

Semi-Supervised Learning

Admin Notes

Schedule

Today and next class (group meetings first hour, lecture immediately following)

- Today: Semi-supervised learning

Last four lectures (review, projects, and final)

- 6 Nov.: Sultan's review (1:30-3:15)
 Open session (3:15 – 4:15)
 (Project – Shakarian; Final – Sultan)
- 13, 20 Nov.: Project presentations
- 27 Nov.: No class
- 4 Dec.: Final exam (during normal class time)

Review From Last Time

- pLSA

Probabilistic Latent Semantic Analysis

Think: Topic ~ Factor

- Now let us have K topics as well:

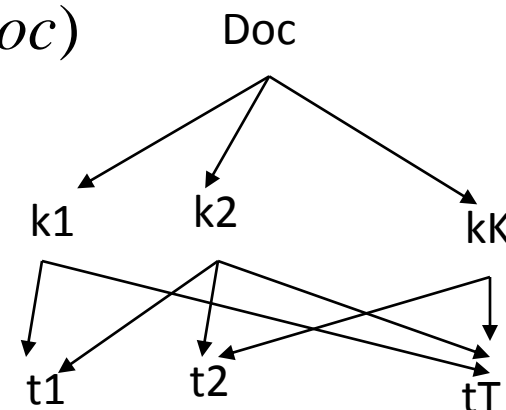
$$P(term_t | doc) = \sum_{k=1}^K P(term_t | topic_k) P(topic_k | doc)$$

The same, written using shorthands :

$$P(t | doc) = \sum_{k=1}^K P(t | k) P(k | doc)$$

So by replacing this, for any doc in the collection ,

$$P(doc) = \prod_{t=1}^T \left\{ \sum_{k=1}^K P(t | k) P(k | doc) \right\}^{X(t, doc)}$$



Which are the parameters of this model?

Probabilistic Latent Semantic Analysis

- The parameters of this model are:
 - $P(t|k)$
 - $P(k|doc)$
- It is possible to derive the equations for computing these parameters by Maximum Likelihood.
- If we do so, what do we get?
 - $P(t|k)$ for all t and k , is a term by topic matrix
(gives which terms make up a topic)
 - $P(k|doc)$ for all k and doc , is a topic by document matrix
(gives which topics are in a document)

