Mine the Easy, Classify the Hard: A Semi-Supervised Approach to Automatic Sentiment Classification

Sajib Dasgupta and Vincent Ng
Human Language Technology Research Institute
University of Texas at Dallas

Sentiment Classification

- Task
  - Classify a review as "thumbs up" or "thumbs down"

  - **Can we build a high-performance sentiment classification system with limited labeled data**

?

Sentiment Classification is Tough!

- **Sentimental ambiguity**
  - A review may contain both positive and negative words!

the sound from my system did seem to be a little better
(the CD's were not skipping as much). but the bottom line is it
didn't fix the problem as the CDs are still skipping noticeably,
although not as bad as before. ...

- **A machine is more likely to label it positive.**

Sentiment Classification is Tough!

- **Sentimental ambiguity**
  - A review may contain both positive and negative words!

> the sound from my system did seem to be a little better (the CD's were not skipping as much). but the bottom line is it didn't fix the problem as the CDs are still skipping noticeably, although not as bad as before. ...

It's the bottom line that determines sentiment!

- **But ... it's actually a negative review!**

Semi-Supervised Sentiment Classification

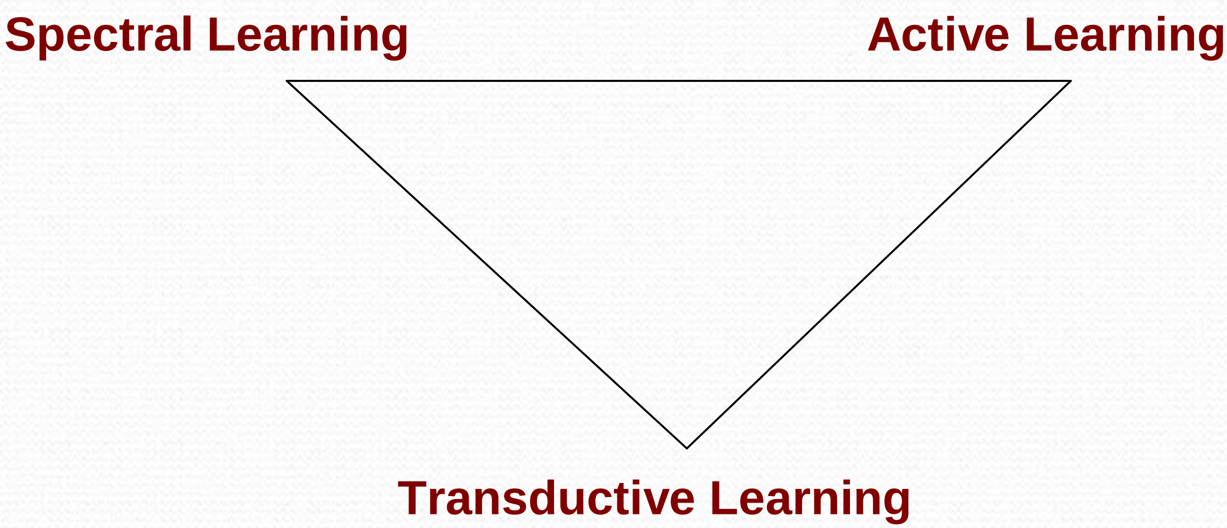
How to handle ambiguity

?

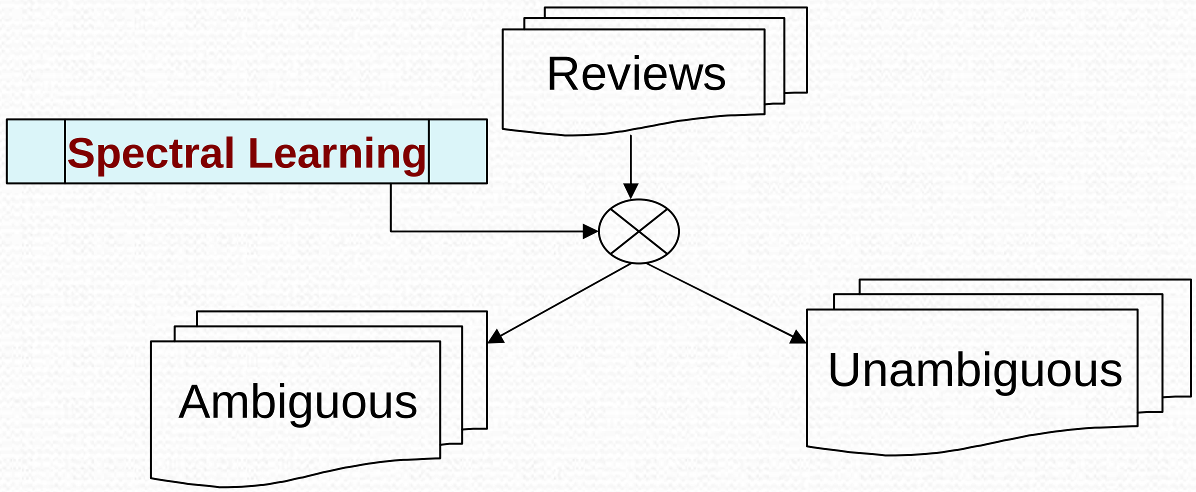
Semi-Supervised Sentiment Classification

- A new semi-supervised classification approach
  - **"Mine the Easy, Classify the Hard"**
    - exploits the fact that reviews are sentimentally ambiguous

