FACE DETECTION USING LARGE MARGIN CLASSIFIERS

Ming-Hsuan Yang *Dan Roth* *Narendra Ahuja*
Department of Computer Science and Beckman Institute
University of Illinois at Urbana-Champaign, Urbana, IL 61801

ABSTRACT

Large margin classifiers have demonstrated their advantages in many visual learning tasks, and have attracted much attention in vision and image processing communities. In this paper we apply and compare two large margin classifiers, Support Vector Machines and Sparse Network of Winnows, to detect faces in still gray scale images. Furthermore, we study the theoretical frameworks of these classifiers and analyze the empirical results. Experiments on a test set of 24,045 images exhibit good generalization and robustness, and conform to theoretical analysis.

1. INTRODUCTION

Machine learning and statistical methods have become popular as tools for addressing a variety of visual learning problems, ranging from object recognition, pedestrian detection, to face detection. There are, however, very few attempts to address theoretical issues and, in particular, study the suitability of different learning algorithms to different vision problems. The goal of this paper is to present a theoretical account of a learning approach and its suitability to visual recognition. We use tools from computational learning theory [14, 15] in order to study the properties of two successful learning approaches. The algorithms are evaluated on a visual learning problem - face detection - and the theoretical generalization properties, along with our analysis of the data are used to explain the prediction performance and discuss the suitability of the approaches to visual learning problems. The learning approaches we study are Support Vector Machines (SVMs) [15] and the SNoW learning architecture [12]. Both have been studied extensively recently and have shown good empirical performance on several visual learning problems. We study both generalization and computational efficiency issues and derive conclusions that are relevant to further use of these learning methods in visual recognition tasks.

Several theoretical results have suggested that these two approaches have incomparable generalization performance that depend on well defined properties of the domain and the target concept. We study these properties and conclude that the face detection data suggests that the SNoW based approach should have advantages in terms of generalization. In addition to generalization, the two learning approaches can also be measured in terms of efficiency. This is important especially when one wants to “blow” up the feature space in order to increase the expressivity of the features and allow a linear classifier to discriminate faces from non-faces, or to discriminate between objects. In this case, we argue that the SVM approach is advantageous. We argue that in order to fully exploit the nice generalization properties of SNoW, images should be represented using features that give rise to a fairly small number of *active* features in each image. That is an attempt should be made to use representations that are not pixel based, but rather based on sparser phenomenon in the image, such as edges, conjunctions of those or other types of features. We show some preliminary results that support this and suggest several directions for future work.

2. LARGE MARGIN CLASSIFIERS

Most efficient learning methods known today, including many probabilistic classifiers, make use of a linear decision surface over the feature space. Among these methods we focus here on SVMs and SNoW which have demonstrated good empirical results in vision and natural language processing problems [10, 16]. SNoW and SVM, are representatives of two different classes of linear classifiers/regressors. SNoW is based on Winnow, a multiplicative update rule algorithm [8, 7]; SVMs are based on perceptron [11], an additive update rule. Although SVMs can also be developed independently of the relation to perceptron, for the sake of our theoretical analysis viewing them as a large margin perceptron [4, 3] is important. Moreover, recent results [5] have shown that the generalization properties of SVMs are dominated by those of large margin perceptron, and therefore it is sufficient here to study those.

2.1. The SNoW Learning Architecture

The SNoW (Sparse Network of Winnows) learning architecture is a sparse network of linear units over a common pre-defined or incrementally learned feature space. Nodes in the input layer of the network represent relations over the input instance and are being used as the input features. Each linear unit is called a *target node* and represents a concept of interest over the input. In the application described here, target nodes could represent an object in terms features extracted from the 2D image input, a face, or a non-face. In the current presentation we assume that all features are binary (in $\{0, 1\}$), although SNoW can take real numbers as input. An input instance is mapped into a set of features which are active (with feature value 1) in it; this variable size representation is presented to the input layer of SNoW and propagates to the target nodes. Target nodes are linked via weighted edges to (some of) the input features.

