Machine Learning
Semisupervised Learning

Paul Swoboda

Lecture 17, 17.12.2018
What is semi-supervised learning (SSL)? What is transduction?

The cluster/manifold assumption

Graph-based SSL using regularized least squares
  1. Interpretation in terms of label propagation
  2. Interpretation in terms of a data-dependent kernel

Experiments
Why semi-supervised learning?

- Human labels can be expensive and time consuming,
- There is a lot of unlabeled data around us e.g. images and text on the web. The knowledge about the unlabeled data “should” be helpful to build better classifiers,

**Distinction from weakly supervised learning**

- one uses weaker information than full supervision e.g. instead of pixel-wise accurate object labels you just have bounding box containing the object.
What is semi-supervised learning?

Input space $X$, Output: $\{-1, 1\}$ (binary classification):

- a **small** set $L$ of labeled data $(X_l, Y_l)$,
- a **large** set $U$ of unlabeled data $X_u$.
- notation: $n = l + u$, total number of data points. $T$ denotes the set of all points.

e.g. a small number of labeled images and a huge number of unlabeled images from the internet.

**Definition:**

- **Transduction:** Prediction of the labels $Y_u$ of the unlabeled data $X_u$,
- **SSL:** Construction of a classifier $f : X \rightarrow \{-1, 1\}$ on the whole input space (using the unlabeled data).
Is it always helpful?

No!

Because:
- in order to deal with a small amount of labeled data we have to make strong assumptions about the underlying joint probability measure $P(X, Y)$ e.g. a relation of $P(X)$ and $P(Y|X)$.

But:
- empirical success of SSL methods shows that unlabeled data can improve performance.
- nice application of SSL (Levin et al. 2006) in user-guided image segmentation (foreground / background).
Matting

Left: Input Image with user labels, Right: Image segmentation
The obvious one - **Self Training**

- use labeled data to build classifier,
- the unlabeled points on which the classifier is most “confident” are added to the label set,
- repeat until all points are labeled.

**Problem:**

- Wrongly assigned labels in the beginning can spoil the whole performance.
- How should we measure the confidence in the labels?
Other more principled approaches to SSL:

- Co-Training,
- Transductive SVM,
- Harmonic function,
  Regularized least squares with the graph Laplacian,
  Label Propagation
  $\Rightarrow$ Different aspects of the same graph based method
- Low Density Separation

$\Rightarrow$ in this lecture we treat the graph-based methods using Laplacian regularization.
$\Rightarrow$ graph-based methods are very flexible (can be applied on any kind of data).
**Cluster assumption:** points which can be connected via (many) paths through high-density regions are likely to have the same label.
Manifold assumption: each class lies on a separate manifold.
Cluster/Manifold assumption: points which can be connected via a path through high density regions on the data manifold are likely to have the same label.

⇒ Use regularizer which prefers functions which vary smoothly along the manifold and do not vary in high density regions.
**Problem:** We have only (a lot of) unlabeled and some labeled points and no information about the density and the manifold.
**Approach:** Use a graph to approximate the manifold (and density).
How to build such graphs?

**Neighborhood graphs:**
Given similarity $s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ or dissimilarity measure $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Denote by $\text{kNN}(X_i)$ the $k$ most similar or least dissimilar points.

- **k-nearest neighbor graphs:** connect points $X_i$ to $X_j$ if
  - $X_j \in \text{knn}(X_i) \Rightarrow$ kNN-graph (directed)
  - $X_i \in \text{kNN}(X_j)$ and $X_j \in \text{kNN}(X_i)$ (mutual) $\Rightarrow$ mutual kNN-graph.
  - $X_i \in \text{kNN}(X_j)$ or $X_j \in \text{kNN}(X_i) \Rightarrow$ symmetric kNN-graph.

The symmetric and mutual kNN-graph are undirected.

