|  |
| --- |
| **CS5785 Final Project Report: Image Classification with Ensemble Classifier** |

**Anonymous Authors**

Cornell Tech

111 8th Ave. #302, New York, NY

**Abstract**

In order to classify 3,000 images into 200 classes, we used three provided data sets, and generated an additional one based on unlabeled images. Then we trained a separate one-against-one multiclass SVM classifier on each of the four data sets and used a voting ensemble to improve the prediction. We achieved an accuracy of 54% (top 10) on the kaggle competition leaderboard.

**1 Introduction**

Image recognition has been an area of intense study in machine learning. Currently the most successful model on this task is deep convolutional neural network. Trained on 1.2 million images, AlexNet achieved a prediction accuracy of 62.5% for a 1,000-class classification problem [1].

In the CS5785 final project, we are provided with three different data sets for 3,000 images: pre-trained features from convolutional neural network, handed-tuned features from bag-of-word SIFT descriptors and binary attributes, in addition to 10k unlabeled images. We are expected to use the methods we learned in class to come up with a best classifier to classify these 3,000 images into 200 classes.

Image recognition are usually trained with millions of images such as ImageNet. One of the major challenges in our project is the scarcity of data. 3,000 training data with 200 classes left us with 15 positive training data for each class. A second challenge would be “the curse of dimensionality”. The provided feature sets have dimensions of: 4,096, 4,096 and 102. Even we use only one feature set, the number of dimensions still exceeds the number of training examples. This will make the training data even sparser. A third challenge is how to combine the three different data sets to improve performance.

In our ensemble model, we addressed these challenges by: training a separate classifier for each feature set, make use of unlabeled data, and use voting ensemble to improve the performance.

**2 The Dataset**

**2.1 Labeled and unlabeled data sets**

3k labeled training data set: the training set for the competition includes 3,000 images belonging to 200 categories. There are exactly 15 images for each category.

10k unlabeled data set: in additional to the labeled training data, we are provided with

10,000 images with no label. However, each of the 10,000 images has five captions.

**2.2 Features**

For training and testing data sets, we are provided with three set of features: 4,096 pre-trained features from convolutional neural network (CNN features); 4,096 hand-tuned features from bag-of-word SIFT descriptors with spatial pyramid (BOW features); 102 binary attributes. For the additional 10k unlabeled data, only CNN features and BOW features are available. Some details of the feature sets are listed below in Table 1.

Table. 1 Data sets overview

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CNN features** | **BOW features** | **Attributes** |
| **Type** | Continuous | Continuous | Binary |
| **Ranges** | (-25.2 , 23.3) | (0 , 0.67) | (0 , 1) |
| **Dimension** | 4,096 | 4,096 | 102 |
| **3k images** | yes | yes | yes |
| **10k images** | yes | yes | no |

CNN features: Since CNN features are pre-trained using convolutional neural network (AlexNet), the features of CNN should represent an abstraction of some patches of the original image. We tried to visually interpret the CNN features but we lack the information to make meaningful inference from the feature itself.

BOW features: as described in the competition, BOW features are hand-tuned features from bag-of-word Scale Invariant Feature Transform (SIFT) descriptors in a spatial pyramid. SIFT descriptors are basically features that trained to detect certain objects in an image, and these features are presented in a bag-of-word fashion [2].

Attribute features: the attribute features are binary and there are 102 different attributes. Each image has one or more attributes in it.

**3 Methods**

We use the 10k unlabeled images to generate a new set of features (7,509 features) for the 3k training data. We will call this new data set “tenk”. For each of the four data sets (CNN, BOW, attributes, and tenk), we trained an SVM classifier. We did 5-fold cross validation on the training set to choose the hyperparameter for the SVM classifier (including scaling options, and kernel choice), and to compute the weight of each individual classifier. Then we combine the four classifiers by a voting ensemble based on the weights and make prediction for the test set.

**3.1 Feature Preprocessing**

**3.1.1 Create “tenk” features for labeled data**

We decided to generate a feature set based on the 10k alexnet and bag of words dataset and use this feature set with an SVM classifier to generate another set of prediction probabilities. First, we stack the training, test and 10k datasets including alexnet and SIFT features. Then, we perform a spectral clustering on the dataset. The spectral clustering produces an affinity matrix which can be accessed as an attribute. This affinity matrix gives the affinity between each row of the data. We then extract the affinity of the train data with 10k data (3000x10000 matrix) and test data with 10k data (1000x10000 matrix). These affinity matrices are treated as weighting matrices for feature extraction. We use the bag of words model with the caption data to generate 7509 feature vectors, each representing a word, for the 10k dataset. Finally, to arrive at the training and test featureset, we take the dot product of the weighting matrices with the 10k bag of words featureset. Taking the dot product allows a weighted sum of the bag of words features to be taken for each row in the training and test datasets. The final training and test feature set using clustering are matrices of size (3000x7509) and (1000\*7509) respectively.

**3.1.2 Feature structure**

Since all features are in high dimensional space, we looked at the linear dependence of each feature data set. We generated screeplot (principal component vs. variance explained) for each of the feature set of the training data. Figure 1 shows the comparison:

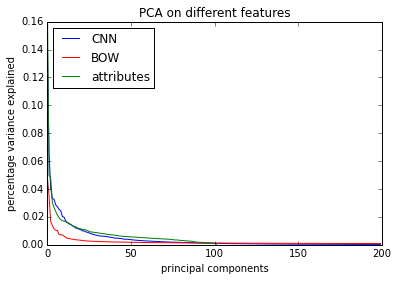


Figure 1. Screeplots for three feature data sets

For CNN features, the top 120 principal components explain 90.0% of the total variance; for BOW features, the top 120 principal components explain only 37.9% of the total variance, And we would need 1300 principal components to explain up to 90% of the total variance; for the attributes, the top 65 principal components explain 89.9% of the total variance. The PCA analysis suggested that CNN features are highly correlated. From the result, we can see that PCA is optional for dimension reduction of CNN features.