Let $\mathcal{A}_t = \{i_1, \ldots, i_m\}$ be the set of features that are active in an example and are linked to the target node t. Then the linear unit corresponding to t is *active* iff $\sum_{i \in \mathcal{A}_t} w_i^t > \theta_t$, where w_i^t is the weight on the edge connecting the ith feature to the target node t, and θ_t is the threshold for the target node t. Each SNoW *unit* may include a collection of subnetworks, one for each of the target relations but all using the same feature space. A given example is treated autonomously by each target unit; an example labeled t may be treated as a positive example by the t unit and as a negative example by the rest of the target nodes in its subnetwork. At decision time, a prediction for each subnetwork is derived using a winner-take-all policy. In this way, SNoW may be viewed as a multi-class predictor. In the application described here, we may have one unit with target subnetworks for all the target objects or we may define different units, each with two competing target objects.

SNoW's learning policy is on-line and mistake-driven; several update rules can be used within SNoW, but here we concentrate on the one which is a variant of Littlestone's Winnow multiplicative update rule [8]. The Winnow update rule has, in addition to the threshold θ_t at the target t, two update parameters: a *promotion* parameter $\alpha > 1$ and a *demotion* parameter $0 < \beta < 1$. These are being used to update the current representation of the target t (the set of weights w_i^t) only when a mistake in prediction is made. Let $\mathcal{A}_t = \{i_1, \ldots, i_m\}$ be the set of active features that are linked to the target node t. If the algorithm predicts 0 (that is, $\sum_{i \in \mathcal{A}_t} w_i^t \leq \theta_t$) and the received label is 1, the active weights in the current example are *promoted* in a multiplicative fashion: $\forall i \in \mathcal{A}_t, w_i^t \leftarrow \alpha \cdot w_i^t$. If the algorithm predicts 1 ($\sum_{i \in \mathcal{A}_t} w_i^t > \theta_t$) and the received label is 0, the active weights in the current example are *demoted*: $\forall i \in \mathcal{A}_t, w_i^t \leftarrow \beta \cdot w_i^t$. All other weights are unchanged. As will be clear below, the key feature of the Winnow update rule is that the number of examples required to learn a linear function grows linearly with the number n_r of *relevant* features and only logarithmically with the total number of features. This property seems crucial in domains in which the number of potential features is vast, but a relatively small number of them is relevant. Moreover, in the sparse model, the number of examples required before converging to a linear separator that separates the data (provided it exists) scales with $O(n_r \log n_a)$. Winnow is known to learn efficiently any linear function (in general cases efficiency scales with the margin) and to be robust in the presence of various kinds of noise and in cases where no linear function can make perfect classifications, while still maintaining its abovementioned dependence on the number of total and relevant attributes [9, 7].

2.2. Large Margin Perceptron and SVMs

In this section we briefly present perceptron and SVM; the presentation concentrates on the linearly separable case, although it can be extended to the more general case.

The perceptron also maintains a weight vector w and, given an input vector x^i, predicts that x^i is a positive example iff $w \cdot x^i > \theta$. Like Winnow, the perceptron's update rule is also an on-line and mistake driven, and the only difference between them is that the weight update rule of perceptron is additive. That is, if the linear function misclassified an input training vector x^i with true label y^i (here we assume for notational convenience that $y^i \in \{-1, +1\}$) then we update each component i of the weight vector w by: $w_j \leftarrow w_j + \eta x^i y^i$, where η is the learning rate parameter.

Like Winnow, the Perceptron is also known to learn every linear function, and in the general case, the number of mistakes required before it converge to a hyperplane that separates the data depends also on the margin in the data, that is, on $\max x^i \cdot y^i$, where $y^i \in \{-1, +1\}$ is the true label of the example x^i.

Linear separability is a rather strict condition. One way to make methods more powerful is to add dimensions of features to the input space. Usually, if we add enough new features, we can make the data linearly separable; if the separation is sufficiently good, then the expected generalization error will be small, proved that we do not increase the complexity of instances too much by this transformation. However, from a computational point of view this could be prohibitively hard. This problem can be sometimes solved by the kernel trick. Aizerman et. al have suggested this methods and showed that it can be combined with perceptron [1]. Boser et. al showed that the same holds for SVMs [2]. As will be clear later, the kernel trick serves to aid efficiency, in case there is a need to work in a higher dimensional space; however, the generalization properties, in general, depend on the effective, high dimensional, feature space in which the linear classifier is determined.