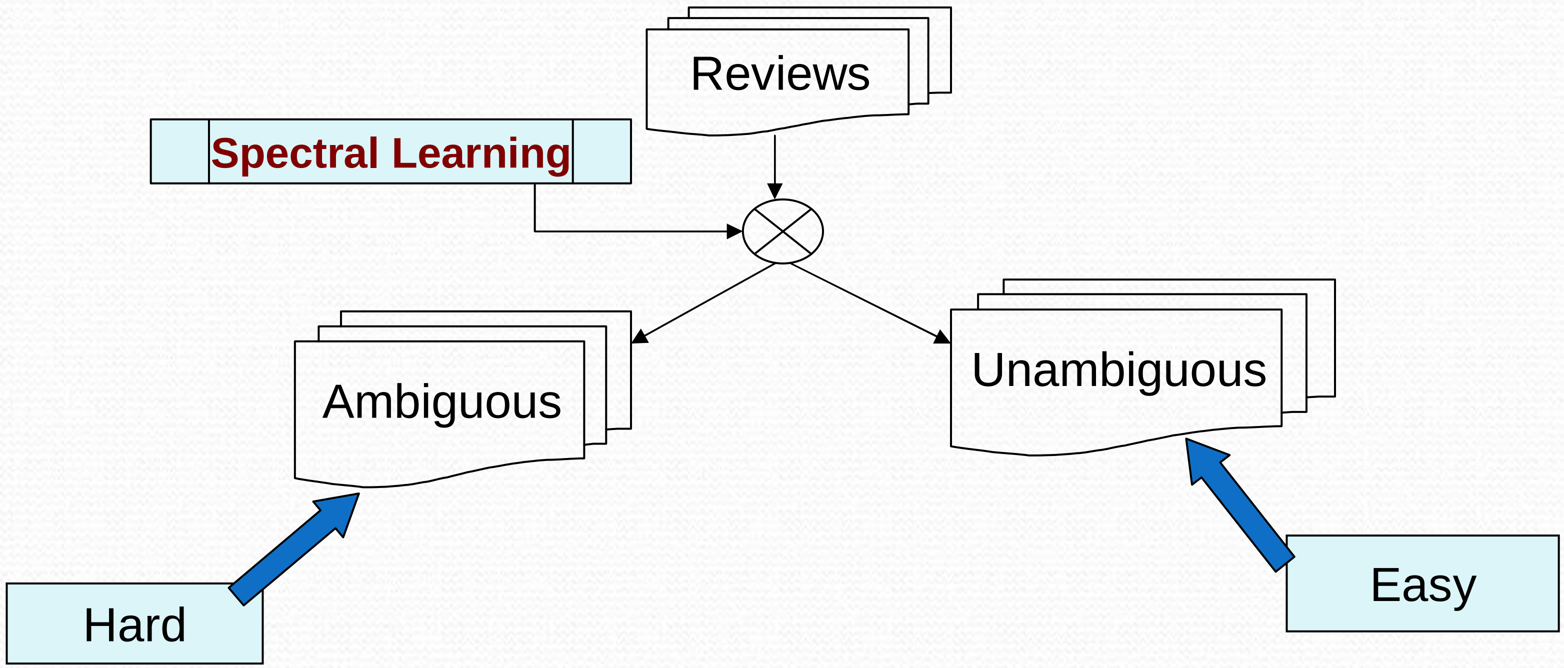
Our Semi-supervised Approach



Main Idea

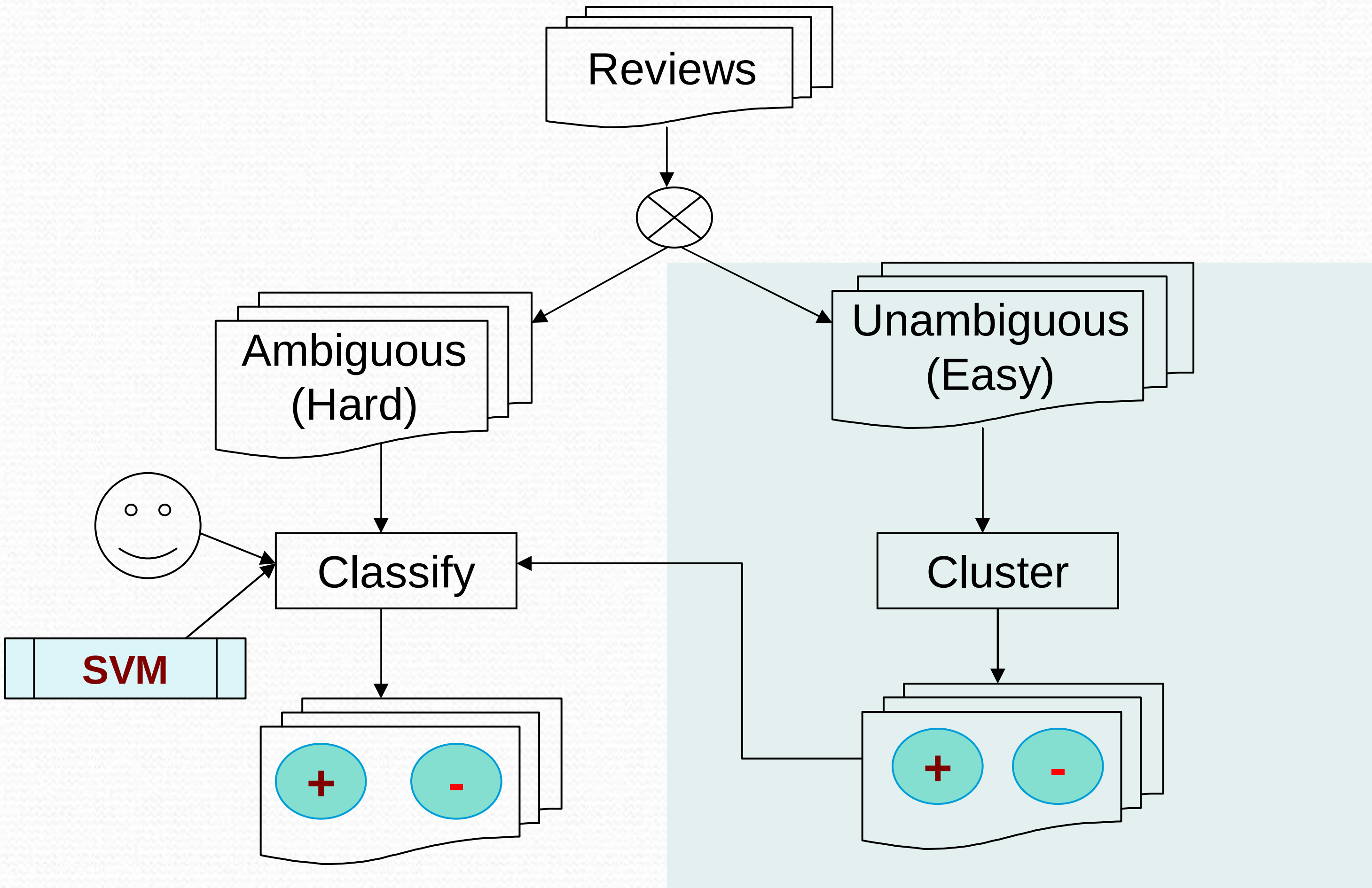


Main Idea



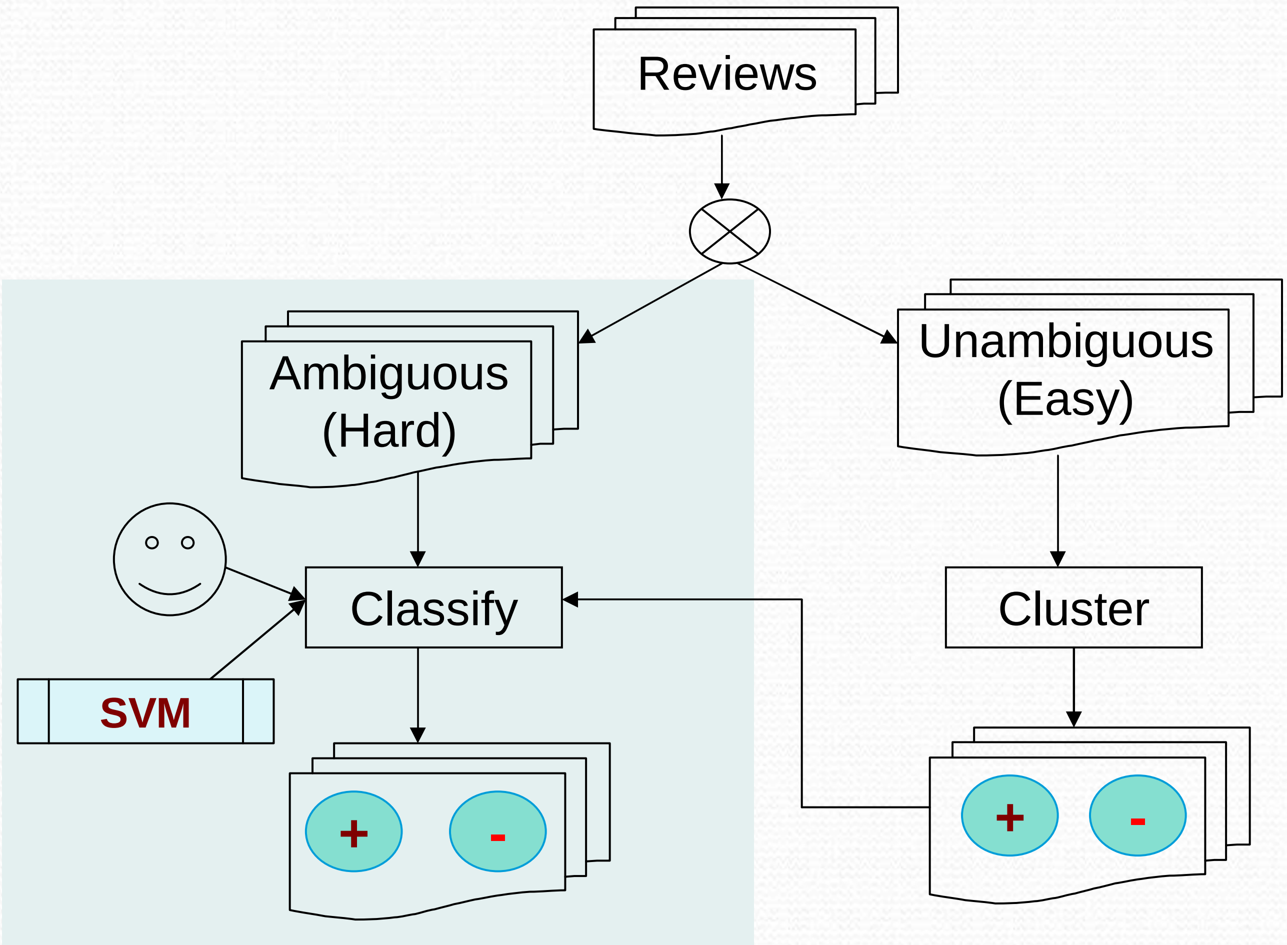
Ambiguous reviews need to be handled with care