Review From Last Time

- LDA

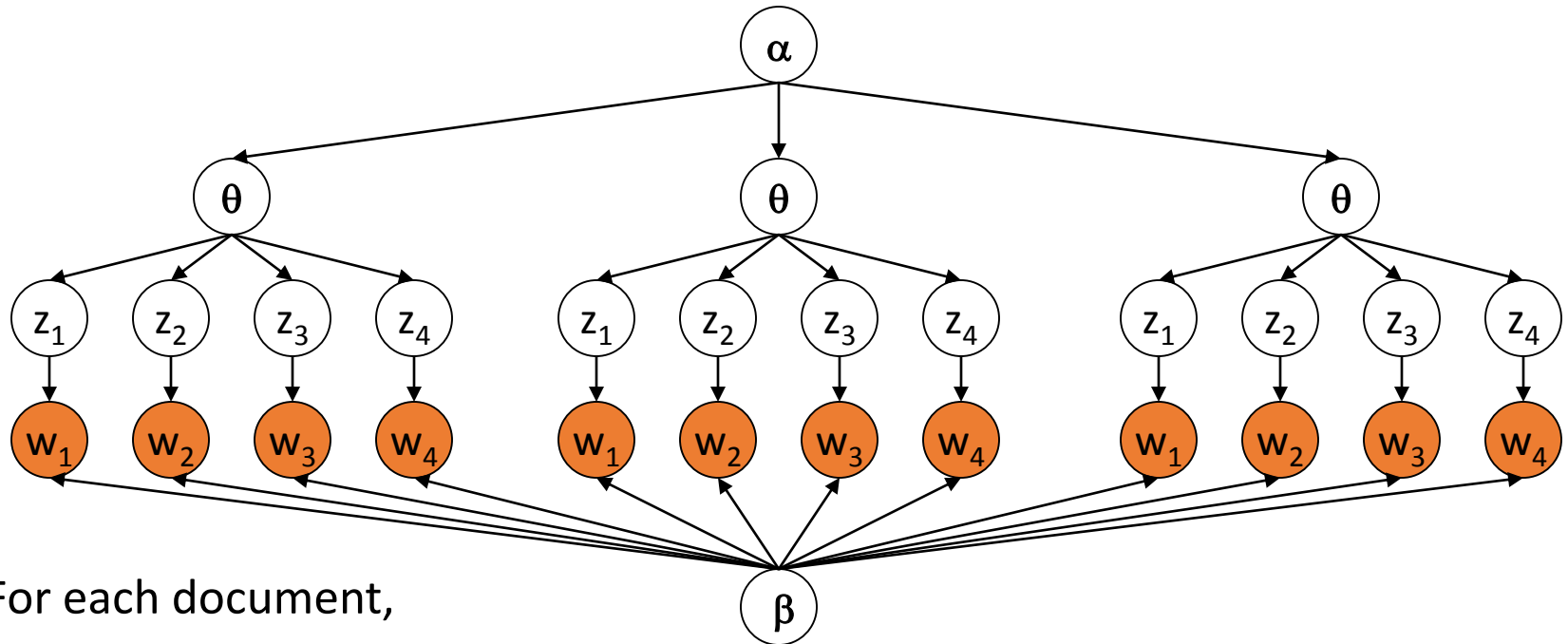
Motivations for LDA

- In pLSI, the observed variable d is an index into some training set. There is no natural way for the model to handle previously unseen documents.
- The number of parameters for pLSI grows linearly with M (the number of documents in the training set).
- We would like to be Bayesian about our topic mixture proportions.

LDA assumes the following generative process for each document \mathbf{w} in a corpus D :

1. Choose $N \sim \text{Poisson}(\xi)$.
2. Choose $\theta \sim \text{Dir}(\alpha)$.
3. For each of the N words w_n :
 - (a) Choose a topic $z_n \sim \text{Multinomial}(\theta)$.
 - (b) Choose a word w_n from $p(w_n | z_n, \beta)$, a multinomial probability conditioned on the topic z_n .

The LDA Model



- For each document,
- Choose $\theta \sim \text{Dirichlet}(\alpha)$
- For each of the N words w_n :
 - Choose a topic $z_n \sim \text{Multinomial}(\theta)$
 - Choose a word w_n from $p(w_n | z_n, \beta)$, a multinomial probability conditioned on the topic z_n .

Semi Supervised Learning

- Bing Liu's slides

Outline

- Fully supervised learning (traditional classification)
- Partially (semi-) supervised learning (or classification)
 - Learning with a small set of labeled examples and a large set of unlabeled examples (**LU learning**)
 - Learning with positive and unlabeled examples (no labeled negative examples) (**PU learning**).

Learning from a small labeled set and a large unlabeled set

LU learning

Unlabeled Data

- One of the bottlenecks of classification is the labeling of a large set of examples (data records or text documents).
 - Often done manually
 - Time consuming
- Can we label only a small number of examples and make use of a large number of unlabeled examples to learn?
- Possible in many cases.

Why unlabeled data are useful?

- Unlabeled data are usually plentiful, labeled data are expensive.
- Unlabeled data provide information about the joint probability distribution over words and collocations (in texts).
