Semi-supervised Learning

FIT5149 Week10

Outline

- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

Outline

- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

Learning Problems

- Supervised learning:
  - Given a sample consisting of object-label pairs (x_i,y_i), find the predictive relationship between objects and labels.
- Un-supervised learning:
  - Given a sample consisting of only objects, look for interesting structures in the data, and group similar objects.
- What is Semi-supervised learning?
  - Supervised learning + Additional unlabeled data
  - Unsupervised learning + Additional labeled data

Motivation of SSL

- Supervised Learning models require labeled data
- Learning a reliable model usually requires plenty of labeled data
- Labeled Data: Expensive and Scarce
- Unlabeled Data: Abundant and Free/Cheap
    - E.g., webpage classification: easy to get unlabelled webpages

Example of hard-to-get labels

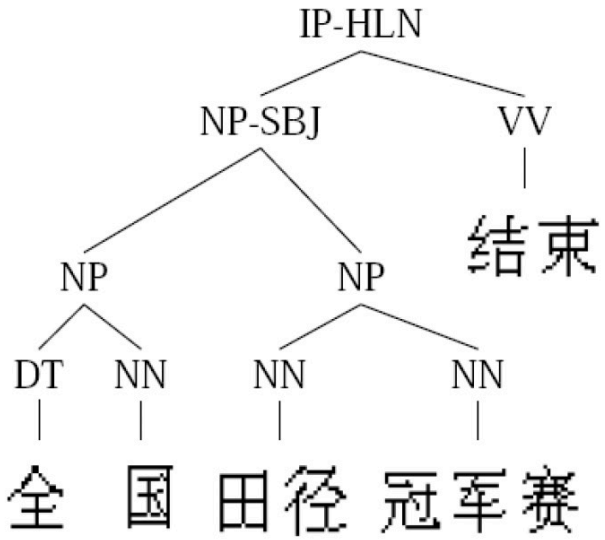
- Task: speech analysis
  - Switchboard dataset
  - telephone conversation transcription
  - 400 hours annotation time for each hour of speech

film => f ih_n uh_gl_n_m

be all => bcl b iy iy tr ao tr ao l dl

Example of hard-to-get labels

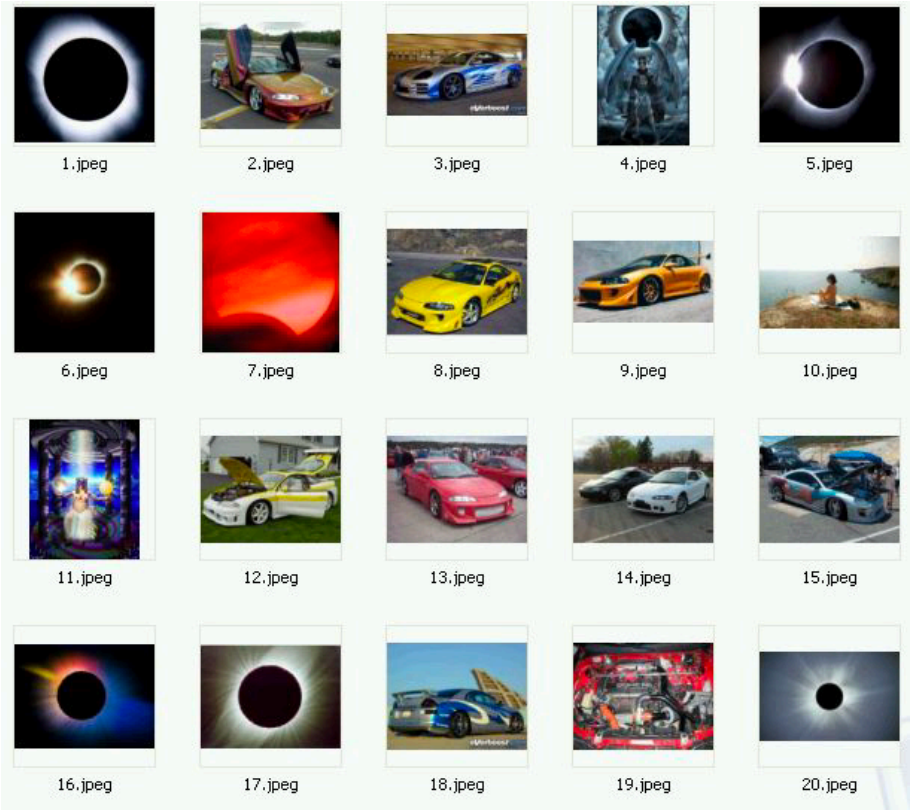
- Task: natural language parsing
  - Penn Chinese Treebank
  - 2 years for 4000 sentences



“The National Track and Field Championship has finished.”

Example of not-so-hard-to-get labels

- Task: image categorization of “eclipse”



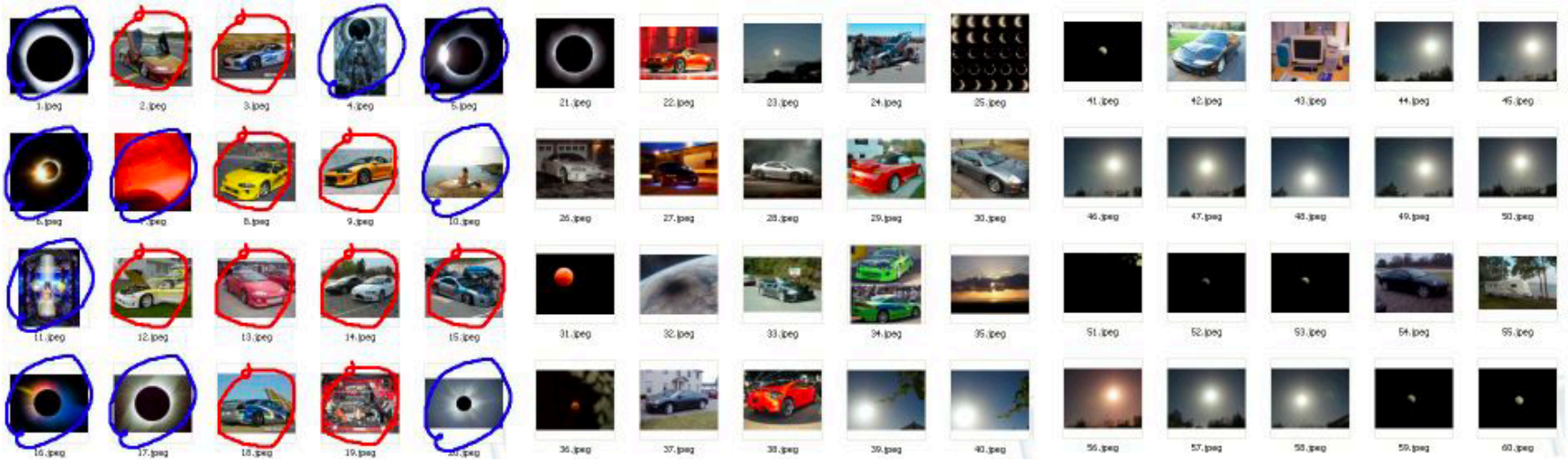
Example of not-so-hard-to-get labels

- Task: image categorization of “eclipse”

There are ways like the ESP game (www.espgame.org) to encourage “human computation” for more labels.

Introduction to SSL

nonetheless...



Today we will learn how to use unlabelled data to improve classification.

Introduction to Semi-supervised Learning (SSL)

- General Idea: Learning from both labeled and unlabeled data
- Semi-supervised Classification/Regression
  - Given: Labeled training data $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{L}$

    Unlabeld data $\mathcal{U} = \{\mathbf{x}_j\}_{j=L+1}^{L+U}$

    (usually U >>L)
  - Goal: Learn a classifier f better than using labeled data alone

Semi-supervised Learning (SSL) vs Transductive Learning

- Transductive: Produce label only for the available unlabeled data.
  - The output of the method is not a classifier.

- Inductive: Not only produce label for unlabeled data, but also produce a classifier.

- What we focus on today is inductive semi-supervised learning..

Outline

- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Training
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

Expectation Maximization (EM)

- Use EM to maximize the joint log-likelihood of labeled and unlabeled data:

$$\sum_i \log\Big(P(y_i|\pi)P(x_i|y_i,\theta)\Big)+$$

L_l : Log-likelihood of labeled data

$$\sum_j \log\Big(\sum_y P(y|\pi)P(x_j|y,\theta)\Big)$$

L_u : Log-likelihood of unlabeled data

Stable Mixing of Information

- Use λ to combine the log-likelihood of labeled and unlabeled data in an optimal way:

$$(1-\lambda)L_l + \lambda L_u$$

- EM can be adapted to optimize it.

- Additional step for determining the best value for λ.

EM_λ Operator

- E and M steps update the value of the parameters for an objective function with particular value of λ.

