Data Mining Proposal

Luis Duque
Joa Jin
Ethan Leeman
Yingnan Liu
Sophia Zheng

March 27, 2017

1 Introduction

For our project, we propose competing in the most recent Kaggle competition, "Quora Question Pairs." Quora is a question and answer website where users can both submit and answer questions in a collaborative manner. While handling a large volume of queries, duplicate questions are asked often, making it difficult for popular, accepted answers to be found. Currently Quora uses a Random Forest model to identify identical questions, and proposed this Kaggle competition to tackle this machine learning problem in natural language processing.

2 Data

Kaggle provides a training dataset and a test dataset. In the training set, there are approximately 400,000 question pairs and the target variable, whether Kaggle feels the questions are duplicates. Some questions appear multiple times in different pairs. For example, the questions "How can you determine the first ionization energy of lithium?" and "How is the ionization energy of silicon determined?" are distinct while the questions "Will Donald Trump shut down the internet?" and "If Donald Trump becomes president will we lose the internet?" are duplicate.

One issue with the data is the subjectivity of the target variable. For example, the question pair "I am 24. Is it too late to get into medicine?" and "Is it too late to study medicine at 23?" could be reasonably interpreted as duplicate or as distinct (note that it is marked as distinct). Additionally, there is some natural human error and noise in the data. The question pair "What is the cultural shock?" and "What is Culture Shock?" is marked as distinct, while these are clearly the same question. Another issue is that the characters are in unicode, and sometimes questions with non-english words appear.

The test data is about 2,000,000 question pairs and the competitors are asked to submit a probability of being duplicate, and the submissions are scored on Logarithimic Loss.

3 Possible Approaches

3.1 by Ethan and Luis

Kaggle provides a discussion board, and rewards competitors for providing helpful guides and comments to other commenters. In particular, there is a "beginner's guide" by "shubh24" which provides many techniques in natural language processing.

- One first idea is to take every sentence, create a set of every k strings of words, after stripping the stopwords, and compare the overlap by some metric. For $k = 1$ we would measure how distinct the words are in each question, $k = 2$ would be word-pairs, and so forth. One key issue is that a single word, even if the rest of the sentence is identical, can drastically change the question, as in the lithium vs. silicon example above.

- Wordnet is an English dictionary which also contains a directed graph showing relationships between words. One possible feature we could make would be to find the rare words and see how distant they are. This would try to find questions that are asked with different synonyms or similar phrases.

- –Ethan stopped here–

A much longer $\LaTeX\,2_\varepsilon$ example was written by Gil [?].

—word embedding and subsequent methods begins here by Joa and Yingnan–

3.2 Neural Network approach

Among previous methods on neural network, the first step is usually transferring the words in a question into vectors using word embedding method. Word embedding is a general method used for similar algorithms that embeds words into a vector space with several hundred of dimensions. These vectors captures semantics and analogies of different words. In other words, the similarity between the words' definitions are denoted by the distance in vector the space. Currently, there are trained models including Googles WordVec, Stanford Universitys GloVe, Gensim and Deeplearning4j. Principal Component Analysis (PCA) and T-Distributed Stochastic Neighbour Embedding (t-SNE) are both used to reduce the dimensionality of word vector spaces and visualize word embeddings and clusters.

To judge whether two sentences (vector sets) are duplicates belongs to natural language sentence matching (NLSM) problem. For NLSM problem, there are two types of deep learning frameworks. One is based on the **Siamese**, the other is based on **matching-aggregation**. For Siamese framework, two sentences are encoded into two vectors individually and the matching decision is based on these two vectors. This method has smaller model and easier to train, but hasnt included the interaction between two sentences. For matching-aggregation framework, the two sentences are matched at the beginning, including word-by-word and/or phrase-by-sentence, and then the results are aggregated into vectors. This method has included the interactions between two sentences, but usually requires relatively larger model.

The first algorithm "Siamese"[**?**] is consist of two weight-sharing convolutional neural networks (CNNs). Each takes one input and projects it into lower-dimensions based on the CNN, where similar items are contracted and dissimilar ones are dispersed over the learned space. And there is a final layer to solve the sentence paring problem. The model is small and easy to train, and adaptive to various purposes. However, the algorithm does not incorporate interactions between sentences, and hence the matching-aggregation is developed.

For matching-aggregation framework, another possible approaching method is described in Wang, Z s paper published recently.[**?**] Basically, they have introduced a bilateral multi-perspective matching (BiMPM) method with 5 levels. Word Representation is used to convert words to vectors. Context Representation Layer is used to encode contextual information. Matching Layer is used to compare the similarity of the two sentences. Here one time-step of one sentence is compared against all time-step of the other sentence. **"Aggregation Layer"** is used to convert matching vectors into fix-length vectors. **"Prediction Layer"** is used to evaluate the conditional probability.