- **epsilon-graphs:** connect points $X_i$ and $X_j$ if
  - dissimilarity: $d(X_i, X_j) \leq \varepsilon$,
  - similarity: $s(X_i, X_j) \geq 1 - \varepsilon$,

  Assumption: $\max_{x,y} s(x, y) = \max_x s(x, x) = 1$.

The epsilon-graph is undirected.
How to build such graphs?

Weighted neighborhood graph:

- Gaussian weights (single scale):
  \[ w(X_i, X_j) = e^{-\frac{d(X_i, X_j)^2}{\sigma^2}}, \]
  where \( \sigma^2 = \frac{1}{n(n-1)} \sum_{i \neq j} d(X_i, X_j)^2 \) or chosen by cross-validation.

- Gaussian weights (adaptive scaling)
  \[ w(X_i, X_j) = e^{-\lambda \frac{d(X_i, X_j)^2}{\sigma^2_k}}, \]
  where e.g. \( \sigma^2_k = \frac{1}{2} (\text{dist}_k(X_i) + \text{dist}_k(X_j)) \) and \( \text{dist}_k(X_i) \) is the distance of \( X_i \) to its \( k \)-nearest neighbor and \( \lambda \) is either one or chosen by cross-validation.

- Other user-defined measures...
Define a regularization functional which penalizes functions which vary in high-density regions.

\[
\langle f, \Delta f \rangle = \langle f, (D - W)f \rangle = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2,
\]

where \(D = d_i \delta_{ij}\) with \(d_i = \sum_{j=1}^{n} w_{ij}\) and the graph Laplacian is defined as \(\Delta = D - W\).

For the \(\epsilon\)-neighborhood graph one can show (Bousquet, Chapelle and H.(2003), H.(2006)) under certain technical conditions that as \(\epsilon \to 0\) and \(n\epsilon^m \to \infty\) (\(m\) is dimension of the manifold).

\[
\lim_{n \to \infty} \frac{1}{n\epsilon^{m+2}} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 \sim \int_{M} \|\nabla f\|^2 p(x)^2 \, dx
\]
Regularized least squares

Transductive Learning via regularized least squares:


\[
\arg \min_{f \in \mathbb{R}^n, \ f_L = Y_L} \sum_{i,j \in T} w_{ij} (f_i - f_j)^2 .
\]

Belkin and Niyogi (2003):

\[
\arg \min_{f \in \mathbb{R}^n} \sum_{i \in L} (y_i - f_i)^2 + \frac{\lambda}{2} \sum_{i,j \in T} w_{ij} (f_i - f_j)^2 .
\]


\[
\arg \min_{f \in \mathbb{R}^n} \sum_{i \in T} (y_i - f_i)^2 + \frac{\lambda}{2} \sum_{i,j \in T} w_{ij} \left( \frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2,
\]

where \( y_i = 0 \) if \( i \in U \).
Regularized least squares

\[
\arg \min_{f \in \mathbb{R}^n} \sum_{i \in T} (y_i - f_i)^2 + \lambda \sum_{i,j \in T} w_{ij} \left( \frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2,
\]

where \( y_i = 0 \) if \( i \in U \). Note that

\[
f^T (\mathbb{I} - D^{-1/2} WD^{-1/2}) f = \frac{1}{2} \sum_{i,j \in T} w_{ij} \left( \frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2.
\]

The solution \( f^* \) can be found as:

\[
f^* = \left( \mathbb{I} + \lambda (\mathbb{I} - D^{-1/2} WD^{-1/2}) \right)^{-1} Y
\]

or with \( S = D^{-1/2} WD^{-1/2} \) and \( \alpha = \frac{\lambda}{1+\lambda} \) (0 < \( \alpha < 1 \)),

\[
f^* = \frac{1}{1+\lambda} \left[ \mathbb{I} - \frac{\lambda}{1+\lambda} S \right]^{-1} Y = (1 - \alpha)[\mathbb{I} - \alpha S]^{-1} Y.
\]
Interpretation of the solution $f^*$ in terms of label propagation:

$$f^* = (1 - \alpha) \left[ I - \alpha S \right]^{-1} Y$$

One can show $\left[ I - \alpha S \right]^{-1} = \sum_{r=0}^{\infty} \alpha^r S^r$ if $|\alpha| \|S\| < 1$.

