Image Retrieval System with An Ensemble Model

Jia, Haochen Zhao,Harrison Zhu,Yuxiang
Cornell Tech
2 West Loop Road, New York, New York,10044

Abstract

As the final project of Machine Learning Class, we are challenged for predicting the fittest pictures by given descriptions. We build a combine-based method to optimize the performance. We implement the PCA, TF-IDF and MLP methods and gain a score of 0.414 in Kaggle.

1 Introduction

With the development of computer science and information system, people gain more and more information from images. Consequently, people need the image retrieval technology to manage and retrieve images from natural language. Traditionally, people will use SVM and HOG to retrieval, but the result is unsatisfied. In recent years, with the development of Deep Learning, people can more efficiently gain features of images via ResNet, with a longer training time, but simultaneously improve the accuracy of retrieving.

2 Problem Clarification

2.1 Description

This Kaggle Competition requests an image searching engine which can retrieve the 20 nearest images from the target natural language phrases.

2.2 Data

We are provided 2 datasets, a training set with 10000 elements and a testing set with 2000 elements. For each element of the training set, we have a 224x224-pixel JPG image, a list of tags, a five-sentence description, a 1000-dimensional FC1000 feature and 2048-dimensional Pool5 feature of the image generated by ResNet. Similarly, for each element in the testing set, we get the picture, tags, FC1000 features and pool5 features. We are requested for 20 nearest pictures in the testing set by each provided descriptions.

3 Model Structure

We formulate such image retrieval system in generally 2 ways: Text-based retrieval and Content-based retrieval. Because every image matches a series of descriptions, text-based retrieval could efficiently inquire the inherent relationship between descriptions and tags. This model is relatively easy to be formulated and add-on other methods like N-grams. But the short come of this strategy is also obvious: Tags may be hard to demonstrate all features of the picture (Too much information losses). Content-based model, which inquires the connection between features and description, could sharply augment the accuracy of retrieving. However, it requests time to train learning models. Luckily, we are provided features and tags in Kaggle. Thus, we will try to solve the problem based on these two methods. Figure 1 represents our flowchart.

Submitted to 31st Conference on Neural Information Processing Systems (NIPS 2017). Do not distribute.

