Faces in the News

Mithun Jacob Snehit Neenakri
University of Pennsylvania
Philadelphia, PA 19104
mithunj@seas.upenn.edu snehitn@seas.upenn.edu

Abstract:

This report explores the use of the Naïve Bayes classifier and the SVM classifier as the basis for person identification. Different machine learning algorithms have previously been used for this purpose but the team's implementation of Naïve Bayes classifier for text classification and the SVM classifier for image classification gave reasonably good results when tested on the training set provided for the project. The effects of various forms of attribute selections are investigated and implemented to obtain the advanced system. It is also shown how the document model approach to the probability calculation in the Naïve Bayes approach can improve accuracy of the system and a component based approach improves the accuracy of the SVM system.

1. Introduction

The development of efficient image and text classification algorithms have been in progress over the last 20 years and still continues to be a highly important endeavor for a number of applications. This report showcases our attempt at making a system which can recognize the a news personality based on the image/text data with a total accuracy of around 83%.

We are using two classification systems in tandem to determine the identity of the person in the given news clip. SVM has been used for face recognition and a Naïve Bayes classifier has been implemented for text classification.

2. Text classification

Text classification has always been the foreground for all machine learning algorithms to test themselves on. Text categorization is the problem of assigning predefined categories to free text documents. There are several similar tasks such as text filtering and routing. All of these tasks require text classifiers that decide which class is more relevant to a given document or which document is more relevant to a fixed user interest. Thus, text classifiers should be able to rank categories given a document and rank documents given a class. A growing number of statistical learning methods have been applied to these problems in recent years, including nearest neighbor classifiers, Naive Bayes classifiers, and support vector machines, etc. Among these methods, Naive Bayes text classifiers have been widely used because of its simplicity although they have been reported as one of poor-performing classfiers in text categorization task. Since several studies show that naive Bayes performs surprisingly well in many other domains, In this project we have used it to implement text classification.

2.1. The Naïve Bayes Text Classifier

For text classification, a naive Bayes is used to calculate probabilities of a class given a document. Similar to the traditional probabilistic information retrieval model, what we are concerned about is not the probability itself calculated by the naive Bayes formula but the ability to give higher scores to more relevant documents given a class or to more relevant classes given a document.

A Naive Bayes classifier models a joint distribution over a label Y and a set of observed random variables, or *features*, **{F_1,F_2,....F_n}**, using the assumption that the full joint distribution can be factored as follows:

$$P(F_1...Fn,Y) = P(Y)\prod_i P(F_y \mid Y)$$

To classify a datum, we can find the most probable class given the feature values for each pixel:

$$P(y \mid f_1,...., f_m) = \frac{P(f_1,...., f_m \mid y)P(y)}{P(f_1,...., f_m)}$$

$$= \frac{P(y)\prod_{i=1}^{m} P(f_i \mid y)}{P(f_1,...., f_m)}$$

$$\arg\max_y P(y \mid f_1,...., f_m) = \arg\max_y \frac{P(y)\prod_{i=1}^{m} P(f_i \mid y)}{P(f_1,...., f_m)}$$

$$= \arg\max_y P(y)\prod_{i=1}^{m} P(f_i \mid y)$$

Because multiplying many probabilities together often results in underflow, we will instead compute log probability which will have the same argmax:

$$\arg\max_y P(y \mid f_1,...., f_m) = \arg\max_y (\log(P(y)) + \sum_{i=1}^{m} \log(P(f_i \mid y)))$$

2.1. Parameter Estimation

Our naive Bayes model has several parameters to estimate. One parameter is the prior distribution over labels (digits, or face/not-face), **P(Y)**.

We can estimate **P(Y)** directly from the training data:

$$\hat{P}(y) = \frac{c(y)}{n}$$

where ***c(y)*** is the number of training instances with label y and n is the total number of training instances.

The other parameters to estimate are the conditional probabilities of our features given each label y: **$P(F_i \mid Y = y)$**. We do this for each possible feature value $(f_i \in 0,1)$.

$$\hat{P}(F_i = f_i \mid Y = y) = \frac{c(f_i, y)}{\sum_{fi} c(f_i, y)}$$

where **$c(f_i, y)$**is the number of times pixel **F_i** took value **f_i** in the training examples of class Y.

2.2. Smoothing

Your current parameter estimates are *unsmoothed*, that is, you are using the empirical estimates for the parameters **$P(f_i\ y)$**. These estimates are rarely adequate in real systems. Minimally, we need to make sure that no parameter ever receives an estimate of zero, but good smoothing can boost accuracy quite a bit by reducing overfitting.