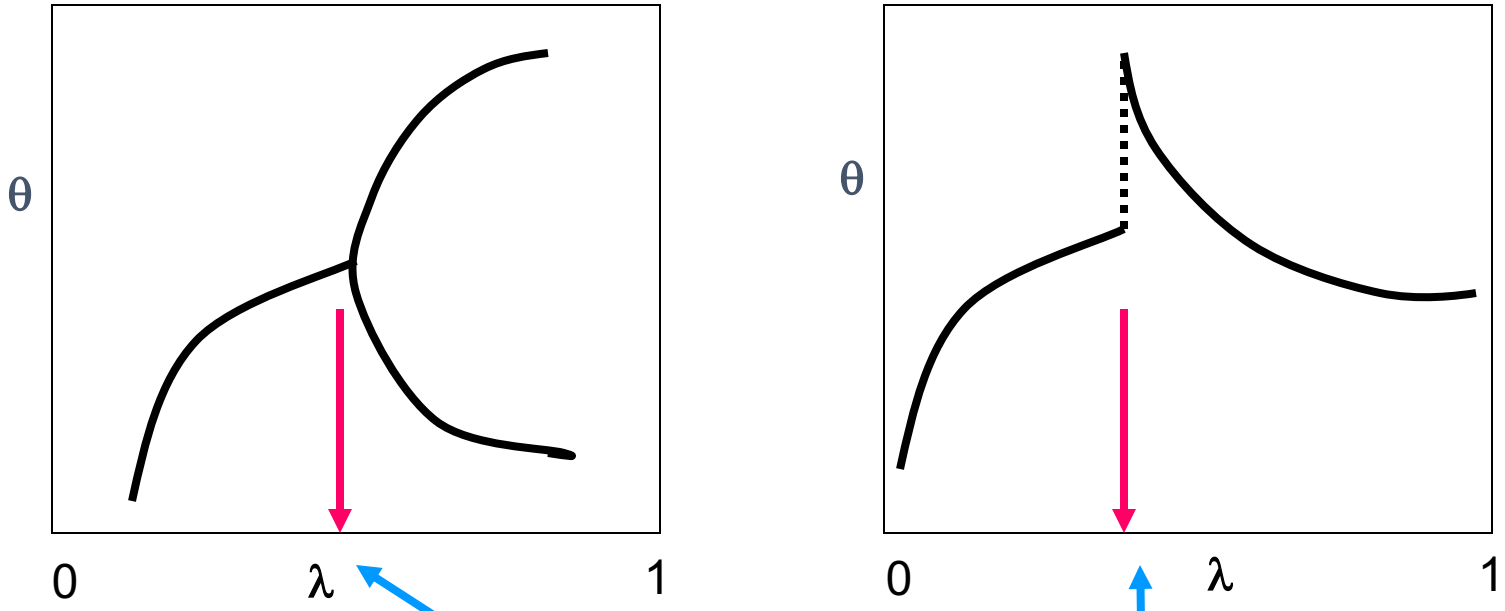
- Name these two steps together as EM_λ operator:

$$\theta^{new} = EM_\lambda(\theta)$$

- The optimal value of the parameters is a fixed point of the EM_λ operator:

$$\theta = EM_\lambda(\theta)$$

Path of solutions



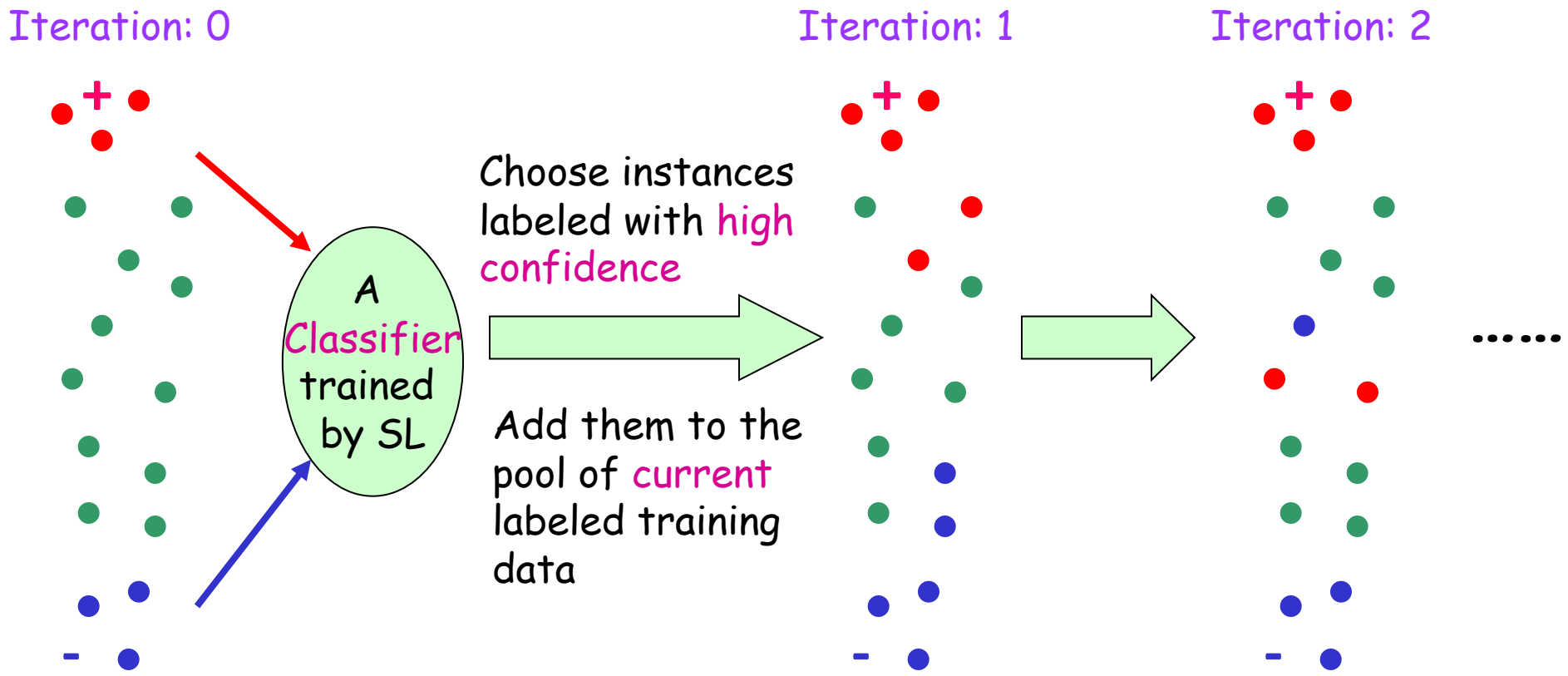
- How to choose the best λ ? $(1-\lambda)L_l + \lambda L_u$
  - By finding the path of optimal solutions as a function of λ
    - Choosing the first λ where a bifurcation or discontinuity occurs; after such points labeled data may not have an influence on the solution.
  - By cross-validation on a held out set.

Outline

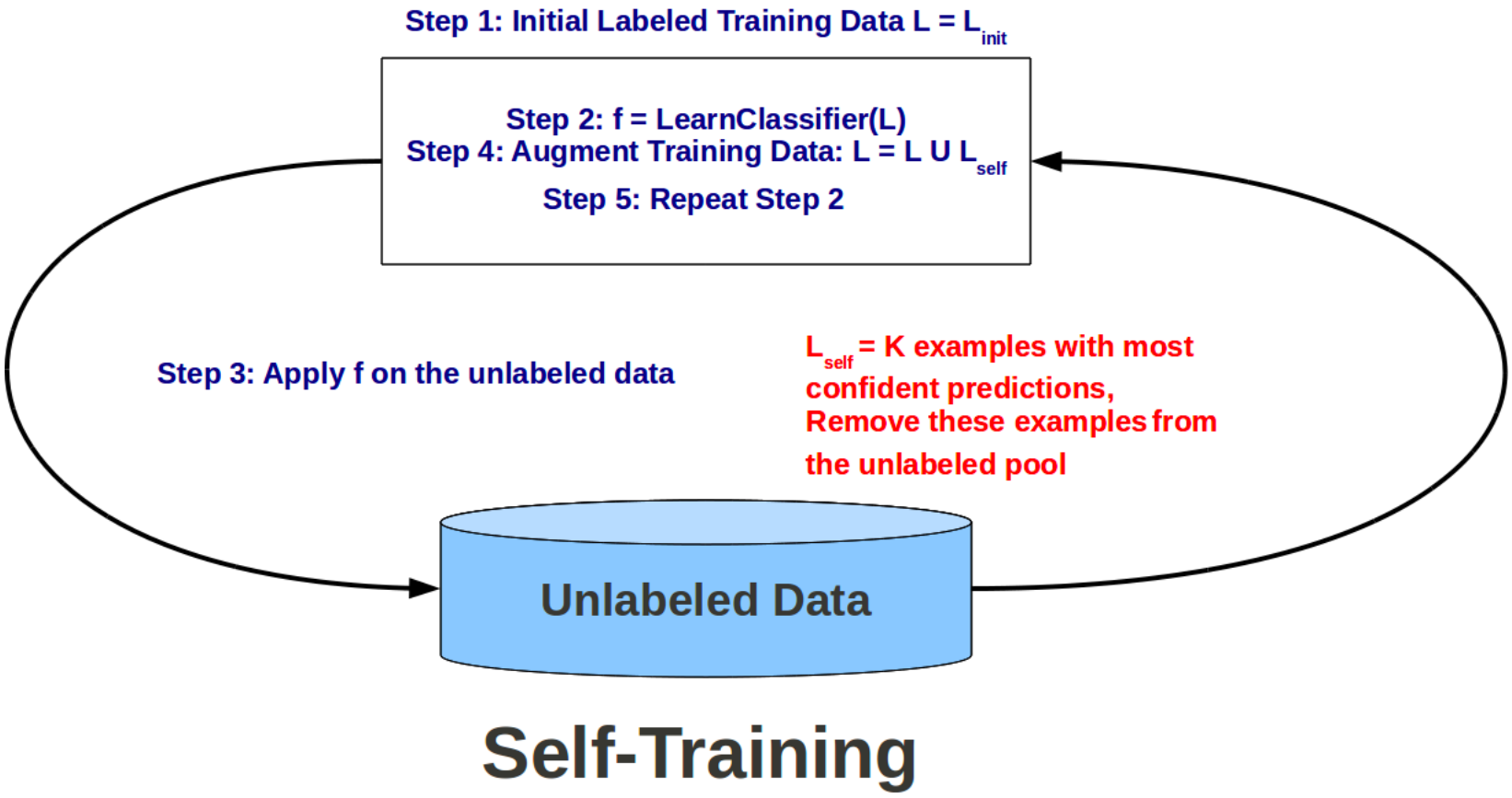
- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

