Iterative Hybrid Algorithm for Semi-supervised Classification

Martin Saveski*
University Pierre and Marie Curie
4 place Jussieu 75005 Paris - France
saveski.martin@gmail.com

Abstract

In the typical supervised learning scenario we are given a set of labeled examples and we aim to induce a model that captures the regularity between the input and the class. However, most of the classification algorithms require hundreds or even thousands of labeled examples to achieve satisfactory performance. Data labels come at high costs as they require expert knowledge, while unlabeled data is usually cheap and easy to obtain. The aim of semi-supervised learning is to build models using both labeled and unlabeled data. In this study, we propose an Iterative Hybrid Algorithm which blends a generative and discriminative model with the goal to benefit from the advantages of both and to make most use of the unlabeled data. We conduct experiments on a synthetic data set, which allows us to easily observe the behavior of the method and compare its performance with two other methods, the Hybrid Model proposed in [2] and Entropy Minimization [6]. We observe that when there are only few labeled examples, the Iterative Hybrid Algorithm achieves better performance than the Entropy Minimization method, but is outperformed by the Hybrid Model. However, as the number of labeled examples increases the difference between the two methods diminishes. The performance of the Entropy Minimization method is still behind the other two methods.

1 Introduction

In the classical supervised learning classification framework, a decision rule is to be learned from a labeled training set. We assume, we are given a data set comprising n vectors $X = \{x_1, \ldots, x_n\}$ together with the corresponding labels $C = \{c_1, \ldots, c_n\}$, in which we assume that the vectors, and their labels, are drawn independently from the same fixed distribution. Our goal is to learn a model governed by a set of parameters θ, which when given a new unlabeled instance $\hat{x}$ can be used to infer its label $\hat{c}$, by assigning the class which maximizes $p(\hat{c}|\hat{x}, \theta)$.

Most of the supervised learning methods, however, require hundreds or even thousands of labeled instances to produce accurate models. Sometimes these labels come at little or no cost at all, for example when we flag emails as spam or when we rate movies on our favorite social networking Web site. But many times, obtaining labels can be time-consuming, difficult, and expensive, as they require the efforts of experienced human annotators. On the other hand, acquiring large amounts of unlabeled data may be relatively easy and at very low costs. The goal of semi-supervised learning, is by using large amount of unlabeled data, together with the labeled data, to build better classifiers.

In machine learning, we distinguish two categories of probabilistic models: generative and discriminative. Generative models are built to model how samples from a particular class are generated, while the discriminative models are concerned with defining the boundaries between the classes. Each of them has its own strengths. The generative models have very strong modeling power and

*Under the supervision of prof. Thierry Artières

in particular they can easily handle missing values, which makes them interesting for the problem of semi-supervised learning. The discriminative models, on the other hand, tend to achieve better classification accuracies. This has motivated many researchers to propose hybrid methods which blend these two approaches and aim to benefit from the advantages of both. One such approach has been proposed in [9] and applied for classification of sequences. In a similar vein, in this study we propose an Iterative Hybrid Algorithm for vectorial data. In this simpler setting, we aim to get better understanding of the approach and ideas for further development. We conduct experiments with synthetic two-dimensional data, which allows us to clearly obverse its decisions and compare its with other methods.

As an academic exercise, the main objectives of this study are: *i*) to develop deep understanding in the theoretical foundations of several semi-supervised learning methods, *ii*) to develop their implementations and conduct various experiments, and *iii*) to compare their behavior and results.

The remainder of this report is organized as follows. In section 2, we further detail the difference of generative and discriminative models to motivate a hybrid approach. In section 4, we present our proposal for an Iterative Hybrid Algorithm for vectorial data. In section 5, we describe two other semi-supervised learning approaches which we aim to compare with and in section 6 we present the experimental results. Finally, in section 7 we conclude.

2 Generative and Discriminative Models

In machine learning, probabilistic models are described as belonging to one of the two categories: generative or discriminative. Generative models are built to understand how samples from a particular class were generated, while discriminative models are concerned with defining the boundaries between the classes.

The generative probabilistic models are developed to capture the interactions between all the variables of a system, in order to be able to synthesize the possible states of the system. This is achieved by defining a probability distribution p modeling inputs, hidden variables, and outputs jointly. In the context of classification, the input is a description of a data instance x and the output is the label c of this instance. Generative models define the joint probability $p(x, c|\theta)$, where θ represents the parameters of the model. One can also model expert knowledge about the problem by defining a prior probability $p(\theta)$, over the set of parameters θ. Most generative probabilistic models are trained by, what is often called *generative learning*, i.e. maximizing the full joint distribution with respect to the parameters θ as:

$$L_G(\theta) = p(X, C, \theta) = p(\theta) \prod_{n=1}^{N} p(x_n, c_n|\theta_{c_n}).$$

This formulation can be easily extended to make use of unlabeled data by taking the marginal distribution over the class c. If we consider $L = \{X_L, C_L\}$ to be the labeled data, and $U = X_U$ to be the unlabeled data, we can rewrite $L_G(\theta)$ as:

$$\begin{aligned} L_G(\theta) &= p(L, U, \theta) = p(X_L, C_L, X_U, \theta) \\ &= p(\theta)\, p(X_L, C_L, X_U|\theta) = p(\theta)\, p(X_L, C_L|\theta)\, p(X_U|\theta) \\ &= p(\theta) \prod_{n \in L} p(x_n, c_n|\theta_{c_n}) \prod_{m \in U} p(x_m|\theta) \\ &= p(\theta) \prod_{n \in L} p(x_n, c_n|\theta_{c_n}) \prod_{m \in U} \left(\sum_c p(x_m, c|\theta_c) \right). \end{aligned}$$