Main Idea

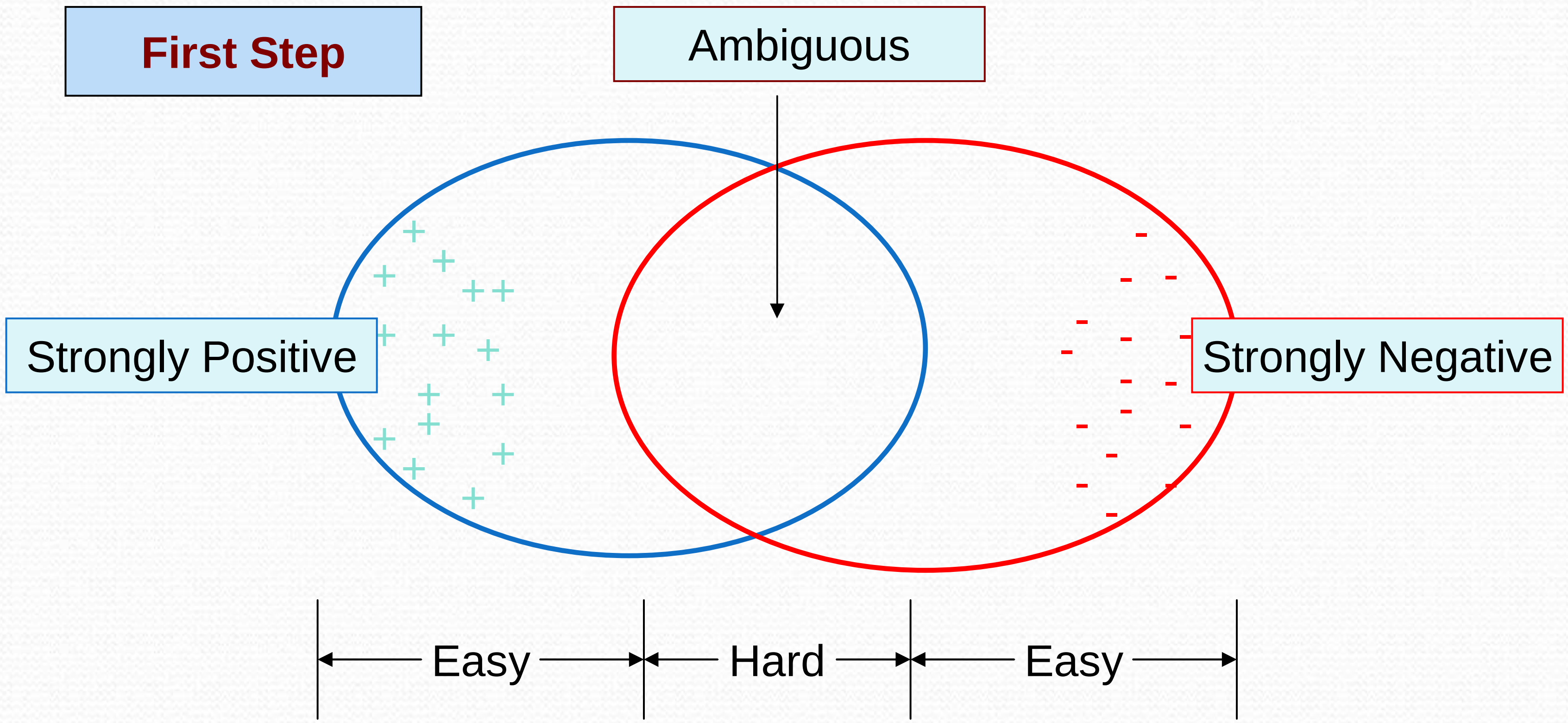


Reviews
Ambiguous
(Hard)
Unambiguous
(Easy)
Classify
Cluster
SVM
+
-
+
-

Main Idea

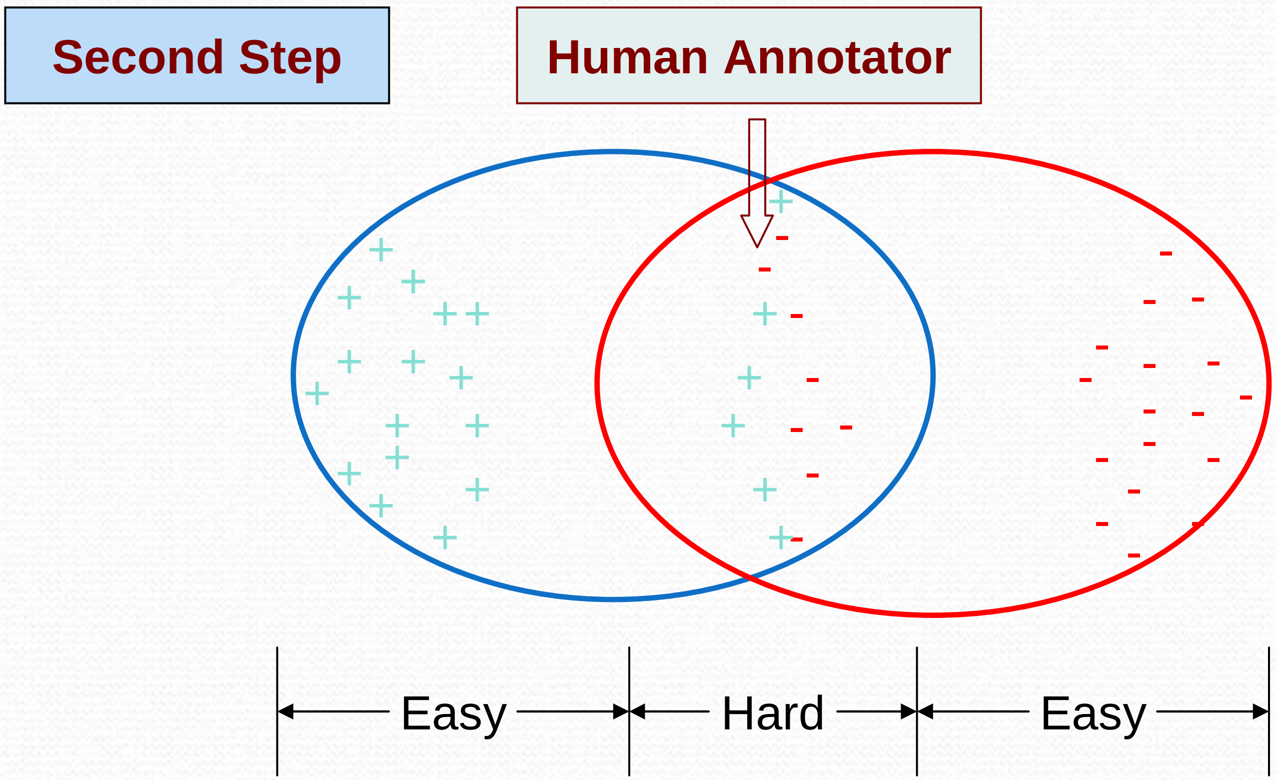

Reviews
Ambiguous
(Hard)
Unambiguous
(Easy)
Classify
Cluster
SVM
+
-
+
-

Main Idea



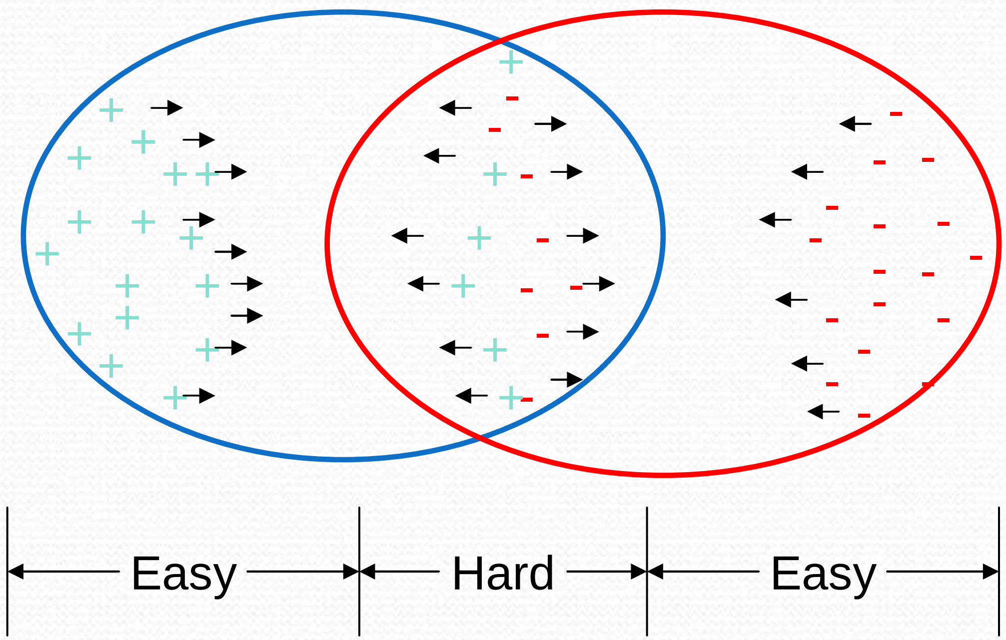
First Step
Ambiguous
Strongly Positive
Strongly Negative
Easy
Hard
Easy

Main Idea



Main Idea

Third Step



A Three-Step Approach

- **Identify and cluster unambiguous data points**
- Hand-label a few ambiguous data points
- Classify the remaining ambiguous data points

Identify and Cluster Unambiguous Data Points

- **Completely unsupervised**
- **Features: Bag of words**
- **Motivated by spectral techniques**

Identify and Cluster Unambiguous Data Points

- **We extend Ng et al's (2002) spectral clustering algorithm to**
  - separate ambiguous data
  - cluster the unambiguous data by sentiment

Ng et al's Spectral Clustering Algorithm

- **Algorithm for 2-way clustering**
- Given a data matrix D (dimension: n x f),

Ng et al's Spectral Clustering Algorithm

- **Algorithm for 2-way clustering**
- Given a data matrix D (dimension: n x f),
  - **Form similarity matrix S=∅(D) (dimension: n x n)**

We used dot product

Ng et al's Spectral Clustering Algorithm

- **Algorithm for 2-way clustering**
- Given a data matrix D (dimension: n x f),
  - Form similarity matrix S=∅(D) (dimension: n x n)
  - **Form diagonal matrix G (dimension: n x n), where $G(i,i)$=sum of the i-th row of S**
  - **Form Laplacian matrix $L = G^{-1/2}\ S\ G^{1/2}$**

Ng et al's Spectral Clustering Algorithm

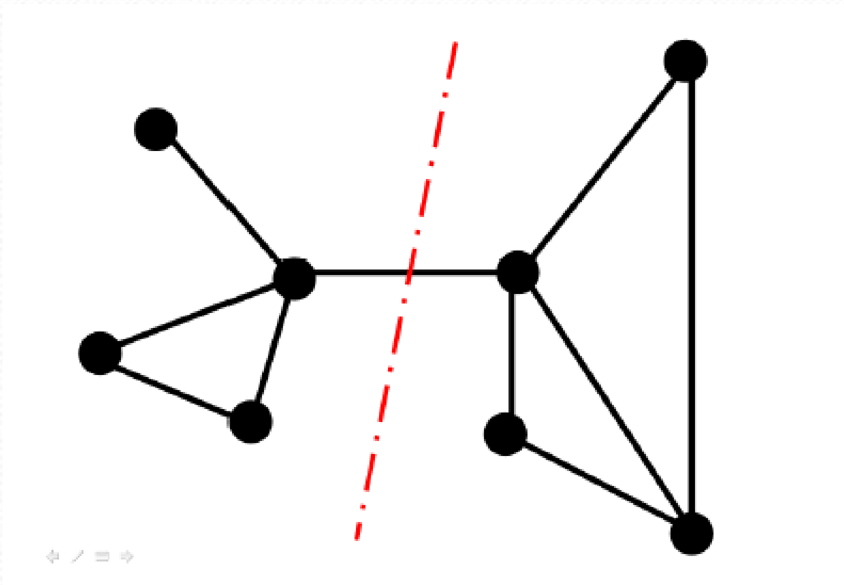
- **Algorithm for 2-way clustering**
- Given a data matrix D (dimension: n x f),
  - Form similarity matrix $S=\varnothing(D)$ (dimension: n x n)
  - Form diagonal matrix G (dimension: n x n),
    where $G(i,i)$=sum of the i-th row of S
  - Form Laplacian matrix $L=G^{-1/2}\, S\, G^{1/2}$
  - **Find the eigenvector *e* corresponding to 2^nd largest eigenvalue of *L***

Ng et al's Spectral Clustering Algorithm

- **Algorithm for 2-way clustering**
- Given a data matrix D (dimension: n x f),
  - Form similarity matrix S=∅(D) (dimension: n x n)
  - Form diagonal matrix G (dimension: n x n),
    where $G(i,i)$=sum of the i-th row of S
  - Form Laplacian matrix $L = G^{-1/2} S G^{1/2}$
  - Find the eigenvector e corresponding to 2^{nd} largest eigenvalue of L
  - **Use 2-means to cluster *n* data points using *e*.**

Why 2nd Eigenvector?

