Midway Report: Semi-Supervised Learning of Pen-Based OCR.

Joshua Brakensiek, Jacob Imola, Sidhanth Mohanty

1 Background

Handwriting recognition is a classic machine learning problem. Generally, an image of a digit is inputted to an algorithm and classified. Instead of taking this approach, we used two data sets, "Pen-Based Recognition of Handwritten Digits Data Set" [AA98] and "UJI Pen Characters (Version 2) Data Set" [LPM+08] from the UCI Machine Learning Repository [Lic13], which consist of pixel coordinates that the writers' pens took at certain time intervals. Each data set has over 10,000 data points. The first data set consists entirely of drawings of the digits $0, \dots, 9$, while the second data set uses a much richer variety of characters. We seek to explore if adding the extra information of time while sacrificing some detail from the shape of the digits can produce a viable semi-supervised model. We implement our learning algorithms in Python 2.x, using on the NumPy, SciPy, and Scikit-learn libraries (and possibly other libraries as we see fit). The models we try are Transductive Support Vector Machines and graph-based kernel methods and regularization. Throughout this paper, let $X = \{x_1, x_2, \dots, x_k\}$ be the labeled training data, $Y = \{y_1, y_2, \dots, y_k\}$ be the corresponding labels, $X^* = \{x_1^*, x_2^*, \dots, x_n^*\}$ be the unlabeled training data, and $Y^* = \{y_1^*, y_2^*, \dots, y_n^*\}$ be variables representing the unlabeled training data's labels. Now, we will give a description of the models:

Recall that an SVM reduces to the following primal problem:

$$\arg \min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{k} \zeta_i \tag{1}$$

$$\text{subject to} \quad \forall_{i=1}^{k} : y_i(wx_i + b) \geq 1 - \zeta_i$$
$$\forall_{i=1}^{k} : \zeta_i > 0$$

A TSVM's primal problem allows the unlabeled data to take on any label, and hence reduces to a similar problem [Joa99a]:

$$\arg \min_{w,b,\zeta,\zeta^*,Y^*} \frac{1}{2} w^T w + C \sum_{i=1}^{k} \zeta_i + C^* \sum_{i=1}^{n} \zeta_i^* \tag{2}$$

$$\text{subject to} \quad \forall_{i=1}^{k} : y_i(wx_i + b) \geq 1 - \zeta_i$$
$$\forall_{i=1}^{n} : y_i^*(wx_i^* + b) \geq 1 - \zeta_i^*$$
$$\forall_{i=1}^{k} : \zeta_i > 0$$
$$\forall_{i=1}^{n} : \zeta_i^* > 0$$

Here, C and C^* are parameters that control the type of fit we get.

2   Related Work

The TSVM algorithm that we implemented requires that in the unlabeled data, the number of data-points classified is constrained to be exactly equal to some parameter num_+. [J$^+$] shows that this special kind of TSVM is equivalent to solving a s-t min cut problem, where the size of the partitions are fixed beforehand, and explores an algorithm called spectral graph transducer, that does not require the sizes of the partitions to be fixed beforehand.

Closely related to the graph interpretation of semi-supervised SVMs are methods intersecting at spectral graph theory and semi-supervised learning like Laplacian SVMs and Manifold regularization (see [BNS]). This method assume a distribution P over $X \times \mathbb{R}$ from which labeled examples (x, y) are drawn, and assumes that the marginal distribution of P on X, denoted P_X, has the structure of a Riemannian manifold and uses this model as a basis for a family of methods.

Another related semi-supervised learning technique that could be applied to TSVMs is co-training, introduced in [BM98]. This technique splits each unlabeled example into multiple 'views' (an example as specified in [BM98] would be how webpages can be partitioned into the content on that page and content on other pages that link to it), and two learning algorithms are trained on different views. Then, the labels provided by the algorithms on previously unlabeled examples are used to increase the labeled data available to the other algorithm to train on.

[Joa99b] describes an algorithm called SVM^{light}, which is faster than traditional algorithms for training semi-supervised SVM that scales well with a large amount of data, but sacrifices on accuracy.

3   Methods

3.1   TSVM

One TSVM method we use is an SVM with local search which was first described in [Joa99a]. We use a more generalized version of an SVM:

$$\arg \min_{w,b,\zeta,\zeta^*} \frac{1}{2} w^T w + C \sum_{i=1}^{k} \zeta_i + C_+^* \sum_{i=1}^{n} \zeta_i^* [y_i^* == 1] + C_-^* \sum_{i=1}^{n} \zeta_i^* [y_i^* == -1] \qquad (3)$$

$$\begin{aligned}
\text{subject to} \quad & \forall_{i=1}^{k} : y_i(wx_i + b) \geq 1 - \zeta_i \\
& \forall_{i=1}^{n} : y_i^*(wx_i^* + b) \geq 1 - \zeta_i^* \\
& \forall_{i=1}^{k} : \zeta_i > 0 \\
& \forall_{i=1}^{n} : \zeta_i^* > 0
\end{aligned}$$

Let's denote an SVM with such an objective function as $SVM(C, C_+^*, C_-^*, T^*)$. The user inputs C and C^* defined in (2) and two additional paremeters: num_+, which is the number of Y^* that will be equal to 1, and ϵ which is a weight to be used later. First, a regular SVM is trained with X, Y, and C, using the objective function of (1), and we find the margin distances of X^*. We take the num_+ most positive margin distances and classify them as 1 in Y^*. Then, we initialize two weights, $C_-^* = \epsilon$ and C_+^* such that $\frac{C_+^*}{C_-^*} = \frac{num_+}{n-num_+}$. Then, we call $SVM(C, C_+^*, C_-^*, Y^*)$, and we greedily find two indices i and j such that $y_j^* = -y_i^*$, $\zeta_i^* > 0$ and $\zeta_j^* > 0$ and $\zeta_i^* + \zeta_j^* > 2$, then we know that flipping y_i^* and y_j^* will reduce (3) and preserve the number of positive num_+ examples. We do this as much as possible, and then set $C_+^* = \max\{2C_+^*, C_+^*\}$ and $C_-^* = \max\{2C_-^*, C_-^*\}$. This increases the importance of the unlabeled data, because its label accuracy should be increasing. Finally, we call $SVM(C, C_+^*, C_-^*, Y^*)$ again, with the updated values of C_+^*, C_-^*, and Y^*, and repeat until $C_-^* = C_+^* = C^*$.

