Image Searching Engine:A Brilliant Name for Our Supper Awesome Paper

Your names and group number

December 5, 2017

Abstract

In this paper, we proposed a solution to deal with an image searching problem....

1 Introduction

In this paper, we are trying to build an image searching engine that can search for relevant images given a natural language query. For instance, if a user types "dog jumping to catch frisbee," Our system will rank-order the most relevant images from a candidates' pool and return the top twenty images that are relevant.

In this paper, we will introduce the experiments we performed involved KNN, random forest, neural network and SVM.

???????For example, you can cite the Nano 3 Lecture notes [1].

1.1 Problem Definition

D**uring training**, we have a dataset of 10,000 samples. Each sample has the following data available: A 224x224 JPG image. A list of tags indicating objects appeared in the image. Feature vectors extracted using ResNet. A five-sentence description.

During testing, our system matches a single five-sentence description against a pool of 2,000 candidate samples from the test set. Each sample has: A 224x224 JPEG image. A list of tags for that image and the ResNet feature vectors for that image.

Output: For each description, our system will rank-score each testing image with the likelihood of that image matches the given sentence. Then the system returns the name of the top 20 relevant images, delimited by space.

For Evaluation: There are 2,000 descriptions, and for each description, we compare against the entire 2,000-image test set. That is, rank-order test images for each test description. Then we use MAP@20 as the evaluation metric. If the corresponding image of a description is among the algorithm's 20 highest scoring images, this metric gives a certain score based on the ranking of the corresponding image as follow.

$$Score = \frac{20 + 1 - i}{20} \tag{1}$$

1.2 Related Work

Will talk a little bit of how KNN, random forest and neural network are important in doing image searching

2 Models Architecture

2.1 Data Preprocessing

For this project, data preprocessing takes up a large portion of the work and by choosing better strategies of preprocessing, we could achieve better performance of the overall model. Here are the few preprocessing strategies we used.

For the five-sentence description, we (a)converted to lower case, (b)stripped punctuations, (c)removed stop words, (d)converted all words to stem words. This is just a regular preprocessing strategy for text input. After we reviewed some literatures, we found that IFTDF would be an useful tool to perform preprocessing on small chunks of text. To be continued....

2.2 Building Features

Since we need to retrieve images based on five-sentences of description queries, so we need to build the feature vectors for these queries. In addition, to perform a KNN, we need to compare each query feature with every other candidate's feature. And we do not have the descriptions in the candidates' data. Therefore we need to perform a mapping that maps the description of the input queries with the tags of the candidates'. We will discuss this part later. So we are building both of these feature vectors in a same way. Therefore we built feature vectors for both input descriptions and candidates' tags.

2.3 Mapping Descriptions to Tags

By mapping description to tags, we could express both of the input queries and the candidates in a same format of features where of these two

We built a dictionary of unique tags from tag data of the training set. Then we built the feature vector for the test data and candidate data through Bag of Words model(??). By now, we have mapped the description data of a query to tags.

2.4 The KNN Approaches

For the naïve version of our solution, we are using a KNN approach. And we split the task into 4 steps:

1. data preprocessing
2. building feature vectors for test queries and candidates
3. calculate pairwise similarities between queries and candidates
4. retrieve the top 20 nearest neighbor from the candidates' pool

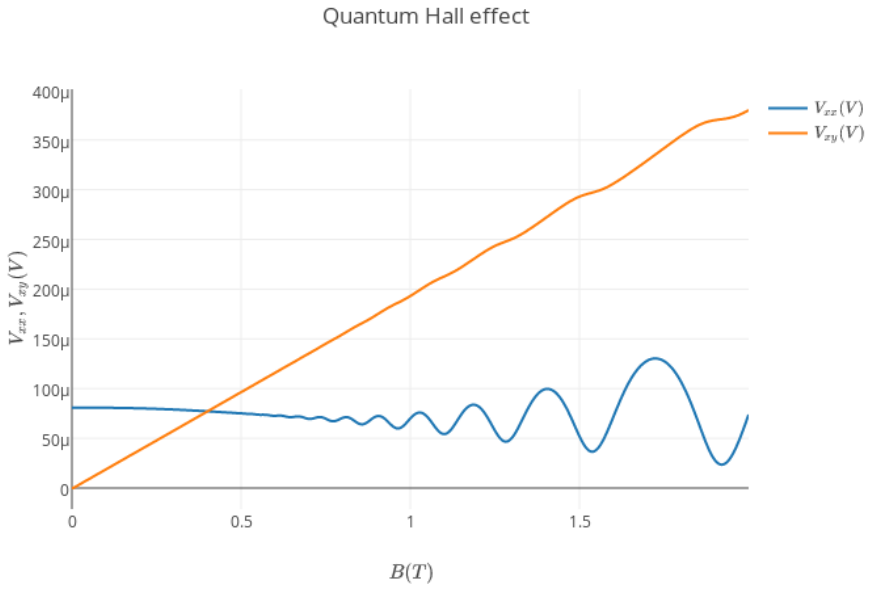

Figure 1: Raw (unprocessed) data. Replace this figure with the one you've made, that shows the resistivity.

2.4.1 Cosine Similarity

After the last step, we got the feature vectors for the test data and candidate data. Then we compute the cosin distance or Euclidean distances between a test sample and all the candidates. After this step, we simply sort the distance and find the top 20 neighbors as the output data.

2.4.2 Classic KNN

2.5 The Neural Network Model

2.6 The Random Forest Model

3 Experiment

3.1 Our Steps of Refining the Models

4 Results

Show a graph of the longitudinal resistivity (ρ_{xx}) and Hall resistivity (ρ_{xy}) versus magnetic field, extracted from the raw data shown in figure 1. You will have the link to the data in your absalon messages, if not e-mail Guen (guen@nbi.dk). Explain how you calculated these values, and refer to the theory.

4.1 Quantum regime

Calculate n_s for the high-field regime. Show a graph of the longitudinal conductivity (ρ_{xx}) and Hall conductivity(ρ_{xy}) **in units of the resistance quantum** ($\frac{h}{e^2}$), depicting the integer filling factors for each plateau. Show a graph of the plateau number versus its corresponding value of $1/B$. From this you can determine the slope, which you use to calculate the electron density. Again, calculate the uncertainty for your obtained values.

5 Conclusion

References

[1] K. Grove-Rasmussen og Jesper Nygård, *Kvantefænomener i Nanosystemer*. Niels Bohr Institute & Nano-Science Center, Københavns Universitet