Discriminative models, as opposed to generative ones, are built to capture the boundaries between the different possible output states of a system, without any interest of modeling the distribution of the inputs. This is achieved by defining a probability distribution p over the output c conditioned on the input x. This can be expressed as $p(c|x, \theta)$, where θ are the parameters of the model. Having the training set $\{X, C\}$, the discriminative models are trained by maximizing the following objective

function with respect to the parameters θ:

$$L_D(\theta) = p(C|X, \theta) = \prod_{n=1}^{N} p(c_n|x_n, \theta).$$

As discriminative models directly focus on the boundaries between classes, they usually achieve better classification performance. This is the main reason why discriminative models have been widely and successfully used.

While, both methods have different advantages, there is no easy way to combine them. This the main challenge we wish to undertake in this study, with a particular attention to the application of hybrid approaches to the problem of semi-supervised learning.

3 Related Work

Although, in this study we focus on the use of hybrid methods for semi-supervised learning, we would like to point out that the problem of semi-supervised learning has been well-studied in the literature and many other approaches have been proposed. We provide a small overview of the main categories of methods. Detailed survey of the literature in the field is given in [12].

Methods using *Expectation Maximization* with generative mixture models have shown to improve performance by using unlabeled data, in the cases when the classes produce well clustered data. Nigam *et al.* [7] apply the EM algorithm on mixture of multinomial distributions for the task of text classification. They showed that the resulting classifiers perform better than those trained only on labeled data. Baluja in [1] uses the same algorithm on a face orientation discrimination task.

Co-training [4] assumes that the features in data set can be split in two sets, each of which is sufficient to train a good classifier and they are conditionally independent of one another given the class. Initially, two separate classifiers are trained on the labeled data with each feature set. Then, each classifier classifies the unlabeled data and "teaches" the other classifier with the unlabeled examples it feels most confident about. Each classifier is re-trained with the new labels and the process is repeated.

Graph-based semi-supervised methods define a graph where the nodes are labeled and unlabeled examples in the dataset, and edges reflect similarity of the examples. Different algorithms operating on the graph have been proposed, including [11, 3]. These methods achieve good performance when similar instances in the data set have similar labels.

Self-training is another commonly used technique. A classifier is first trained with the small amount of labeled data and then it is used to classify the unlabeled data. Then, the most confident unlabeled points, with their predicted labels, are added to the training set. The classier is re-trained and the procedure is repeated. This method has been successfully applied to several natural language processing tasks in [8, 10].

We mostly relate our work with the hybrid methods proposed in [2] and [5]. The hybrid model from [2] is described in details in section 5 and compared to our method in section 6. Another similar approach is introduced by Bouchard *et al.* in [5]. They propose a convex combination of the objective functions of the generative and discriminative model.

4 Iterative Hybrid Algorithm

In this section, we propose an iterative algorithm for blending generative and discriminative models for semi-supervised learning. The main idea is to use the generative model to incorporate the additional information brought by the unlabeled data (U), and to use the discriminative model to achieve better classification accuracy. We also want to make sure that the two models are not far apart from one another by making constrains on their parameters.

Initially, we train a generative model only on the labeled data (L), and then we use this optimal solution as a starting position for training a generative model both on L and U. In the main loop of the algorithm, we train a discriminative model on L, constraining its values to be close to the

ones of the generative model. Then, we label part of U and together with L, and we use it to train a generative model. We repeat this procedure a number of iterations or until convergence.

More formally, the method can be described as follows. The parameters of the discriminative model are denoted as θ, and the parameters of the generative model are denoted as $\tilde{\theta}$.

1. Learn $\tilde{\theta}$ on $L \rightarrow \tilde{\theta}^{(0)}$, by maximizing the following objective function:
$$\sum_{x \in L} \log p(c|x, \tilde{\theta})$$
2. Learn $\tilde{\theta}$ on $L \cup U \rightarrow \tilde{\theta}^{(1)}$, starting from $\tilde{\theta}^{(0)}$, maximizing:
$$\sum_{x \in L} \log p(x|c, \tilde{\theta}) + \lambda \sum_{x \in U} \log \sum_{c'} p(x|c', \tilde{\theta})$$
3. Loop n number of iterations, or until convergence:
   3.1. Learn θ on $L \rightarrow \theta^{(i)}$, starting from $\tilde{\theta}^{(i)}$, maximizing:
   $$-\frac{1}{2}||\theta - \tilde{\theta}^{(i)}||^2 + \sum_{x \in L} \log p(c|x, \theta)$$
   3.2. Use $\theta^{(i)}$ to label part of $U \rightarrow U_{Labeled}$, where the labels are assigned as:
   $$x \rightarrow c = \arg\max_c p(c|x, \theta^{(i)})$$
   3.3. Learn $\tilde{\theta}$ on $L + U_{Labeled} \rightarrow \tilde{\theta}^{(i)}$, maximizing:
   $$\sum_{x \in L} \log p(x|c, \tilde{\theta}) + \lambda \sum_{x \in U_{Labeled}} \log p(x|c, \tilde{\theta})$$

The only hyper parameter in the model is λ, which controls the influence of the unlabeled data. Larger values of λ imply more influence of U. The final output of the method are the parameters of the discriminative model $\tilde{\theta}$ and they are used to predict the labels of new instances.