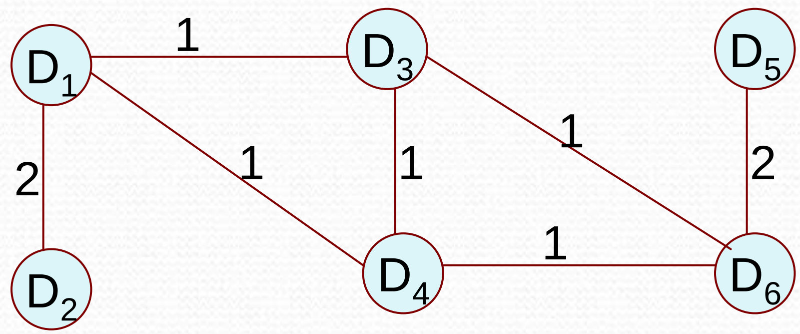
- Shi and Malik (2000): Normalized Cut and Image Segmentation
- 2nd eigenvector of the Laplacian induces the **normalized mincut** of a graph formed from the similarity matrix S.



An Example

	-			+		
	f1	f2	f3	f4	f5	f6
D1	1	1	1	0	0	0
D2	1	1	0	0	0	0
D3	0	0	1	0	1	0
D4	0	0	1	1	0	0
D5	0	0	0	1	0	1
D6	0	0	0	1	1	1

Data Matrix

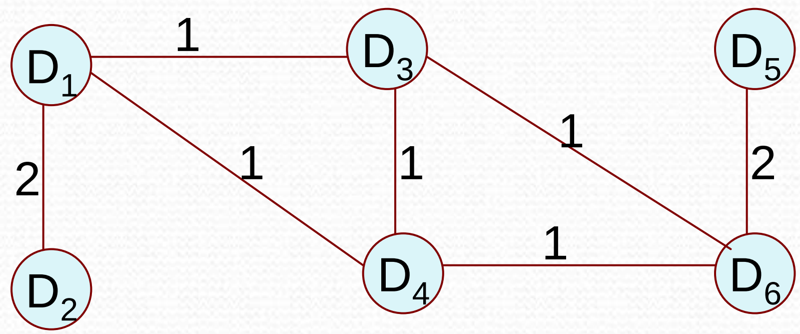


Similarity Graph

An Example

	-			+		
	f1	f2	f3	f4	f5	f6
D1	1	1	1	0	0	0
D2	1	1	0	0	0	0
D3	0	0	1	0	1	0
D4	0	0	1	1	0	0
D5	0	0	0	1	0	1
D6	0	0	0	1	1	1

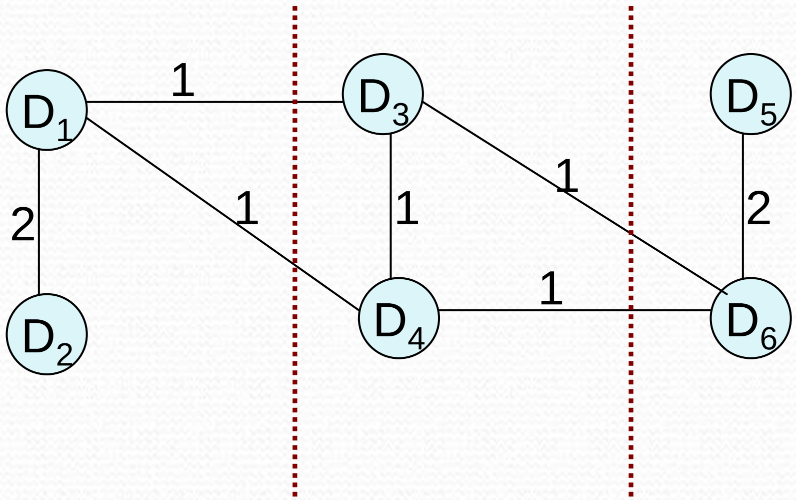
Data Matrix



Similarity Graph

An Example

- Similarity Graph

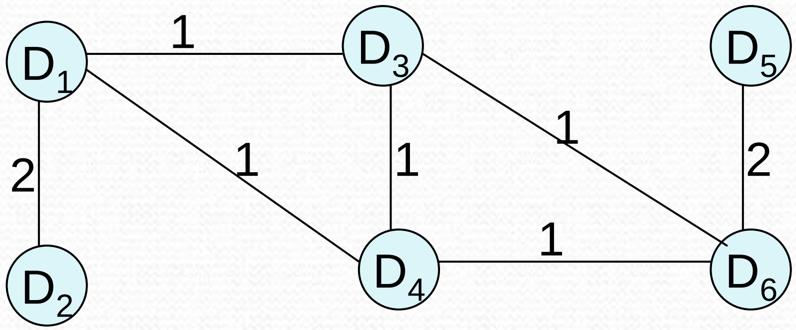


Two possible normalized mincut partitions

Let’s see what partition 2nd eigenvector produces

An Example

- Similarity Graph

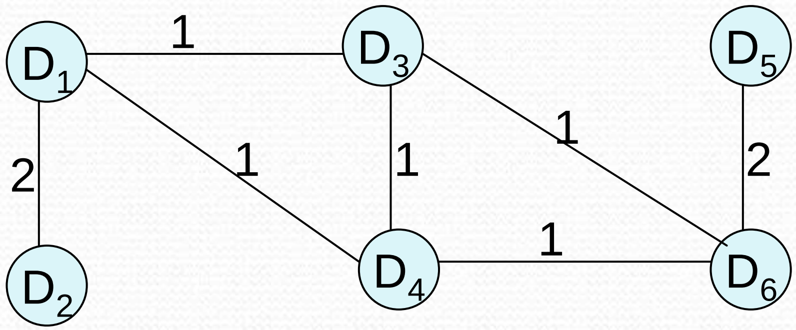


- Top 2 eigenvectors of its Laplacian

$$
\begin{array}{c c}
D_1 & \\
D_2 & \\
D_3 & \\
D_4 & \\
D_5 & \\
D_6 &
\end{array}
\begin{pmatrix}
0.6362 & -0.7709 \\
0.4161 & -0.7502 \\
0.4838 & 0.0135 \\
0.8082 & 0.2593 \\
0.5130 & 0.5637 \\
0.6973 & 0.6605
\end{pmatrix}
$$

An Example

- Similarity Graph

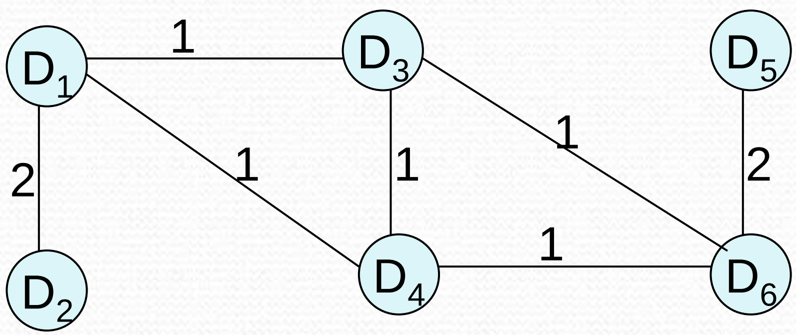


- Top 2 eigenvectors of its Laplacian

$$\begin{array}{c} D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \\ D_6 \end{array} \begin{pmatrix} 0.6362 & -0.7709 \\ 0.4161 & -0.7502 \\ 0.4838 & 0.0135 \\ 0.8082 & 0.2593 \\ 0.5130 & 0.5637 \\ 0.6973 & 0.6605 \end{pmatrix} \begin{array}{l} \left.\begin{array}{c} \\ \\ \end{array}\right\} \boxed{-} \\ \left.\begin{array}{c} \\ \\ \\ \\ \end{array}\right\} \boxed{+} \end{array}$$

