Machine Learning Engineer Nanodegree

Capstone Proposal

Gennady Lungu
March 31, 2019

Proposal

The following Kaggle competition has drawn my attention and I propose to participate in it by providing a Late Submission (competition already closed): https://www.kaggle.com/c/quora-insincere-questions-classification. Even though the competition is closed I find Natural Language Processing Tasks really fascinating and want to try out what I've learned on a real-world project like this.

Domain Background

The competition is initiated by Quora. Quora is a platform that empowers people to learn from each other. On Quora, people can ask questions and connect with others who contribute unique insights and quality answers. Since Quora's main idea is people learning from each other by asking questions and providing answers, and since virtually any question can be asked, some posts may present a problem. It is those posts that camouflage as a question yet aim at promoting a pre-conceived point of view or seek to slander a group of people without any intent of asking a question or learn something.

In my opinion this presents a challenging problem that is not easy to tackle with conventional linguistic methods, because connotations, irony, sarcasm etc. are not easily caught with direct grammar analysis. On the other hand, it seems that Machine Learning tools should be able to successfully tackle it given enough data (and Quora seems to be a great source of such data).

Problem Statement

Given a question (interrogative sentence) classify it as sincere or insincere. By insincere we will mean questions that are "founded upon false premises, or that intend to make a statement rather than look for helpful answers". Some characteristics that can signify that a question is insincere:

- Has a non-neutral tone
  - Has an exaggerated tone to underscore a point about a group of people
  - Is rhetorical and meant to imply a statement about a group of people
- Is disparaging or inflammatory
  - Suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype
  - Makes disparaging attacks/insults against a specific person or group of people
  - Based on an outlandish premise about a group of people
  - Disparages against a characteristic that is not fixable and not measurable
- Isn't grounded in reality
  - Based on false information, or contains absurd assumptions
- Uses sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers

A dataset of over 1,300,000 questions marked as insincere or sincere is provided by Quora. This training data includes the question that was asked, and whether it was identified as insincere (target = 1). The ground-truth labels contain some amount of noise: they are not guaranteed to be perfect.

The task is to train a model on this dataset, and then apply this model to a test set to see how well it generalizes to questions it has never seen before.

Datasets and Inputs

The following files are provided as inputs.

- **train.csv** - the training set
- **test.csv** - the test set
- **enbeddings** – the word embeddings (word to vector encodings that seek to encode closely related words as closely located vectors) from four well-known sources that can be used along with the dataset when building or training the model. These are as follows:
    - GoogleNews-vectors-negative300 - https://code.google.com/archive/p/word2vec/
    - glove.840B.300d - https://nlp.stanford.edu/projects/glove/
    - paragram_300_sl999 - https://cogcomp.org/page/resource_view/106
    - wiki-news-300d-1M - https://fasttext.cc/docs/en/english-vectors.html

The training set itself contains the following data fields:

- **qid** - unique question identifier
- **question_text** - Quora question text
- **target** - a question labeled "insincere" has a value of `1`, otherwise `0`

Solution Statement

The solution is going to use a Recurrent Neural Network built in Python with Keras on top of Tensorflow. The Neural Network shall output the probability of a sentence being insincere. If produced value is less than 0.5 our final prediction will be 0 (sincere), and 1 (insincere), otherwise.

Benchmark Model

In the benchmark model, it is proposed to use so called word n-grams (in the fields of computational linguistics and probability, an n-gram is a contiguous sequence of n items from a given sample of text or speech). For example, in “What is your last name?” we have the following 3-grams: “what is your”, “is your last” and “your last name”.

For each n-gram we shall calculate the number and, more importantly, the frequency of its occurrence in ‘sincere’ and ‘insincere’ questions. Those n-grams whose frequency in insincere questions is at least one order of magnitude (i.e. at least 10 times) bigger than in sincere ones, will be considered “target” n-grams.

Finally, we will check each test question on occurrences of target n-grams, and if more than one target n-gram is contained within a question, we shall predict it as ‘insincere’, and ‘sincere’ otherwise. It is proposed to use 3-grams in the benchmark model.

Evaluation Metrics

Submissions are evaluated on F1 Score between the predicted and the observed targets on a test dataset. (This is clearly quantifiable, measurable and replicable).

Project Design

The project design is better described in terms of project workflow.

- **Data Preprocessing** (same preprocessing shall be applied to data before both training and prediction):
  - **Special characters** - this shall include quotes and other special character removal, converting to lower case etc.
  - **Max length** - question length shall be limited to some max value, that fits most of the questions. This may involve finding outliers (unusually long questions). If no outliers exist, max length shall be simply max length over all dataset.
  - **Embeddings** – after preprocessing, each word in a question shall be converted into a vector according to a given embedding. If N is the embedding dimension, and M is the max word length, then prepared input data shall have shape (X, N, M), where X is the number of records.
- **Train-Validation split** – prepared input data shall then be split into Training and Validation sets. Validation set is needed to see how the training process is doing (e.g. early stopping), and for selecting the best model in case of hyper-parameter tuning. Another strategy could be to use Cross-Validation, but it will require more training time, so shall be decided later based on that.
- **Training** - the training data shall then be fed into the RNN for training. At this point the idea is to use several LSTM layers, followed by one or two fully-connected ones. The hyper parameters, like number of layers, learning rate, batch size etc would be left for hyper-parameter tuning process.
- **Validation and hyper-parameter tuning**– check performance on the validation set, then change a given hyper-parameter and go back to training phase. Stop when achieved satisfactory performance, i.e. better F1 score that the one achieved on the benchmark model.
- **Prediction** – generate predictions and commit to Kaggle to be tested on their test set. In case performance on Kaggle test set is too poor, a complete Network re-design might be required and the process shall be repeated again.