Creating an Image Search Engine Using Machine Learning Models

Abstract

The purpose of this document is to provide an in-depth description of the different machine learning methods used to develop a large-scale image search engine. This means that for each description, a search engine would return a series of images that best match the given description. This report provides performance results for the most accurate methods, and presents a final machine learning system that consists of Bag of Words, PCA, Kernel Ridge Regression, and Cosine Similarity as a solution with 39% test set accuracy.

1 Introduction

The purpose of this work is to build a large-scale image search engine that searches for images relevant to a given natural language query and return them in order of similarity.

The competition was held in Kaggle (1) and can be found on (2). The training dataset consisted of 10,000 samples and the testing dataset consisted of 2,000 samples, where each sample consists of a 224x224 JPG image, a list of tags for objects appearing in the image, a 5 sentence description, and both intermediate and final feature vectors extracted using a neural network with ResNet-101 architecture (3). When testing, the search engine must match each descriptor to the 2,000 samples in the test set, and return the top 20 images that best match the description. The evaluation metric used was Mean Average Precision at 20 (MAP@20). This method gives a score of $score = \frac{20+1-i}{20}$ for an image ranked i-th in the top 20 results.

Different methods were tested to increase the accuracy. The best method consisted of using PCA on a concatenated feature vector of fc1000 and pool5 image features as well as a Bag of Words of the tags. A Kernel Ridge Regression model was trained using a Bag of Words for the image descriptions as input and the concatenation of features and tags as output. This method yielded 34% accuracy during cross-validation and 39% accuracy upon submission.

2 Background/Related Work

The problem of image searching is complex, with many Convolutional Neural Networks (CNN) for fine-grain image classification existing (4). The image search engine developed for this project uses image features extracted by using ResNet, which is a well-known CNN. Therefore, the combination of MLP with CNN presented in this and other articles was implemented and tested for accuracy. MLP seemed initially to be the best choice for our model because it allows us to use a Bag of Words (to numerically represent text) and compare it to these ResNet feature vectors. However, the existence of several types of data including image tags, fc1000 features, and pool5 features, gave the opportunity to explore other methods to find the best images to match to descriptions.

Kernel Ridge Regression was explored as it has been used widely in the literature of image classification (5). Using multi-modal Kernel Ridge Regression, feature vectors of images were predicted. Cosine distances were calculated afterwards to create an image search engine to return the 20 best images that match input text descriptions.

Next, concatenating the Bag of Words encoded tags, and the final and the intermediate feature vectors provided a chance to exploit all the available resources. The literature explains how PCA is one of the most successful ways to make sense of high-dimensional data by reducing dimensionality and noise, leaving only the most variant and most significant vectors (6).

Additionally, similarity measures are required for comparison between images and text or images and images. One measure, cosine similarity has been used for image applications such as face recognition (7). However, k-nearest-neighbors (KNN) and sum of squared distance are also techniques that can help find the distance between vectors and should be explored. Thus, both techniques were explored.

3 Experiments

Experiments to map the descriptions to the images and/or tags followed these guidelines. The first experiments attempted to build a model to compare the ResNet-101 features to the descriptions. Then, the option to relate documents with tags was explored. Finally, both the documents and the tags were used together, by concatenating their vectors. The best model was determined based on the experiments detailed in the following sections.

3.1 Preprocessing

All experiments were subject to the same description preprocessing methods, each with some form of variation, mainly in the use of either Bag of Words or Word2Vec.

For each image, the descriptions were aggregated into a single lowercase description. These were stripped the sentences of stop words and punctuation as recommended by the research papers. Each word was lemmatized in the interest of contextualizing the description with meaningful words (i.e nouns, adjectives, etc.) Finally conjunctions were removed as they are non-descriptive words.

A Bag of Words that counted the number of occurrences of each word was created from the flattened descriptions. This was transformed this into a normalized feature vector. Similarly, feature vectors for the tags were created. However, for each tag (key: value), only the value of the tags was used.

