Machine Learning Engineer Nanodegree

Capstone Proposal

Akila Jeeson Daniel

September 20, 2017

Domain Background

Natural Language Processing (NLP) is field in computer science/artificial intelligence which deals with interaction of computers with human natural languages. There are different topics of research such as language understanding, replication, summarizing, translations. I am interested in the language understanding part of the problems. One of the fundamental problems to solve is how two sentences are similar. This comes under the topic of Paraphrase Matching in NLP. We are trying to quantify the similarity in the intent of two sentences as understood by a human reviewer. This is a current research problem with many solutions being proposed on the academia standard dataset as Microsoft Research Paraphrase Corpus (https://aclweb.org/aclwiki/Paraphrase_Identification_(State_of_the_art)).

Problem Statement

The problem statement we are trying to solve in this project is - 'How similar are two questions?'. This specific problem was addressed as a 2016 Kaggle competition organised by Quora. As a baseline approach, we can convert sentences to vectors, use cosine distance between vectors to measure sentence similarity and then train a basic machine learning model to predict duplicates. But with more advanced word vectorization techniques and newer machine learning algorithms, we aim to improve the accuracy of our duplicates predictions.

Datasets and Inputs

The dataset we are going to use for our analysis is the Quora Question Pairs dataset (https://www.kaggle.com/quora/question-pairs-dataset). The problem we are trying to solve The dataset consists of 404350 question pairs which as labeled as duplicates (1) or not (0) which 149306 are in the 'are duplicates' class. There are 789801 different questions in the dataset.

Solution Statement

We can improve on our benchmark model by using advanced word to vector conversion models such as word2vec and then use an advanced deep learning model such as LSTM or GRU to better predict if the questions are duplicate or not within each pair. My project is inspired by - Bogdanova 2015 (https://aclweb.org/anthology/K15-1013), Addair 2016 (https://web.stanford.edu/class/cs224n/reports/2759336.pdf), Homma et al. 2017 (https://web.stanford.edu/class/cs224n/reports/2748045.pdf)

Benchmark Model

Our benchmark model would be built as follows -

1. split the sentences to words

2. remove stop words and punctuation

3. convert sentences to vectors with weights by tf-idf

4. Find cosine similarity distance

5. use logistic regression to determine at what threshold of cosine similarity distance, the accuracy and F-score of prediction of duplicates, reach a maximum value.

Evaluation Metrics

To evaluate the performance of our models, we use accuracy and F-score which can be defined as -

TP - Number of objects in Positive class predicted by the model to be in the positive class.

FP - Number of objects in Negative class predicted by the model to be in the positive class.

TN - Number of objects in Negative class predicted by the model to be in the negative class.

FN - Number of objects in Positive class predicted by the model to be in the negative class.

Accuracy = TP + TN/ (TP + TN + FP + FN)

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

F-score = 2 * (Recall * Precision) / (Recall + Precision)

Project Design

For this project, I intend to use Python 2.7 and libraries such as scikit-learn, Tensor-flow.

Our dataset consists of more than 400,000 question pairs which are labeled as duplicates or not. I intend to split it as test and training datasets using a 30% split.

For feature extraction, we split the sentence to words, remove stop words and punctuations, compute the frequency of words in our entire dataset and use that to build word vectors of sentences using term frequency - inverse document frequency as weights for words. Then using cosine similarity, we measure the distance between the word vectors of sentences. Then we use a simple machine learning model such as logistic regression to determine at what value of cosine similarity can be used as a threshold for our predictions. We aim to use accuracy and F-score during k-fold cross validation to determine our threshold value.

Next as an improvement to our baseline model, we improve on vector representation of sentences using Word2Vec model (https://www.tensorflow.org/tutorials/word2vec). Also, for our training algorithm, we aim to use Deep Learning algorithms such as such as LSTM or GRU. We believe these changes in our methods will improve our accuracy an F-score significantly.