The Yarowsky Algorithm

(Yarowsky 1995)



Self Learning



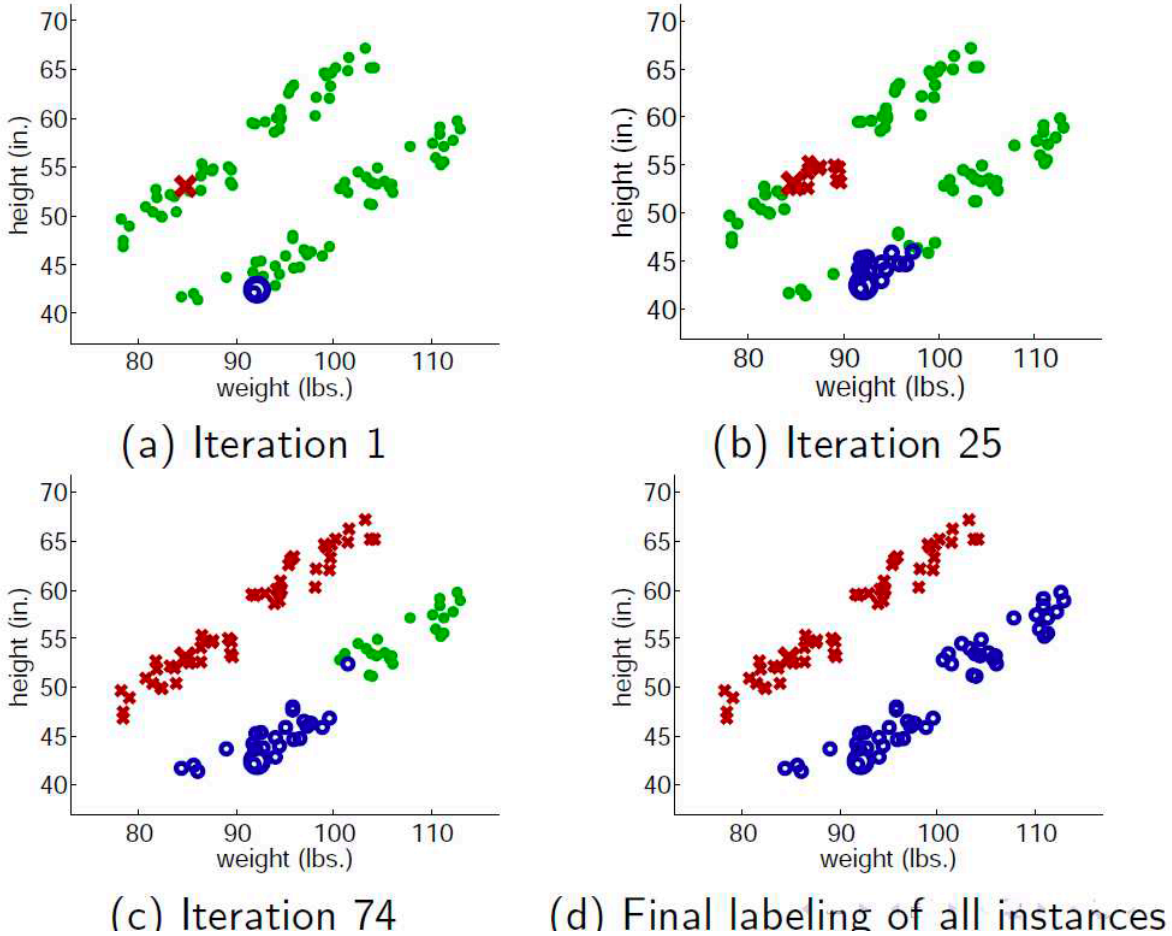
Self Learning

- Assumption: One's own high confidence predictions are correct.

- Variations in Self Training
  - Add a few most (x, f(x)) to labelled data
  - Add all (x, f(x)) to labeled data
  - Add all (x, f(x)) to lableled data, weigh each by confidence
- Can be used with any supervised learner.
- Caution: Prediction mistake can reinforce itself

Self Learning: A Good Case

- Base learner: KNN classifier

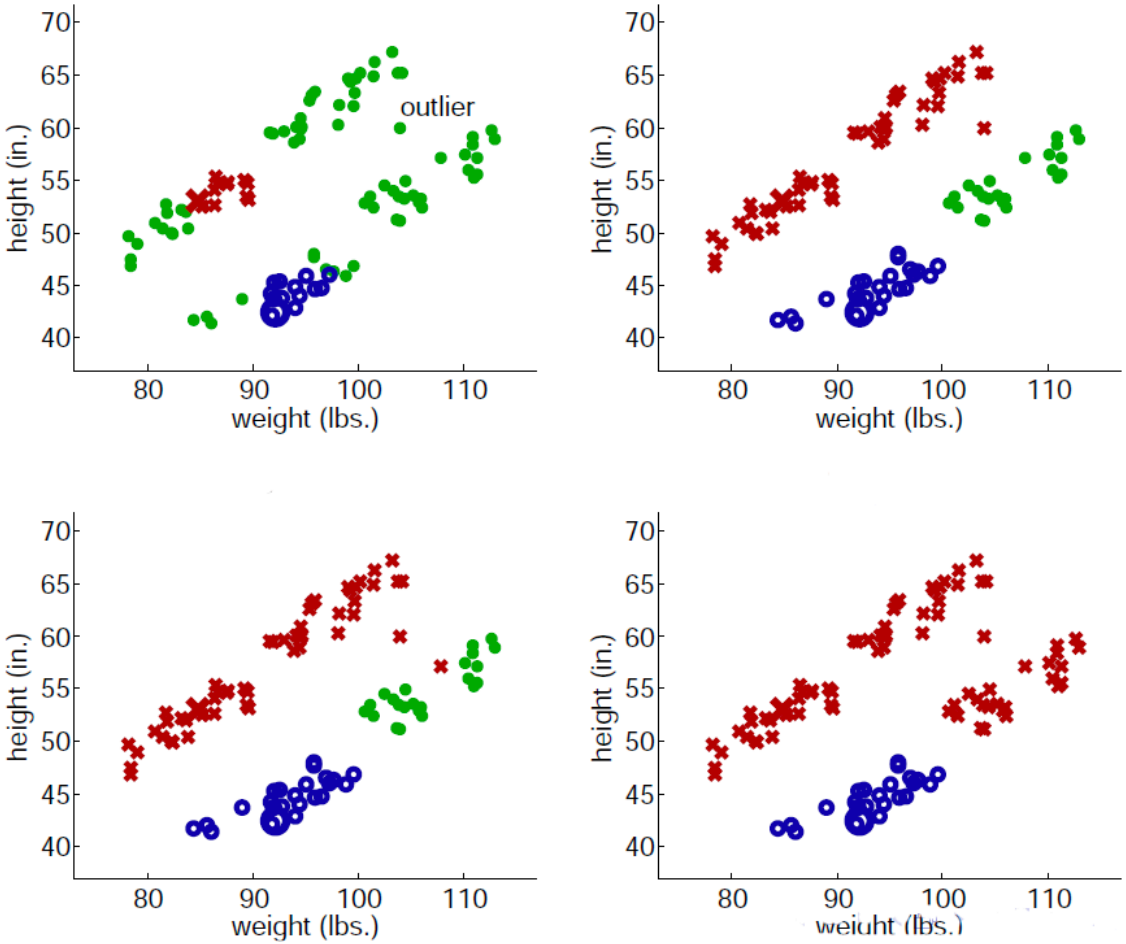


(a) Iteration 1 (b) Iteration 25

(c) Iteration 74 (d) Final labeling of all instances

Self Learning: A Bad Case

- Things can go wrong if there are outliers. Mistakes get reinforced



Self Learning: Pros and Cons

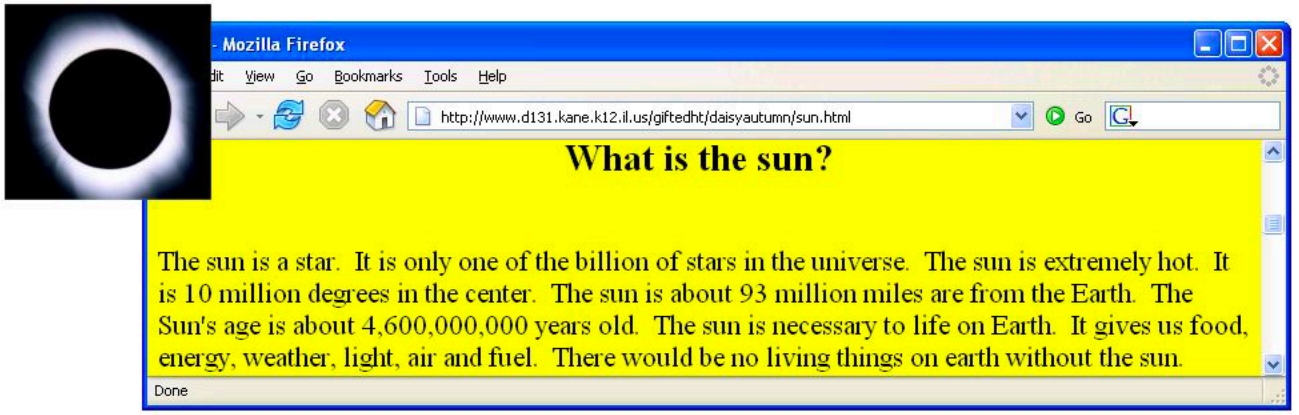
- Pros:
  - The simplest semi-supervised learning method.
  - A wrapper method, applies to existing (complex) classifiers.
  - Often used in real tasks like natural language processing.
- Cons:
  - Early mistakes could reinforce themselves.
  - Cannot say too much in terms of convergence.
    - But there are special cases when self-training is equivalent to the Expectation-Maximization (EM) algorithm.
    - There are also special cases (e.g., linear functions) when the closed-form solution is known.

