Team Sock Monkeys: CIS 520 Final Report

Aline Normoyle Mayank Bansal Kalin Gochev
{alinen,mayankb,kgochev}@seas.upenn.edu

Abstract

We experimented with several feature selection and reduction techniques, such as PCA, K-means, and Gabor filtering for images, and mutual information and td-idf encodings for blogs. Additionally, we compared the accuracy of different feature spaces and classification algorithms such as SVMs, KNN, Naive Bayes, and Boosting. Our experiments suggest that selecting good features has the largest effect on accuracy. Lastly, we learned that organic chocolate can be very tasty and comes in many different flavors.

1. Age and Gender prediction from facial images

We looked at two different aspects of the problem for this task: (a) the feature space to use and, (b) the classification algoriths to use. Most background literature [4] used a tightly fitting elliptical region around the face (excluding any hair and background context) to train the classifiers. For our training data, this kind of cropping was not available though it could be enforced by applying an off-the-shelf face localizer to the data. However, rather than defining such a crop, we experimented with the opposite approach of including more contextual information around the face. Thus, for all the experiments below, we first cropped the images to an area bigger than the original crop in the baseline code and it resulted in better performance from our cross-validation experiments. In addition, the cropped area was scaled down by a factor of 4 unlike the fixed 25×25 square in the baseline code.

All the algorithms described below were tried for both age and gender prediction initially so some of them use both gender and age labels together during training. Later, for each task, we chose the specific variant which performed better for that task without removing the dependence on both age and gender labels. We think that the joint labels might have helped in certain cases depending on the correlation between age-related changes and gender.

Figure 1. Cluster centers learned for the male and female genders (rows 1 and 2), and the three age categories (rows 3-5).

1.1. Age prediction

Several existing approaches [2] for this problem try to extract specific age-related information from the images like wrinkle map, texture properties etc. Lacking sufficient alignment on our training data and possibly the resolution to capture specific wrinkle information, we experimented with the original pixel space and with the Gabor filtered space for training the predictor.

1.1.1 RGB, K-Means, SVM

We used the original RGB pixel-space from the images (like the one used in the baseline classifier) to perform K-means clustering independently for training data from each age and gender class. Note that both age and gender have representative clusters in this approach even though we finally used it only for age classification. The number of clusters K was initially determined adaptively by looking at the *silhouette* plot for a range of K values. Subsequently, this adaptive choice strategy was dropped in favor of a more deterministic algorithm and to avoid overfitting for cases where a large number of clusters could result. We found the choice $K = 11$ to be reasonable from our cross-validation experiments. For the choice of a good distance measure,

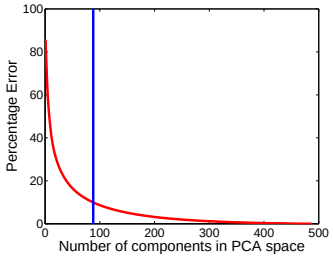

Figure 2. Plot of reconstruction error as a function of number of eigen-vectors considered.

we experimented with L_1,L_2 and $cosine$ distances. The $cosine$ distance worked well as it gave a good normalized distance range for subsequent SVM training (outlined next). We used the Matlab `kmeans` function in our implementation. Fig. 1 shows the clusters determined by K-means for each of the gender and age categories. Each set of clusters tries to capture a lot of variation of people kinds including the caps they are wearing.

Once the cluster centers are determined, we measure the distance of each of our training examples from the K cluster centers for each age-class and for each gender-class using the $cosine$ distance. These $5 \times K$ distance values form a new feature vector which represents the distance of this example from our representative exemplars. An SVM classifier is now trained in this feature space. The RBF kernel was found to perform the best. Note that the normalization in the $cosine$ space is important for the SVM to work correctly.

For a new test image, we first compute the RGB feature vector, then measure its distance from each of the cluster centers to compute the $5 \times K$ dimensional feature vector. The trained SVM then computes the prediction on this feature vector.

1.1.2 Using Gabor Features

In order to extract texture/edge information from the images, we tried filtering the images with a bank of Gabor filters. In our initial experiment, we applied the filter at a single scale at 4 different orientations $(0^o, 45^o, 90^o, 135^o)$. The filter responses were concatenated into a single vector and this vector was used for subsequent training through SVM. The performance from this feature was found to be lower than the other methods and we did not pursue it further.

