Capstone proposal

Zoe Zhou

November 29th, 2017

Domain Background

NLP is the acronym of Natural Language Processing. Using Machine Learning in this area, computers can "read" articles, extract useful information from unstructured Data, classify news, "understand" meanings and calculate words similarity, and even more, computers are capable of judging the sentiment of reviews.

Generally, machine learning algorithms require the input to be represented as a fixed-length feature vector. When it comes to texts, one of the most common fixed-length features is bag-of-words. BoW is a simple but applicable method in classifying articles, such as spam emails detection. Despite its popularity, BoW pays heavy attention to words' frequency and has two major weaknesses: ignores the ordering of words and the meaning of words. These weaknesses has greatly limited BoW's competence in text mining.

In 2013, Google published Word2vec method[1], a deep learning achievement in NLP, which assigned computers new abilities to understand words internal and external meanings. To everybody's surprise, without learning any special semantic or syntax rules, just by inputting lots of articles to a shallow-layered neural network model, computers could know words' intension so well that they can even achieve this famous equation: "king - man + woman = queen".

On 16 May 2014 , Quoc V. Le, Tomas Mikolov published the famous article "Distributed Representations of Sentences and Documents"[2], in which they developed an efficient way inspired by word2vec to represent features of paragraphs in a dense fixed-length vector. That is the origin of today's Doc2vec algorithm. By

1 https://rare-technologies.com/word2vec-tutorial/
2 https://cs.stanford.edu/~quocle/paragraph_vector.pdf

using Doc2vec algorithm, engineers could gain efficient text features and achieve a new state-of-art in text classification and sentiment analyzing.

Problem Statement

In this capstone project, I will dive into an interesting competition on Kaggle, "bag of words meet bag of popconrn", you can click on the title to read more details about the competition.

The goal of this competition is to predict sentiment of all the movie reviews in test data. The inputs are labeled and unlabeled movie reviews dataset from IMDB. This is a binary classification problem.

Datasets and Inputs

The dataset for this project can be found in the data section of this competition on the kaggle platform.

Data Set

The data set comes from IMDB, which is specially selected for this sentiment analysis competition. The labeled training dataset consists fo 25000 IMDB movie reviews with a binary sentiment field, which is 1 when the IMDB rating <5 and 0 while rating >=7. Notice, the dataset has been specially filtered to make sure:

1. No individual movie has more than 30 reviews,
2. There is no intersection between the movies in the labeled training dataset and the movies in test set.

File descriptions

- labeledTrainData - The training set with a labeled field – 'sentiment'. The file has 25,000 rows of movie reviews. The positive and negative classes in this dataset are well balanced, 50% vs 50%.
- testData - The test set with only an id and text for each review. The file has 25,000 rows. Our goal is to predict the sentiment for each review.

- unlabeledTrainData - An extra training set with 50,000 rows of movie reviews. Because Doc2vec is an unsupervised learning algorithm, we could make good use of this data set to help generate better feature vectors.

Data fields

- id - Unique ID of each review. There are two parts in ID, movie id + review index.
- sentiment - Sentiment of movie reviews; 1 for positive reviews and 0 for negative reviews
- review - Text of the movie reviews; This is the most important field we have, we will extract feature vectors for each review from this text info.

Solution Statement

To get a better result, I will use Doc2vec algorithm to extract and represent features of movie reviews. There're different models using different parameters, such as PV-DMC (paragraph2vec distributed memory concatenation), PV-DMM(paragraph2vec distributed memory average), PV-DBOW(paragraph2vec distributed bag of words), and so on. I'd like to try each of them to select the best one.

Because Doc2vec is an unsupervised learning algorithm, we can use unlabeled training reviews and even the test reviews to gain paragraph2vec vectors for movie reviews. Using this consideration, we could maximize the corpus to better represent our text features.

Besides, I will try and compare multiple classification algorithms, such as logistic regression, naïve bayes, support vector machine, ensemble method, neural network,   to see in text sentiment classification scenario, which classification algorithm predict the best coping with doc2vec feature vectors. I believe the whole process of this capstone project will be full of challenge and joy.

Benchmark Model

This competition’s benchmark model is a “Word2Vec - Bag of Centroids” model, which has an AUC of 0.84528, ranking after the No417.

Evaluation Metrics

Submissions are judged on area under the ROC(**receiver operating characteristic curve**) curve.
Related metrics are :
true positive rate (TPR, sensitivity, recall)
false positive rate (FPR, 1- specificity)

Project Design

The main workflow of this project is as following :

When you resubmit, please add a bit more detail to your workflow. For instance, what are all of the steps that you'll take during pre-processing? What are all of the supervised learning techniques you plan on trying? How will you tune your supervised learning algorithms?

We understand you can't possibly know how your solution will turn out...you haven't done it yet! However, we need to make sure that you're on the right track and won't try things that waste your time or won't work out.

Keep in mind that this is a great opportunity to bounce lots of ideas off of the reviewers. You can get feedback without putting in much work. It's also a good idea to build some backup plans into your workflow in case something doesn't work. You don't want to get stuck...

1. import all data of reviews
2. normalize text into words and review paragraphs
    a) cleaning: remove markup, control characters, non-alphabet characters;
    b) transforming: uppercase to lowercase;
    c) tokenizing: change reviews to wordlist
    d) removing stop words: this step could be optional
3. train paragraph2vec using Doc2vec

4. tune train models and parameters
    a) splitting the training dataset for cross validation, keeping the class balance across each subset;
    b) tune learning rate and try grid search of doc2vec parameters
    c) calculate error rate and compare models
5. choose the best Doc2vec model to infer features of test movie reviews
6. try and compare different classification algorithms with feature vectors from the following classification algorithms:
    a) Logistic Regression;
    b) Gaussian Naïve Bayes;
    c) Ensemble Method;
    d) K-Nearest Neighbors (K Neighbors)
    e) Support Vector Machines (SVM)
    f) Neural Network
7. choose the best classifier to predict sentiment of the test set and upload result to kaggle
8. get feedback from kaggle and tune model
9. summary of achievements and potential further improvements