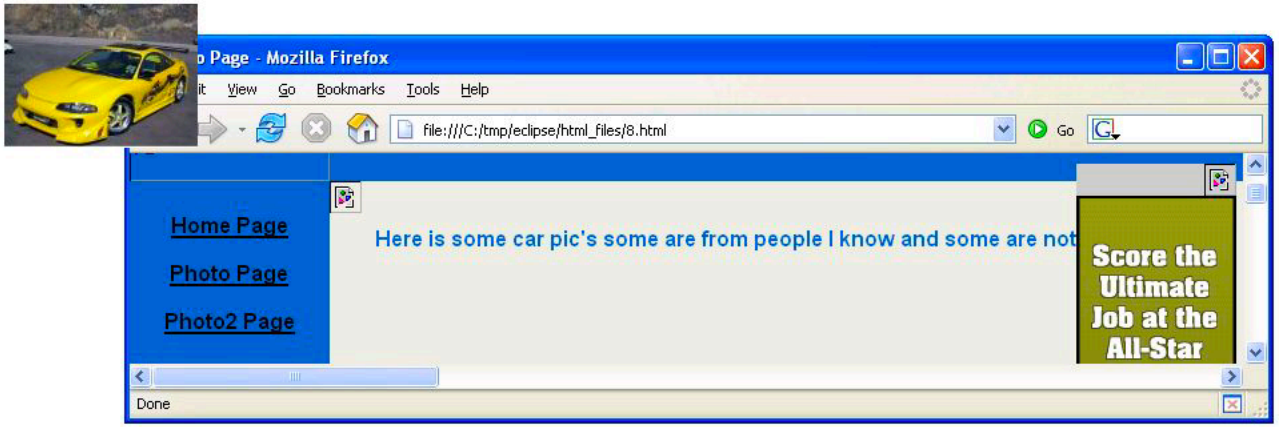
Outline

- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

Co-Training

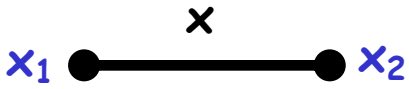
- Two views of an item: image and HTML text





Co-Training

- Instances contain two sufficient sets of features
  - i.e. an instance is x=(x_1,x_2)
  - Each set of features is called a View



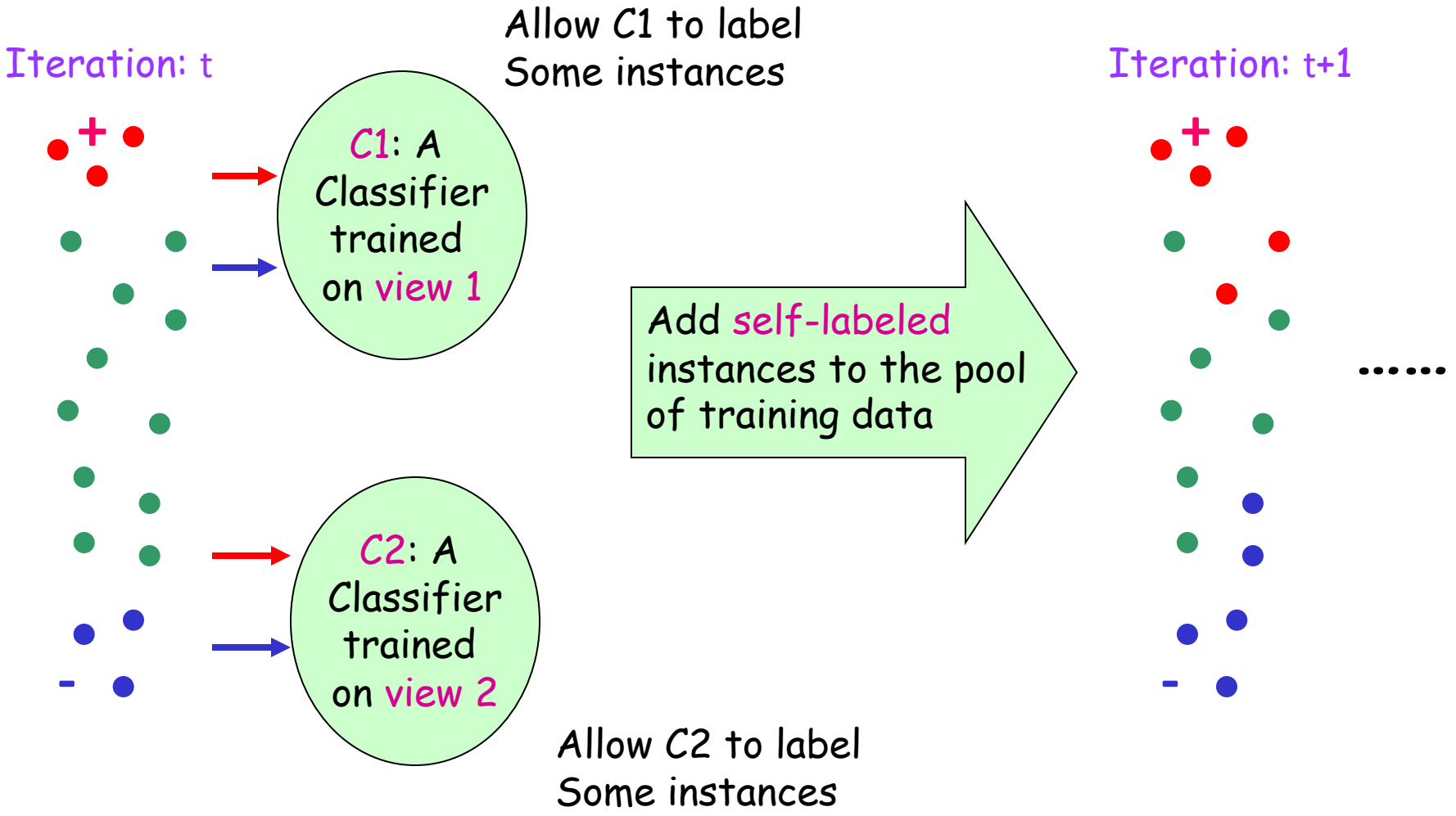
- Two views are independent given the label:

$$P(x_1|x_2, y) = P(x_1|y)$$
$$P(x_2|x_1, y) = P(x_2|y)$$

- Two views are consistent:

$$\exists c_1, c_2 : c^{opt}(x) = c_1(x_1) = c_2(x_2)$$

Co-Training


Iteration: t
Allow C1 to label
Some instances
C1: A Classifier trained on view 1
Add self-labeled instances to the pool of training data
Iteration: t+1
.....
C2: A Classifier trained on view 2
Allow C2 to label
Some instances

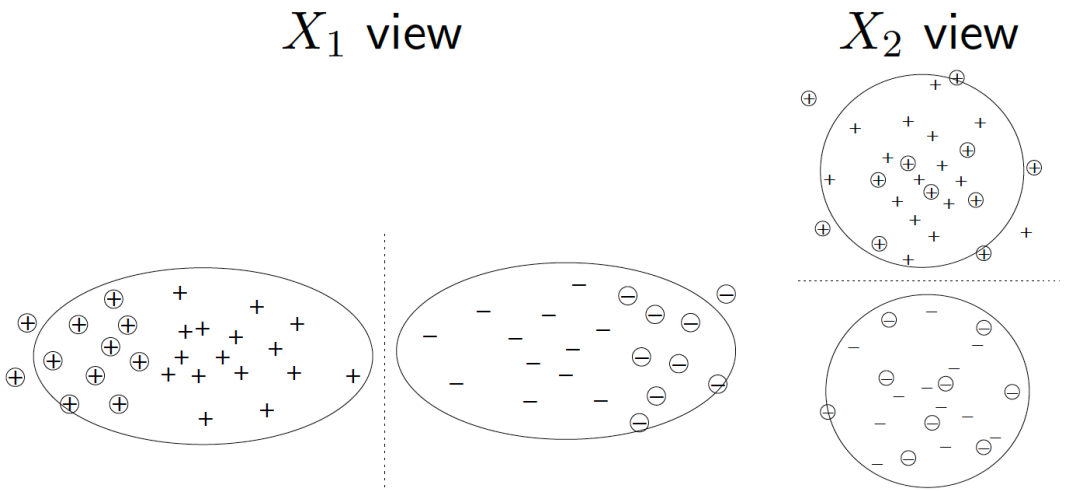
Co-Training Algorithm

Co-training algorithm

1. Train two classifiers: $f^{(1)}$ from $(X_l^{(1)}, Y_l)$, $f^{(2)}$ from $(X_l^{(2)}, Y_l)$.
2. Classify X_u with $f^{(1)}$ and $f^{(2)}$ separately.
3. Add $f^{(1)}$'s k-most-confident $(x, f^{(1)}(x))$ to $f^{(2)}$'s labeled data.
4. Add $f^{(2)}$'s k-most-confident $(x, f^{(2)}(x))$ to $f^{(1)}$'s labeled data.
5. Repeat.

