CIS520 Project Report: Team k-Center

Kook Jin Ahn[*] Huen Oh[†]

December 2, 2009

1 Overview

Implemented Classifiers: We implemented the following classifiers:

1. Gaussian Naive Bayes,
2. Kernel Logistic Regression, and
3. AdaBoost(Weighted Naive Bayes, Decision Stumps).

Our goal in this project was to test as many classifiers and features as possible. For that, we created a small set of training data which consists of 1/10 of training set. This data was used for check the correctness of implementations and the running time before we perform actual cross validation. If the result revealed a problem with either accuracy or running time, we did not perform cross validation.

In Section 2 and Section 3, we discuss the details about features and classifiers we implement. We focused on blogs data because most blog features take less time to generate than images features. Also the number of features in blogs is usually smaller than the number of features in images. So it is easier to test classifiers' performance.

Cross Validation: We evaluated our progress with cross validation results. We split the data into 5 folds. We trained a classifier with 4 of them and test on the other fold. We repeated the test on all 5 folds and averaged the results. Since we reduced the size of training data for cross validation, we expected that the actual test accuracy would be higher than the cross validation result.

The cross validation code that we used is different from the cross validation code for SVM given in starter kit. For blogs data, we had to remove words that appeared only in the test fold to be correct. For images data, SVM cross validation code partitions data randomly regardless of names of the input. This enables classifiers to overfit to the training data and consequently gives a higher cross validation result than the actual test result. We partition the data based on the names.

Even though we use the same algorithm and parameters without randomization, the cross validation results vary around two to three percentage points depending on the partition. Also some classifiers chooses parameters with on their own cross validation processes. For example, we found that the baseline classifier for images data sometimes chooses a different parameter. We picked the fastest algorithm when the cross validation result is similar to each other so that we can run more experiments and optimize the parameters.

[*]E-mail: kookjin@cis.upenn.edu
[†]E-mail: huen@seas.upenn.edu

2 Blogs

Features: We added 2-grams(two consecutive words) as binary features. The problem was that we had too many features. Kernel Logistic Regression and SVM couldn't handle those features due to memory limitation. In addition, PCA was also not applicable for the same reason. So we first reduced the number of words and 2-grams based on the number of occurrences. Another method we tried is entropy-based method. We computed the conditional entropy of each word and picked top 750 words. Since 2-grams are highly related to features and the total number of features is still high for SVM, we applied PCA on the data. Table 1 shows the result of PCA.

positive	negative
'and the', 'the', 'DON'T'	'I don't', 'sad I', 'I NOT'
'the', 'sad the', 'and the'	'concious', 'immigrants', '. immigrants'
'looking', 'Whooo', 'P'	'personally', 'Modules', 'people'
'Whooo', 'our', 'mother'	'I NOT', 'linking', '2 a'
'I do', 'am', 'do',	'ga', 'now', 'Texas'

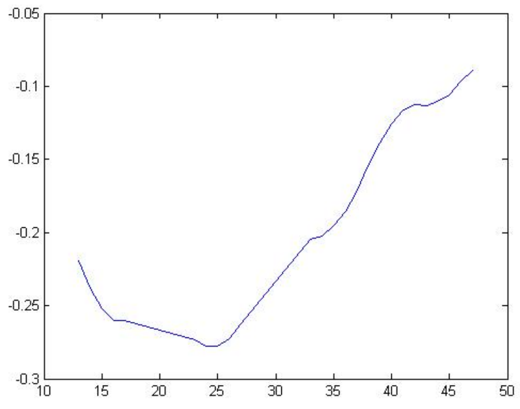
Table 1: The first five dimensions by PCA on bag of words and 2-grams. Each column indicates words and 2-grams that are highly correlated to one dimension.

We also tested porter stemmer [2] before we apply the above process. However, the result was not better while the running time increases. Our guess was that one important criteria for distinguishing age or gender is frequency of uppercase words and grammatical correctness. For example, Although 'DON'T' is related to 'I don't', they are actually in the opposite direction of one dimension in the result of PCA. When we used a stemmer, we used lowercases. It might have hurt the accuracy by removing the difference between 'DON'T' and 'don't'.