- We will use text classification to study this problem.

Labeled Data

Unlabeled Data

Documents containing “homework”
tend to belong to the positive class

DocNo: k ClassLabel: Positive

.....

.....homework....

...

DocNo: m ClassLabel: Positive

.....

.....homework....

...

DocNo: n ClassLabel: Positive

.....

.....homework....

...

DocNo: x (ClassLabel: Positive)

.....

.....homework....

...lecture....

DocNo: y (ClassLabel: Positive)

.....lecture.....

.....homework....

...

DocNo: z ClassLabel: Positive

.....

.....homework....

.....lecture....

How to use unlabeled data

- One way is to use the EM algorithm
 - **EM: Expectation Maximization**
- The EM algorithm is a popular iterative algorithm for maximum likelihood estimation in problems with missing data.
- The EM algorithm consists of two steps,
 - **Expectation step**, i.e., filling in the missing data
 - **Maximization step** – calculate a new maximum *a posteriori* estimate for the parameters.

Incorporating unlabeled Data with EM

(Nigam et al, 2000)

- Basic EM
- Augmented EM with weighted unlabeled data
- Augmented EM with multiple mixture components per class

Algorithm Outline

1. Train a classifier with only the labeled documents.
2. Use it to probabilistically classify the unlabeled documents.
3. Use ALL the documents to train a new classifier.
4. Iterate steps 2 and 3 to convergence.

Basic Algorithm

Algorithm EM(L, U)

- 1 Learn an initial naïve Bayesian classifier f from only the labeled set L (using Equations (27) and (28) in Chap. 3);
 - 2 **repeat**
 - // E-Step
 - 3 **for** each example d_i in U **do**
 - 4 Using the current classifier f to compute $\Pr(c_j|d_i)$ (using Equation (29) in Chap. 3).
 - 5 **end**
 - // M-Step
 - 6 learn a new naïve Bayesian classifier f from $L \cup U$ by computing $\Pr(c_j)$ and $\Pr(w_i|c_j)$ (using Equations (27) and (28) in Chap. 3).
 - 7 **until** the classifier parameters stabilize
- Return the classifier f from the last iteration.