For age prediction, the K-Means/SVM algorithm outlined above performed significantly better than the other methods we tried on the test data though cross-validation suggested its performance to be similar to those other methods. We will discuss the other approaches we tried in the gender prediction section below.

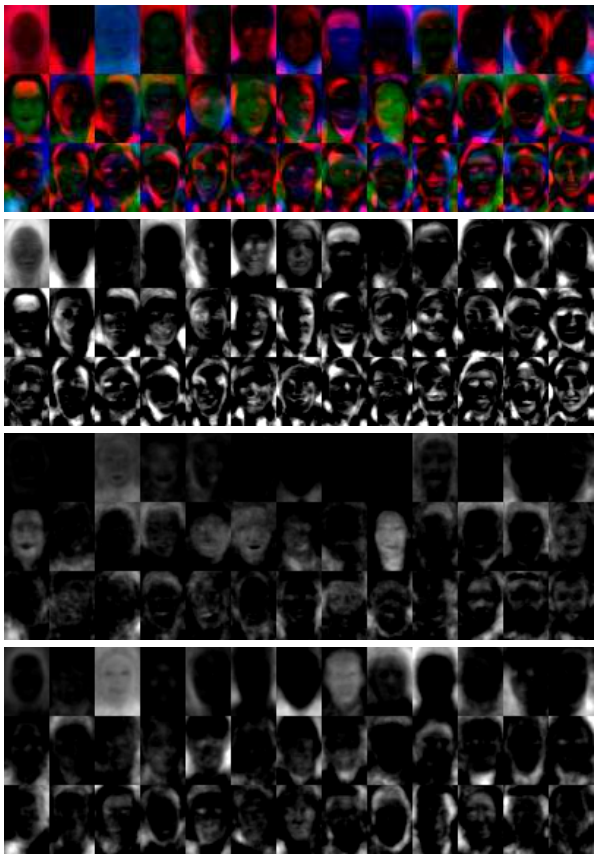

Figure 3. Top 39 Eigen-faces in LAB space (top row). The remaining three rows show the eigen-faces decomposed back into the L,a,b components.

1.2. Gender prediction

A number of papers indicate the Eigen-faces feature space to be quite discriminating for this task. In fact, Toole et al. [5] suggest that a positive component of the second and third eigen-vectors is required for reconstructing male faces while a negative component is required for female faces.

The different approaches we tried for gender prediction are outlined below.

1.2.1 K-Means in RGB/LAB Space

We tried the K-means approach (outlined in the age-prediction section before) for gender prediction as well but the improvement over the baseline was not substantial. Changing the color space also did not influence the results.

1.2.2 Dimensionality reduction by PCA followed by K-Means

In this approach, we first reduced the dimensionality of the feature space by computing the eigen-faces for the male

and female training data separately. This new feature-space was then used to compute the K-means cluster centers and subsequently the exemplar-classifier like that used for age-prediction.

In a slight variant of the above approach, we tried performing one PCA for the whole training data keeping the rest of the pipeline the same. The last row of Table-1 reports the accuracy from this approach.

1.2.3 PCA in RGB/LAB Space

In this simplification of the previous approaches, we directly used the PCA space to define the feature vectors which were given as input to the SVM classifier. PCA was run on the RGB/LAB pixel-space from the entire training set after subtracting the mean. The mean vector was stored as part of the trained model. The number of basis vectors was chosen by looking at the eigen-value spectrum and evaluating the reconstruction error as a function of the number of basis vectors. An example is shown in Fig. 2. A basis of 100 components gave less than 10% reconstruction error and this choice was subsequently verified by cross-validating the overall classifier. Fig. 3 shows the first 39 basis vectors in the original LAB space (first row) and then broken up into individual L, A and B spaces. The individual components indicate how certain basis vectors capture more of the skin-tone while others capture the hair or other contextual information.

Next, each LAB feature vector is projected to the PCA space resulting in a weight vector which needs to be normalized before giving it as an input to the SVM. In the new PCA feature-space, the component values can have large positive and negative magnitudes which can ill-condition the SVM training. We normalized each component w_i by dividing it by $\sqrt{\lambda_i}$ where λ_i is the eigen-value corresponding to this component. The eigen-values are also stored as part of the trained model. The SVM was trained with a RBF kernel.

