Identify Duplicate Questions in Quora

Venkata Sai Abhishek Gogu

Capstone Proposal

Machine Learning Engineer Nanodegree

1. Domain Background:

This is a classic Natural Language Processing problem popularly known as NLP. NLP is a way for computers to analyze, understand, and derive meaning from human language in a smart and useful way. The study of NLP techniques has been there around for more than 50 years and grew out of the field of linguistics with the increase in computing power. Before going on to statistical and probabilistic models, hard hand-written rules along with decision tree algorithms have been used which is not general robust to natural-language[1] variation. But, these models have used machine learning algorithms which automatically focus on the most common cases, and more robust to unfamiliar input. Also, these models have an extensive use of context of the language. Most commonly research tasks in NLP are Lemmatization, Stemming, Parts-of-speech tagging, Parsing and so on. There are a lot of open source tool-kits available to perform these tasks such as Apache CoreNLP, Natural Language Tool-Kit(NLTK), Stanford NLP, MALLET and the list[2] goes on.

In the last decade, question and answer websites like Quora, stack exchange, yahoo answers, so on., put the world's wealth of information at our fingertips, but still are generally quite primitive when it comes to answering specific questions posed by humans. As the information is growing, it is necessary to remove the information which depicts the similar content/meaning already posed/exists earlier. Simply, these questions lead us to design a clever system which can predict whether this question is already in the system or not by using context, the semantics of the question.

As per recent web trends, data on web[3] is growing exponentially and at this growth rate, one needs to have use big data technologies to store and retrieve data fast and efficiently. Still, it needs labor, time and money to set up everything and still it is not reliable. Instead, regulating the duplicate/same-intent data entering into the system will help to overcome these issues to a maximum extent.

References:

1. *Wikipedia, "https://en.wikipedia.org/wiki/Natural-language_processing"*
2. *Wikipedia, list of popular toolkits," https://en.wikipedia.org/wiki/Outline_of_natural_language_processing#Natural_language_processing_toolkits"*
3. *Data on web, "https://thenextweb.com/contributors/2017/04/11/current-global-state-internet/#.tnw_G0fBh235"*

2. Problem Statement:

Quora is a platform to ask questions and connect with people who contribute unique insights and quality answers. This empowers people to learn from each other and to better understand the world. Over 100 million people visit Quora[4] every month, so it's no surprise that many people ask similarly worded questions. Multiple questions with the same intent can cause seekers to spend more time finding the best answer to their question and make writers feel they need to answer multiple versions of the same question.

Quora needs a model to identify duplicate questions by applying advanced Machine Learning techniques along with Natural Language Processing techniques to classify whether question pairs are duplicates or not. Doing so will make it easier to find high-quality answers to questions resulting in an improved experience for Quora writers, seekers, and readers. Given a question pair, we need to find that second question is a duplicate/same-intent of the first one or not.

3. Datasets and Inputs:

This project uses Kaggle Quora Question Pairs competition data-set (link). This contains train data and test data. Train data contains 404290 question pairs and it sizes around 64 MB. Among them about 36.9% are duplicate pairs and it contains 537933 number of questions out of which 111780 of them have appeared multiple times. Each question in question pair is accompanied with question text involved in the question of size around 10 to 170 characters. Fields of the input data:

- *id* - the id of a training set question pair
- *qid1, qid2* - unique ids of each question (only available in train data)
- *question1, question2* - the full text of each question
- *is_duplicate* - the target variable, set to 1 if question1 and question2 have essentially the same meaning, and 0 otherwise.

In test data, 2345796 question pairs of data and about of 315 MB of size is given and need to identify whether the second question is duplicate of the first one or not.

4. Solution Statement:

The most obvious solution is pre-processing questions text using NLP techniques and applying Machine Learning algorithms to the processed data. Here, we need to find is_duplicate or not. So, clearly, it is a classification problem with supervised learning technique. As there are question pair ids, we can preprocess data using the count of occurrences of these numbers and some more NLP methods like stemming, lemmatization so on. Few of the Machine Learning algorithms include Linear Classification, Logistic Classification, Neural Networks, Ensemble methods so on can be used to solve this problem.

References:

4. Kaggle, https://www.kaggle.com/c/quora-question-pairs

5. Benchmark Model:

For this problem statement, I would like to choose cosine similarity as my benchmark model. This technique involves, strip the stop words, stem the remaining, convert it into term frequency vectors and do a simple Cosine similarity[5] test.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Here, the attribute vectors *A* and *B* are usually the term frequency[6] vectors of the question texts. It ranges from 0 to 1, 1 indicates both are similar and 0 indicates not similar. This will provide very small accuracy rate as it lacks semantic understanding (there might be two questions with a high percentage of common words, but different meanings) and it checks for only syntactic equivalence. Based on this accuracy, I would like to design a better model which understands semantics of the question pairs with high accuracy rate.

6. Evaluation Metrics:

As it is a supervised classification problem, using Logarithmic loss[7] or simply log-loss is used as an evaluation metric. Log Loss quantifies the accuracy of a classifier by penalizing false classifications. Minimizing the Log Loss is basically equivalent to maximizing the accuracy of the classifier. Also,

- Precision
- Recall
- F1 measure
- Mean Squared Error

can be used as an evaluation metric.

References

5. *Cosine Similarity, https://en.wikipedia.org/wiki/Cosine_similarity :*
6. *Wikipedia, Term Frequency, "* https://en.wikipedia.org/wiki/Tf-idf "
7. *Log loss, wiki.fast.ai, http://wiki.fast.ai/index.php/Log_Loss*

7. Project Design:

This project involves natural/general approach of solving Machine Learning problems. The general steps involved are:

a. Data Preprocessing
b. Model Selection
c. Model Evaluation

All these are further divided into sub-tasks and explained in detail below.

a. Data Preprocessing:

Before developing a model, this is important to know that Machine Learning Algorithms learn from data. So, feeding right data to the model is always key factor. Hence, preprocessing the data will likely to help achieve more consistent and better results. Preprocess can be done in following steps:

i. Data visualization – Provide insights about data and describe about data. Usually done using graphs, statistics
ii. Formatting, cleaning and sampling – Do the cleaning and structuring text. Usually perform stemming, remove Nan or Empty lines and other NLP techniques.
iii. Feature Transformation – Some models only accept numerical values. So this step involves converting to words to vectors(word2vec) where words are features.
iv. Generalization – Out of all the features, some features might need scaling, normalizing, aggregating or Generalization. Using PCA, ICA techniques will help to complete this step. This is to be performed only if needed.

According to data, if needed some more additional preprocessing tasks can be performed.

b. Model Selection:

After getting the data, its important to pick the right algorithm for the data. There are various algorithms out there that works for this dataset. Experimenting with different models and selecting the best for this case is a challenging task. At the end of this step, we got our model ready for use.

c. Model Evaluation:

After we got the model, we can either perform model tuning by trying out with different parameters to increase performance of our algorithm. Finally, evaluate our model against our test data.