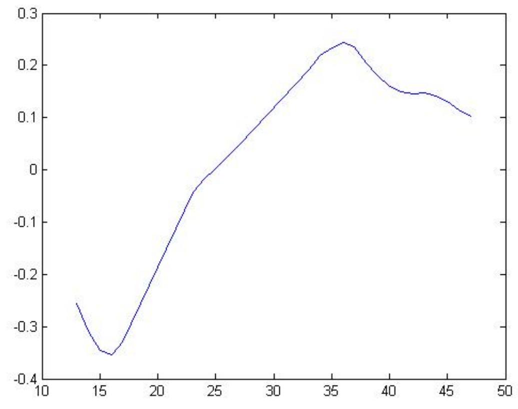
Another feature we tried was the probability distribution of words over age. We classified ages into three groups. But the actual training data gave us more accurate data. A blog of 17-year-old person gave some information about 23-year-old person. We constructed a map $f_X(Y) = P(X|Y)$ where X is a binary feature that indicates if a word is in the blog and Y indicates exact ages. We smoothed the function $f_X(Y) = P(X|Y)$ by adding virtual weight on nearby ages, i.e., if we are given 17-year-old person's blog that contains X, we also add a smaller weight to 16-year-old, 15-year-old, and so on. In order to get a more meaningful data, we applied PCA on $\log P(X|Y)$. We applied log function because we wanted to add those values at the end and multiplication of probabilities made more sense than addition of probabilities. Figure 1 shows the results. This feature is very fast to generate and it describes the training set very well even with a small number of dimensions. With only 15 dimensions, we obtained over 95% training accuracy. On the other hand, it is susceptible to overfitting too. The cross validation result using this feature was not much better than others.

We also implemented string kernel. But it was discarded due to its speed. Even C language implementation of string kernel takes around 5 minutes to train for a small training set. It would take approximately 500 minutes for the real training set. We even discarded rarely used words to improve the performance while not losing much information. However, the accuracy was much worse than the baseline and the speed was still a problem.

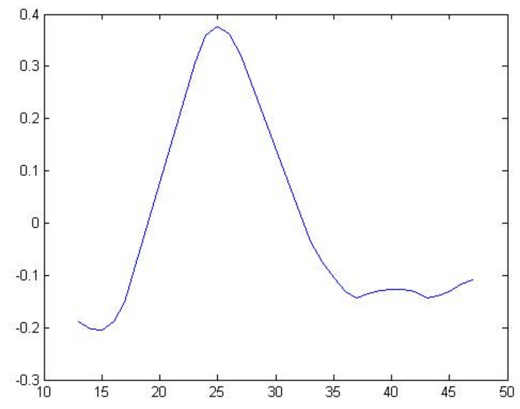
Classifiers: We tested SVM, Kernel Logistic Regression, Gaussian Naive Bayes, AdaBoost with weighted Naive Bayes, and a combination of Gaussian Naive Bayes and Naive Bayes. Kernel Logistic Regression was too slow to use. AdaBoost with weighted Naive Bayes did not give a good



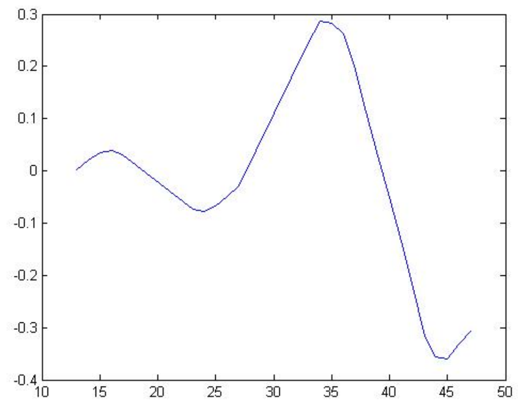
(a) Interwined, jousts, knights



(b) organizations, 1993, husband



(c) roommate, addictive, Manhattan



(d) rates, i'm, hoping

Figure 1: Probability of word occurrence over ages.

result on a small test set.