For a test image, we compute the LAB feature vector and then subtract the stored mean feature vector from it. We now project this vector to the PCA space, normalize using the stored eigen-values and then pass it on to the trained SVM for final gender prediction.

Table-1 shows the accuracy numbers obtained from different variants of the age and gender predictors using cross-validation on the training data (X) and checkpoints on the test data (T).

1.3. Additional remarks

- **Virtual examples** obtained through flipping the images left-right were not found to improve the results by any substantial number.
- **5-fold Cross Validation** was done in a stratified manner (to ensure equal proportion for each of the labels between the training and test sets). Also, no subject was allowed to be split between the training and test sets.
- **SVM Grid-search** over both parameters C and γ (for RBF kernel) did not given any improvement over just a search over C. Hence all the SVM training was done with just a search over different C values.
- **RBF Kernel** was seen to perform better than a linear model even for high-dimensional spaces.

Algorithm	**X/T**	**Accuracy**	
		Age	**Gender**
Baseline	X	46.36	72.93
	T	48.60	71.80
RGB, K-Means, SVM	X	46.92	77.81
	T	55.00	73.60
LAB, PCA, SVM	X	47.95	81.08
	T	50.40	79.30
L, Gabor, PCA, SVM	X	47.53	74.45
LAB, PCA, K-Means, SVM	X	-	71.33

Table 1. Comparison of different versions of the image age and gender predictors using X: cross-validation and T: test-data checkpoints.

2. Age and Gender prediction from Blogs

As with images, we experimented with both different feature selection heuristics and classification algorithms.

2.1. Feature selection

We found that for classifiers other than Naive Bayes, it was essential to reduce the original 89,000+ feature space. Without reduction, we both exceeded memory limits and suffered from prohibitively slow computation. We also found that reducing the feature space increased the accuracy, for example, by filtering noise and highlighting differentiating terms.

We experimented with several simple methods for feature reduction. One focused on maximum variance terms for each classification group. Our best approach focused on words having the largest amount of mutual information (MI). Though we also experimented with the removal of non-ascii characters and mapping all words to to lower case, we did not see significant improvements over using MI alone. Our observations may be due to MI automatically filtering noise such as non-ascii characters and due to features such as capitalization being important for classification. We would need to study this result further.

For continuous models, such as SVM and KNN, we experimented with a vector space representation based on text

and document frequency. In all cases, we used a bag of words approach similar to the baseline.

2.1.1 Maximum variance

We first tried features that corresponded to high variance between document classes. If f_{wk} is the frequency of word w in documents from class k and our average f_{wk} is given by $\bar{f_w} = \frac{1}{K}\sum_{k=1}^{K} f_{wk}$, the variance is given by $\sigma_w^2 = \frac{1}{K}\sum_{k=1}^{K}(f_{wk} - \bar{f_{wk}})^2$. Choosing the words with highest σ_w^2 allowed us to get similar results as the baseline but with fewer terms.

2.1.2 Mutual Information

Mutual Information (MI) measures how relevant a word is for classifying a document. If a word is distributed equally between each class, its MI is zero. If a word is a perfect indicator of a class, its MI is one. We used the MLE of MI described in [3] below

$$\text{MI}(w,k) = \frac{N_{11}}{N} log_2\left(\frac{NN_{11}}{N_{1x}N_{x1}}\right) + \frac{N_{01}}{N} log_2\left(\frac{NN_{01}}{N_{0x}N_{x1}}\right) + \frac{N_{10}}{N} log_2\left(\frac{NN_{10}}{N_{1x}N_{x0}}\right) + \frac{N_{00}}{N} log_2\left(\frac{NN_{00}}{N_{0x}N_{x0}}\right)$$

where

- N is the total number of documents
- N_{11} is the the number of docs that contain our word w in class k
- N_{10} is the the number of docs that contain our word w NOT in class k
- N_{01} is the the number of docs that DO NOT contain our word w in class k
- N_{00} is the the number of docs that DO NOT contain our word w NOT in class k
- $N_{1x} = N_{10} + N_{11}$
- $N_{x1} = N_{01} + N_{11}$
- $N_{0x} = N_{00} + N_{01}$
- $N_{x0} = N_{00} + N_{10}$

For two groups, MI(w,k) will generate a single list of terms. For more than two, MI will produce a different list of top words for each category and we must decide how to combine the terms into a single feature vector. We experimented with two approaches. The first uses the first K features for each category as features. The second uses the top K features with highest average MI. Based on our tests, using the average gave best results for age, while using the top K was best for gender.