Fig. 5.1. The EM algorithm with naïve Bayesian classification

Basic EM: E Step & M Step

$$\Pr(c_j | d_i; \hat{\Theta}) = \frac{\Pr(c_j | \hat{\Theta}) \Pr(d_i | c_j; \hat{\Theta})}{\Pr(d_i | \hat{\Theta})} \quad (29)$$

E Step:

$$= \frac{\Pr(c_j | \hat{\Theta}) \prod_{k=1}^{|d_i|} \Pr(w_{d_i,k} | c_j; \hat{\Theta})}{\sum_{r=1}^{|C|} \Pr(c_r | \hat{\Theta}) \prod_{k=1}^{|d_i|} \Pr(w_{d_i,k} | c_r; \hat{\Theta})},$$

M Step:

$$\Pr(w_t | c_j; \hat{\Theta}) = \frac{\lambda + \sum_{i=1}^{|D|} N_{ti} \Pr(c_j | d_i)}{\lambda |V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N_{si} \Pr(c_j | d_i)}. \quad (27)$$

$$\Pr(c_j | \hat{\Theta}) = \frac{\sum_{i=1}^{|D|} \Pr(c_j | d_i)}{|D|}. \quad (28)$$

The problem

- It has been shown that the EM algorithm in Fig. 5.1 works well if the
 - The two mixture model assumptions for a particular data set are true.
- The two mixture model assumptions, however, can cause major problems when they do not hold. In many real-life situations, they may be violated.
- It is often the case that a class (or topic) contains a number of sub-classes (or sub-topics).
 - For example, the class Sports may contain documents about different sub-classes of sports, Baseball, Basketball, Tennis, and Softball.
- Some methods to deal with the problem.

Weighting the influence of unlabeled examples by factor μ

New M step:

$$\Pr(w_t | c_j) = \frac{\lambda + \sum_{i=1}^{|D|} \Lambda(i) N_{ti} \Pr(c_j | d_i)}{\lambda |V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} \Lambda(i) N_{ti} \Pr(c_j | d_i)}, \quad (1)$$

where

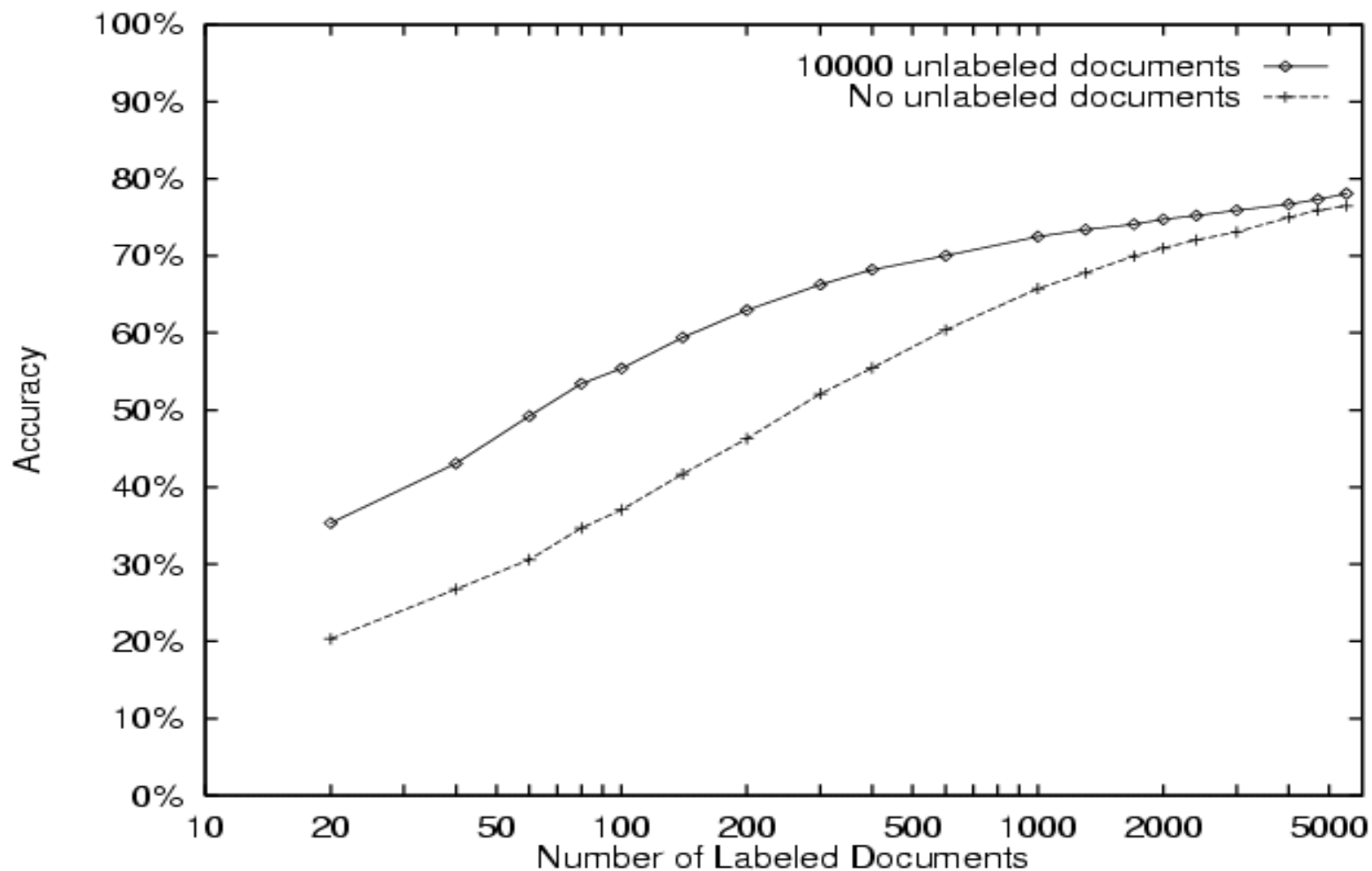
$$\Lambda(i) = \begin{cases} \mu & \text{if } d_i \in U \\ 1 & \text{if } d_i \in L. \end{cases} \quad (2)$$