SVMs, or batch large margin classifiers can be derived directly from a large margin version of perceptron (which we do not describe here; see e.g., [17]) using a standard way to convert the on-line algorithm to a batch algorithm. This is done in order to convert the mistake bounds that are typically derived for on-line algorithms to generalization bounds that are of more interest (e.g. [4]). However, for completeness, we briefly explain the original, direct derivation of SVMs. SVMs can be derived directly from the following inductive inference. Given a labeled set of training samples, an SVM finds the optimal hyperplane that correctly separates the data points while maximizing the distance of either class from the hyperplane (maximizing the margin). Vapnik shows that maximizing the margin is equivalent to minimizing the VC dimension and thus yield best generalization results [15]. Computing the best hyperplane is posed as a constrained optimization problem and solved using quadratic programming techniques. The optimal hyperplane is defined by

$$\min \frac{1}{2} w^2, \text{ subject to } \quad y^i(w^T x^i + b) \geq 1 \; \forall i = 1, \ldots, M$$

where b is a bias term computed from the margin.

Finally we note that although large margin perceptron and SVMs are very related, it turns out that the generalization bounds of the large margin perceptron are slightly better than those of SVMs and therefore we will use those in our analysis. Although these are worst case bounds, they have already be shown to be quite representative in some experiments using synthetic data [7], so we can use them to guide our understanding.

3. GENERALIZATION AND EFFICIENCY

We consider two issues when comparing learning algorithms (e.g., SNoW and SVM): generalization and efficiency.

3.1. Generalization Error Bounds

Learning systems use training data in order to generate a hypothesis, but the key performance measure one cares about is actually how well they will perform on previously unseen examples. Generalization bounds are derived in order to estimate, given the performance on the training data, what will be the performance on previously unseen examples. The assumption underlying the derivation of generalization bounds is the basic assumption of the PAC learning theory [14], that the test data is sampled from the same (unknown) distribution from which the training data was sampled. In the following we preset two theorems, one describing the generalization error bound of large margin classifiers (e.g., SVMs) and the corresponding theorem for the multiplicative update algorithm (e.g., Winnow). The first is a variant of Theorem 4.19 about the in [3, 17]:

Theorem 1 *If the data is L_2 norm bounded as $||x||_2 \leq b$, then consider the family Γ of hyperplanes w such that $||w||_2 \leq a$. Denote by $err(w)$ the misclassification error of w with the true distribution. Then there is a constant C such that for any $\gamma > 0$, with probability $1 - \eta$ over n random samples, any $w \in \Gamma$ satisfies:*

$$err(w) \leq \frac{k_\gamma}{n} + \sqrt{\frac{C}{\gamma^2 n} a^2 b^2 \ln(\frac{nab}{\gamma} + 2) + \ln\frac{1}{\eta}}$$

where $k_\gamma = |\{i : w^T x^i y^i < \gamma\}|$ is the number of samples with margin less than γ

Similarly we present a generalization bound for Winnow family of algorithms (e.g., SNoW). Derivations of this theorem can be found in [3, 17].

Theorem 2 *If the data is L_∞ norm bounded as $||x||_\infty \leq b$, then consider the family Γ of hyperplanes w such that $||w||_1 \leq a$ and $\sum_j w_j \ln(\frac{w_j ||\mu||_1}{\mu_j ||w||_1}) \leq c$. Denote by $err(w)$ the misclassification error of w with the true distribution. Then there is a constant C such that for any $\gamma > 0$, with probability $1 - \eta$ over n random samples, any $w \in \Gamma$ satisfies:*

$$err(w) \leq \frac{k_\gamma}{n} + \sqrt{\frac{C}{\gamma^2 n} b^2 (a^2 + ac) \ln(\frac{nab}{\gamma} + 2) + \ln\frac{1}{\eta}}$$

where $k_\gamma = |\{i : w^T x^i y^i < \gamma\}|$ is the number of samples with margin less than γ

In order to understand the relative merits of the algorithms, a closer look at the above bounds shows that, modulo some unimportant terms, the error bounds E_a and E_m for the additive algorithms and the multiplicative algorithms scale with:

$$E_a \approx ||w||_2^2 \max_i ||x^i||_2^2,$$

and

$$E_m = 2 \ln 2n ||w||_1^2 \max_i ||x^i||_\infty^2.$$

where w is the target hyperplane.