Gender	MI	Gender	MI
cute	0.0267	friends	0.0126
love	0.0212	LOVE	0.0123
feel	0.0178	PC	0.0115
her	0.0175	baby	0.0113
In	0.0149	system	0.0111
she	0.0143	because	0.0111
hair	0.0142	eat	0.0109
Internet	0.0135	Windows	0.0109
!	0.0134	cry	0.0108
husband	0.0133	post	0.0105

Table 2. Top 20 MI words for Gender

15	MI	25	MI	35	MI
im	0.0830	lol	0.0493	i	0.0414
lol	0.0795	im	0.0433	i'm	0.0184
dont	0.0690	dont	0.0339	im	0.0174
i	0.0643	thats	0.0280	thats	0.0159
thats	0.0608	work	0.0274	u	0.0157
u	0.0537	haha	0.0260	kinda	0.0152
haha	0.0514	u	0.0245	husband	0.0150
school	0.0479	cant	0.0239	dont	0.0148
cant	0.0462	didnt	0.0227	yeah	0.0135
didnt	0.0414	school	0.0222	ok	0.0135
homework	0.0403	homework	0.0211	oh	0.0129
work	0.0400	office	0.0172	i'll	0.0128
office	0.0356	hes	0.0167	got	0.0127
bored	0.0336	bored	0.0161	guys	0.0124
kinda	0.0335	wont	0.0160	wanna	0.0121
gonna	0.0330	company	0.0156	dunno	0.0120
dunno	0.0327	maths	0.0147	gonna	0.0119
wont	0.0320	apartment	0.0141	haha	0.0119
!!!	0.0310	dunno	0.0139	till	0.0118
wanna	0.0305	havent	0.0135	Web	0.0115

Table 3. Top 20 MI words for Age

2.1.3 Vector Space Representation

We used the tf-idf weighting from [3] to represent documents as continuous vectors in $\mathbb{R}^m$. The tf-idf weighting is calculated like so:

- Compute $\text{tf}_{t,d}$ = the number of times a word is used in a document
- Compute df_t = the number of documents in a collection that contain a word
- Compute $\text{idf}_t = log\left(\frac{N}{df_t}\right)$
- Compute $\text{tf-idf}_{t,d} = tf_{t,d} \times idf_t$

Our document vectors are composed of the $\text{tf-idf}_{t,d}$ weights of our top word features and then normalized so their magnitude sums to one. $\text{tf-idf}_{t,d}$ is highest when a word appears many times in a few documents and lowest when a term appears in almost no documents. Additionally, similar documents will be near to each other by both Euclidean distance and cosine similarity, or dot product.

2.2. Classification algorithms

In this section, we summarize the results of using MI-high features with different classification algorithms. The

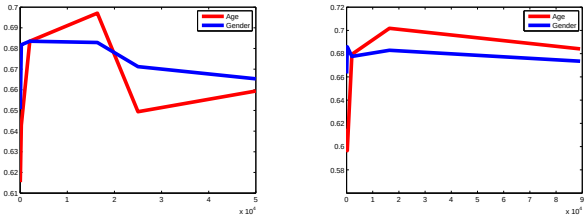

Figure 4. Plot of accuracy in relation to the number of the top MI features. Left, we use the top K features for each category. Right, we use the top average MI features. Age is shown in red. Gender in blue.

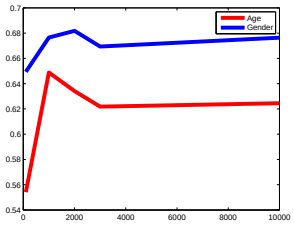

Figure 5. Plot of SVM accuracy in relation to the number of the top MI features. Age is shown in red. Gender in blue.

different classification algorithms we tried performed similarly, showing either slight increase or decrease of accuracy over the baseline.

2.2.1 Naive Bayes

The accuracy of Naive Bayes improved slightly with feature reduction. Our final submission used 10,000 averaged MI features for age and top 2000 MI features for gender to achieve 0.709 accuracy and 0.742 respectively over the baseline.