The basic smoothing method we'll use here is *Laplace Smoothing* which essentially adds k counts to every possible observation value:

$$P(F_i = f_i \mid Y = y) = \frac{c(F_i = f_i, Y = y) + k}{\sum_{f_i} c(F_i = f_i, Y = y) + k}$$

If k=0 the probabilities are unsmoothed, as k grows larger the probabilities are smoothed more and more. You can use your validation set to determine a good value for k (note: you don't have to smooth P(C)).

This concludes the discussion regarding the baseline system implemented for text classification using the Naïve Bayes text classifier.

2.3. Advanced system

For the advanced system we used the document model approach to the Naïve Bayes calculation.

Document Model Based Parameter Estimation

Our first hypothesis is that the occurrence of term *wk* in short document should be treated as more weighted event than that in long document. For example, in Fig. 1, it is our intuition that w2 is more likely term in class *cj* than w1, because short documents usually have only the important terms and w2 appears twice in the short document. In addition, log-odds ratio of terms should increase according to the number of documents which belog to *cj* and have the term. From this point of view, w3 should be low-weighted. We suggest a document modelbased parameter estimation(DMBE), which calculates class model parameters by expanding it with document model parameters.

DMBE estimates class model parameters as follows:

LHki means likelihood of *wk* in *di* and *P*(*di*|*cj*) is considered as a uniform distribution. However, it may also reflect the importance of specific training documents if one use different *P*(*di*|*cj*) for each training document. Document model parameter *LHki* may be estimated with maximum likelihood estimation(MLE). For the same purpose with the traditional multinomial naïve Bayes, we estimate these likelihoods as follows rather than MLE:

$$LH_{ki} = \begin{cases} \frac{\theta + TF_{ik}}{\theta \cdot |V| + dl_i} & \text{if } w_k \in d_i \\ \frac{\theta}{\theta \cdot |V| + \text{avdl}} & \text{otherwise} \end{cases}$$

where, *dli* and avdl means document length of *di* and average document length over the collection, respectively. DMBE has two distinctive features. First, this estimation method makes a distinction between term occurrences in short documents and those in long documents. Second, it gives default weights for missing terms in each document, and finally tend to have higher log-odds ratio than the traditional estimation. As shown in Fig. 2, log-odds ratio values by DMBE is a little higher than those by the conventional estimation, and it makes the scores of the long documents more higher than that of the short documents. Another difference between two curves is that DMBE curve is more close to the normal distribution than Multinomial curve. In other words, DMBE makes more smooth distribution.

We also tried experimenting with removing words which occur many times unnecessarily like ‘is’ ‘but’ ‘as’ and other frequent words but since the training set provided for the project doesn’t provide a very high frequency of these strings, their exclusion doesn’t cause much of a change so we decided to go without it.

By including the above system in our text classification system, we were able to substantially increase the accuracy of the text classification system.

3.1. The SVM Face Classifier

The face recognition techniques used in the baseline face classifier implements a global approach to face recognition since a single feature vector which represents the whole face (Fig. 3.1) is used as input to the classifier. For the advanced system, a single feature vector representing the whole image as well as some portions of the face was used since a purely global approach would render the system highly sensitive to image variations caused by changes in the pose of the face.

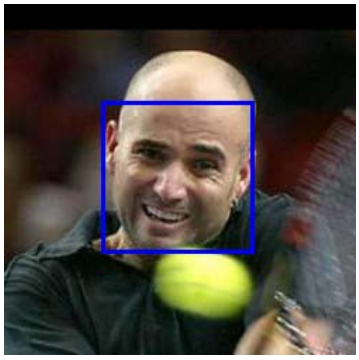


Fig 3.1. Global Approach

In order to generate the input for the advanced system, the red, green and blue pixel values representing the entire face, eyes and bridge, hair and nose were normalized in size and combined into a single feature vector. Since the bounding face box in each image was provided, isolating these individual features (approximately) involved running a number of tests on various images, until a set of bounding rectangles for each feature relative to the face box was determined. The features included in the input to the advanced system are shown in Fig. 3.2.

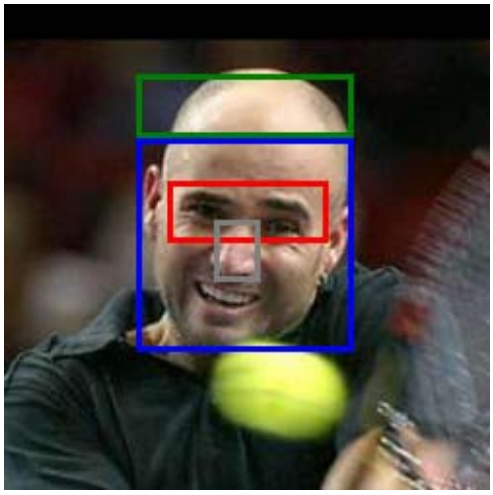


Fig 3.2. Global-Component Approach

.A one-versus-one strategy is used so that for the 10-person task, 45 SVMs are trained to differentiate one class from every other.