In addition, one may imagine different strategies of choosing which portion of U is going to be labeled in each iteration of the main loop. One can simply take random instances or choose the ones for which the discriminative model is most confident about. We are in favor of the second idea.

5 Other approaches

In this section, we describe two other semi-supervised learning methods, with which we wish to compare our results. The first is a hybrid model similar in nature to the one we propose. However, unlike the iterative approach, it aims in optimizing a hybrid objective function. The second, belongs to the family of entropy minimization methods.

Note the distinction we make between hybrid algorithm and hybrid model. Hybrid algorithms refer to the algorithms involving two or more models influencing each other, while hybrid models involve a multi-criteria objective function.

5.1 Hybrid Model

This method is developed by Bishop and Lasserre in [2]. They introduce a multi-criteria objective function which combines, in a principled way, generative and discriminative models by using specific priors linking the parameters of the two models.

Given the data set $X = \{X_L, X_U\}$, containing both labeled and unlabeled data, where the classes of the labeled points are denoted as C_L, they propose maximizing the following objective function:

$$\begin{aligned} q(X, C, \theta, \tilde{\theta}) &= q(X_L, C_L, X_U, \theta, \tilde{\theta}) \\ &= p(\theta, \tilde{\theta})\, p(C_L|X_L, \theta) p(X_L, X_U|\tilde{\theta}) \\ &= p(\theta, \tilde{\theta}) \prod_{n \in L} p(C_n|X_n, \theta) \prod_{m \in L \cup U} p(X_m|\tilde{\theta}) \end{aligned}$$

where, θ and $\tilde{\theta}$ are the parameters of the discriminative and the generative model, respectively.

Assuming, uniform prior probability of the classes, we can rewrite the above expression as:

$$p(\theta, \tilde{\theta}) \prod_{n \in L} \left(\frac{p(x|c, \theta)}{\sum_{c'} p(x|c', \theta)} \right) \prod_{m \in L \cup U} \left(\sum_{c'} p(x|c', \tilde{\theta}) \right).$$

the derivation of $p(c|x, \theta)$ and $p(x, \tilde{\theta})$ is given in appendix B.1.

A prior, proposed in [2], which interpolates smoothly between the generative and discriminative limits with hyper parameter σ, can be defined as:

$$p(\theta, \tilde{\theta}) \propto \frac{1}{\sigma} \exp\left(-\frac{||\theta - \tilde{\theta}||^2}{2\sigma^2} \right).$$

To make it easy to compare with the different parameter settings of the other methods, we can express σ in terms of $\alpha \in [0, 1]$ as:

$$\sigma^2(\alpha) = \left(\frac{\alpha}{1 - \alpha} \right)^r$$

where, for $\alpha \to 0, \sigma^2 \to 0$, results in pure generative model, and when $\alpha \to 1, \sigma^2 \to \infty$, results in pure discriminative model.

For classification of new instances only the θ parameters are used assigning the most probable class c as:

$$x \to c = \arg\max_c p(c|x, \theta) = \arg\max_c p(x|c, \theta).$$

Full derivation is given in appendix B.1.

5.2 Entropy Minimization Method

This method is introduced in [6] by Grandvalet and Bengio. They use the label entropy on unlabeled data as a regularizer. By minimizing the entropy, the method assumes a prior which prefers minimal class overlap. They propose maximizing the following objective function:

$$\sum_{x \in L} \log p(c|x, \theta) + \lambda \sum_{x \in U} \sum_{c' \in C} p(c'|x, \theta) \log p(c'|x, \theta)$$

where, λ is a hyper parameter of the model controlling the influence of the unlabeled data.

6 Experiments

In this section, we illustrate the behavior and measure the performance of each of the methods described using an example based on synthetic data. The data is chosen to be as simple as possible and so involves vectors x_n which have only two dimensions/features, and belong to one of two classes. This allows us to easily visualize the data, in a two-dimensional Euclidean space, and to perform many experiments with all the methods in a reasonable amount of time. Data is generated from two Gaussian distributions, one for each class. The class-conditional densities $p(x|c)$, have the same variance on the y axis, but are horizontally elongated.

In each model, we represent the class-conditional density using an isotropic Gaussian distribution. As this distribution does not capture the horizontal elongation of the true class distributions, it represents a form of a model mis-specification. The parameters of the models are the means and the variances of the Gaussian distributions. To simplify, we assume uniform prior probability of each classes.

The training data set consists of 200 instances per class, where only several of them are labeled from each class, and the testing data set consists of 200 instances per class. Each experiment is run with different random initialization and all methods are tested. The parameters are initialized by setting the means of the isotropic Gaussians to the mean of the labeled instances, and setting the variances to one.