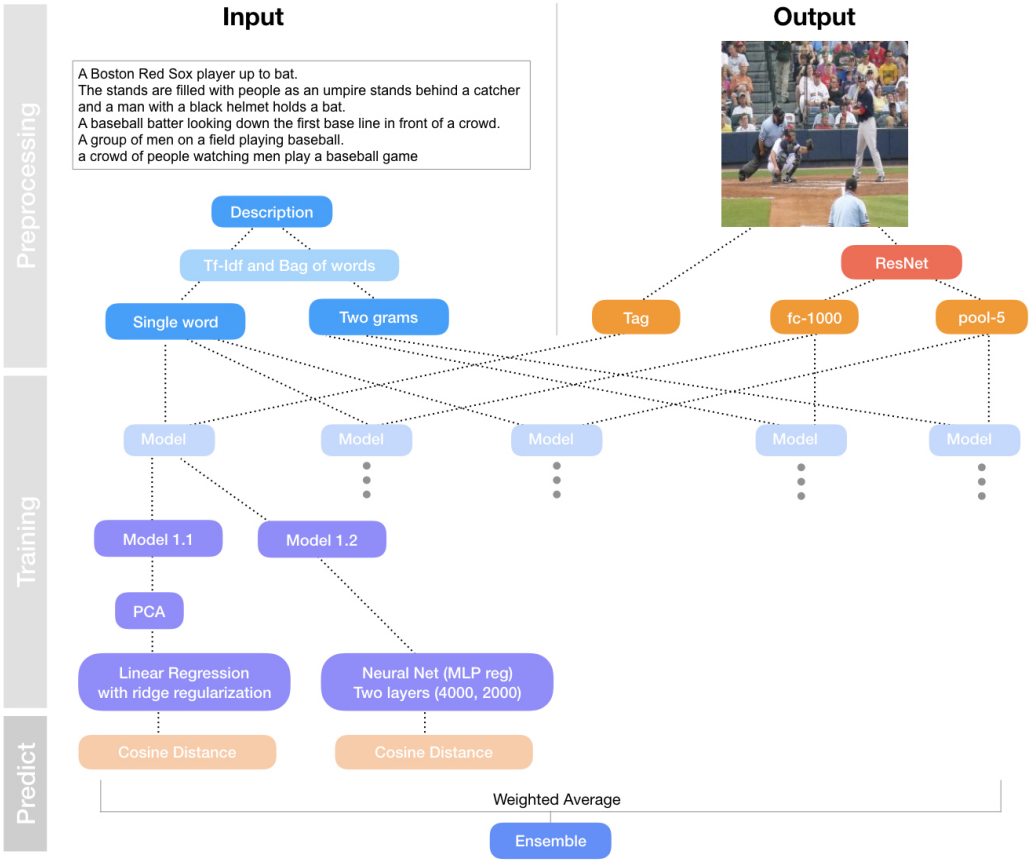

Figure 1: Flow Chart

4 Text Based Retrieval

In this method, we have approximately 6000-dimensional vectors for description. We wish to train a model to inject description of tags. Then, we use tags to discover similar pictures by finding the distance of each sample picture of target descriptions and sort them in the end.

4.1 Datamation/ Preprocessing

Datamation should be the first thing to do intuitively. We do follow steps to transfer descriptions in natural language to vectors or matrices:

1. Combine five descriptions of each image in one line.
2. Use Stop-Words package to drop all less-meaning words, and split each phrase into a list of words.
3. Use Snowball Stammers to revise all words.
4. Create a dictionary to descriptions of the training set.
5. Attribute each list of description words in the training set and testing set with the same dictionary by trainning set and get vectors of descriptions.
6. Attribute each list of tags in testing set and training set to the dictionary,and get vectors of descriptions.
7. Normalize each vector by TFIDF methods.

- Similarly, we repeat the whole process with 2-Gram method and apply step 1,4,5,6 to tags.

4.2 TF-IDF

TF-IDF (term frequency-inverse document frequency) is a numerical statistical method which intends to reflect the importance of a word to a document or corpus. We will apply this method for weighting words in the dictionary.

1. Term frequency is a scaler for the frequency of certain words, we will use the following function to define the significance of each phrase:

$$tf(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{max[f_{t',d} : t' \in d]}$$

2. Inverse document frequency is used to recognize the diminish the weight of the extremely high-frequency word, because some words with extremely high frequency, like "the", provide no information. The function is defined as following:

$$idf(t,D) = log\frac{N}{|[d \in D : t \in d]|}$$

3. Finally, we multiply the result of TF and IDF, and get the high weight phrase only in high term frequency but low document frequency:

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D)$$

4.3 MLP Regression

We define input as X, which are vectors of descriptions in this case and define output as Y, which are vectors of tags. Then we apply the Artificial Neural Network to train the model: This is a multi-class regression, so, we apply the softmax function to count the highest probability: where z_i are i th class input, and k is the number of classes.

$$softmax(z)_i = \frac{e^{z_i}}{\sum_{l=1}^{k} e^{z_l}}$$

The loss function of MLP regression based on Square Error function:

$$Loss(\hat{y}, y, W) = \frac{1}{2}||\hat{y} - y||_2^2 + \frac{1}{2}\alpha||W||_2^2$$

where $\alpha||W||_2^2$ is a sort of complex model penalty in L2-regularization term. Finally, we adjust the W based on the loss function.

4.4 Dimensional reduction

Because dealing with 6000-dimensional vectors is not only burdensome but also unnecessary, we consider to use PCA method to reduce the dimension of vectors to 2840 (Because the intersection between dictionaries of training set and testing set is about 2840). It is really practical because it decreases the time consuming to about 1/10 of time on training models. Also, this kind of dimensional reduction does not hurt the accuracy too much.

4.5 Tags Modifications

When carefully scrutinized all tags and some descriptions, we realized some of similar words or synonyms cannot be sorted into the dictionary. For improving the performance of our model, we import the package of nltk to compare the similarity in each combination of word x with tag y in the dictionary. If the similarity beyond the threshold (we set 0.3), we will directly match attribute description y to tag x. But this method is outperformed my our final model.