3.2   Spectral Graph Transducer

Another method we are using is a semi-supervised version of k-Nearest Neighbors known as the Spectral Graph Transducer, as described in [J$^+$]. First, a similarity graph is constructed where the

vertices are the datapoints (both labeled and unlabeled) and have edges weighted by some notion of similarity, like distance in a metric space. And under the 3 goals of achieving low training error, having the transductive algorithm match its inductive counterpart, and achieving approximately the same expected ratio between positive and negative examples in both labeled and unlabeled data, [J$^+$] reduces the problem of weighted kNN to approximating the unconstrained ratio cut on the similarity graph.

Let L^+ denote the set of labeled positive examples, and let L^- denote the set of labeled negative examples. The unconstrained ratio cut problem can be stated as follows.

$$\arg \max_{(G^+, G^-)} \frac{\text{cut}(G^+, G^-)}{|G^+||G^-|}$$

$$\text{subject to} \quad x \in G^+ \text{ if } x \in L^+$$
$$x \in G^- \text{ if } x \in L^-$$

Here, $\text{cut}(G^+, G^-)$ denotes the sum of weights of edges between G^+ and G^-. As described in [Dhi01], the problem can be equivalently written as

$$\arg \min_{\vec{z}} \frac{\vec{z}^T L \vec{z}}{\vec{z}^T \vec{z}}$$

$$\text{subject to} \quad z_i \in \{\gamma_+, \gamma_-\}$$

where L is the Laplacian matrix of the graph G, $\gamma_+ = \sqrt{\frac{|L^-|}{|L^+|}}$ and $\gamma_- = -\sqrt{\frac{|L^+|}{|L^-|}}$. This problem, though NP-hard, can be approximated well. The approximation arises in considering the real relaxation of the problem where one minimizes $\vec{z}^T L \vec{z}$ under the constraints that $\vec{z}^T 1 = 0$ and $\vec{z}^T \vec{z} = n$, for which the solution happens to be the second largest eigenvalue of L.

Under the constraint that elements in L^+ must be in G^+ partition and elements in L^- must be in G^- partition, the objective function we must minimize changes to $\vec{z}^T L \vec{z} + c(\vec{z} - \vec{\gamma})^T C(\vec{z} - \vec{\gamma})$, the original objective with an added quadratic penalty, with c being the tradeoff parameter and $\vec{\gamma}$ being a vector with dimension equal to the number of datapoints where γ_i is γ_+ if the i-th datapoint is in L^+, γ_- if the datapoint is in L^- and 0 otherwise.

The algorithm for approximating the minimum of the objective is described in [J$^+$].

4 Experiments

We tested the searching algorithm used in [Joa99a] on the dataset found in [AA98]. We trained 10 different TSVMs, one to distinguish one digit from the rest of the data. To classify a given test point, we found the signed distance from the point to the decision boundary of each TSVM and found the maximum. We tried using three different kernels for our TSVM: linear, Gaussian, and sigmoid. The Gaussian and sigmoid kernels caused our TSVMs to predict all test points as -1, achieving 90% accuracy. We believe this is because the two kernels can express functions that are too complicated for the noise and high-dimension of this data set. We had more luck with the simple linear kernel, and some results are plotted in the tables below:

[Insert table with a few of the classification accuracies of the TSVMs]

Finally, we produced a noticeable relationship between percent of labeled data and classification accuracy

[Insert graph of classification accuracy vs percent of labeled data]

[Insert comments about digits that the TSVM often mistook for another]

5 Future Plans

We plan to test the Spectral Graph Transducer algorithm described in section 3.2.

Do feature selection on the data. Try different kernels for the TSVM, something more advanced than a linear kernel and not as advanced as a Gaussian kernel. Doing feature selection could help us find a good kernel. Change the algorithm of [Joa99a] by not enforcing exactly num_+ examples to be classified as +1. Try different values for the parameters in the algorithm. Try more methods.

References

[AA98]   E Alpaydin and Fevzi Alimoglu. Uci pen-based recognition of handwritten digits data set. 1998.

[BM98]   Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[BNS]   Misha Belkin, Partha Niyogi, and Vikas Sindhwani. On manifold regularization.

[Dhi01]   Inderjit S Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274. ACM, 2001.

[J+]   Thorsten Joachims et al. Transductive learning via spectral graph partitioning.

[Joa99a]   T Joachims. Transductive inference for text classifcation using support vector machines. 1999.

[Joa99b]   Thorsten Joachims. Making large scale svm learning practical. Technical report, Universität Dortmund, 1999.

[Lic13]   M. Lichman. UCI machine learning repository, 2013.

[LPM+08]   David Llorens, Federico Prat, Andrés Marzal, Juan Miguel Vilar, María José Castro, Juan-Carlos Amengual, Sergio Barrachina, Antonio Castellanos, Salvador España Boquera, JA Gómez, et al. The ujipenchars database: a pen-based database of isolated handwritten characters. In *LREC*, 2008.