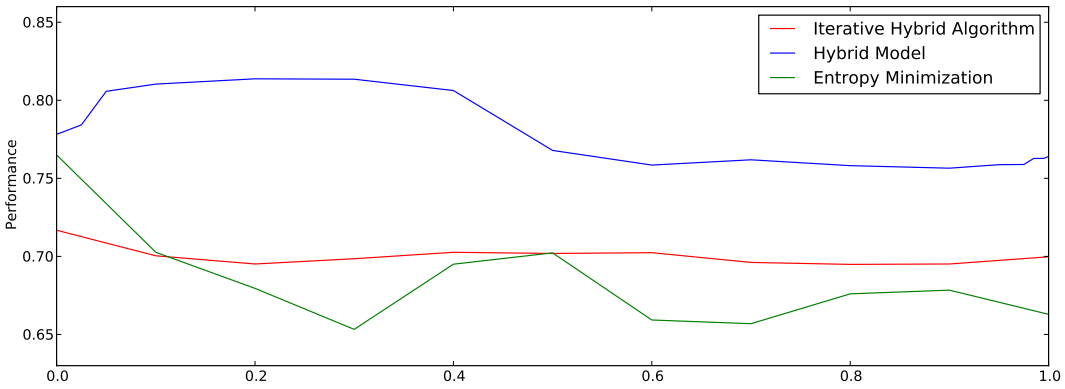

Figure 1: Average performance of each method in 25 runs with random initializations, where only two points are labeled. The x axis shows the parameter values and the y axis the performance achieved. Note that the values of the parameters of each method do not have the same semantics and are not directly comparable. Nonetheless, we can compare their overall performance.

6.1 Experiments with only two labeled points

We ran 25 experiments with fixed values of the parameters of each method and we measured the classification performance. For the Iterative Hybrid Algorithm and the Entropy Minimization, we set the values of λ in the range of $[0, 1]$ with step of 0.1. For the Hybrid Model, as suggested by the authors, we test the following values of parameter α: $\{0, 0.005, 0.025, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.975, 0.985, 0.995, 1\}$.

In figure 1, we show an average of the performance of each method, averaged over all runs with the same value of the parameter. It is important to note that the values of the parameters of each method, do not have the same semantics and are not directly comparable. For instance, in the Hybrid Model $\alpha = 0$ corresponds to the use of purely generative model, while in the Iterative Hybrid Algorithm and Entropy Minimization methods $\lambda = 0$ corresponds to no influence of the unlabeled data in the final model. Nonetheless, the figure illustrates the overall performance of each method and allows us to compare them.

We can note the dominance in the performance of the Hybrid Model over the two other methods. The Iterative Hybrid Algorithm, on the other hand, outperforms Entropy Minimization over all values of the parameters, except for $\alpha = 0$. The improved performance at $\alpha = 0$ is not surprising as, if we take a closer look at the objective functions of each method, we can notice that this value of α in the Entropy Minimization corresponds to a purely discriminative model and in the Iterative Hybrid Algorithm corresponds to a mixture of both discriminative and generative model, which we know is misspecified.

Taking a closer look at the performance curves of each run in figure 2, we can note that it is very hard to fix the parameter of each method. In other words, the best performance of the methods in each run is obtained for different values of the parameters. Moreover, we observe that the Entropy Minimization method is very unstable and its performance may drastically change with small changes of the parameter value. This shows that the method is very sensitive and lacks robustness, making its use in practical scenarios very hard.

This, however, is not the case with the two other methods. Although, we do not know how to fix the parameter values, we can note that their performance is not sensitive to the small changes of the parameters. In practical use, one may image an iterative process where we start at one value of parameter and by adjusting it we can converge to a satisfactory performance.

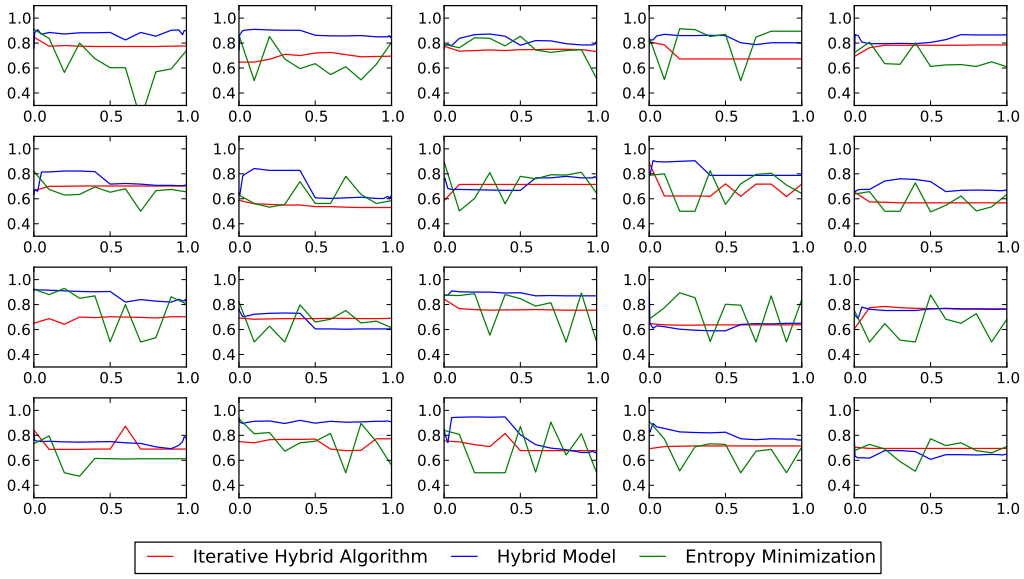

Figure 2: Performance curves over all runs. Note that the parameters have different semantics for each method and are not directly comparable.

6.2 Particular cases

To develop a deeper understanding of the difference between the behavior of the Iterative Hybrid Algorithm and the Hybrid Model, we have decided to look closer at some particular cases. In specific, the cases where the decision boundary induced by the labeled points is far from the real boundary between the classes.