Solution $f^*$ can be interpreted as the limit $f^* = \lim_{t \to \infty} f_t$ of the iterative scheme $f_t$, typically $f_0 = Y$,

$$f_{t+1} = \alpha S f_t + (1 - \alpha) Y \quad \Rightarrow \quad f_{t+1} = \alpha^t S^t f_0 + (1 - \alpha) \sum_{r=0}^{t} (\alpha S)^r Y,$$

where $\lim_{t \to \infty} \alpha^t S^t f_0 = 0$. 

Paul Swoboda (Lecture 17, 17.12.2018)
Random walks on a graph

Given a weighted, undirected graph with \( n \) vertices we define the matrix \( P \),

\[
P = D^{-1} W,
\]

\( P \) is a stochastic matrix:
- \( P \) is a \( n \times n \)-matrix,
- \( P_{ij} \geq 0, \ \forall \ 1 \leq i, j \leq n, \)
- \( \sum_{j=1}^{n} P_{ij} = 1. \)

**Interpretation:**
\( P_{ij} \) is the probability to go to vertex \( j \) when the current vertex is \( i \).

\[
P_{ij} = P(X_{t+1} = j \mid X_{t} = i).
\]
Random walks on a graph II

**Probability measure** $p_i(t) = P(X_t = i)$ on the graph at time $t$:

$$\sum_{i=1}^{n} p_i(t) = 1.$$  

**One step of the random walk:**

$$P(X_{t+1} = j) = p_j(t+1) = \sum_{i=1}^{n} p_i(t)P_{ij} = \sum_{i=1}^{n} P(X_{t+1} = j \mid X_t = i)P(X_t = i).$$

This is again a probability measure,

$$\sum_{j=1}^{n} p_j(t+1) = \sum_{j=1}^{n} \sum_{i=1}^{n} p_i(t)P_{ij} = \sum_{i=1}^{n} p_i(t) \sum_{j=1}^{n} P_{ij}$$

$$= \sum_{i=1}^{n} p_i(t) = 1.$$

This is a **Markov stochastic process** since the probability to do the next step just depends on the current probability measure on the graph and not on previous states.
Stationary distribution $\pi$: A probability distribution $\pi$ is stationary if

$$\pi_j = \sum_{i=1}^{n} \pi_i P_{ij}.$$ 

Results:

- For an undirected graph there exists a not necessarily unique stationary distribution,

$$\pi_i = \frac{d_i}{d}, \text{ where } d = \sum_{i=1}^{n} d_i,$$

and $d_i = \sum_{j=1}^{n} w_{ij}$ (degree function).

- For an undirected graph the random walk converges to the stationary distribution if the graph is connected and non-bipartite. In this case the stationary distribution is unique.
Relation to random walks

The solution is given by

\[
f^* = (1 - \alpha) \left[ I - \alpha S \right]^{-1} Y = \frac{\sum_{r=0}^{\infty} \alpha^r S^r}{\sum_{r=0}^{\infty} \alpha^r} Y
\]

Using \( S = D^{-1/2} WD^{-1/2} \) we get with the stochastic matrix \( P = D^{-1} W \),

\[
S = D^{1/2} PD^{-1/2} \quad \text{and} \quad S^r = D^{1/2} P^r D^{-1/2}.
\]

Plugging the expression for \( S^r \) into the equation for the solution \( f \),

\[
f^* = D^{1/2} \frac{\sum_{r=0}^{\infty} \alpha^r P^r}{\sum_{r=0}^{\infty} \alpha^r} D^{-1/2} Y
\]
Harmonic function