An Example

- Similarity Graph

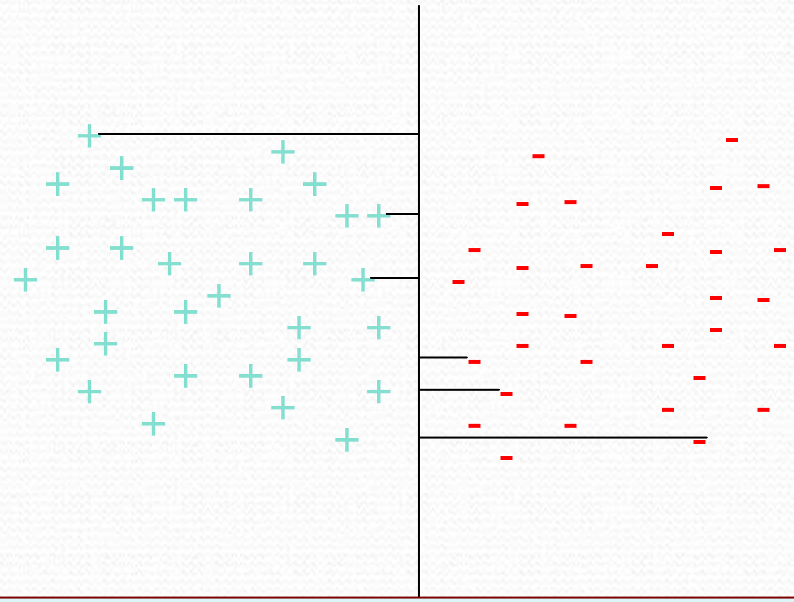


- Top 2 eigenvectors of its Laplacian

$$\begin{array}{c} D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \\ D_6 \end{array} \begin{pmatrix} 0.6362 & -0.7709 \\ 0.4161 & -0.7502 \\ 0.4838 & 0.0135 \\ 0.8082 & 0.2593 \\ 0.5130 & 0.5637 \\ 0.6973 & 0.6605 \end{pmatrix} \begin{array}{l} \left.\begin{array}{l} \\ \\ \end{array}\right\} \boxed{-} \\ \left.\begin{array}{l} \\ \\ \end{array}\right\} \boxed{\textbf{Ambiguous}} \\ \left.\begin{array}{l} \\ \\ \end{array}\right\} \boxed{+} \end{array}$$

Identify Unambiguous Reviews

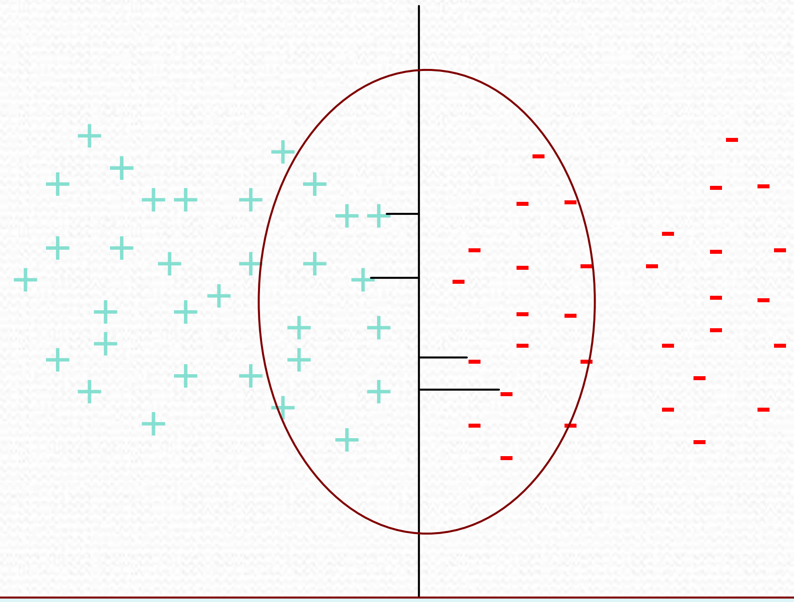
- How to separate unambiguous reviews?



Orthogonal projections on a learned-dimension

Identify Unambiguous Reviews

- How to separate unambiguous reviews?



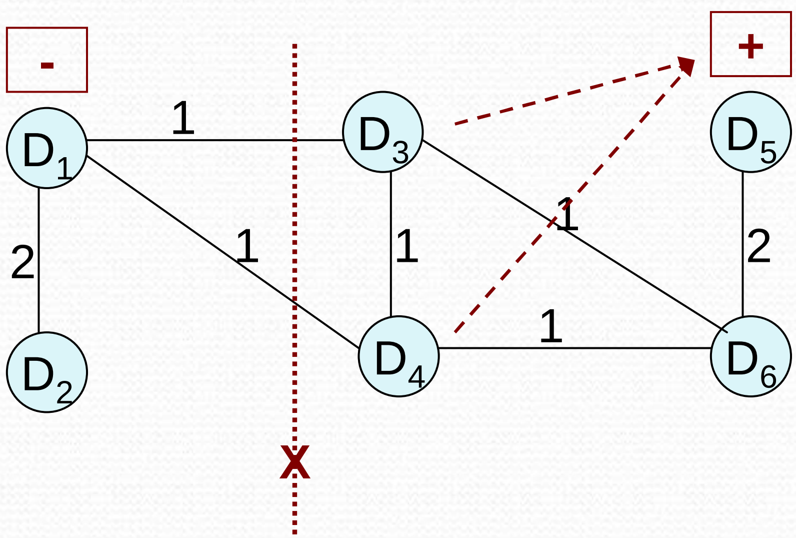
Ambiguous points have small projections

Our Clustering Algorithm

- Two important ideas
  - **Separate ambiguous points from unambiguous points**
  - **Iterative clustering**

Two Important Ideas

- Why we need to handle ambiguous points carefully?

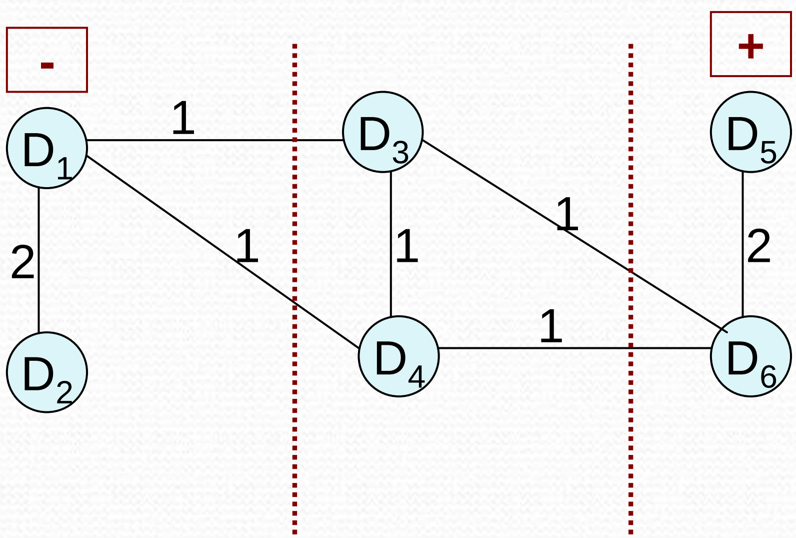


$$
\begin{array}{l}
D1 \\ D2 \\ D3 \\ D4 \\ D5 \\ D6
\end{array}
\begin{pmatrix}
0.6362 & -0.7709 \\
0.4161 & -0.7502 \\
0.4838 & 0.0135 \\
0.8082 & 0.2593 \\
0.5130 & 0.5637 \\
0.6973 & 0.6605
\end{pmatrix}
$$

- **2^{nd} eigenvector puts D_3 and D_4 into + class.**

Two Important Ideas

- Why we need to handle ambiguous points carefully?

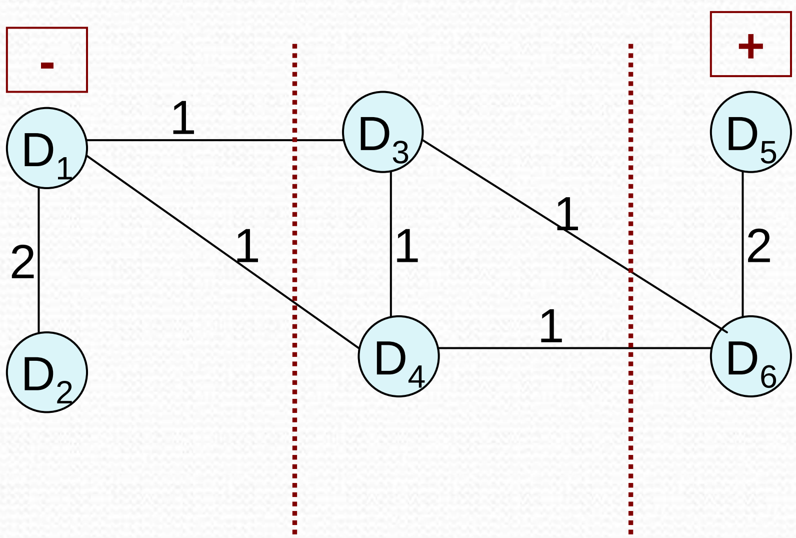


	-			+		
	f1	f2	f3	f4	f5	f6
D1	1	1	1	0	0	0
D2	1	1	0	0	0	0
D3	0	0	1	0	1	0
D4	0	0	1	1	0	0
D5	0	0	0	1	0	1
D6	0	0	0	1	1	1

- **What if f3 feature is strong?**

Two Important Ideas

- Why we need to handle ambiguous points carefully?



	f1	f2	f3	f4	f5	f6
D1	1	1	1	0	0	0
D2	1	1	0	0	0	0
D3	0	0	1	0	1	0
D4	0	0	1	1	0	0
D5	0	0	0	1	0	1
D6	0	0	0	1	1	1