The prior probability also needs to be weighted.

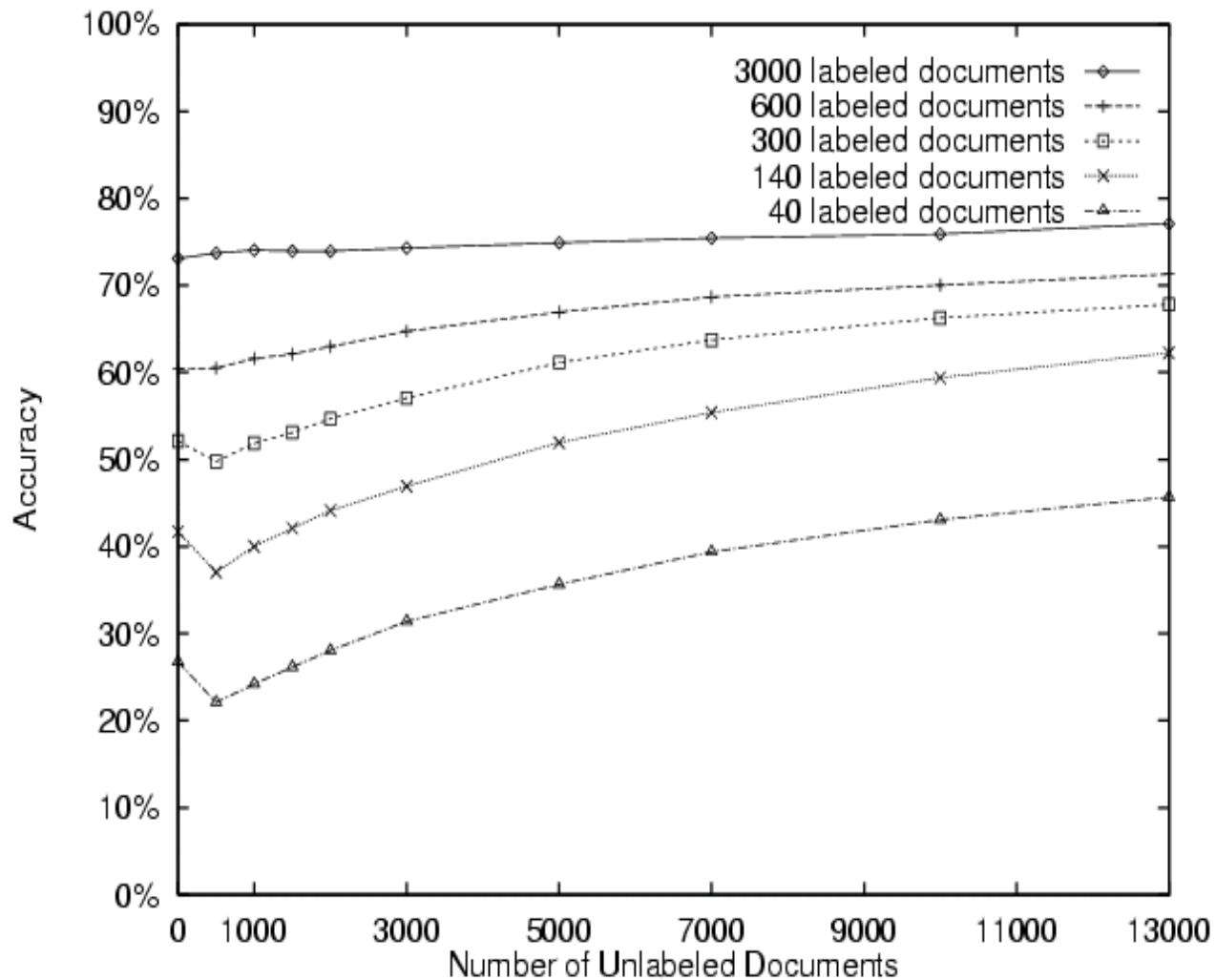
Experimental Evaluation

- Newsgroup postings
 - 20 newsgroups, 1000/group
- Web page classification
 - student, faculty, course, project
 - 4199 web pages
- Reuters newswire articles
 - 12,902 articles
 - 10 main topic categories

20 Newsgroups



20 Newsgroups



Another approach: Co-training

- Again, learning with a small labeled set and a large unlabeled set.
- The attributes describing each example or instance can be partitioned into two subsets. Each of them is sufficient for learning the target function.
 - E.g., hyperlinks and page contents in Web page classification.
- Two classifiers can be learned from the same data.

Co-training Algorithm

[Blum and Mitchell, 1998]

Given: labeled data L ,

unlabeled data U

Loop:

Train h_1 (e.g., hyperlink classifier) using L

Train h_2 (e.g., page classifier) using L

Allow h_1 to label p positive, n negative examples from U

Allow h_2 to label p positive, n negative examples from U

Add these most confident self-labeled examples to L

Co-training: Experimental Results

- begin with 12 labeled web pages (academic course)
- provide 1,000 additional unlabeled web pages
- average error: learning from labeled data 11.1%;
- average error: co-training 5.0%

	Page-base classifier	Link-based classifier	Combined classifier
Supervised training	12.9	12.4	11.1
Co-training	6.2	11.6	5.0

When the generative model is not suitable

- **Multiple Mixture Components per Class** (M-EM). E.g., a class --- a number of sub-topics or clusters.
- Results of an example using 20 newsgroup data
 - 40 labeled; 2360 unlabeled; 1600 test
 - Accuracy
 - NB 68%
 - EM 59.6%
- Solutions
 - **M-EM** (Nigam et al, 2000): Cross-validation on the training data to determine the number of components.
 - **Partitioned-EM** (Cong, et al, 2004): using hierarchical clustering. It does significantly better than M-EM.

Summary

- Using unlabeled data can improve the accuracy of classifier when the data fits the generative model.
- Partitioned EM and the EM classifier based on multiple mixture components model (M-EM) are more suitable for real data when multiple mixture components are in one class.
- Co-training is another effective technique when redundantly sufficient features are available.

Learning from Positive and Unlabeled Examples

PU learning

Learning from Positive & Unlabeled data

- **Positive examples**: One has a set of examples of a class P , and
- **Unlabeled set**: also has a set U of unlabeled (or mixed) examples with instances from P and also not from P (*negative examples*).
- **Build a classifier**: Build a classifier to classify the examples in U and/or future (test) data.
- **Key feature of the problem**: no labeled negative training data.
- We call this problem, **PU-learning**.