Semi-supervised learning as finding a harmonic function with boundary conditions:

\[
\arg \min_{f \in \mathbb{R}^n, f_L = Y_L} \sum_{i,j \in T} w_{ij} (f_i - f_j)^2 = \langle f, \Delta f \rangle.
\]

The solution can be found as:

\[
f_L = Y_L, \quad \Delta f = 0.
\]

This leads to

\[
f_U = (D_{UU} - W_{UU})^{-1} W_{UL} Y_L = (I_{UU} - P_{UU})^{-1} P_{UL} Y_L.
\]

where \( P = D^{-1} W \) is the stochastic matrix of the random walk associated to the undirected graph.
Interpretation of the solution in terms of a random walk:

\[
f_U = (D_{UU} - W_{UU})^{-1} W_{UL} Y_L = (I_{UU} - P_{UU})^{-1} P_{UL} Y_L.
\]

We will use \((I_{UU} - P_{UU})^{-1} = \sum_{s=0}^{\infty} (P_{UU}^s)\). Then we get for a point \(i \in U\),

\[
(f_U)_i = \sum_{k \in L} \sum_{j \in U} \sum_{s=0}^{\infty} (I_{UU} - P_{UU})_{ij}^{-1} (P_{UL})_{jk} (Y_L)_k
\]

\[
= \sum_{k \in L} \sum_{j \in U} \sum_{s=0}^{\infty} (P_{UU}^s)_{ij} (P_{UL})_{jk} (Y_L)_k
\]

\[
= \sum_{k \in L_+} \sum_{j \in U} \sum_{s=0}^{\infty} (P_{UU}^s)_{ij} (P_{UL})_{jk} - \sum_{k \in L_-} \sum_{j \in U} \sum_{s=0}^{\infty} (P_{UU}^s)_{ij} (P_{UL})_{jk}
\]

\[
= P(\text{hits positive points} \mid \text{started in } i) - P(\text{hits negative points} \mid \text{started in } i).
\]
Do you trust all your labels?

Relaxed version of the approach of Belkin et al:

\[
\arg\min_{f \in \mathbb{R}^n} \sum_{i \in L} (y_i - f_i)^2 + \frac{\lambda}{2} \sum_{i,j \in T} w_{ij}(f_i - f_j)^2,
\]

where \(\lambda > 0\) is the regularization parameter.

Extremal equations with \(\Delta = D - W\):

\[
(\mathbb{1} + \lambda \Delta) f = Y, \text{ on the labeled points},
\]
\[
\lambda \Delta f = 0, \text{ on the unlabeled points}.
\]

With \(Y_i = 0\) if \(i\)-th point and \((\mathbb{1}_L)_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and } i \text{ is labeled}, \\ 0 & \text{if } i \text{ is unlabeled}. \end{cases}\),

\[
(\mathbb{1}_L + \lambda \Delta) f = Y.
\]
All approaches can also be interpreted as kernel machines. Let $\Delta^\dagger$ be the pseudo-inverse of the graph Laplacian. Then

$$K = \Delta^\dagger,$$

is a (data-dependent) kernel on $n$ points. Let $f_i = \sum_{j=1}^{n} \alpha_j k(x_i, x_j)$. Then

$$f^\top \Delta f = \alpha^\top K^T \Delta K \alpha = \alpha^\top K \alpha.$$

The structure of the graph influences significantly the result. For high-dimensional data one can improve the performance by using “Manifold Denoising” as a preprocessing method.
Experiments

- DemoSSL
- Graph structure has large influence on result (mainly unexplored area in machine learning),
- Result “can” be pretty stable with respect to the location of the labeled points,
- If cluster assumption is not valid then SSL does not help (in the worst case it yields even a worse performance).
- for a few labeled points (say 10 times the number of classes) cross validation works already pretty well.