From the theorems, the main difference between SVM and SNoW is the data assumption. If the data is L_2 norm bounded and there is a small L_2 norm hyperplane, then SVM is suitable for the problem. On the other hand, Winnow is suitable for a problem where the data is L_∞ norm bounded and there is a small L_1 norm hyperplane. Theoretical analysis indicates that the advantage of the Winnow family of algorithms (e.g., SNoW) over Perceptron family of algorithms (e.g., SVM) requires the data to have small L_∞ norm but large L_2 norm. Numerical experiments in [7] have confirmed the claim above and demonstrated the generalization bounds are quite tight.

3.1.1. Experiment I: Generalization

To understand and analyze the performance of SNoW and SVM, we perform numerous experiments on face detection. The training set consists of 6,977 images (2,429 faces and 4,548 non-faces), and the test set consists of 24,045 images (472 faces and 23,573 non-faces). Our training and test sets are similar to the ones used in [6] which also show that SVMs with the feature representation of normalized intensity values perform better than the ones with Harr wavelet and gradient representations. In our experiment, each image is normalized to 20×20 pixels and processed with histogram equalization and quantization (50 rather than 256 scales). Figure 1 shows some face images in the training and test sets. We use the the normalized intensity values as representations for SVM with linear kernel. For SNoW, we also use normalized intensity values as features of images, which we call linear features.

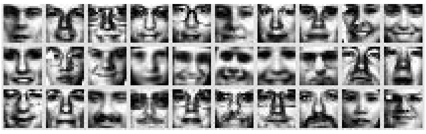

Fig. 1. Sample face images: each image is normalized to 20×20 pixels with histogram equalization.

For the baseline study where SNoW and SVM have the same feature representation, i.e., normalized intensity values, SNoW clearly outperforms SVM as shown by the lower two ROC curves in Figure 2. For visual pattern recognition, most data dimension is not useful as demonstrated in the Eigenface [13] approach and others. Many studies have also shown that the target hyperplane function in visual pattern

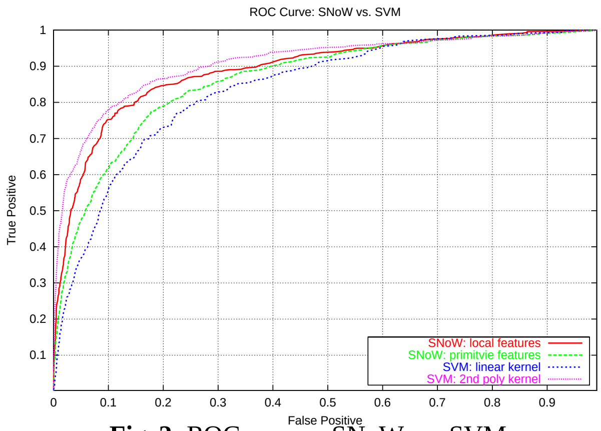

Fig. 2. ROC curves: SNoW vs. SVM

recognition is usually sparse. Consequently, the target hyperplane has a relatively small L_2 norm and relatively small L_1 norm. Under such situations, the Perceptron does not have any theoretical advantage over Winnow. Thus it is not surprising to see that the Winnow family and the Perceptron family of algorithms perform equally well. For the experiment with linear features (i.e., normalized intensity values), the L_2 norm is on the average 10.2 times larger than the L_∞ norm. The number of active features in the final hyperplane of SNoW is very sparse, i.e., 1.6% of all possible features. The number of support vectors is also sparse, i.e., 5% of all the training examples. The empirical results show that SNoW outperforms SVM (shown in lower two ROC curves in Figure 2) and match the predictions of the theorems well.

