Kaggle: YELP-challenge

Roos van der Donk (s4166418), Laurens Hagendoorn (s4624858), Diede Kemper (s4056566), Daniëlle Tump (s4538102) *Machine Learning in Practice, Master Artificial Intelligence, Radboud University*

1. Problem Description

Yelp is a corporation that gathers crowd-sourced photos and reviews about local businesses (http://www.yelp.com). In its competition, Yelp has challenged Kagglers to build a model that automatically tags businesses with multiple predefined labels using a set of user-submitted photos per business. The train data contains 2,000 businesses with around 250,000 photos and the test data contains 10,000 businesses with around 250,000 photos. Labels were only known for the businesses in the training set.

2. Approach

To get to know the problem and data, Yelp's benchmark of training on color features was first replicated. Then, a pre-trained convolutional neural network (Simonyan and Zisserman (2014)[1]) was applied to extract a vector of 4096 features per photo. Multiple models were applied to these features. At the end, all models were combined in an ensemble. Now, the individual models will be described.

2.1 Statistics

This simple model computed the frequency of each class label in the train set. These frequencies were then applied to label the test set: If at least 50% of the businesses in the train set had a label, the label was given, otherwise it was not. Thus, every test business was assigned the same set of labels. The script to compute the frequencies was found on the Kaggle Scripts section for this challenge.

2.2 Random Forest + Color features

Per business, the following features were extracted: The mean and standard deviation of the red, green and blue values, the number of photos per business, and the mean and standard deviation of the height and weight of the photos. These features were then used to train a random forest, with 10 decision trees.

2.3 SVM + Average caffe feature per business

The average feature values of the caffe features were calculated per business. A poly-kernel SVC model was trained on these values to predict the labels assigned to the businesses of the testset.

2.4 SVM + Caffe feature per photo

A linear SVM was trained on the caffe features to predict per photo the labels of the corresponding business. Per business, all predicted labels were summed and normalized and another poly-kernel SVC model was trained on these data to predict the business label.

2.5 HBOW-EM but not quite

According to Amores (2013) the Histogram Bag of Words method combined with gaussian mixture models using Expectation Maximisation gave the best performance on multi-instance image datasets. However, the gaussian mixture model was too memory intensive on data of this size. Therefore, it was decided to use mini-batch k-means

1. We have made changes to the following code for our means: `https://github.com/avisingh599/visual-qa/blob/master/scripts/extract_features.py`

instead. Clusters were created for all caffe features. Per business, it was counted how many of its photos were in each cluster. These data were normalized and used to train a SVM to predict the business labels.

2.6 Ensemble

All models were combined in an ensemble, which based the final output on a weighted combination of the label probabilities given by the models. To find the optimal weights, a validation set was extracted from the train set, such that the accuracy of the ensemble could be computed. All individual models were trained on the remaining part of the train data.

3. Results

Table 1 shows the performance of our models. The SVM trained on the average caffe feature per business was the best individual model. The ensemble gave the best score in the end. The weights of the ensemble with the best final performance were: 0.06 for the statistics model, 0.75 for the SVM per business average caffe feature model and 0.19 for the SVM per photo caffe feature model. On the private leaderboard of the YELP-competition, the ensemble had a slightly higher score (0.8049), giving our team the 87th position in the ranking.

Model	F1-score
Statistical model	0.609
Color feature model	0.6048
SVM per business average caffe feature	0.7863
SVM per photo caffe feature	0.75713
HBOW-EM caffe feature	0.68559*, 0.75007**
Ensemble of above models	0.79759

Table 1: The F1-score on the public leaderboard of the submissions of our models. * SVM with poly kernel. ** SVM with linear kernel.

4. Discussion

As the results show, we have created multiple successful models. Combining them into an ensemble gave even slightly better results. During our project we have encountered the power of ensembles. At some point, our SVM model contained a bug which caused it to give a F1-score of 0.68. By combining this SVM with the statistical model, the F1-score increased to 0.77. Thus, as we have seen, an ensemble enables the models to correct for each others weaknesses.

In the last week, we found out that the performance of SVMs with a linear kernel was much better than the SVMs with a poly kernel. However, we could not use the linear kernel SVM in the emsemble, because it only outputted labels and not probabilities. Unfortunately, we did not have time left to figure out a solution to this problem. Also, if more time was available, we would have optimized the parameters of the individual models and varied the used classifiers. For instance, we could have applied a random forest to the caffe features.

Due to inexperience and difficulty in setting up Python, GPU Cartesius and Caffe, this project had a very slow and tough start. As soon as we had generated the caffe features, the project accelerated. In the end, we are very proud to see our final results.

5. Contributions

Danielle

Explore the dataset. Get Caffe working. Explore the use of batch jobs. Present flashtalk. Implement SVM for the mean feature vectors. Implement SVM for H-BOW(EM) method. Implement SVM for per-photo feature vector.

Diede

Team captain. Implement random forest for color features. Generate caffe feature vectors on the server. Compute quartiles and maximum for feature vector per business (did not work out). Present final presentation.

Laurens

Compute color features per business. Explore online server cartesius. Explore the use of batch jobs. Explore H-BOW(EM) method. Present flashtalk. Implement H-BOW(EM) method.

Roos

Explore the dataset. Preprocess images. Generate caffe feature vectors on the server. Implement SVM for per-photo feature vector. Implement ensemble. Present final presentation.

Acknowledgments

We would like to thank Maaike de Boer for her coaching support during this project.

References

Jaume Amores. Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*, 201:81–105, 2013.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.