Co-Training Assumptions

- Feature split $x=(x_1,x_2)$ exits
- x_1 or x_2 alone is sufficient to train a good classifier
- x_1 and x_2 are conditionally independent given the class



Pros and cons of co-training

- Pros
  - Simple wrapper method. Applies to almost all existing classifiers
  - Less sensitive to mistakes than self-training
- Cons
  - Natural feature splits may not exist
  - Models using BOTH features should do better

Variants of co-training

- Co-EM: add all, not just top k
  - Each classifier probabilistically label unlabeled set
  - Add (x, y) with weight P(y|x)
- Fake feature split
  - create random, artificial feature split
  - apply co-training
- Multiview: agreement among multiple classifiers
  - no feature split
  - train multiple classifiers of different types
  - classify unlabeled data with all classifiers
  - add majority vote label

Outline

- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

Graph Based Semi-supervised Learning

- Graph based approaches exploit the property of label smoothness
- Idea: Represent each example (labeled/unlabeled) as vertices of some graph, the labels should vary smoothly along the graph
  - Nearby vertices should have similar labels
- This idea is called Graph-based Regularization

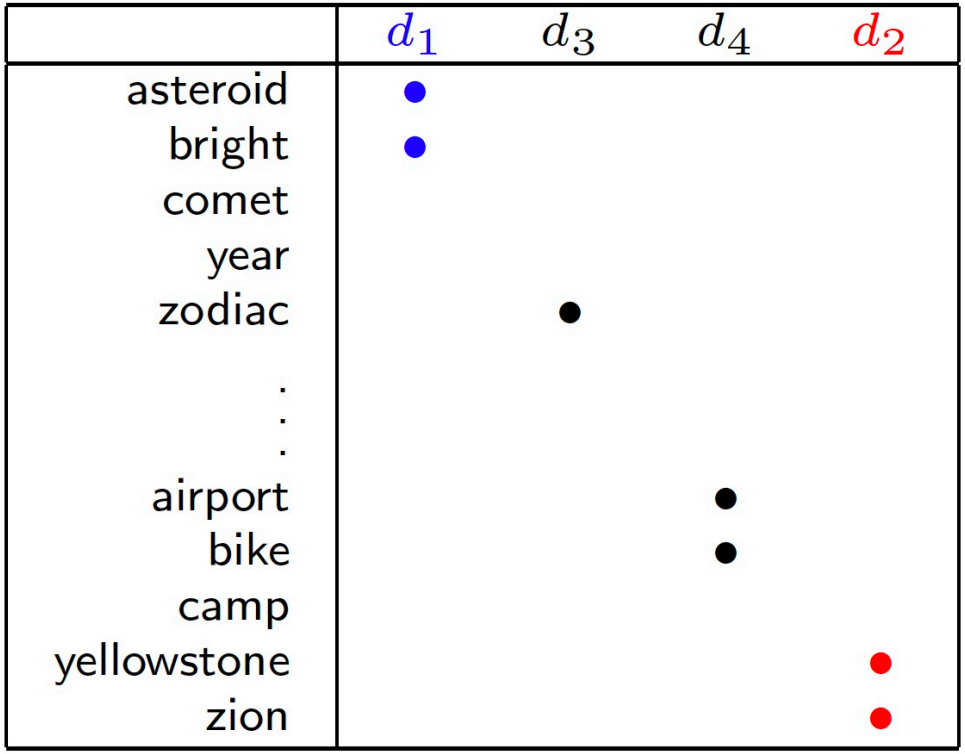
Graph Based Semi-supervised Learning

- Example 1: Text classification
- Similarity measured by content word overlap

	d_1	d_3	d_4	d_2
asteroid	●	●		
bright	●	●		
comet		●		
year				
zodiac				
.				
.				
.				
airport				
bike				
camp			●	
yellowstone			●	●
zion				●

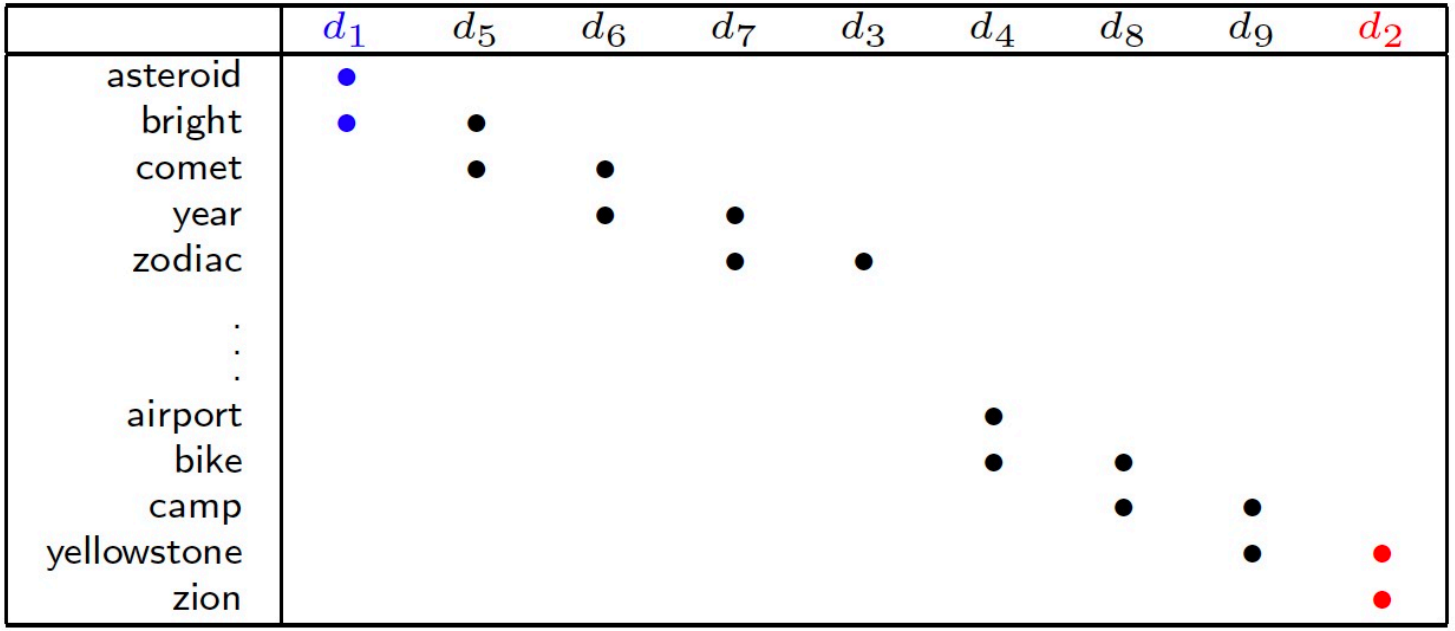
Graph Based Semi-supervised Learning

- Example 1: Text classification
- When labeled data alone fails: No overlapping words!



Graph Based Semi-supervised Learning

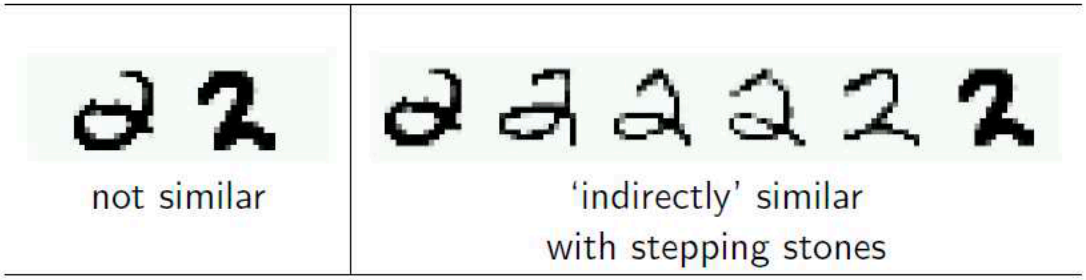
- Example 1: Text classification
- Unlabeled data as stepping stones: Labels "propagate" via similar unlabeled articles.



Graph Based Semi-supervised Learning

- Example 2: Handwritten digit classification

Handwritten digits recognition with pixel-wise Euclidean distance

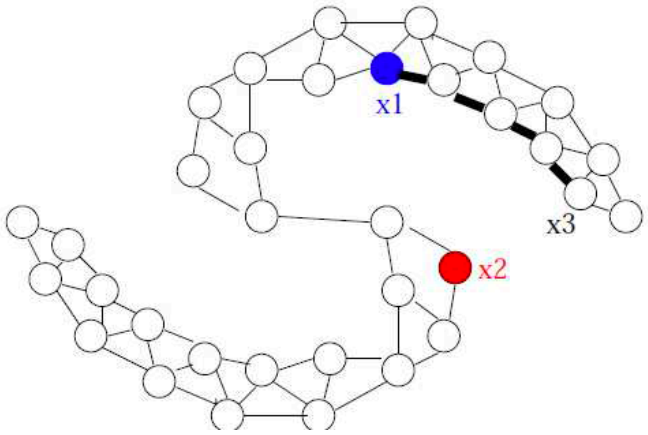


not similar

'indirectly' similar
with stepping stones

Graph Based Semi-supervised Learning

- Nodes: $X_l \cup X_u$
- Edges: similarity weights computed from features, e.g.,
  - k-nearest-neighbor graph, unweighted (0, 1 weights)
  - fully connected graph, weight decays with distance $w = \exp\left(-\|x_i - x_j\|^2/\sigma^2\right)$
  - ϵ-radius graph
- Assumption Instances connected by heavy edge tend to have the same label.