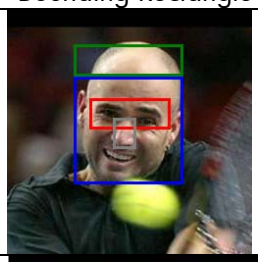
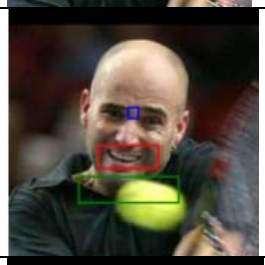
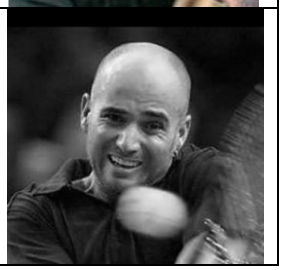
3.2. Kernel Selection

By exhaustively testing different kernels, a linear SVM classifier with C = 1worked much better than a range of non-linear RBF, sigmoid or polynomial kernels.

3.3. Feature Selection

A number of features were considered for the advanced system and are listed below in Table 1 along with their bounding rectangles.

Table 1

No.	Feature	Bounding Rectangle
1 2 3 4	Hair Face Eyes-Bridge Nose	
5 6 7	Bridge Mouth Clothes	
8 9	Grayscale HSV	

Features 1,2,3 and 4 were finally selected for the component based approach. Features 5,6, and 7 turned out to be inaccurately positioned most of the time and generally decreased the accuracy of the system. Grayscale and HSV features did not perform well either.

The selected features were combined into a single vector and normalized using the following:

$$MAX = \max(sample) MIN = \min(sample)$$
$$DEL = MAX - MIN$$
$$if\ DEL$$
$$for\ i\ in\ sample$$
$$sample = \frac{i - MIN}{DEL}$$

Storing the MAX, MIN and DEL values before normalizing the feature vector greatly reduced processing speed.

4. Results

In the combined system, a weight of 0.644 was assigned to the scores generated by the NB text classifier and 0.356 to the scores generated by the SVM image classifier. The accuracy obtained for the baseline and advanced systems have been displayed in the Appendix for each person in our training set.

By observing the results, it is apparent that the NB classifier performs remarkably well in the baseline as well as the advanced system whereas the SVM demonstrates a marked improvement in the advanced system. The data used to plot the results have been listed in Table 2.

5. Conclusion

We achieved 83.75% overall accuracy with the combined use of the text and image classifiers in the advanced system and 79.75% for the baseline system. There is room for much improvement in the SVM classifier since it could be combined with a face detection algorithm to precisely determine the bounding rectangles of each feature instead of using approximate values.

In all our tests we, used 10-fold cross-validation with 80% training data and 20% testing data. It is apparent that using a global-component based approach greatly improved the accuracy of the SVM image classifier and that using document model based parameter estimation also improved the NB text classifier.

References

Shen, Jiang: *Improving the Performance of Naive Bayes forText Classification*, CS224N Spring 2003

Rennie, Shih, Teevan, Karge: *Tackling the Poor Assumptions of Naive Bayes Text Classifiers*, Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

Anthony, Gregg, Tshilidzi: *Image Classification Using SVMs: One-against-One Vs One-against-All,* Department of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg.