Word2Vec (8), an alternate vector representation of data, was used to quantitatively describe each preprocessed description. The same was done for tags when needed.

3.2 Multi-Layer Perceptron - MLP

The multi-layer perceptron model from sklearn(6) was trained with the fc1000 feature vectors and the documents encoded as Bag of Words. Then, the 20 best images were selected using KNN.

At an attempt to improve the outcome, various experiments were run using different number of neurons (10, 100, 200, and 300) and hidden layers (1, 2, or 3 layer(s)), and a Word2Vec representation of the descriptions. Cross-validation was used to determine that the model with highest performance had a single layer of 300 neurons.

3.3 Ensemble Methods

Two create an ensemble method, two separate approaches were taken and both their outputs were considered to map 20 images to a description. The following two methods carry out this process.

3.3.1 SVM and Ridge Regression Tags with Multi-layer Perceptron Image Features

The models trained were an SVM and a Ridge Regressor. Each was combined with the MLP network trained to predict the image feature vectors being given the image descriptions.

Each model first predicted the description based on the tags, and the MLP used the predicted description to predict the images based on the fc1000 image features. At this point, there were two matrices representing the mapping of one description to 20 images. For each description, images that were in the same set of predictions were compiled into a single matrix. This process did not guarantee 20 images. Therefore, if there were less than 20 images in both sets of predictions, the top predictions of the MLP network that used the SVM descriptions and were not in the predictions of the other network were added to the final prediction matrix until there were 20 images mapped to a description. This would ensure each description had 20 images mapped to it.

3.3.2 Weighted Average Method for Pairwise-Comparison

For this method, the output of two different approaches were combined and averaged in a weighted manner.

First, a Bag of Words of descriptions was used to train a MLP that predicted feature vectors. A distance matrix was calculated from these predictions and the real image fc1000 feature vectors. After that, a pairwise distance matrix was calculated between two Bag of Words, that of the descriptions and that of the image tags.

Finally, a weighted average matrix of these two distance matrices was computed, doubling the importance to the Bag of Words pairwise matrix than the MLP distance matrix. Each row was sorted in ascending order to rank the 20 images with the closest distances.

3.4 PCA & Kernel Ridge Regression

Kernel Ridge Regression was used to train four models to predict image features. Each model was trained with a respective matrix of feature vectors of reduced dimensions. These were reduced using PCA with a variance of 0.97.

The first model only predicted fc1000 final features. The second model only predicted pool5 intermediate features. The third model predicted a concatenation of fc1000 final and pool5 intermediate features. The last model predicted a concatenation of fc1000 normal, pool5 intermediate features, and a bag of words of the tags. Then, the distance matrix was calculated using the cosine distance function from the scipy module. These distances would be sorted in ascending order. The lowest 20 distances were chosen to represent the 20 images that best match to the description. Of the four models, the last three provided the highest accuracies and their predictions were submitted to Kaggle (results shown in Table 1).

3.5 Other

Several other regression methods in the sklearn library were tested such as Linear Regression, Logistic Regression, Gaussian Naive Bayes, KNN Regression and Random Forest Regression (9). For these models the fc1000 image features were used with a Bag of Words description vectors. However, these models did not perform well and therefore their were not included in this report.