We observe that in such cases the overlap on the x axis between the labeled points of each class plays an important role. When there is no overlap, in overall both methods tend to achieve good performance. Although this depends from one case to another. We exhibit one case where the

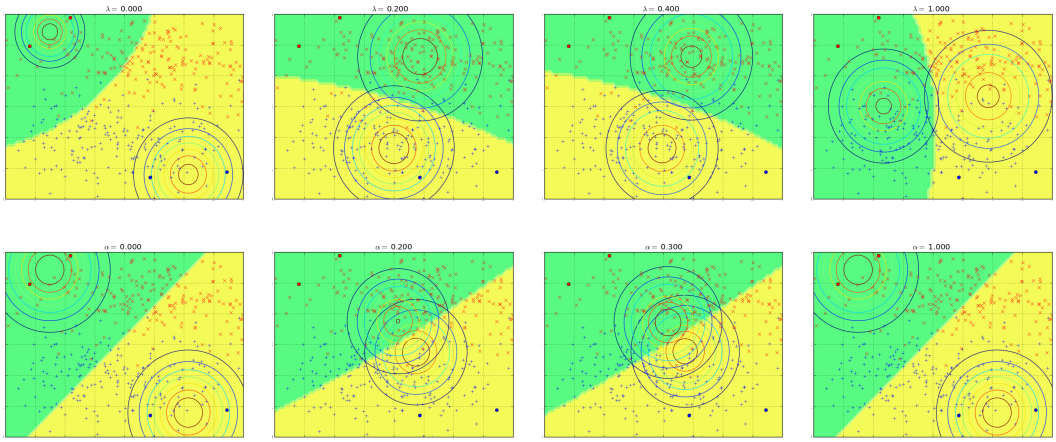

Figure 3: A case where there is no overlap between the labeled points of each class on the x axis. The Iterative Hybrid Algorithm is shown on the top and the Hybrid Model on the bottom. Each figure corresponds to a different value of the parameter, increasing from left to right. The crosses denote the unlabeled points, while the dots denote the labeled points. The green area corresponds to the points that are assigned to the red class, while the yellow area corresponds to the points which are assigned to the blue class. The circles denote the contours of the isotropic Gaussians. The two methods take a different path. Only the Iterative Hybrid Algorithm, for $\lambda = 0.2$, converges close to the original boundary between the classes.

Iterative Hybrid Algorithm is dominant (figure 3), and one case where both converge to a satisfactory solution (figure 4).

However, in the cases where there is an overlap on the x axis, we observe that the Hybrid Model is dominant. Figure 5, shows one such example. We believe that this behavior of the Iterative Hybrid Algorithm happens because the model of one of the classes, in particular the one whose labeled instances are further apart on the x axis, in the early iterations converges to a higher variance of the isotropic Gaussian causing it to incorrectly label instances from the other class. This error is reinforced in the later iterations of the algorithm, causing a dominance of the model of one class and convergence to an unsatisfactory solution. This is illustrated in figure 5.

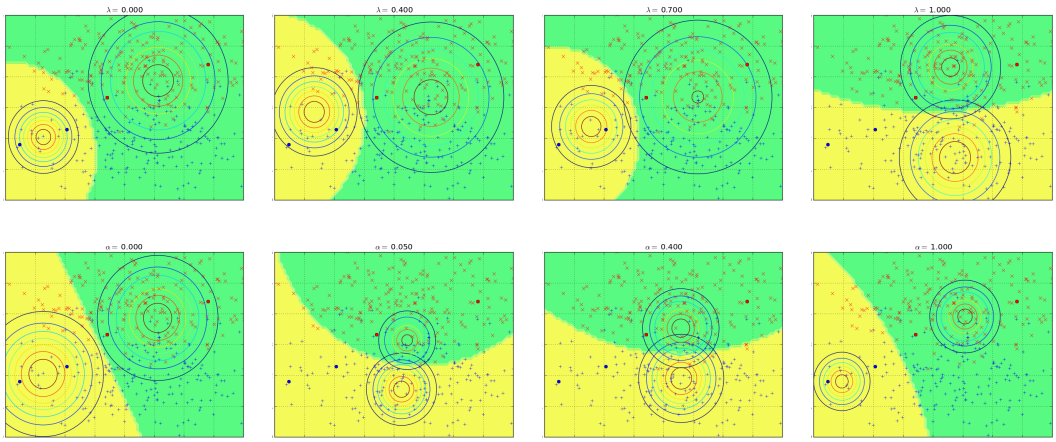

Figure 4: Another case where there is no overlap between the labeled points of each class on the x axis. The Iterative Hybrid Algorithm is shown on the top and the Hybrid Model on the bottom. Each figure corresponds to a different value of the parameter, increasing from left to right. Both methods (for $\lambda = 1$ and $\alpha = 0.4$) converge to solutions close to the original boundary between the classes.