Generally, SVM was the best when we used the result of PCA. The result obtained from Gaussian Naive Bayes on PCA data was slightly worse than baseline(Naive Bayes). We also combined Gaussian Naive Bayes and Naive Bayes. If they did not agree, we used the classifier that gives higher $P(X, Y)$. This sightly improved the baseline but not statistically meaningful amount.

Cross Validation Results: Table 2 shows the cross validation results for features and classifiers.

3 Images

Features: For images, we mainly tried three features. The first one is canny edge detection. We detected edges in a image and scale the image into a smaller image. After that, we gave the result with RGB values. The second one is normalized RGB. Instead of the original RGB values, we normalize each channel with $(x - \mu)/\sigma$ where μ is the mean value and σ is the standard deviation of the channel values. Unfortunately, this worsened the result.

Classifiers+Features	Age	Gender
Baseline	66.82	67.24
SVM+BW	69.94	66.53
SVM+PBG(e)	71.00	68.41(*)
SVM+PAD	71.41(*)	N/A
GNB+PBG	65.29	64.76
GNB+NB+PBG	67.29	67.41

Table 2: Cross Validation Results. Classifiers: SVM=Support Vector Machine, NB=Naive Bayes, GNB=Gaussian Naive Bayes. Features: BW=Bag of Words, PBW=PCA on BW, PBG=PCA on BW and 2-grams, (e)=Entropy version, PAD=Age Distribution of Words. (*)=Submitted.

The third one is age distribution of pixels in edge detection results. Once we detect edges whose result is a set of binary pixels, we can treat them as binary features and process them as in blogs data to reduce dimensions. As in the previous case, it overfits to the training set too easily. In this case, we combined it with RGB data in order to avoid overfitting a little. Figure 2 shows some dimensions of the result. Faces represents corresponding distribution of pixels. One interesting part we found is that very old people have slimmer faces according to this. The first dimension excludes people who are over 70 years old. The corresponding image shows that their chins are concentrated near edges of the box. On the other hand, the second and the third dimensions have chins distributed about 10 pixels inside the box.

For age, age distribution worked best. For gender, edge detection result gave the best cross validation result. We also added some virtual examples such as flipped images and rotated images. But it also worsened the cross validation result a little bit while slowed down the training due to increase in the number of examples.

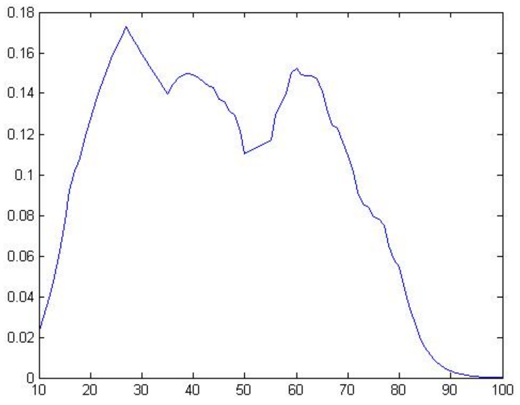
Although we did not cross validate the result, we also tested Leung-Malik filter bank [1]. Each filter gives a set of numbers for an image. We use these numbers as features. A problem with these features is that it takes too long to generate and we do not know which filter would be important. So we could not test all the filters. For these filters, we used AdaBoost with decision stumps.

Classifiers: We mainly used SVM since our KLR implementation was too slow and we did not have good features for Naive Bayes and Gaussian Naive Bayes. As mentioned above, we tested AdaBoost with Leung-Malik filter bank [1]. However, it was slow while it almost did not give any information. Gender accuracy using AdaBoost was about 50% which means that it is basically same as a random classifier.