- **Discriminative systems (e.g., SVM) better for complex feature space**

Our Clustering Algorithm

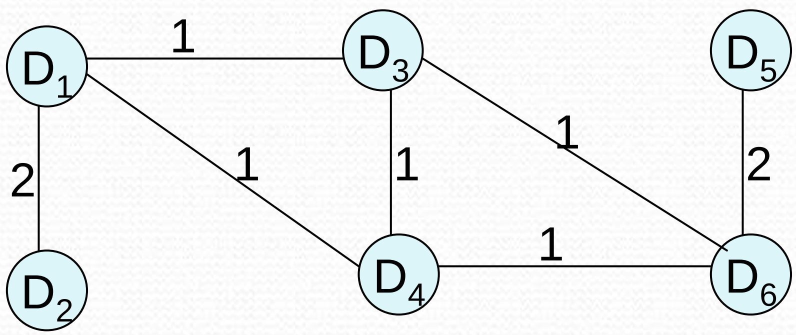
- Two important ideas
  - Separate ambiguous points from unambiguous points
  - **Iterative clustering**

Two Important Ideas

- **Sparse graph** ideal for clustering
- We **remove ambiguous points iteratively** to make the graph more sparse

Two Important Ideas

- **Sparse graph ideal for clustering**

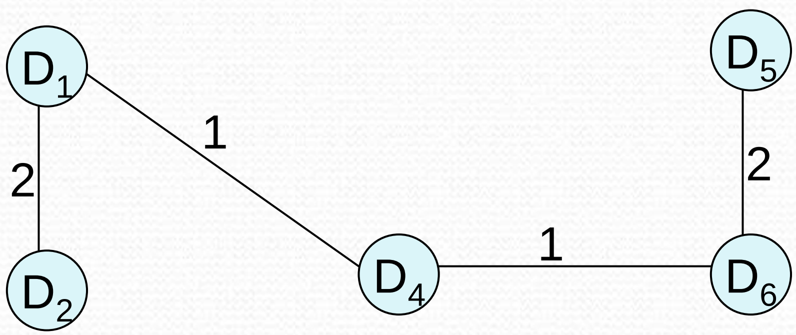


$$
\begin{array}{c}
D1 \\ D2 \\ D3 \\ D4 \\ D5 \\ D6
\end{array}
\begin{pmatrix}
0.6362 & -0.7709 \\
0.4161 & -0.7502 \\
0.4838 & 0.0135 \\
0.8082 & 0.2593 \\
0.5130 & 0.5637 \\
0.6973 & 0.6605
\end{pmatrix}
$$

- **We remove ambiguous points iteratively to make the graph more sparse**

Two Important Ideas

- **Sparse graph ideal for clustering**



$$
\begin{array}{c}
D1 \\ D2 \\ D3 \\ D4 \\ D5 \\ D6
\end{array}
\begin{pmatrix}
0.6362 & -0.7709 \\
0.4161 & -0.7502 \\
0.4838 & 0.0135 \\
0.8082 & 0.2593 \\
0.5130 & 0.5637 \\
0.6973 & 0.6605
\end{pmatrix}
$$

- **Remove D_3**

Two Important Ideas

- **Sparse graph ideal for clustering**

$$
\begin{array}{l}
D1 \\
D2 \\
D4 \\
D5 \\
D6 \\
\\
\end{array}
\begin{pmatrix}
0.6362 & -0.7709 \\
0.4161 & -0.7502 \\
0.8082 & 0.2593 \\
0.5130 & 0.5637 \\
0.6973 & 0.6605 \\
 &
\end{pmatrix}
$$

- **Remove D_4**

Our Algorithm

- Two important ideas
  - **Separate ambiguous points from unambiguous points**
  - **Iterative clustering**

Our Algorithm

- Given a data matrix D,
  1. Form similarity matrix S, diagonal matrix G, *and* Laplacian matrix $L = G^{-1/2} S G^{1/2}$

Our Algorithm

- Given a data matrix *D*,
  1. Form similarity matrix *S*, diagonal matrix *G*, *and* Laplacian matrix $L = G^{-1/2} S G^{1/2}$
  2. Find the top 5 eigenvectors of *L*
  3. Row-normalize the eigenvectors
  4. Pick eigenvector ***e*** that produces min normalized cut

Our Algorithm

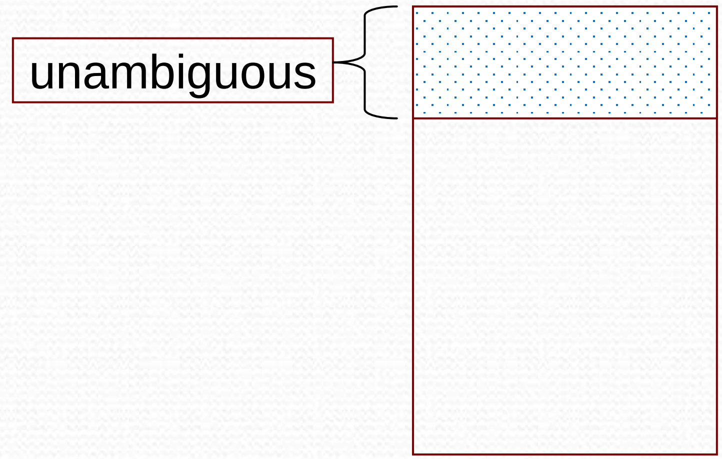
- Given a data matrix D,
  1. Form similarity matrix S, diagonal matrix G, *and* Laplacian matrix $L=G^{-1/2}\ S\ G^{1/2}$
  2. Find the top 5 eigenvectors of L
  3. Row-normalize the eigenvectors
  4. Pick eigenvector ***e*** that produces min normalized cut
  5. Sort D according to ***e***, remove β ambiguous points in the middle
  6. If $|D|= \alpha$ goto Step 7; else goto Step 1

Our Algorithm

- Given a data matrix *D*,
  1. Form similarity matrix *S*, diagonal matrix *G*, *and* Laplacian matrix $L=G^{-1/2}\ S\ G^{1/2}$
  2. Find the top 5 eigenvectors of *L*
  3. Row-normalize the eigenvectors
  4. Pick eigenvector ***e*** that produces min normalized cut
  5. Sort *D* according to ***e***, remove β ambiguous points in the middle
  6. If $|D|= \alpha$ goto Step 7; else goto Step 1
  7. Use 2-means to cluster α data points using ***e***

Parameters

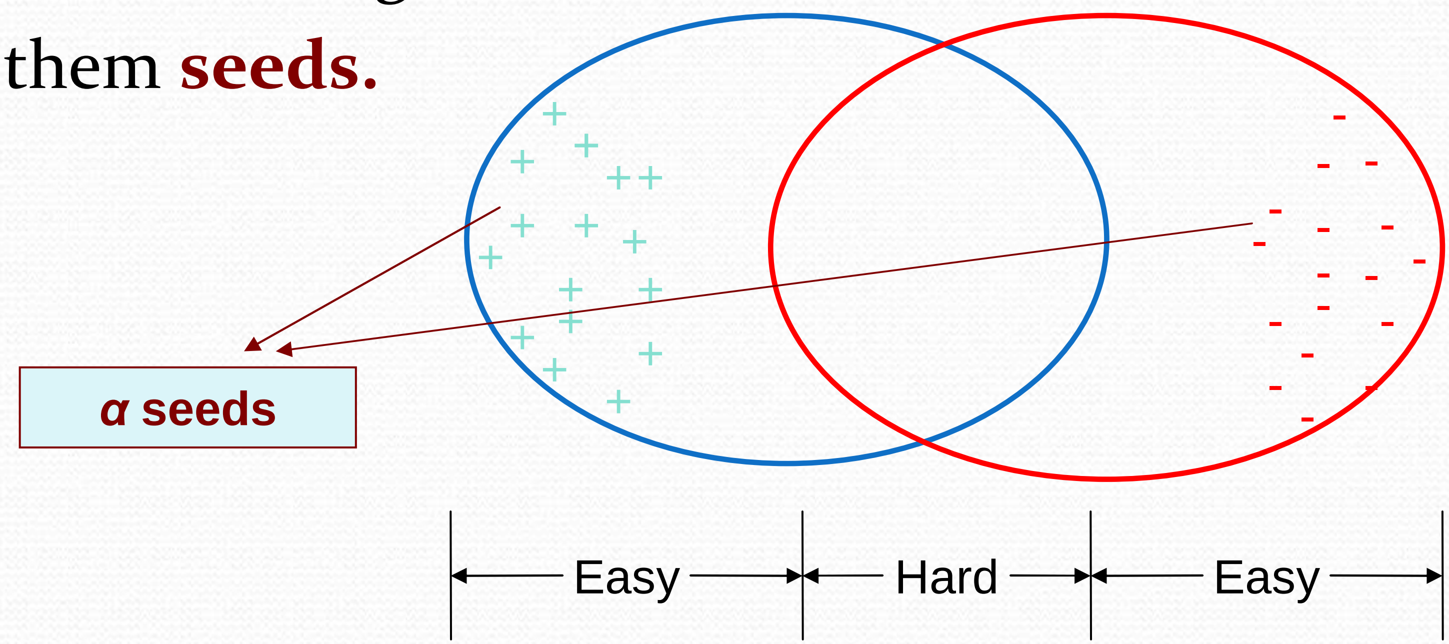
- Stopping criteria
  - $\alpha = n/4$
  - At least 25% are unambiguous
- Step size:
  - β=50
  - We drop 50 ambiguous data points in iteration