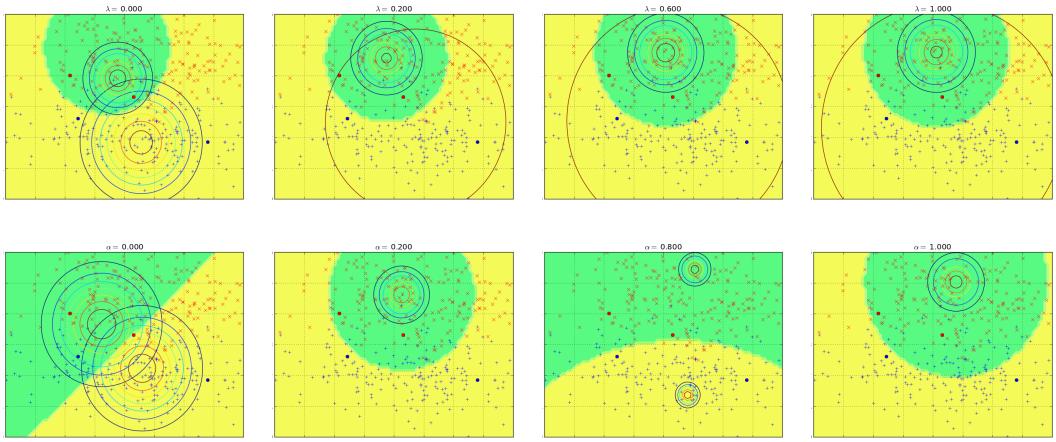

Figure 5: A case where there is an overlap overlap between the labeled points of each class on the x axis. The Iterative Hybrid Algorithm is shown on the top and the Hybrid Model on the bottom. The Iterative Hybrid Algorithm correctly classifies the labeled points, but fails to converge to the real boundary between the classes. However, the Hybrid Model for $\alpha = 0.8$ converges to a satisfactory solution.

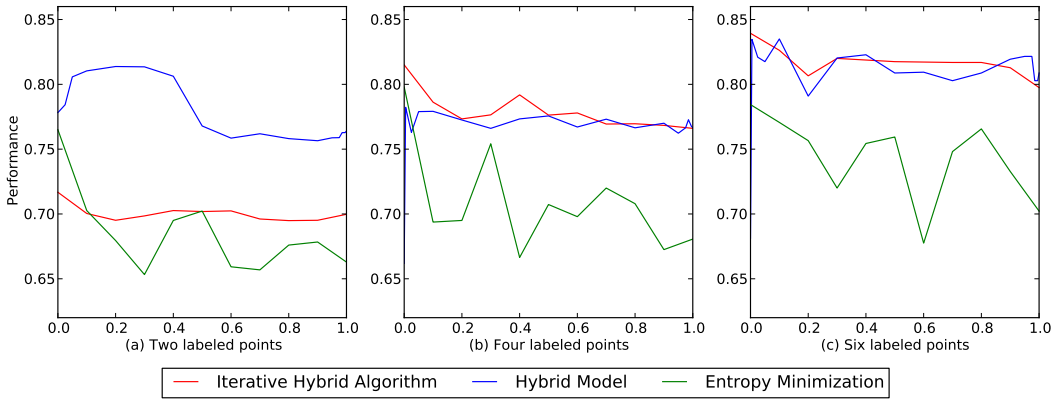

Figure 6: Comparison of the average performance of each method with different number of labeled points.

6.3 Different number of labeled points

Finally, we observe how the performance of the methods changes when the number of labeled instances increases. Figure 6 depicts the mean performance of all methods in several runs with 2, 4, and 6 labeled examples. As before, we note that the parameters of each method do not have the same meaning and are not directly comparable, but nevertheless we can compare the overall performance of the methods. In appendices C.1 and C.2, we show the performance of each run with 4 and 6 labeled points, as well as the standard deviations of the performance of each method.

It can be observed that as the number of labeled example increases, the difference in the performance between the Iterative Hybrid Algorithm and the Hybrid Model vanishes. The Entropy Minimization method, although has increased performance, is still outperformed by the two other methods.

7 Conclusions

We have studied methods combining generative and discriminative models in the setting of semi-supervised learning. We have proposed an Iterative Hybrid Algorithm and compared its performance with two other methods, the Hybrid Model and Entropy Minimization. We have conducted various experiments with a syntactic data set consisting of two Gaussian distributions. We observe that when number of labeled instances is small the Iterative Hybrid Algorithm dominates the Entropy Minimization method, but it is outperformed by the Hybrid Model. Looking at particular cases, we conclude that in the scenarios where there is an overlap on the x axis between the labeled points of each class the two methods behave differently and the Hybrid Model tends to converge to an satisfactory solution more often. Finally, when the number of the labeled instances increases the difference between the Iterative Hybrid Algorithm and the Hybrid Model diminishes, while both outperform the Entropy Minimization method.

8 Acknowledgments

I would like to thank professor Thierry Artières for the many helpful discussions and his willingness to guide me through this journey in the world of semi-supervised learning and probabilistic models. Merci Beaucoup.

References

[1] S Baluja. Probabilistic modeling for face orientation discrimination: Learning from labeled and unlabeled data. *Advances in Neural Information Processing Systems*, 11:854–860, 1998.

[2] Christopher M Bishop and Julia Lasserre. Generative or Discriminative? Getting the Best of Both Worlds. *Statistics*, 8:3–24, 2007.

[3] Avrim Blum and Shuchi Chawla. Learning from Labeled and Unlabeled Data using Graph Mincuts. In *Science*, pages 19–26. Morgan Kaufmann Publishers Inc., 2001.

[4] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory COLT 98*, COLT' 98, pages 92–100. ACM, ACM Press, 1998.