3.1.2. Experiment II: Efficiency

Since the features in the SVM with polynomial kernel are more expressive than the linear features, we choose to use conjunctions of features to capture local information of image patterns. For each pixel, we represent the conjunction of intensity values of m pixels within a window of $w \times w$ pixels as a new feature value and use them as feature vectors. Each feature value is then mapped to a binary feature using the method discussed in [16] To make sure that the combined computational requirement of SNoW (computational loads of features and training) does not outweigh the one of SVM, we choose to use a small window of 2×2 pixels and conjunctions of 2 pixels.

Figure 2 shows the upper two ROC curves of SVM with second order polynomial kernel and SNoW with conjunction of features. Although SVM performs slightly better than SNoW, we think that SNoW can perform as well as SVM if the feature representation is as powerful as the one in SVM with polynomial kernel. We will discuss these issues in Section 4. The L_2 norm of the local features (generated by 2×2 window) is 2.2 times larger than L_∞ norm. In this case, SVM performs slightly better than SNoW. The results conform to the predictions of the analysis of the theorems which indicates that the advantage of SNoW over SVM requires the data to have large L_2 norm but small L_∞ norm.

4. CONCLUSION

This paper proposes some theoretical arguments that suggest that the SNoW-based learning framework has important advantages for visual learning tasks. Given good experimental results with SNoW on face detection, the main contribution of this work is in providing an explanation for this phenomena - by giving a theoretical analysis and validating it with real world data - and providing ways for thinking about good representations for visual learning tasks. We have shown that SNoW, being based on a multiplicative update algorithm, has some nice generalization properties compared to other learning algorithms used in this domain. On the other hand, algorithms that are based on additive update algorithms, like perceptrons and SVM, have some nice computational properties, stemming from the ability to use the kernel trick and to avoid computing with in very high dimensional data. We then argue that SNoW, with its ability to handle variable size examples does not suffer from the dimensionality of the data but only from the presence of many active features in each examples. Moving to a sparse representation of images, (e.g., edges, conjunctions of those or others families of features studied in computer vision) would allow one to enjoy both worlds - a good generalization performance along with computational efficiency. We believe this to be an important direction for future research.

5. REFERENCES

[1] M. Aizerman, E. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[2] B. E. Boser, I. M. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *COLT 1992*, pp. 144–152.

[3] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other Kernel-based learning methods.* Cambridge University Press, 2000.

[4] Y. Freund and R. Schapire. Large margin classification using the perceptron. *Machine Learning*, 37(3):277–296, 1999.

[5] T. Graepel, R. Herbrich, and R. C. Williamson. From margin to sparsity. In *NIPS 13*. MIT Press, 2001.

[6] B. Heisele, T. Poggio, and M. Pontil. Face detection in still gray images. AI Memo 1687, MIT AI Lab, 2000.

[7] J. Kivinen, M. K. Warmuth, and P. Auer. The Perceptron algorithm vs. Winnow: linear vs. logarithmic mistake bound when few input variables are relevant. *Artificial Intelligence*, 1-2:325–343, 1997.

[8] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[9] N. Littlestone. Redundant noisy attributes, attribute errors, and linear threshold learning using winnow. In *COLT 1991*, pp. 147–156.

[10] C. Papageorgiou and T. Poggio. A trainable system for object detection. *IJCV*, 38(1):15–33, 2000.

[11] F. Rosenblatt. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

[12] D. Roth. Learning to resolve natural language ambiguities: A unified approach. In *AAAI 1998*, pp. 806–813.

[13] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[14] L. G. Valiant. A theory of the learnable. *CACM*, 27(11):1134–1142, 1984.

[15] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[16] M.-H. Yang, D. Roth, and N. Ahuja. A SNoW-based face detector. In *NIPS 12*, pp. 855–861. MIT Press, 2000.

[17] T. Zhang. Some theoretical results concerning the convergence of compositions of regularized linear functions. In *NIPS 12*, pp. 370–376. MIT Press, 2000.