End of first step,

End of First Step

- We have ***α*** unambiguous data points clustered into positive class and negative class.
- We call them **seeds.**

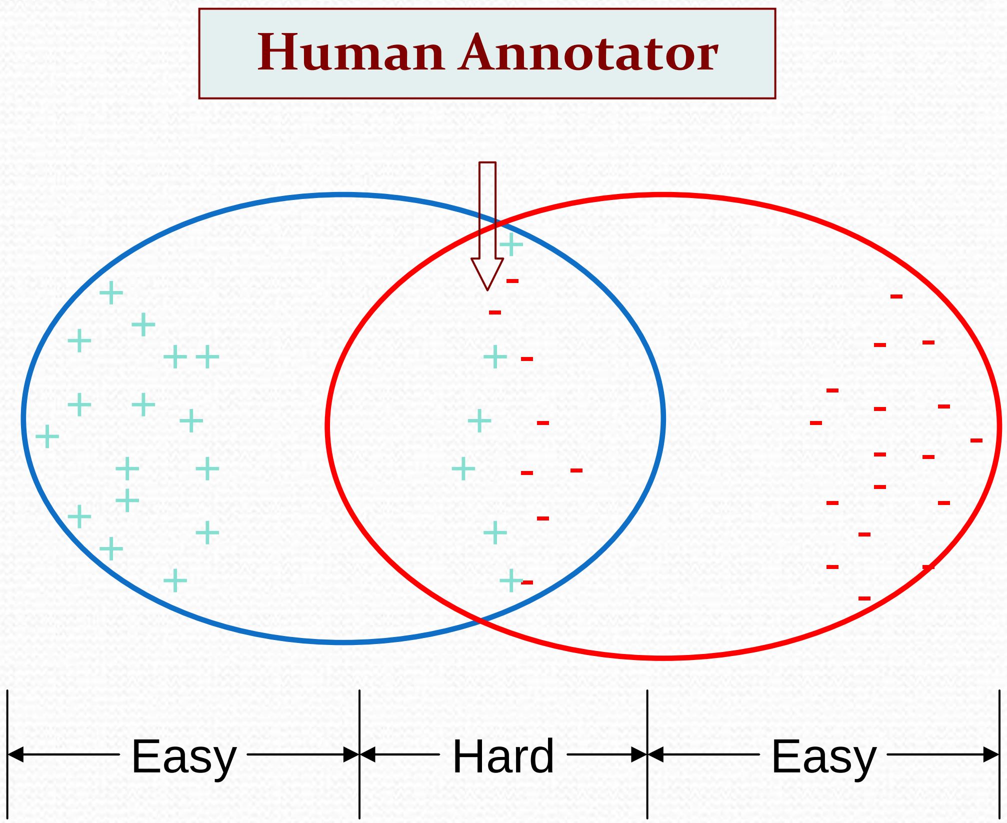


Next we build a classifier based on seeds

A Three-Step Approach

- Identify and cluster unambiguous data points
- **Hand-label a few ambiguous data points**
- Classify the remaining ambiguous data points

Main Idea

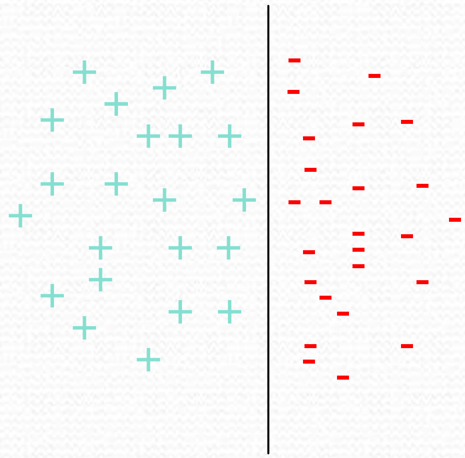


Active Learning

- **Margin-based** (Tong and Koller 2002)
- **Iterative**

Margin-based Active Learning

Unlabeled data



We use SVM to learn the margin of separation

Active Learning

- **Steps:**
  1. **Create a SVM classifier using seeds as training data**

Active Learning

- **Steps:**
  1. Create a SVM classifier using seeds as training data
  2. **Find 10 most uncertain data points from each side of hyperplane**

Active Learning

- **Steps:**
  1. Create a SVM classifier using seeds as training data
  2. Find 10 most uncertain data points from each side of hyperplane
  3. **Ask a human to label these data points**
  4. **Add them to the training data**

Active Learning

- **Steps:**
  1. Create a SVM classifier using seeds as training data
  2. Find 10 most uncertain data points from each side of hyperplane (i.e., those that are closest to hyperplane)
  3. Ask a human to label those 20 data points
  4. Add them to the training data
  5. **Repeat until 100 data points are labeled**

Active Learning

- 100 labeled points only
- Now we have

α automatically acquired labels

+

100 active learning labels

System is more knowledgeable

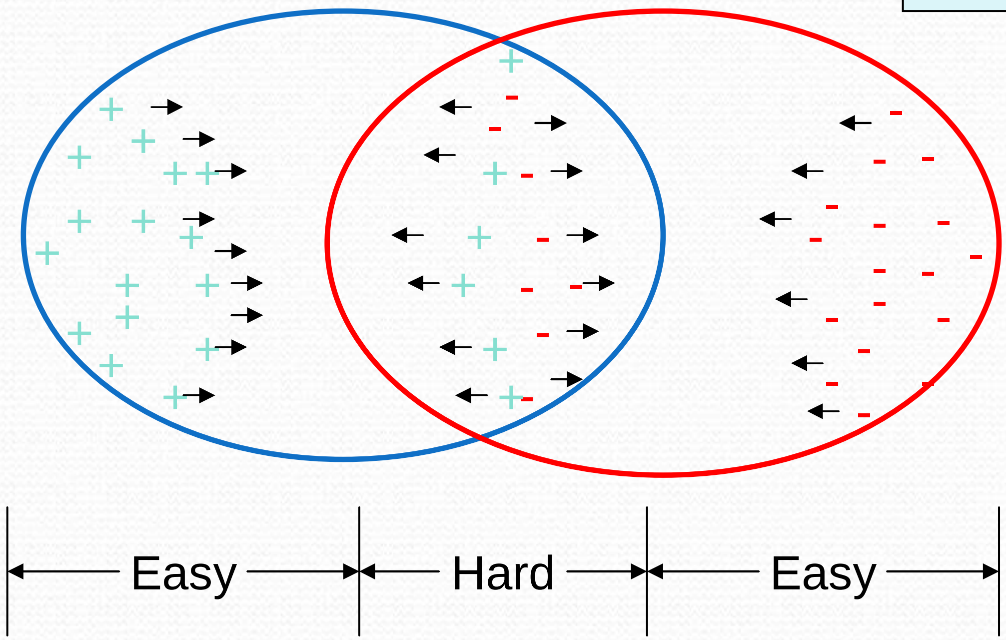
End of second step,

A Three-Step Approach

- Identify and cluster unambiguous data points
- Hand-label a few ambiguous data points
- **Classify the remaining ambiguous data points**

Main Idea

Little Labeled Data
Lots of Unlabeled Data



Semi-supervised Learning

- Transductive SVM
- $(\alpha+100)$ labeled points, $(n-\alpha-100)$ unlabeled points