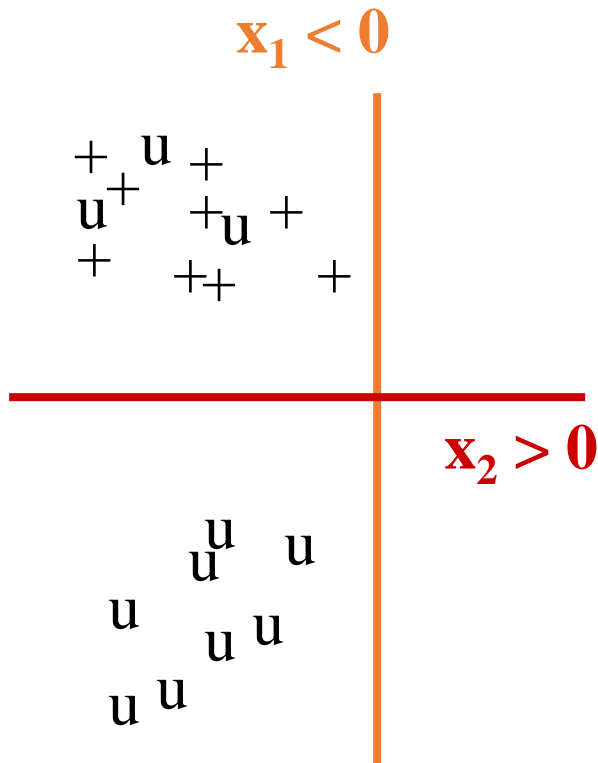
Applications of the problem

- With the growing volume of online texts available through the Web and digital libraries, one often wants to find those documents that are related to **one's work** or **one's interest**.
- **For example, given a ICML proceedings,**
 - **find all machine learning papers from AAAI, IJCAI, KDD**
 - **No labeling of negative examples from each of these collections.**
- Similarly, given one's bookmarks (positive documents), identify those documents that are of interest to him/her from Web sources.

Direct Marketing

- Company has database with details of its customer – **positive examples**, but no information on those who are not their customers, i.e., **no negative examples**.
- Want to find people who are similar to their customers for marketing
- Buy a database consisting of details of people, some of whom may be potential customers – **unlabeled examples**.

Are Unlabeled Examples Helpful?



- Function known to be either $x_1 < 0$ or $x_2 > 0$
- Which one is it?

“Not learnable” with only positive examples. However, addition of unlabeled examples makes it learnable.

Theoretical foundations

- (X, Y) : X - input vector, $Y \in \{1, -1\}$ - class label.
- f : classification function
- We rewrite the probability of error

$$\Pr[f(X) \neq Y] = \Pr[f(X) = 1 \text{ and } Y = -1] + \Pr[f(X) = -1 \text{ and } Y = 1] \quad (1)$$

We have $\Pr[f(X) = 1 \text{ and } Y = -1]$

$$= \Pr[f(X) = 1] - \Pr[f(X) = 1 \text{ and } Y = 1]$$

$$= \Pr[f(X) = 1] - (\Pr[Y = 1] - \Pr[f(X) = -1 \text{ and } Y = 1]).$$

Plug this into (1), we obtain

$$\Pr[f(X) \neq Y] = \Pr[f(X) = 1] - \Pr[Y = 1] + 2\Pr[f(X) = -1 | Y = 1]\Pr[Y = 1] \quad (2)$$

Theoretical foundations (cont)

- $\Pr[f(X) \neq Y] = \Pr[f(X) = 1] - \Pr[Y = 1] + 2\Pr[f(X) = -1 | Y = 1] \Pr[Y = 1]$ (2)
 - Note that $\Pr[Y = 1]$ is constant.
 - If we can hold $\Pr[f(X) = -1 | Y = 1]$ small, then learning is approximately the same as minimizing $\Pr[f(X) = 1]$.
 - Holding $\Pr[f(X) = -1 | Y = 1]$ small while minimizing $\Pr[f(X) = 1]$ is approximately the same as
 - minimizing $\Pr_u[f(X) = 1]$
 - while holding $\Pr_p[f(X) = 1] \geq r$ (where r is recall $\Pr[f(X)=1 | Y=1]$) which is the same as $(\Pr_p[f(X) = -1] \leq 1 - r)$
- if the set of positive examples P and the set of unlabeled examples U are large enough.
- **Theorem 1** and **Theorem 2** in [Liu et al 2002] state these formally in the noiseless case and in the noisy case.

Put it simply

- A constrained optimization problem.
- A reasonably good generalization (learning) result can be achieved
 - If the algorithm tries to minimize the number of unlabeled examples labeled as positive
 - subject to the constraint that the fraction of errors on the positive examples is no more than $1-r$.

An illustration

- Assume a linear classifier. Line 3 is the best solution.