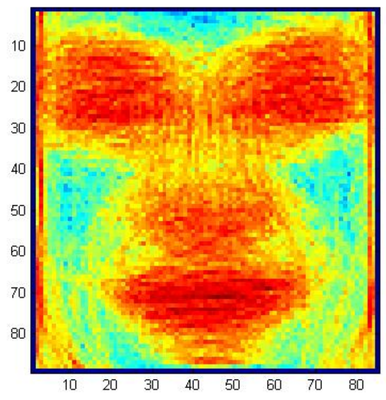
Cross Validation Results: Table 3 shows the cross validation results for features and classifiers.

Classifiers+Features	Age	Gender
Baseline	45.70	74.18
SVM+E	48.16	77.66(*)
SVM+NC	47.34	76.64
SVM+PAD	50.00(*)	N/A

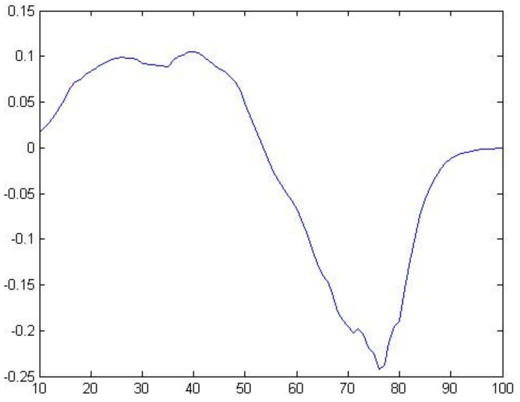
Table 3: Cross Validation Results. Features: E=Edge Detection(Canny), NC=Normalized RGB, PAD=Age Distribution of Edge Detection Result. (*)=Submitted.



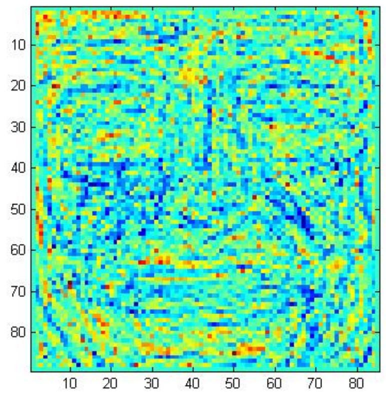
(a) The first dimension(distribution)



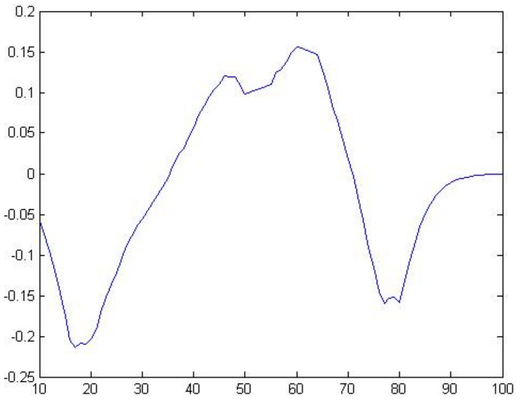
(b) The first dimension(face)



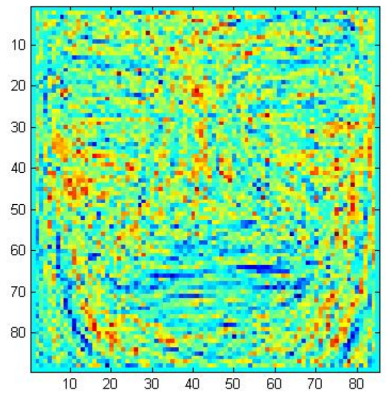
(c) The second dimension(distribution)



(d) The second dimension(face)



(e) The third dimension(distribution)



(f) The third dimension(face)

Figure 2: Probability of pixel occurance over ages.

References

[1] O. U. V. G. Group. Texture classification. `http://www.robots.ox.ac.uk/~vgg/research/texclass/filters.html`.

[2] M. Porter. The porter stemming algorithm. `http://tartarus.org/~martin/PorterStemmer/`.