[5] Guillaume Bouchard. Bias-Variance tradeoff in Hybrid Generative-Discriminative models. *ICMLA 07*, pages 124–129, 2007.

[6] Yves Grandvalet and Yoshua Bengio. Semi-supervised Learning by Entropy Minimization. *Network*, 17(5):529–536, 2005.

[7] K. Nigam, McCallum A. K., S. Thrun, and T Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

[8] E Riloff and J Wiebe. Learning Extraction Patterns for Subjective Expressions. In Ellen Riloff and Janyce Wiebe, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing EMNLP*, pages 105–112. Conference on Empirical Methods in Natural Language Processing, 2003.

[9] Yann Soullard and Thierry Artières. Hybrid HMM and HCRF model for sequence classification. In *European Symposium on Artificial Neural Networks (ESANN)*, 2011.

[10] David Yarowsky. Unsupervised Word-Sense Disambiguation Rivalling Supervised Methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, 1995.

[11] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning from Labeled and Unlabeled Data on a Directed Graph. *Proceedings of the 22nd International Conference on Machine Learning*, pages 1041–1048, 2005.

[12] Xiaojin Zhu. Semi-Supervised Learning Literature Survey Contents. *SciencesNew York*, 2008.

A Implementation

The implementation of all the methods discussed is submitted as a supplementary material. The programing language of choice was *Python*, as it allows rapid development and provides an excellent range of tools for scientific computing. Data generation and manipulation was done with *NumPy*, optimization was done with *SciPy*, and all the plots were produced using *PyPlot*.

B Complements to subsection 5.1, Hybrid Model

B.1 Derivations

If we assume uniform prior probability of each class we can rewrite $p(c|x,\theta)$ and $p(x,\tilde{\theta})$ as:

$$\begin{aligned} p(c|x,\theta) &= \frac{p(x|c,\theta)p(c|\theta)}{p(x|\theta)} = \frac{p(x|c,\theta)p(c|\theta)}{\sum_{c'} p(x|c',\theta)} \\ &= \frac{p(x|c,\theta)p(c,\theta)}{\sum_{c'} p(x|c',\theta)p(c',\theta)} = \frac{p(x|c,\theta)}{\sum_{c'} p(x|c',\theta)} \end{aligned}$$

$$p(x,\tilde{\theta}) = \sum_{c'} p(x,c',\tilde{\theta}) = \sum_{c'} p(x|c',\tilde{\theta})\, p(c',\tilde{\theta}) = \sum_{c'} p(x|c',\tilde{\theta}).$$

For classification, only the θ parameters of the model are used. New instances are classified by assigning the most probable class c as:

$$\begin{aligned} x \to c &= \arg\max_c p(c|x,\theta) \\ &= \arg\max_c \frac{p(x|c,\theta)}{\sum_{c'} p(x|c',\theta)} \\ &= \arg\max_c p(x|c,\theta) \end{aligned}$$

In the first step, we assume uniform prior probability $p(c|\theta)$, for each c. In the second step the denominator $\sum_{c'} p(x|c',\theta)$ is constant over all classes and can be ignored.

C Complements to section 6, Experiments

C.1 Results of the experiments with only two labeled points

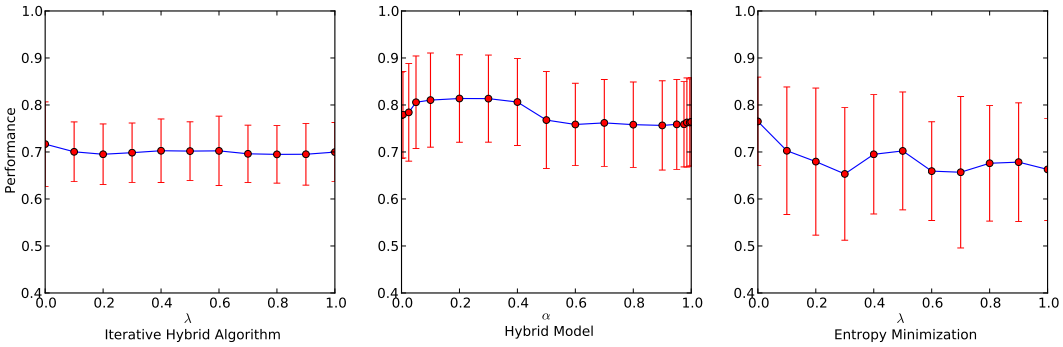

Figure 7: Mean performance and standard deviation of each method, over all runs.