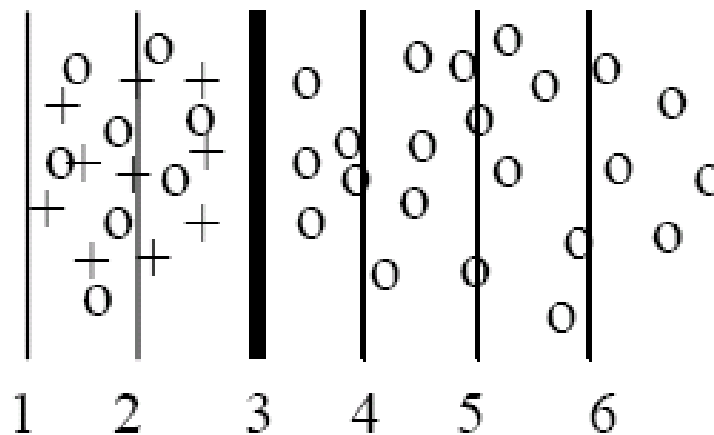



Fig. 5.6. An illustration of the constrained optimization problem

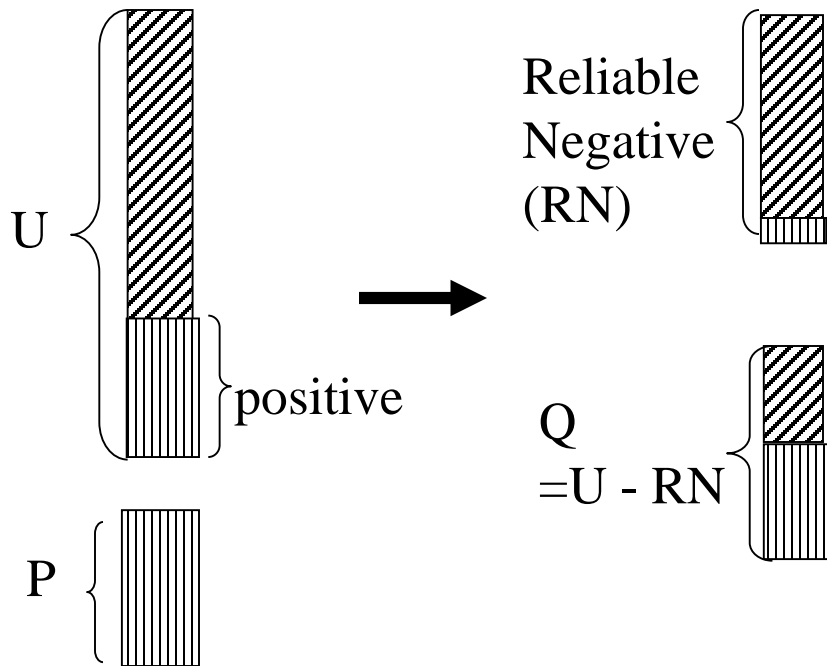
Existing 2-step strategy

- Step 1: Identifying a set of reliable negative documents from the unlabeled set.
 - S-EM [Liu et al, 2002] uses a Spy technique,
 - PEBL [Yu et al, 2002] uses a 1-DNF technique
 - Roc-SVM [Li & Liu, 2003] uses the Rocchio algorithm.
 - ...
- Step 2: Building a sequence of classifiers by iteratively applying a classification algorithm and then selecting a good classifier.
 - S-EM uses the Expectation Maximization (EM) algorithm, with an error based classifier selection mechanism
 - PEBL uses SVM, and gives the classifier at convergence. I.e., no classifier selection.
 - Roc-SVM uses SVM with a heuristic method for selecting the final classifier.

Step 1

Step 2

 positive  negative



Using P, RN and Q
to build the final
classifier iteratively

or

Using only P and RN
to build a classifier

Step 1: The Spy technique

- Sample a certain % of positive examples and put them into unlabeled set to act as “spies”.
- Run a classification algorithm assuming all unlabeled examples are negative,
 - we will know the behavior of those actual positive examples in the unlabeled set through the “spies”.
- We can then extract reliable negative examples from the unlabeled set more accurately.

Step 1: Other methods

- 1-DNF method:
 - Find the set of words W that occur in the positive documents more frequently than in the unlabeled set.
 - Extract those documents from unlabeled set that do not contain any word in W . These documents form the **reliable negative documents**.
- Rocchio method from information retrieval.
- Naïve Bayesian method.

Step 2: Running EM or SVM iteratively

(1) Running a classification algorithm iteratively

- Run EM using P , R_N and Q until it converges, **or**
- Run SVM iteratively using P , R_N and Q until this no document from Q can be classified as negative. R_N and Q are updated in each iteration, or
- ...

(2) Classifier selection.

Do they follow the theory?

- Yes, heuristic methods because
 - Step 1 tries to find some initial reliable negative examples from the unlabeled set.
 - Step 2 tried to identify more and more negative examples iteratively.
- The two steps together form an iterative strategy of increasing the number of unlabeled examples that are classified as negative while maintaining the positive examples correctly classified.

Can SVM be applied directly?