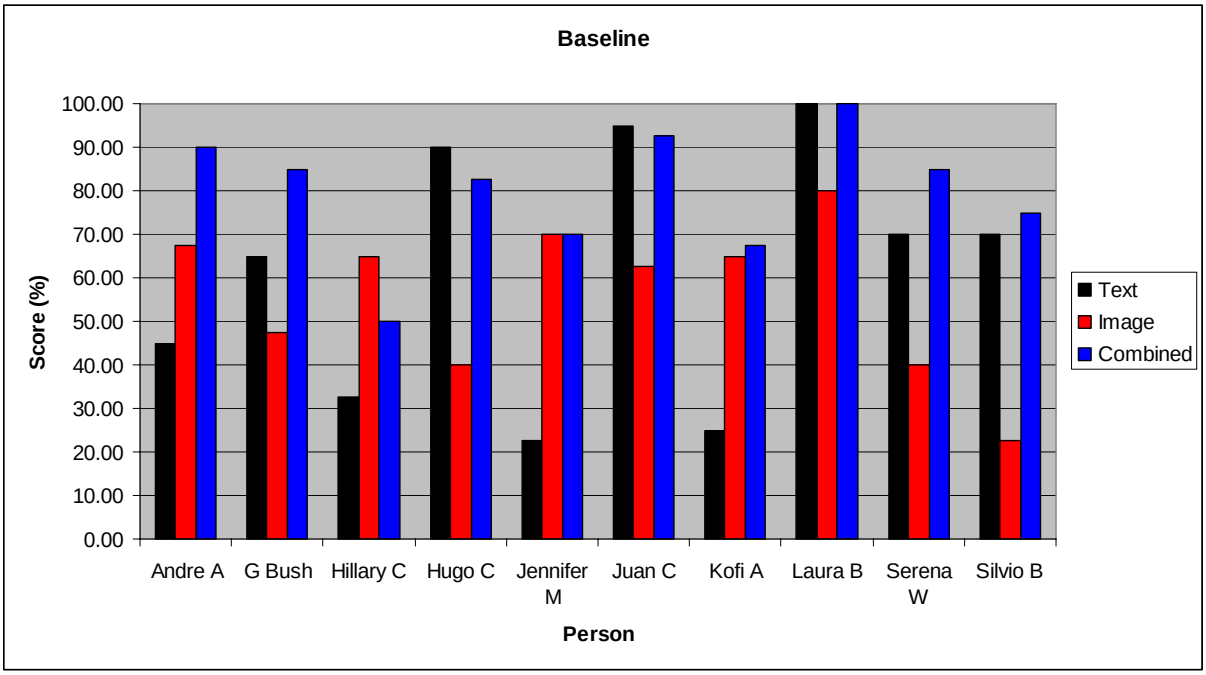
Heisele,Ho,Poggio:*Face Recognition with Support Vector Machines:Global versus Component-based Approach*, Massachusetts Institute of Technology, Center for Biological and Computational Learning

C. Cortes and V. Vapnik. Support vector networks. Machine Learning, 20:1–25, 1995.

Appendix A:

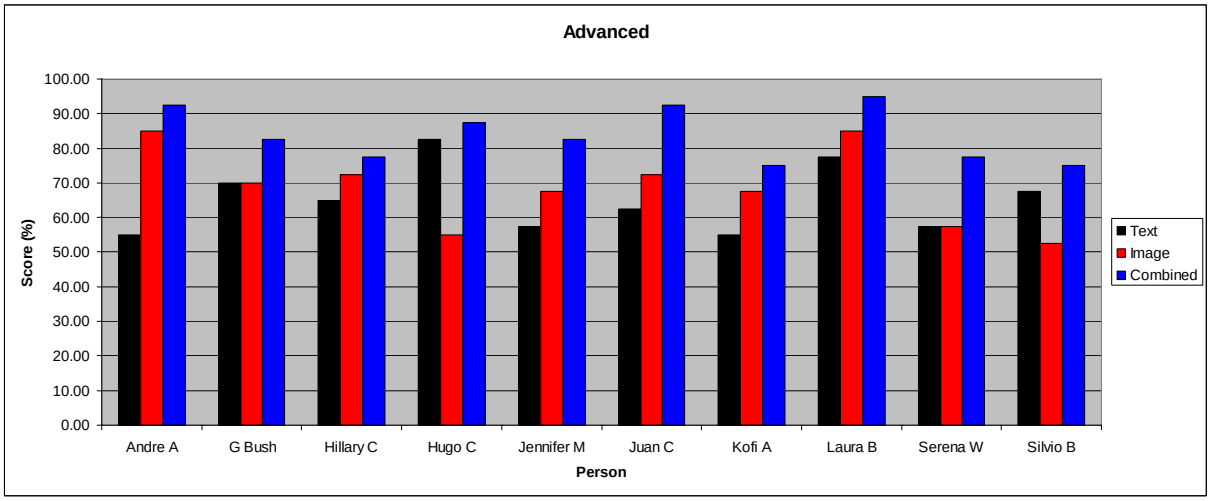
Baseline System Results:

	Text	Image	Combined
Andre A	45.00	67.50	90.00
G Bush	65.00	47.50	85.00
Hillary C	32.50	65.00	50.00
Hugo C	90.00	40.00	82.50
Jennifer M	22.50	70.00	70.00
Juan C	95.00	62.50	92.50
Kofi A	25.00	65.00	67.50
Laura B	100.00	80.00	100.00
Serena W	70.00	40.00	85.00
Silvio B	70.00	22.50	75.00



Advanced System Results:

	Text	Image	Combined
Andre A	55.00	85.00	92.50
G Bush	70.00	70.00	82.50
Hillary C	65.00	72.50	77.50
Hugo C	82.50	55.00	87.50
Jennifer M	57.50	67.50	82.50
Juan C	62.50	72.50	92.50
Kofi A	55.00	67.50	75.00
Laura B	77.50	85.00	95.00
Serena W	57.50	57.50	77.50
Silvio B	67.50	52.50	75.00



Accuracy Comparison:

	Text	Image	Combined
Baseline	61.50	56.00	79.75
Advanced	65.00	68.50	83.75