However, as we will mention later, we addressed this “high dimensional data” issue by using an SVM model, which is robust in high dimensional space.

**3.1.3 Feature scaling**

Since we used support vector machine (SVM) as our classifier. And there has been discussions that SVM performance depends on scaling of classifier [4,5,6] . For each of the three data sets, we did 5-fold cross validation to determine whether to scale the data or not. And the 5-fold cross validation result is in Table 2. Here scaling means zero-centering and unit-variance transformation.

**Table. 2 SVM performance with or without scaling**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **CNN features** | **BOW features** | **Attributes** | **Tenk** |
| **Unscaled** | 35.9% | 12.6% | 19.0% | 15.7% |
| **Scaled** | 37.1% | 12.9% | 25.9% | 29.2% |

Based on the 5-fold cross validation, we choose to do scaling for CNN feature data set, BOW data set, and Tenk data set, but no scaling for attributes data set. Intuitively this also makes sense because attributes is binary data, and scaling will cause it to lose the sparsity property.

**3.2 Training each dataset separately with support vector machine**

Support vector machine (SVM) is a linear classifier that solves the following optimization problem:

For our model, we implemented a linear SVM for CNN dataset and BOW dataset, and a polynomial kernel SVM for attributes data set. The type of kernel is chosen based on 5-fold cross validation results.

We choose to use SVM classifier over other types of classifier for the following reasons:

a. Working with high dimensional data, we hypothesized that support vector machine will work well because the generalization properties of SVM do not depend on the dimensionality of the space [7].

b. The scikit-learn implementation of SVM solved the multi-class classification problem by one-against-one approach. That is, we constructed n\_class\*(n\_class-1)/2 number of classifiers with each classifier classify two different classes [3]. This implementation solved the problem of extremely unbalanced distribution of positive and negative examples in the classification.

To prove this hypothesis, we compared the performance of linear SVM, logistic regression, neural network, and random forest on the CNN data set. And the 5-fold cross validation result is shown in Figure 2.

**3.3 Voting Ensemble**

From Table 2, we can see that SVM gave very different performance for the three different data sets (37.1% for CNN, 12.9% for BOW and 25.9% for Attributes). This means we have different confidence in the prediction of the three of them. In order to make use of this information to improve performance in the test set, we added a voting ensemble on top of the three classifiers. That is, on the training set, we did a five-fold cross validation to choose the hyper parameters for SVM including: scaling, kernels. Also, we produce a weight for each classifier, which is the accuracy of the classifier (0.37, 0.13 and 0.26). Finally, we fit the individual classifiers to the test set and combine the prediction of each classifier by the weight.

**4 Results**

**4.1 SVM versus other classifiers**

We compared the performance of different classifiers using CNN dataset by 5-fold cross validation on the 3k images. And the result is shown in Figure 3.

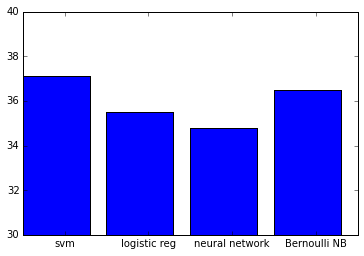


Figure3. Comparison of SVM with other algorithm on CNN data set

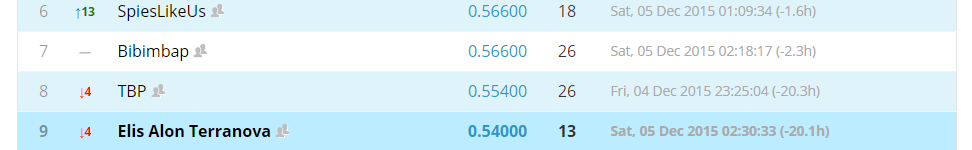
The best performing neural network is used here, which has 1 hidden layer and 500 nodes.

**4.2 Voting ensemble versus single SVM**

We also did a single SVM classifier on the combined features, and the performance is 35.5%, whereas using the voting ensemble, the cross-validation performance is 42.2%.

**4.3 Voting ensemble**

By using a voting ensemble, we achieved a cross validation performance of 42.2% on the training data. And a testing accuracy of 54% on the kaggle leaderboard.



We also did a single SVM classifier on the combined features, and the performance is 35.5%, whereas using the voting ensemble, the cross-validation performance is 42.2%.

**5 Conclusion**

As we stated in the introduction, the major challenge of this learning problem is the scarcity of data, discrepancy in quality of different data sets, and extremely small number of positive examples. We addressed the first challenge by making use of the 10k unlabeled data to create additional bag-of-word features for the 3k training set, addressed the second challenge by using voting ensemble, and the third challenge by doing one-against-one SVM for multi-class classification problem.

These three factors together gave us a model the improvement in prediction performance.

[1] AlexNet

[2] SIFT

[3] sklearn-svm

[4] [www.hpl.hp.com/techreports/2009/HPL-2009-31R1.pdf](http://www.hpl.hp.com/techreports/2009/HPL-2009-31R1.pdf)

[5] [books.nips.cc/papers/files/nips15/AA09.pdf](http://books.nips.cc/papers/files/nips15/AA09.pdf)

[6] [olivier.chapelle.cc/pub/jsm\_fsel.pdf](http://olivier.chapelle.cc/pub/jsm_fsel.pdf)

[7] Vapnik's book "Statistical Learning Theory"