- Can we use SVM to directly deal with the problem of learning with positive and unlabeled examples, without using two steps?
- Yes, with a little re-formulation.

Support Vector Machines

- Support vector machines (SVM) are linear functions of the form $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$, where \mathbf{w} is the weight vector and \mathbf{x} is the input vector.
- Let the set of training examples be $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$, where \mathbf{x}_i is an input vector and y_i is its class label, $y_i \in \{1, -1\}$.
- To find the linear function:

Minimize:
$$\frac{1}{2} \mathbf{w}^T \mathbf{w}$$

Subject to:
$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, n$$

Soft margin SVM

- To deal with cases where there may be no separating hyperplane due to noisy labels of both positive and negative training examples, the soft margin SVM is proposed:

Minimize:
$$\frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^n \xi_i$$

Subject to:
$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \dots, n$$

where $C \geq 0$ is a parameter that controls the amount of training errors allowed.

Biased SVM (noiseless case)

- Assume that the first $k-1$ examples are positive examples (labeled 1), while the rest are unlabeled examples, which we label negative (-1).

$$\text{Minimize: } \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=k}^n \xi_i$$

$$\text{Subject to: } \mathbf{w}^T \mathbf{x}_i + b \geq 1, \quad i = 1, 2, \dots, k-1$$

$$-1(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = k, k+1, \dots, n$$

$$\xi_i \geq 0, \quad i = k, k+1, \dots, n$$

Biased SVM (noisy case)

- If we also allow positive set to have some noisy negative examples, then we have:

$$\text{Minimize:} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C_+ \sum_{i=1}^{k-1} \xi_i + C_- \sum_{i=k}^n \xi_i$$

$$\text{Subject to:} \quad y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \dots, n$$

$$\xi_i \geq 0, \quad i = 1, 2, \dots, n.$$

- This turns out to be the same as the asymmetric cost SVM for dealing with unbalanced data. Of course, we have a different motivation.

Estimating performance

- We need to estimate the performance in order to select the parameters.
- Since learning from positive and negative examples often arise in retrieval situations, we use F score as the classification performance measure $F = 2pr / (p+r)$ (p : precision, r : recall).
- To get a high F score, both precision and recall have to be high.
- However, without labeled negative examples, we do not know how to estimate the F score.

A performance criterion

- Performance criteria $pr/\Pr[Y=1]$: It can be estimated directly from the validation set as $r^2/\Pr[f(X) = 1]$
 - Recall $r = \Pr[f(X)=1 \mid Y=1]$
 - Precision $p = \Pr[Y=1 \mid f(X)=1]$

To see this

$$\Pr[f(X)=1 \mid Y=1] \Pr[Y=1] = \Pr[Y=1 \mid f(X)=1] \Pr[f(X)=1]$$

$$\Leftrightarrow \frac{r}{\Pr[f(X) = 1]} = \frac{p}{\Pr[Y = 1]}$$

//both side times r

- Behavior similar to the F-score ($= 2pr / (p+r)$)

A performance criterion (cont ...)

- $r^2 / \Pr[f(X) = 1]$
- r can be estimated from positive examples in the validation set.
- $\Pr[f(X) = 1]$ can be obtained using the full validation set.
- This criterion actually reflects the theory very well.

Empirical Evaluation

- **Two-step strategy:** We implemented a benchmark system, called **LPU**, which is available at <http://www.cs.uic.edu/~liub/LPU/LPU-download.html>
 - Step 1:
 - **Spy**
 - **1-DNF**
 - **Rocchio**
 - **Naïve Bayesian (NB)**
 - Step 2:
 - **EM with classifier selection**
 - **SVM: Run SVM once.**
 - **SVM-I: Run SVM iteratively and give converged classifier.**
 - **SVM-IS: Run SVM iteratively with classifier selection**
- **Biased-SVM** (we used **SVMlight** package)

Results of Biased SVM

Table 3: Average F scores on the two collections

	γ	Average F score of Biased-SVM	Previous best F score
Reuters	0.3	0.785	0.78
	0.7	0.856	0.845
20Newsgroup	0.3	0.742	0.689
	0.7	0.805	0.774

Summary

- Gave an overview of **the theory** on learning with positive and unlabeled examples.
- Described the existing **two-step strategy** for learning.
- Presented an more principled approach to solve the problem based on a **biased SVM formulation**.
- Presented a performance measure **$pr/P(Y=1)$** that can be estimated from data.
- Experimental results using text classification show the superior classification power of Biased-SVM.
- Some more experimental work are being performed to compare Biased-SVM with **weighted logistic regression** method [Lee & Liu 2003].