4 Results

Table 1: Accuracy of Different Approaches

Model	Parameters	Cross-Validation/Kaggle Accuracy
BoW, MLP	$\alpha = 1e5$ $hidden\_layer\_sizes = (10,)$	9%*, 5.3%
Word2Vec, MLP	$\alpha = 1e5$ $hidden\_layer\_sizes = (10,)$	12.5%*, N/A
Word2Vec, MLP	$\alpha = 1e5$ $hidden\_layer\_sizes = (100,)$	32.2%*, 16.9%
Word2Vec, MLP	$\alpha = 1e5$ $hidden\_layer\_sizes = (300,)$	36%*, N/A
Word2Vec, MLP SVM MultiOutput Regressor on Tags	$\alpha = 1e5$ $hidden\_layer\_sizes = (300,)$	32.3%*, 15.6%
Weighted Average Pairwise-Comparison	$\alpha = 1$	25%*, 13.1%
Ridge Regressor on Tags	$\alpha = 1$	37%, N/A
BoW, PCA on Intermediate features using Kernel Ridge	$\alpha = 1$ PCA variance = 0.97	24.9%, 24.6%
BoW, PCA on concatenated features using Kernel Ridge	$\alpha = 1$ PCA variance = 0.97	23%, 27%
BoW, PCA on concatenated features and tags using Kernel Ridge	$\alpha = 1$ PCA variance = 0.97	34%, 39%

**: Incorrect method used for calculating cross-validation accuracy*

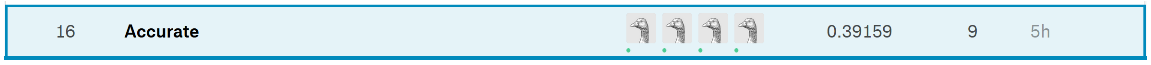
16 **Accurate** 0.39159 9 5h

Figure 1: Submission of BoW, PCA on concatenated features and tags using Kernel Ridge Regression

5 Conclusion

As shown above, the method that resulted in the best accuracy was the BoW, PCA on concatenated features and tags, using Kernel Ridge Regression. During testing, an accuracy of 34% was constantly achieved. A 39% was achieved at the time of the submission, which collocated the prediction in 16th place in the Kaggle public leaderboard. The main attributes of this model is the PCA and concatenation of both feature vectors and image tags. To summarize the model, a BoW was created from the descriptions of an image and concatenated all the features and tags of an image. This results in a very large vector of features and tags so PCA was applied with a variance parameter of 0.97 because it resulted in the best accuracy when doing cross validation. After performing PCA, there were less but more significant dimensions of an image to consider. This data was then used to train a Kernel Ridge Regression with its default parameters to predict a concatenation of feature vectors and tags for any description. During the cross-validation process it was found that the cosine distance was the best metric to determine how close a predicted feature vector was to a real feature vector. Having said that, the cosine distance was calculated between each prediction and the actual value and sorted in order to predict the 20 best images based on the shortest distances.

References

[1] K. Inc. (2019) Kaggle. "https://www.kaggle.com/"[Accessed 12-10-2019].

[2] C. Tech. (2019) Cs5797 fall 2019 final. "https://www.kaggle.com/c/cs5785-fall19-final"[Accessed 12-10-2019].

[3] Ethereon. (2019) Resnet architecture. "http://ethereon.github.io/netscope/#/gist/b21e2aae116dc1ac7b50"[Accessed 12-10-2019].

[4] C. Zhang, X. Pan, H. Li, A. Gardiner, I. Sargent, J. Hare, and P. Atkinson, “A hybrid mlp-cnn classifier for very fine resolution remotely sensed image classification,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 140, pp. 133–144, 06 2018.

[5] X. Zhang, W. Chao, Z. Li, C. Liu, and R. Li, “Multi-modal kernel ridge regression for social image classification,” *Applied Soft Computing*, vol. 67, p. 117–125, 2018.

[6] P. Wang, L. Li, and C. Yan, “Image classification by principal component analysis of multi-channel deep feature,” *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2017.

[7] H. V. Nguyen and L. Bai, “Cosine similarity metric learning for face verification,” *Computer Vision – ACCV 2010 Lecture Notes in Computer Science*, p. 709–720, 2011.

[8] Google. (2019) Word2vec gensim. "https://radimrehurek.com/gensim/models/word2vec.html"[Accessed 12-07-2019].

[9] A. G. e. a. Fabian Pedregosa, Gaël Varoquaux, “Scikit-learn: Machine learning in python,” *Journal of Machine Learning Research*, 2011.