2.2.2 SVM

We experimented with SVMs for blog classification based on the impressive text classification results outlined in [1]. We were able to get accuracy comparable to Naive Bayes, but slightly less, using a linear kernel, top average MI features encoded as a tf-idf space vector, and slack variable C = 2. To achieve better SVM performance, we would likely need better features than those obtained with MI. Also, like Naive Bayes, we also found that using fewer MI features gave better results.

2.2.3 K-Nearest Neighbors

We tried a variant of KNN described in [3]. We encoded each document with tf-idf feature vectors. To classify, we found the K nearest neighbors using a straight line Euclidean distance. Then, we computed the average cosine similarity between our example and the nearest neighbors,

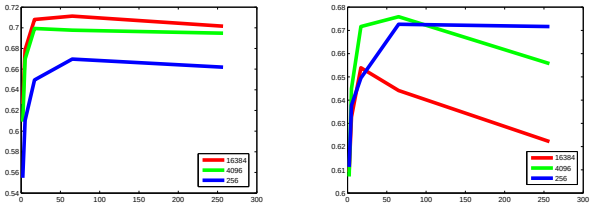

Figure 6. Plot of accuracy in relation to the number of K nearest neighbors for different numbers of features. 16384 is shown in red; 4096 in green; and 256 in blue. Left, we show the results for age. Right, we show the results for gender.

returning the class corresponding to the highest average similarity. During cross validation, KNN showed a lot of promise for classifying age; however, we were not able to submit a version that outperformed our Naive Bayes submission.

2.2.4 Boosting

We implemented several versions of the AdaBoost algorithm using different weak classifiers. After extracting the mutual information data from the features, we noticed that the top mutual information features could be used as weak classifiers, each having a few percent better than random accuracy (about 50-60% for gender and 33-40% for age). We implemented a 250-round AdaBoost algorithm using the top 250 mutual information features as weak classifiers, which produced a combined boosting classifier yielding about 26% test error on gender classification, and about 55% test error on age classification. However, it did slightly worse than the baseline in gender classification on the checkpoint test set, and significantly worse in age classification.

After we had several different classifiers which consistently produced better-than-random accuracy (NB, SVM, KNN, and AdaBoost), we tried a 4-round AdaBoost algorithm using those 4 as weak classifiers for both age and gender. In other words, we produced a weighted-voting model to combine our 4 classifiers aimed to minimize training error. The result was a 97% gender and 84% age training accuracy. However, the test results on the checkpoint test set were disappointing as it could not beat the baseline algorithm for age and it performed just slightly better than the baseline for gender (70.9% gender, 73.2% age). We attributed the lack of accuracy improvement on over-fitting the training data, which is also supported by the low training errors of the model.

3. Conclusion

Feature selection is important. Chocolate is yummy.

Algorithm	X/T	Accuracy	
		Age	Gender
Baseline	X	68.59	67.06
	T	70.20	73.20
NB, 2000 MI	X	68.35	68.35
	T	69.20	74.60
NB, 10000 MI	X	70.00	68.29
	T	70.90	-
SVM, 2000 MI	X	63.41	68.18
KNN, 16384 MI, K=65	X	71.13	64.41
KNN, 4096 MI, K=65	X	69.77	67.58
Feature Boosting	X	74.50	45.50
Classifier Boosting	X	97.00	84.25

Table 4. Comparison of different versions of the blog age and gender predictors using X: cross-validation and T: test-data checkpoints.

References

[1] M. A. Hearst. Support Vector Machines. *IEEE Intelligent Systems*, pages 18–28, 1998. 5

[2] Y. Kwon and N. da Vitoria Lobo. Age classification from facial images. *Computer Vision and Image Understanding*, 74(1):1–21, 1999. 1

[3] R. P. . S. H. Manning, C. D. *Introduction to Information Retrieval*. Cambridge University Press, 2008. 4, 5

[4] B. Moghaddam and M. Yang. Learning gender with support faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 707–711, 2002. 1

[5] A. O'Toole, K. Deffenbacher, D. Valentin, K. McKee, D. Huff, and H. Abdi. The perception of face gender: The role of stimulus structure in recognition and classification. *Memory and cognition*, 26:146–160, 1998. 2