Graph Regularization

- Assume the predictions on the entire data L and U to be defined by function f
- Graph regularization assumes that the function f is smooth
  - Similar examples i and j should have similar predictions f_i and f_j
- Graph regularization optimizes the following objective:

$$\min_f \sum_{i \in \mathcal{L}} (y_i - f_i)^2 + \lambda \sum_{i,j \in \mathcal{L},\mathcal{U}} w_{ij} (f_i - f_j)^2$$

First term: minimize the loss on the labeled data

Second term: ensures smoothness of labels of labeled and unlabeled data

Some graph-based algorithms

- mincut
- harmonic
- local and global consistency
- manifold regularization

Graph-based semi-supervised learning

- Assumption: A graph is given on the labeled and unlabeled data. Instances connected by heavy edge tend to have the same label.

- Pros:
  - Clear mathematical framework
  - Performance is strong if the graph happens to fit the task
  - The (pseudo) inverse of the Laplacian can be viewed as a kernel matrix
  - Can be extended to directed graphs

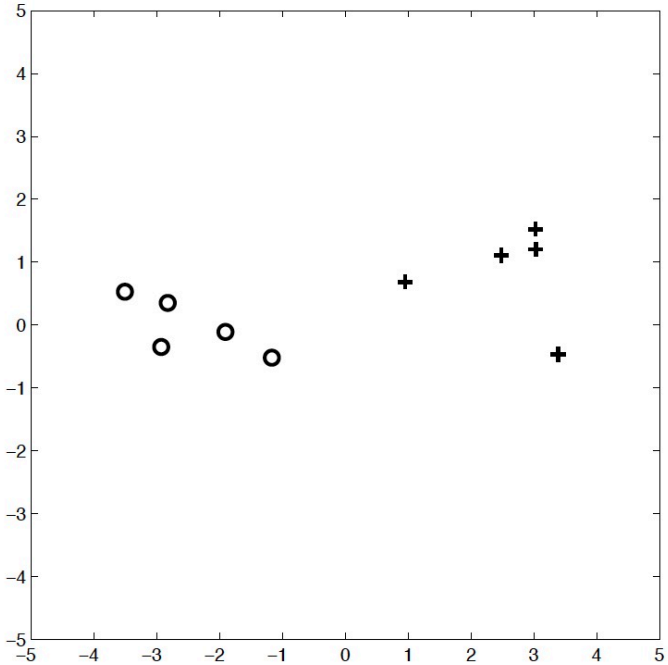
- Cons:
  - Performance is bad if the graph is bad
  - Sensitive to graph structure and edge weights

Outline

- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

A simple example of generative models

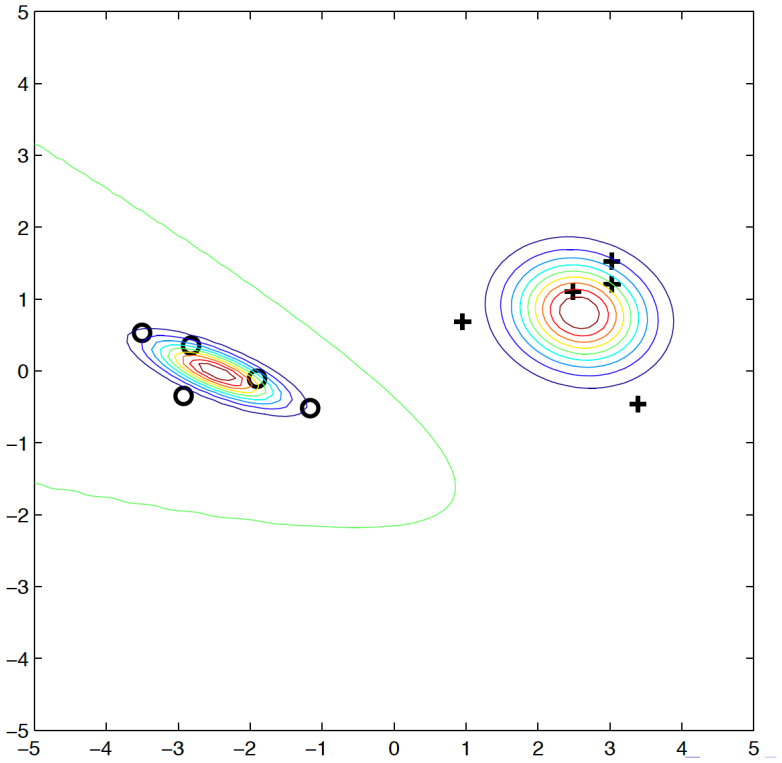
- Labeled data (X_L, Y_L):



Assuming each class has a Gaussian distribution, what is the decision boundary? （LDA or QDA）

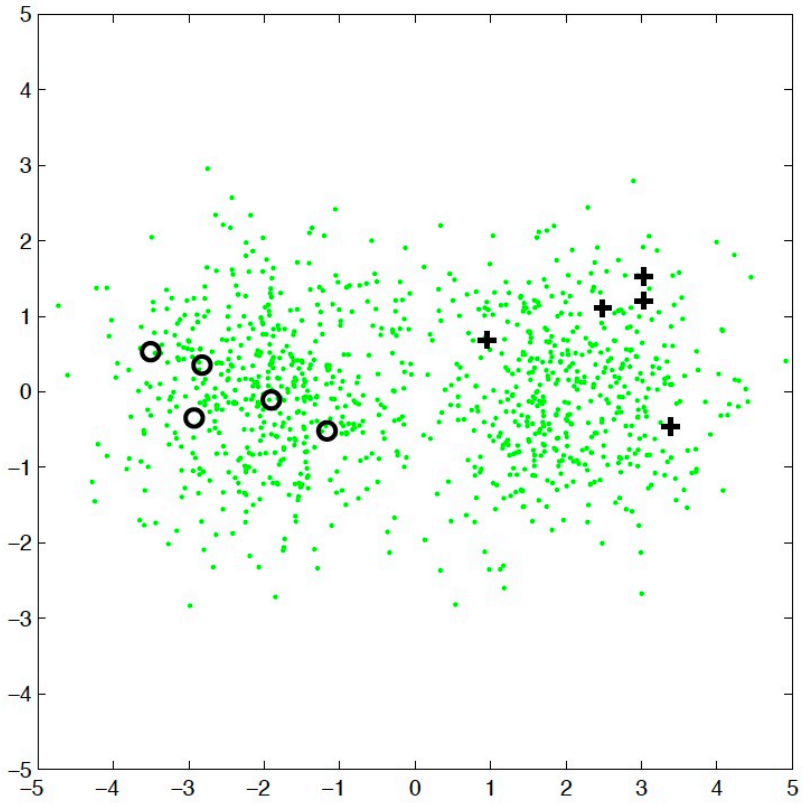
A simple example of generative models

- The most likely model and its decision boundary:



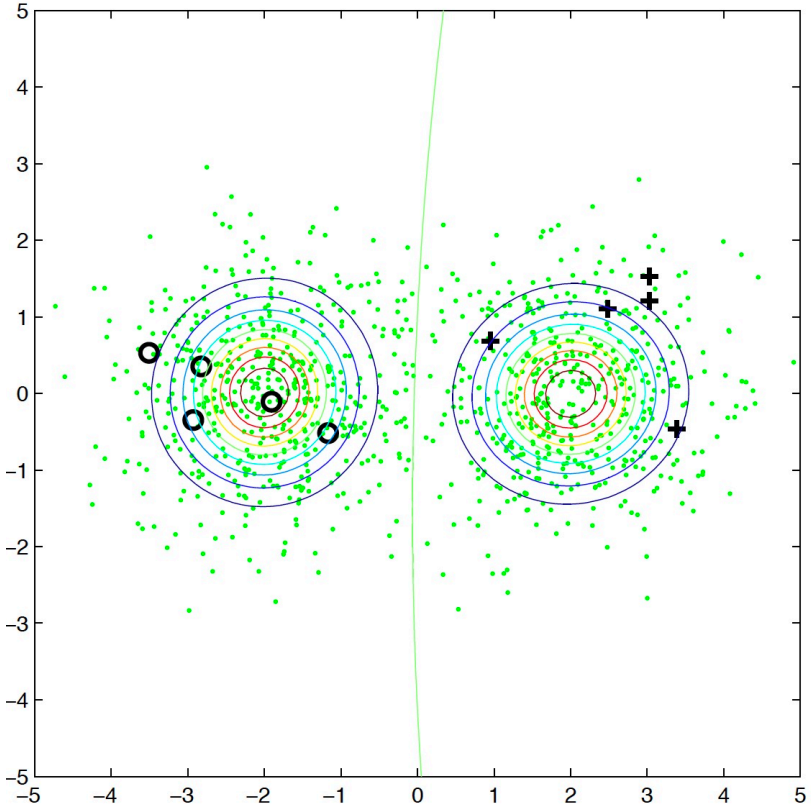
A simple example of generative models

- Adding unlabeled data:



