Back to Linear Classifiers

$$\mathbf{x} = \begin{bmatrix} 0.3 \\ 0 \\ \vdots \\ 0 \\ 0.1 \end{bmatrix}$$

Classifier $\operatorname{sgn}\left[\mathbf{w}\cdot\mathbf{x} + \mathbf{v}\cdot\mathbf{\Phi}^T\mathbf{x}\right]$

- **Source** training: Learn $\mathbf{w}$ & $\mathbf{v}$ together

$$\mathbf{\Phi}^T\mathbf{x} = \begin{bmatrix} 0.3 \\ -1.0 \\ \vdots \\ 0.7 \\ -2.1 \end{bmatrix}$$

- **Target** testing: First apply $\mathbf{\Phi}$, then apply $\mathbf{w}$ and $\mathbf{v}$

Inspirations for SCL

1. **Alternating Structural Optimization (ASO)**
   - **Ando & Zhang** (JMLR 2005)
   - Inducing structures for semi-supervised learning

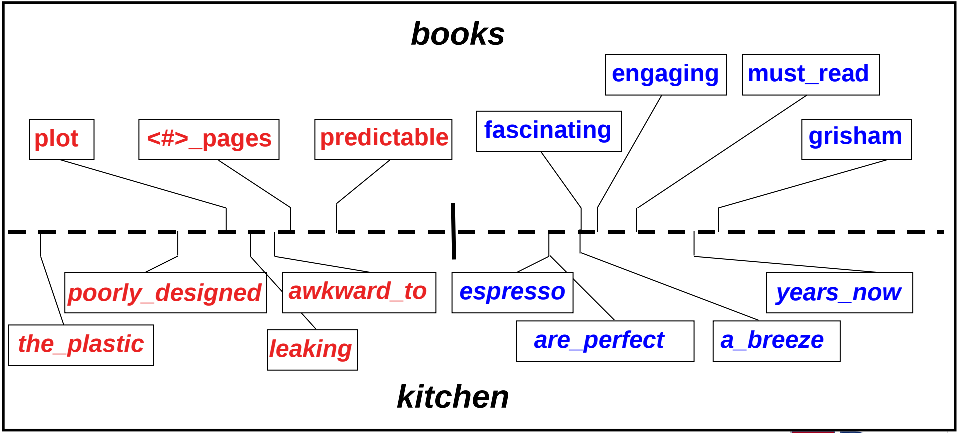
1. **Correspondence Dimensionality Reduction**
   - **Ham, Lee, & Saul** (AISTATS 2003)
   - Learn a low-dimensional representation from high-dimensional correspondences

Sentiment Classification Data

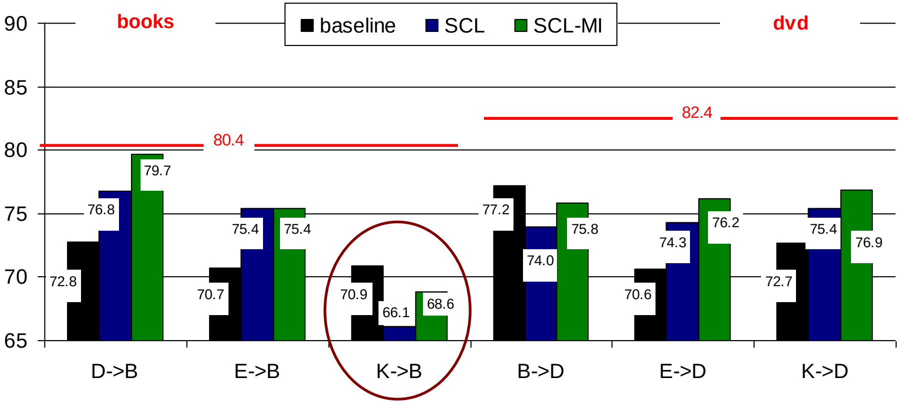
- **Product reviews from Amazon.com**
  - Books, DVDs, Kitchen Appliances, Electronics
  - 2000 labeled reviews from each domain
  - 3000 – 6000 unlabeled reviews
- **Binary classification problem**
  - Positive if 4 stars or more, negative if 2 or fewer
- **Features:** unigrams & bigrams
- **Pivots:** SCL & SCL-MI
- **At train time:** minimize Huberized hinge loss (Zhang, 2004)

Visualizing Φ (books & kitchen)

negative **vs.** **positive**



Empirical Results: books & DVDs



- **Sometimes SCL can cause increases in error**
- **With only unlabeled data, we misalign features**

Summary: Unknown words

- Unknown word: an important problem for domain adaptation
- Instance weighting and generalization can not solve this problem
- One solution: try to learn the relations between known words and unknown words

Summary: Domain Adaptation

Domain adaptation

- An important problem. We only have limited amount of labeled data and there are so many domains.
- Existing Solutions:
  - Instance Weighting
  - Regularization
  - Find the relationship between known words and unknown words

Summary: Domain Adaptation

Domain adaptation

- An important problem. We only have limited amount of labeled data and there are so many domains.
- Existing Solutions:
  - Instance Weighting
  - Regularization
  - Find the relationship between known words and unknown words

Many open problems

- Better techniques
- How to combine those techniques
- Multiple domains adaptation
- ...

Summary: Domain Adaptation

Domain adaptation

- An important problem. We only have limited amount of labeled data and there are so many domains.
- Existing Solutions:
  - Instance Weighting
  - Regularization
  - Find the relationship between known words and unknown words

Many open problems

- Better techniques
- How to combine those techniques
- Multiple domains adaptation
- ...

Thank you!!

Ando, R. K. and T. Zhang (2005).
A framework for learning predictive structures from multiple tasks and unlabeled data.
J. Mach. Learn. Res. 6, 1817–1853.

Bickel, S., M. Brüeckner, and T. Scheffer (2007).
Discriminative learning for differing training and test distributions.
In Z. Ghahramani (Ed.), *Proc. of the International Conference on Machine Learning (ICML)*, pp. 81–88.

Blitzer, J., R. McDonald, and F. Pereira (2006).
Domain adaptation with structural correspondence learning.
In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*.

Chang, M.-W. and D. Roth (2008, April).
Robust feature extension algorithms.
In *Learning Workshop, Snowbird*.

Daumé III, H. (2007, June).
Frustratingly easy domain adaptation.

In *Proc. of the Annual Meeting of the ACL*, pp. 256–263.

Evgeniou, T. and M. Pontil (2004).
Regularized multi–task learning.
In *Proc. of KDD*, pp. 109–117.

Jiang, J. and C. Zhai (2007, June).
Instance weighting for domain adaptation in nlp.
In *Proc. of the Annual Meeting of the ACL*, pp. 264–271.

Shimodaira, H. (2000).
Improving predictive inference un- der covariate shift by weighting the log-likelihood func- tion.
Journal of Statistical Planning and Inference 90.