Semi-supervised Learning

- Transductive SVM
- (α+100) labeled points, (n-α-100) unlabeled points

- **One final step!**

Semi-supervised Learning

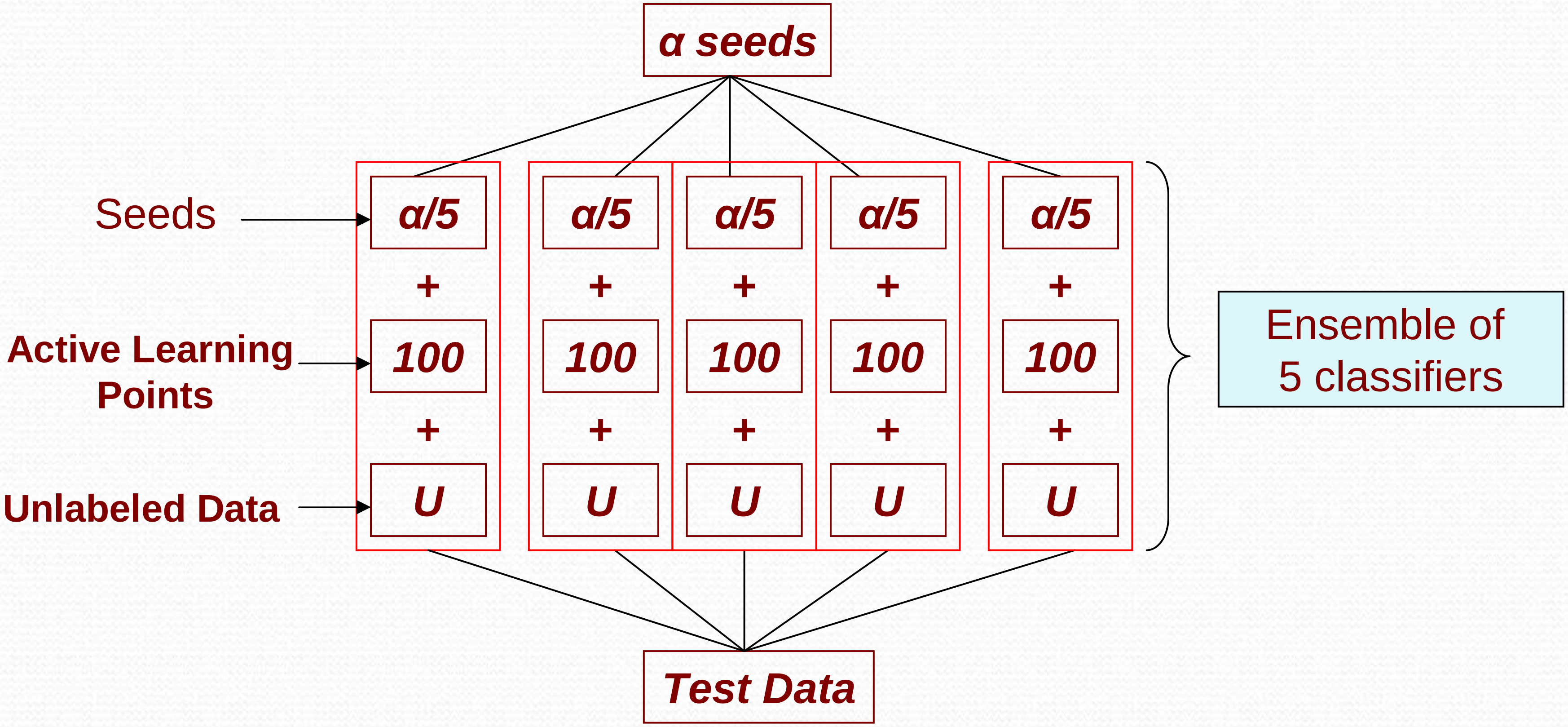
- ***Training Data:***

  α seeds + **100** active learning labels

- Seeds are automatically acquired: not perfectly labeled
- 100 active learning points are perfectly labeled

- The system should be **noise-tolerant** to seeds

Ensembled Transduction



- **Now we move on to evaluation section,**

Evaluation

- **Datasets**
  - Movie (Pang et al., 2002)
  - 4 datasets from Blitzer et al. (2007)
    - Kitchen and Housewares
    - Electronics
    - Books
    - DVD
  - each has 2000 points (1000 positives & 1000 negatives)
- **Evaluation Metrics**
  - Accuracy

Experimental Setup

- 10-fold cross validation
  - Same number of positives and negatives per fold

Three Baselines

- Semi-supervised Spectral Learning (Kamvar et al., 2003)
- Transductive SVM (Joachims, 1999)
- Active Learning (Tong and Koller, 2002)

- All baselines are trained using same amount of labeled and unlabeled data as in our approach

Baseline Accuracies

	MOV	KIT	ELE	BOO	DVD
Spectral Learning	67.3	63.7	57.7	55.8	56.2
Transductive SVM	68.7	65.5	62.9	58.7	57.3
Active Learning	68.9	68.1	63.3	58.6	58.0

Active Learning is the best baseline

Our Approach

- **Step 1:**
  - Train a SVM on α seeds only
- **Step 2:**
  - Train a SVM on α seeds and 100 active-learning points
- **Step 3:**
  - Train ensemble of TSVMs on (α+100) labeled points and (n-α-100) unlabeled points
- $n=2000$ and $\alpha=n/4$ in our experiments

Our Approach: Results

	MOV	KIT	ELE	BOO	DVD
Best Baseline	68.9	68.1	63.3	58.6	58.0
After First Step	69.8	70.8	65.7	58.6	55.8

Even after first step we beat best baseline for 4 datasets

First step does not use any hand-labeled data!

Our Approach: Results

	MOV	KIT	ELE	BOO	DVD
Best Baseline	68.9	68.1	63.3	58.6	58.0
After First Step	69.8	70.8	65.7	58.6	55.8
After Second Step	73.5	73.0	69.9	60.6	59.8

100 Active Learning Points Incorporated

Our Approach: Results

	MOV	KIT	ELE	BOO	DVD
Best Baseline	68.9	68.1	63.3	58.6	58.0
After First Step	69.8	70.8	65.7	58.6	55.8
After Second Step	73.5	73.0	69.9	60.6	59.8
After Third Step	76.2	74.1	70.6	62.1	62.7

Ensembled Transductive Learning

Our Results

	MOV	KIT	ELE	BOO	DVD
Best Baseline	**68.9**	**68.1**	**63.3**	**58.6**	**58.0**
After First Step	69.8	70.8	65.7	58.6	55.8
After Second Step	73.5	73.0	69.9	60.6	59.8
After Third Step	**76.2**	**74.1**	**70.6**	**62.1**	**62.7**

We outperform the best baseline

Additional Experiments

- Importance of seeds
- Importance of ensemble
- Importance of active learning

Analysis 1: Importance of Seeds

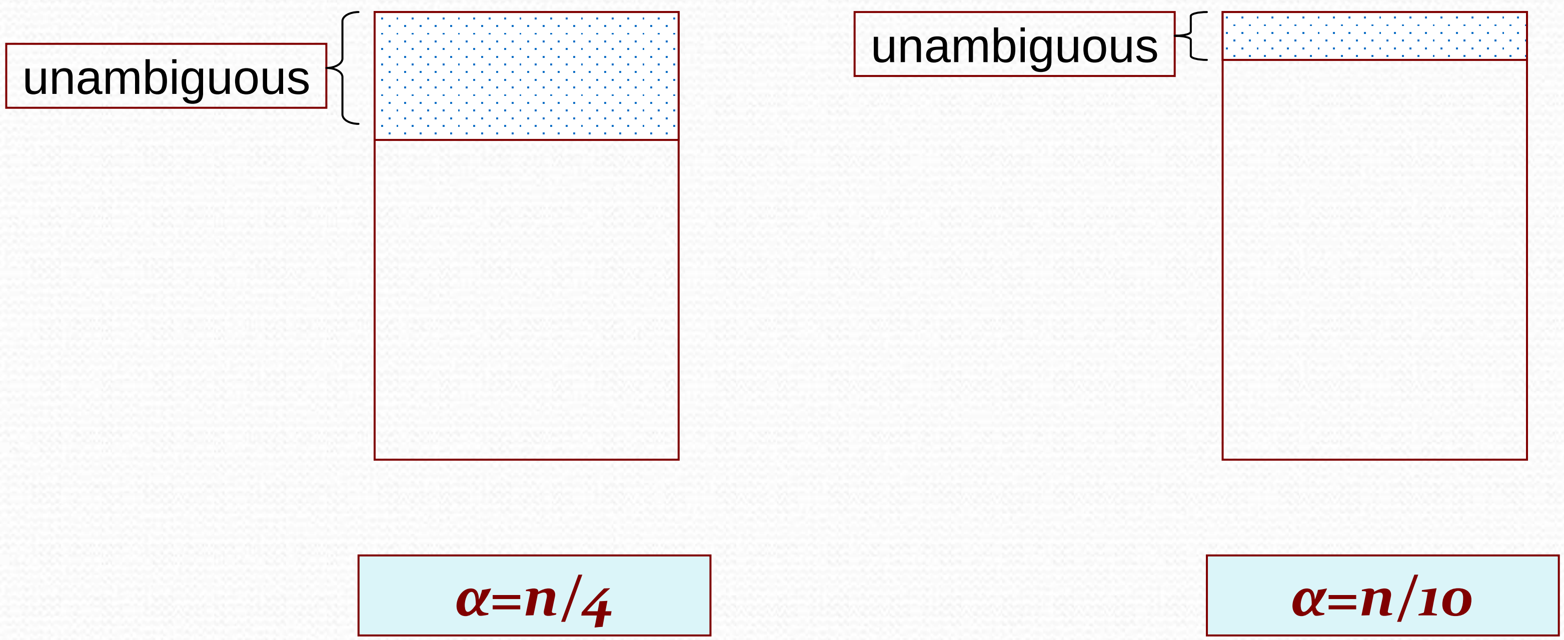
- What if we don't use any seeds?
- Train a transductive SVM on 100 active-learning points only

	MOV	KIT	ELE	BOO	DVD
Our Approach	76.2	74.1	70.6	62.1	62.7
No Seeds	58.3	55.6	59.7	54.0	56.1

Seeds are important

Analysis 1: Importance of Seeds

- What if we use fewer seeds?



Analysis 1: Importance of Seeds

- What if we use fewer seeds?
- ***α=n/10***
- Seeds are **less ambiguous** and **more accurate**

	MOV	KIT	ELE	BOO	DVD
Our Approach	76.2	74.1	70.6	62.1	62.7
Fewer Seeds	74.6	69.7	69.1	60.9	63.3

More seeds are beneficial even if they are noisy

Analysis 2: No Ensemble

	MOV	KIT	ELE	BOO	DVD
Our Approach	76.2	74.1	70.6	62.1	62.7
No Ensemble	74.1	72.7	68.8	61.5	59.9

Ensemble approach is more noise tolerant

Analysis 3: Passive Learning

- What if we choose ambiguous points randomly?
- Replace active learning with passive learning

	MOV	KIT	ELE	BOO	DVD
Our Approach	76.2	74.1	70.6	62.1	62.7
Passive Learning	74.1	72.4	68.0	63.7	58.6

Active Learning is essential

- **Finally,**

Our Semi-supervised Approach

Why not isomap?

Why not Mark Dredze's?

Spectral Learning

Active Learning

Transductive Learning

Why not manifold regularization?

Each step can be improved

- But … our goal is to introduce a new semi-supervised architecture:

Mine the Easy, Classify the Hard

- It's **general:**
  - can be applied to other domains

Thank you

God bless you