A simple example of generative models

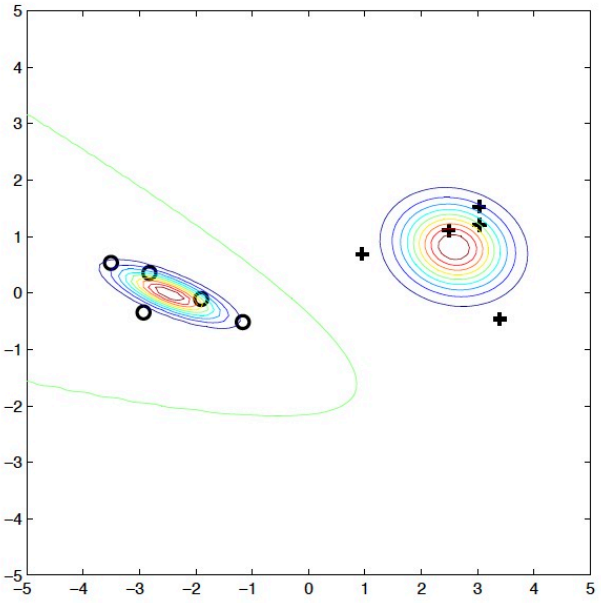
- With unlabeled data, the most likely model and its decision boundary:



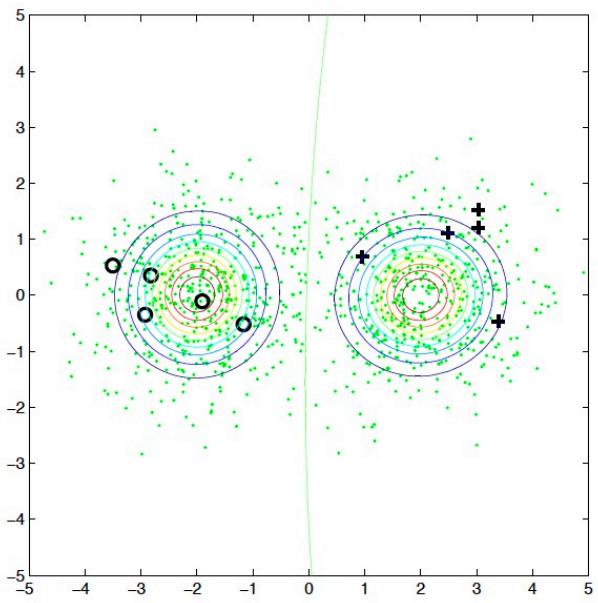
A simple example of generative models

- They are different because they maximize different quantities:

$p(X_l, Y_l|\theta)$



$p(X_l, Y_l, X_u|\theta)$



Generative models for SSL

- Assumption: The full generative model p(X, Y $|\theta$).

Generative model for semi-supervised learning:

- quantity of interest: $p(X_l, Y_l, X_u|\theta) = \sum_{Y_u} p(X_l, Y_l, X_u, Y_u|\theta)$
- find the maximum likelihood estimate (MLE) of θ, the maximum a posteriori (MAP) estimate, or be Bayesian

Examples of some generative models for SSL

- Mixture of Gaussian distributions (GMM)
  - Image classification
  - the EM algorithm
- Mixture of multinomial distributions (Naïve Bayes)
  - Text categorization
  - the EM algorithm
- Hidden Markov Models (HMM)
  - speech recognition
  - Baum-Welch algorithm

Case study:GMM

For simplicity, consider binary classification with GMM using MLE.

- labeled data only
  - $\log p(X_l, Y_l|\theta) = \sum_{i=1}^{l} \log p(y_i|\theta)p(x_i|y_i, \theta)$
  - MLE for θ trivial (frequency, sample mean, sample covariance)
- labeled and unlabeled data

  $$\log p(X_l, Y_l, X_u|\theta) = \sum_{i=1}^{l} \log p(y_i|\theta)p(x_i|y_i, \theta) + \sum_{i=l+1}^{l+u} \log \left(\sum_{y=1}^{2} p(y|\theta)p(x_i|y, \theta) \right)$$

  - MLE harder (hidden variables)
  - The Expectation-Maximization (EM) algorithm is one method to find a local optimum.

The EM algorithm for GMM

1. Start from MLE $\theta = \{w, \mu, \Sigma\}_{1:2}$ on (X_l, Y_l), repeat:
2. The E-step: compute the expected label $p(y|x,\theta) = \frac{p(x,y|\theta)}{\sum_{y'} p(x,y'|\theta)}$ for all $x \in X_u$
   - label $p(y=1|x,\theta)$-fraction of x with class 1
   - label $p(y=2|x,\theta)$-fraction of x with class 2
3. The M-step: update MLE θ with (now labeled) X_u
   - w_c=proportion of class c
   - μ_c=sample mean of class c
   - Σ_c=sample cov of class c

Can be viewed as a special form of self-training.

Pros and Cons of generative models

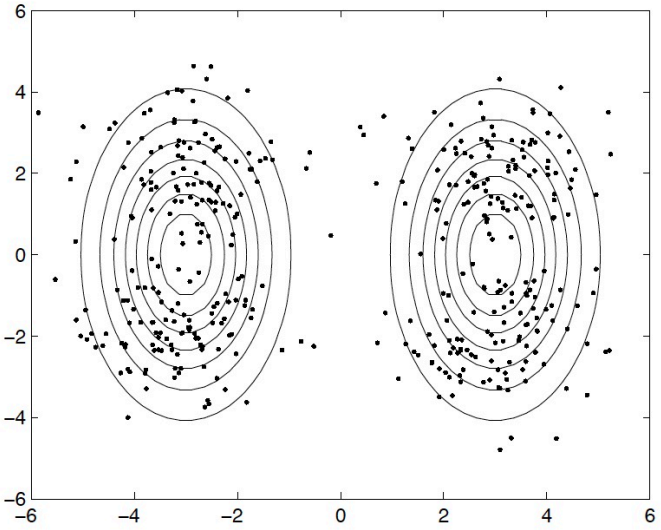
- Pros:
  - Clear, well-studied probabilistic framework
  - Can be extremely effective, if the model is close to correct

- Cons:
  - Often difficult to verify the correctness of the model
  - Model identifiability
  - EM local optima
  - Unlabeled data may hurt if generative model is wrong

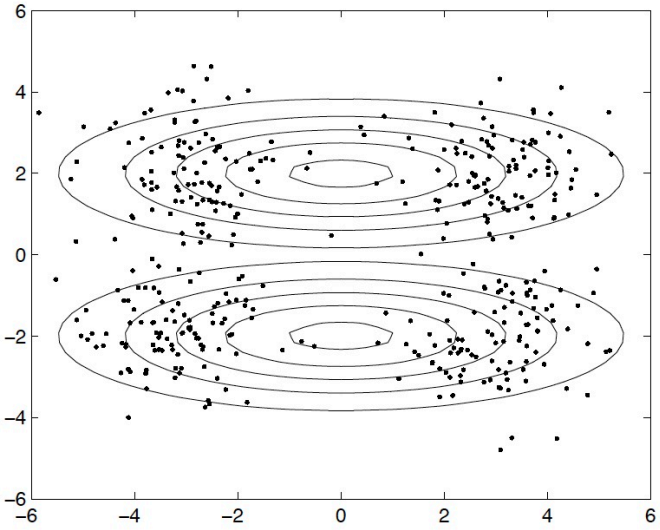
Unlabeled data may hurt Semi-supervised learning

- If the generative model is wrong:

high likelihood
wrong



low likelihood
correct



Heuristics to lessen the danger

- Carefully construct the generative model to reflect the task
  - e.g., multiple Gaussian distributions per class, instead of a single one
- Down-weight the unlabeled data (< 1)

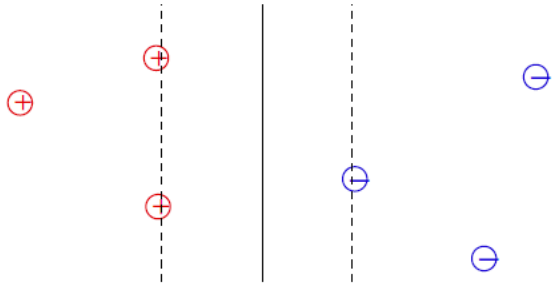
$$\log p(X_l, Y_l, X_u|\theta) = \sum_{i=1}^{l} \log p(y_i|\theta)p(x_i|y_i,\theta) + \lambda \sum_{i=l+1}^{l+u} \log \left(\sum_{y=1}^{2} p(y|\theta)p(x_i|y,\theta) \right)$$

Outline

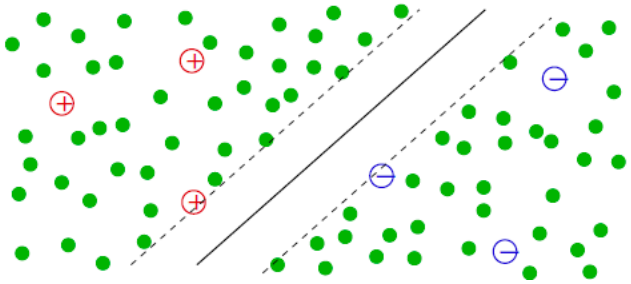
- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