4.6 Distance Functions

In this part, we did a careful research on selecting the distance function for checking similarity. Because in the final step, we need to compare the distance between the description vector of target images and similar images.
Initially, we apply the Euclidean distance on checking the similarity. The score by grading system is a little bit lower than 0.18. (It may be caused by the curse of dimensionality)

$$Dist_{eucledian}(X, Y) = (X - Y)(X - Y)^T$$

Then, we try the L1 distance. The performance is little bit better (slightly higher than 0.18).

$$Dist_{L1}(X, Y) = |X_Y|$$

Finally, we apply the cosine distance, which has the best performance (The score is 0.192).

$$Dist_{cosine}(X, Y) = \frac{X \cdot Y}{||X|| \cdot ||Y||}$$

4.7 N-Gram

We also applied the N-Gram method to improve the accuracy of predictions. We built a 2-Gram diction on each description and traced the result of the model.

4.8 Model 1

For text-based analysis, we design to predict tags by description and compare predicted tags on all images with on target images with cosine distance.
We set a model to match Descriptions and tags with 1 hidden layer and 1500 neutrons with an adaptive learning rate MLP regresses, and the score we get is: 0.2305.

5 Content Based Retrieval

In this section, we will try to train a model to match the feature vectors and descriptions.

5.1 Attempt of PCA

In the beginning, we scrutinize the feature set of FC1000 and Pool5, and realize that all features in Pool5 are not very distinctive. Thus, we use the PCA method to deduct the dimension of features sets, FC1000 and Pool5. Also, we attempt to change coordinate on inputs by PCA. Considering the tradeoff between the accuracy and efficiency, we decide to reduce the dimension of each feature set to 300.

$$BagofWords(ncomponent = 2840) \rightarrow FC_{1000}(ncomponent = 300)$$
$$BagofWords(ncomponent = 2840) \rightarrow Pool_5(ncomponent = 400)$$

However, the performance of both PCA change coordinates on inputs and PCA dimension is only favored by Linear Regression but not good for MLP Regression.

5.2 MLP Regression

We still think the MLP Regression is a good choice for our models because it balances the accuracy and the complexity.

5.3 Model 2 to 5

- Model 2: for content-based analysis, we design to predict features by description and compare predicted features on all images with on target by cosine distance.
  We set an MLP Regression model to match descriptions and Fc-1000 with 2 hidden layer and 4000,2000 neutrons with an adaptive learning rate of MLP Regression:
- Model 3: We did a similar setting on Model 2 but change the input to Pool-5.

- Model 4: We did a similar setting on Model 2 but change the dictionary in a bag of words into 2 grams.
- Model 5: We did a similar setting on Model 3 but change the dictionary in a bag of words into 2 grams.

5.4 Linear Regression

We applied the Linear Regression with ridge regularization cross-validation by following preprocessing:

1. Do PCA on input data without dimension reduction
2. Reduct the dimension of output data to 350.
3. Train Linear Regression with Ridge CV
4. Finally, we predict and figure out minimal cosine distance.

Ridge CV could reduce the effect of multicollinearity because it considers not only the loss function (traditional linear regression) but also the penalty (L2 normalization of betas):

$$min_{f \in F} \frac{1}{N} \sum_{i=1}^{N} Lossfunction + \lambda ||J(f)||$$

L2 normalization carries more information about the correlation between inputs and outputs. It is a better choice as the penalty than L1 distance (the Lasso Regression).

5.5 Model 6 to 9

- Model 6-9: We did PCA on inputs and outputs as the previous section mentioned, and apply Linear Regression respectively:
- Model 6: For content-based analysis, we design to predict features by description and compare predicted features on all images with on target by cosine distance. We set a Ridge Linear Regression model to match descriptions and Fc-1000.
- Model 7: We did similar setting on Model 6 but change the input to Pool-5.
- Model 8: We did similar setting on Model 6 but change the dictionary in bag of words into 2 grams.
- Model 9:We did similar setting on Model 8 but change the dictionary in bag of words into 2 grams.

6 Ensemble

6.1 Weight average

We separately apply our 5 models by cross validation, and get scores S_1 to S_9 respectively. Then we weight our model in following way:

$$score = \sum \frac{S_i}{\sum_i S_i} \cdot model_i$$

This combination is slightly better preformed than the best result of 9 models.