C.2 Results of the experiments with four labeled points

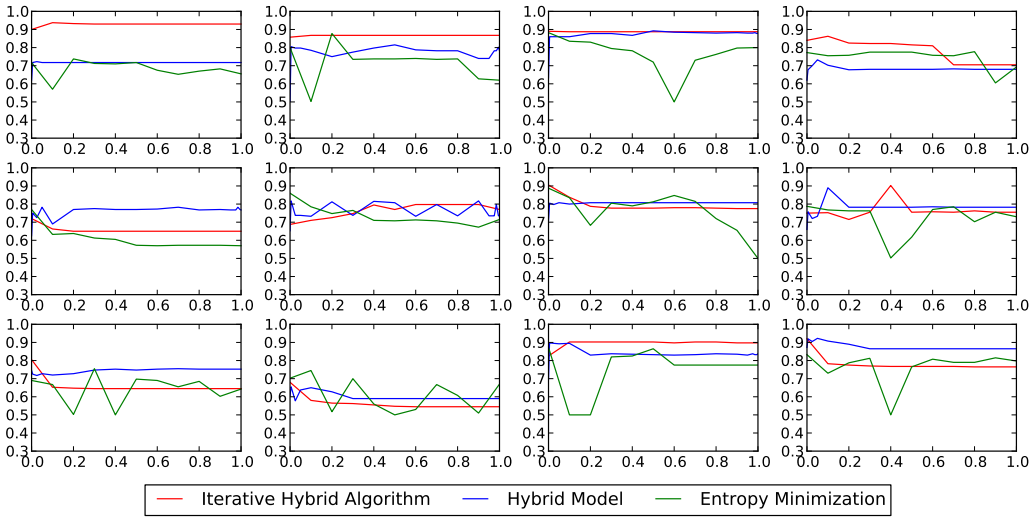

Figure 8: Performance curves over all runs. Note that the parameters have different semantics for each method and are not directly comparable.

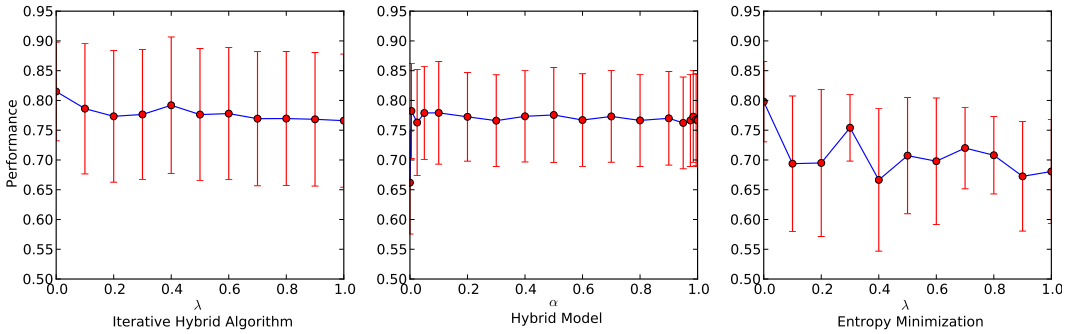

Figure 9: Mean performance and standard deviation of each method, over all runs.

C.3 Results of the experiments with six labeled points

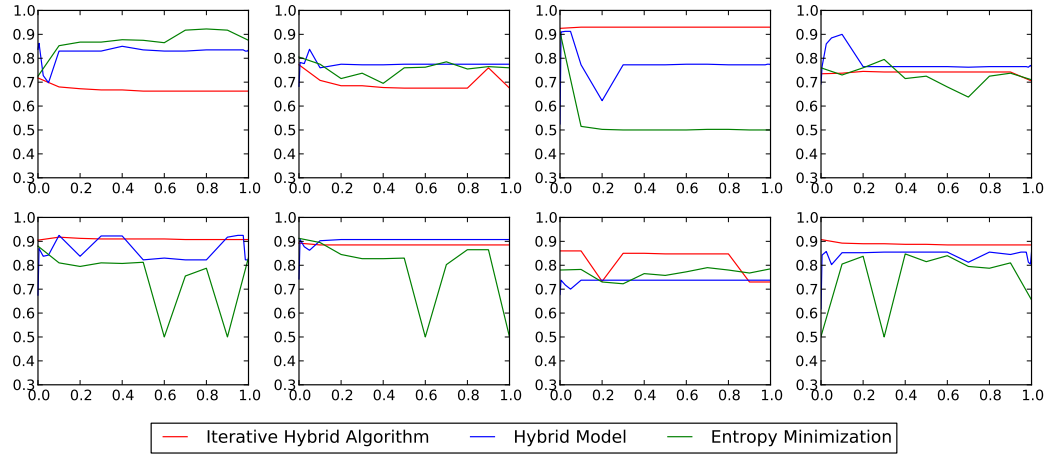

Figure 10: Performance curves over all runs. Note that the parameters have different semantics for each method and are not directly comparable.

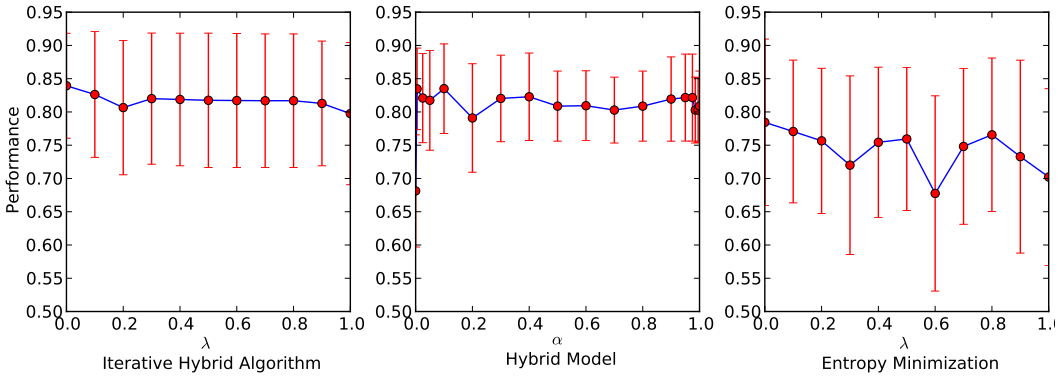

Figure 11: Mean performance and standard deviation of each method, over all runs.