S3VMs

SVMs



Semi-supervised SVMs (S3VMs) = Transductive SVMs (TSVMs)

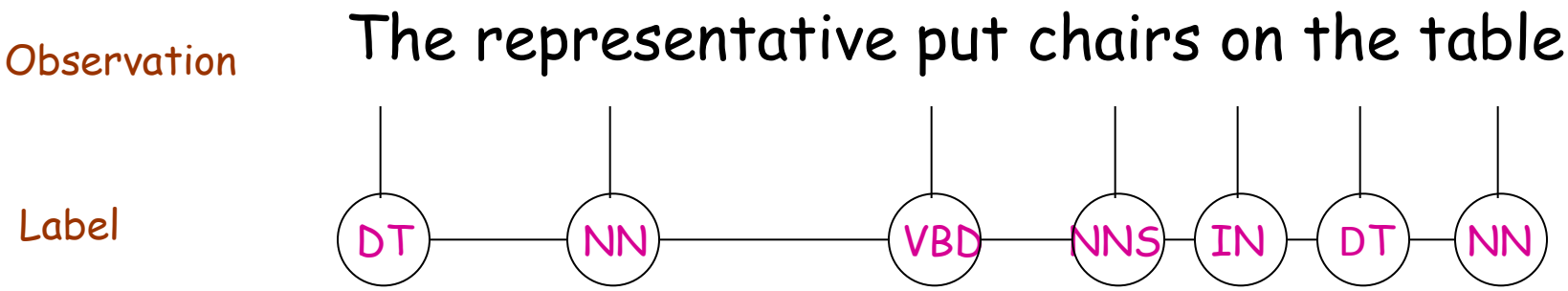


Outline

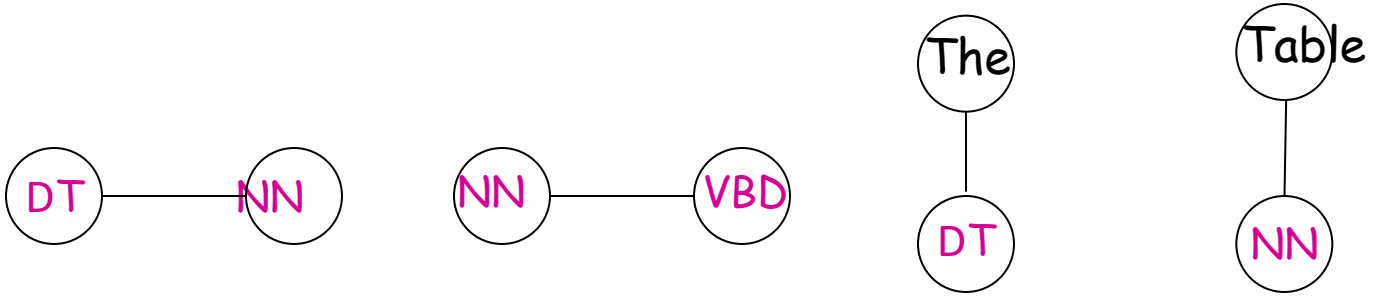
- Introduction to Semi-supervised Learning
- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction
- Summary

Structured Prediction

- Example: Part-of-speech tagging:



- The input is a complex object as well as its label.
  - Input-Output pair (x,y) is composed of simple parts.
  - Example: Label-Label and Obs-Label edges:



Scoring Function

- For a given x, consider the set of all its candidate labelings as Y_x.
  - How to choose the best label from Y_x?
- By the help of a scoring function S(x,y):

$$y = \arg\max_{y' \in Y_x} S(x, y')$$

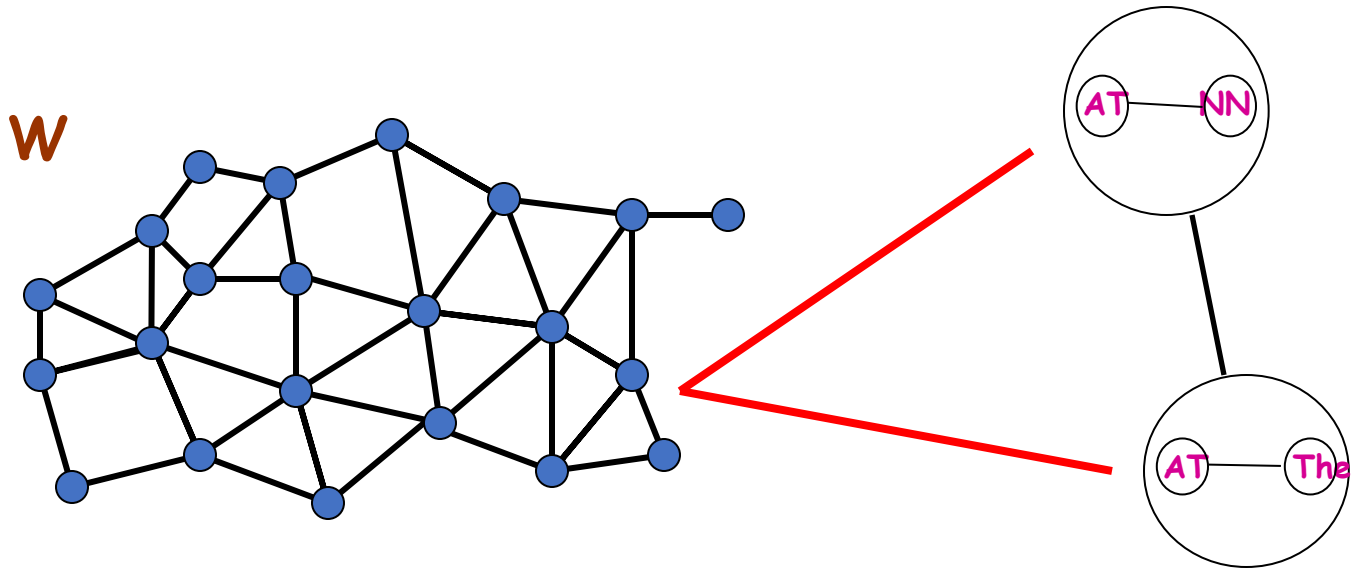
  - Assume S(x,y) can be written as the sum of scores for each simple part:

$$S(x, y) = \sum_{r \in R(x,y)} f(r)$$

    - R(x,y) the set of si
  - How to find f(.)?

Manifold of "simple parts"

(Altun et al 2005)



- Construct d-nearest neighbor graph on all parts seen in the sample.
  - For unlabeled data, put all parts for each candidate.
- Belief: f(.) is smooth on this graph (manifold).

SSL for Structured Labels

- The final maximization problem:

Data dependent regularization

$$\arg\min_{f\in\mathcal{H}} \sum_i loss(f(x_i), y_i) + \lambda_k ||f||_k + \lambda_I \sum_{r,r'} W_{r,r'}(f(r) - f(r'))^2$$

Fitness to Labeled data

Function complexity: Prior belief

Smoothness term: Unlabeled data

- The Representer theorem:

$$f(.) = \sum_{r\in R(S)} \alpha_r k(r, .)$$

  - R(S) is all the unlabeled instances in the sample.
  - Note that f(.) is related to .

$$\alpha = (\alpha_1, ..., \alpha_{R(S)})$$

Modified problem

- Plugging the form of the best function in the optimization problem gives:

$$\arg\min_\alpha \sum_i loss(f_\alpha(x_i), y_i) + \alpha^T \cdot Q \cdot \alpha$$

  - Where Q is a constant matrix.
  - By introducing slack variables ε_i :

$$\arg\min_\alpha \sum_i \varepsilon_i + \alpha^T \cdot Q \cdot \alpha$$

Subject to

$$\forall i, loss(f_\alpha(x_i), y_i) \leq \varepsilon_i$$

Modified problem (cont'd)

- Loss function: $\arg\min_\alpha \sum_i \varepsilon_i + \alpha^T \cdot Q \cdot \alpha$

  Subject to $\forall i, loss(f_\alpha(x_i), y_i) \leq \varepsilon_i$

  - SVM:

$$loss(f_\alpha(x), y) = \max_{y' \in Y_x} \Delta(x, y, y') + S_\alpha(x, y') - S_\alpha(x, y)$$

Hamming distance

  - CRF:

$$loss(f_\alpha(x), y) = -S_\alpha(x, y) + \log \sum_{y' \in Y_x} \left(\exp S_\alpha(x, y') \right)$$

  - Note that an α vector gives the f(.) which in turn gives the scoring function S(x,y). We may write $S_\alpha(x,y)$.

Summary

- Semi-supervised Learning Algorithms
  - Expectation Maximization
  - Self Learning
  - Co-Training
  - Graph-based Methods
  - Generative Models
  - S3VMs
- SSL for Structured Prediction

References

- http://pages.cs.wisc.edu/~jerryzhu/pub/sslicml07.pdf
- http://www.cs.cmu.edu/~10701/slides/17_SSL.pdf
- https://www.cs.cmu.edu/~epxing/Class/10701/slides/semi15.pdf
- https://www.cs.utah.edu/~piyush/teaching/8-11-slides.pdf
- https://www.cs.rutgers.edu/~pa336/mlS16/ml-sp16-lec22.pdf
- http://www.cs.cmu.edu/~ninamf/courses/601sp15/slides/18_svm-ssl_03-25-2015.pdf
- https://www.molgen.mpg.de/3659531/MITPress--SemiSupervised-Learning.pdf