6.2 Final models

We develop a voting system to ensemble results of our models.This system is generally based on AHP (Analytic Hierarchy Process), because we treat this voting system as a group decision system: Assume our model set is L. For description i, and image j, we have a rank k (we treat all k >20 as k=20) in model l. We give the score on this rank: $\frac{1}{R_{ijkl}}$

Then, we can count the total ranks of all models: $\forall j, R_{ijk} = \sum_{\forall l \in L} \frac{1}{R_{ijkl}}$

Based on the total score of each image, we can sort all images, and show highest 20 results for description i.

7 Evaluation

7.1 Our outputs

We randomly pick some sample of our model

- Input: 5 phrases of description:(15.txt)
  1. A pizza sitting on top of a wooden cutting board.
  2. The pizza is loaded with melted cheese and vegetables such as green peppers.
  3. A pizza with bell peppers, meat, and sauce.
  4. A vegetable pizza that has been sliced into pieces.
  5. a pizza and a pizza cutter and some green pepper toppings
- Output: our 8 highest predictions:(Figure 2:15.txt)

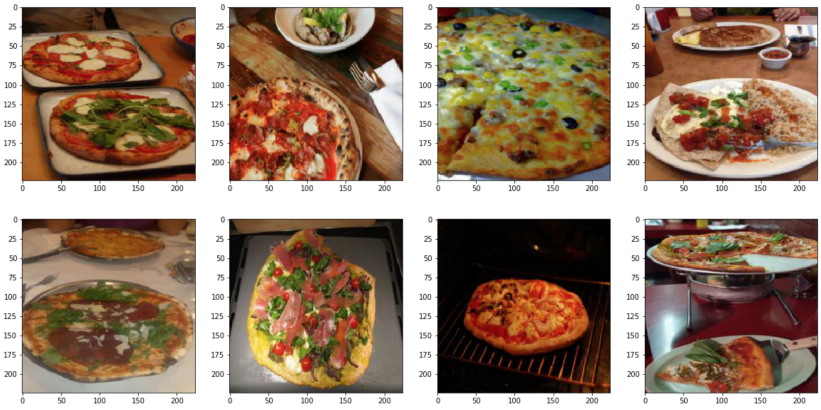

Figure 2: 15.txt

- Input: 5 phrases of description:(17.txt)
  1. A man is holding a couple of traffic lights on the side of the road.
  2. A man carrying two traffic lights on the side of a street.
  3. a man in a tan jacket is carrying a traffic light
  4. a man holding some street lights as he stands next to the road
  5. A man holding the streetlights in his hands near a street
- Output: our 8 highest predictions:(Figure 3:17.txt)

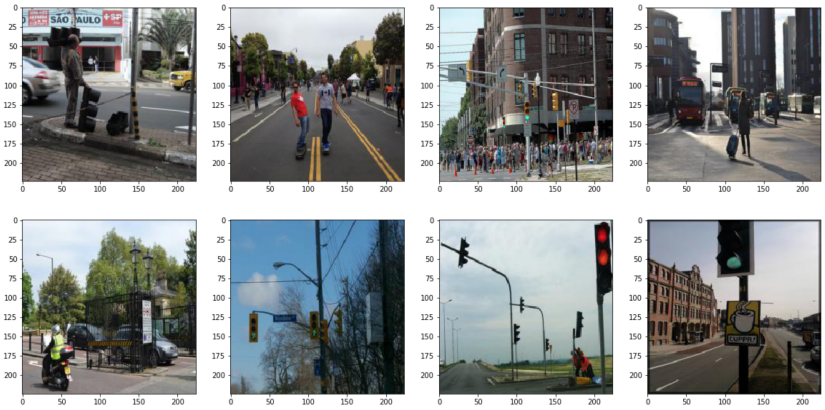

Figure 3: 17.txt

7.2 Kaggle Scoring System

The evaluation system Kaggle applies in this competition is called "MAP @ 20"(Mean Average Precision at 20) on the testing set. For each set of descriptions, there is only one constant corresponding image. Assume this image appears in our i-th rank, we will get a score $\frac{21-i}{20}$, the final score should be the average score of our entire test set.

8 Next Steps

We believe that if possible, we merge all features, FC1000, Pool5, tags, and description together as a high dimension training set, and then, trains a multilayer neural network. Though it is highly hardware consuming, it will definitely perform better. Also, finding other methods to the ensemble is also a good idea. For example, we can do regression methods further than MLP, such as PLS.