Machine Learning Engineer Nanodegree

Capstone Proposal

Eric Restrepo
4/17/2017

Proposal

(approx. 2-3 pages)

Domain Background

(approx. 1-2 paragraphs)

My project will be to use multiple Natural Language Processing (NLP) techniques and supervised learning algorithms to solve the challenge on HackerRank.com at this link: https://www.hackerrank.com/contests/indeed-ml-codesprint-2017/challenges/tagging-raw-job-descriptions

NLP has been in existence as a concept for many centuries, but the first recorded work started in the late 1940's and early 1950's as there was a lot of interest in using NLP as a means to break codes passed accross enemy lines during World War 2. In 1950 Alan Turing, who worked on the code breaking problems during that time, published an article which established the criteria for what is now known as the Turing test. The test was a measure of how intelligent a machine is compared to a human as measured by its ability to be indistinguishable from a human during conversations. Also, in 1954, research called the Georgetown experiment which involved automatic translation of over sixty Russian sentenced into English led the authors to boldly predict that within 3-5 years machine translation would be a solved problem. These early wins in NLP led to more funding for Artificial Intelligence related research which, ultimately fell flat following the ALPAC report in 1966, which told of the disapointing results of more than 10 years of research in the field. Ultimately there were very few new discoveries in NLP during the 1960's and 1970's as most of the work creating artificial chat agents during this time was

Problem Statement

(approx. 1 paragraph)

Indeed.com is a well known job posting website and search engine with thousands of posting for jobs. The problem they would like to address is to more accurrately tag the job postings with some specific tags that will allow users to filter for the types of opportunities that they are searching for. The tags that they would like to automatically assign to a text include these twelve tags:

part-time-job full-time-job hourly-wage salary associate-needed bs-degree-needed ms-or-phd-needed licence-needed 1-year-experience-needed 2-4-years-experience-needed

5-plus-years-experience-needed supervising-job

Datasets and Inputs

(approx. 2-3 paragraphs)

The training dataset consists of 4375 job descriptions with the accompanying tags that are included in each post. The testing dataset 2921 job descriptions with no tags. The dataset was given by hackerrank.com and is downloaded as a zip file called indeed_ml_dataset.zip and it contains a tags.tsv file and a test.tsv. The tags.tsv file has a header row with two columns including "tags" which is a space separated list of tags and "description" which is a job description. https://www.hackerrank.com/contests/indeed-ml-codesprint-2017/challenges/tagging-raw-job-descriptions

This will be a supervised learning problem using tags as the training labels and the descriptions as the training data.

Solution Statement

(approx. 1 paragraph)

The solution would be an algorithm that correctly classifies and tags the job descriptions to at least a human level of accuracy with a minimal amount of training and retraining time for the algorithm and a maximum accuracy.

Benchmark Model

(approximately 1-2 paragraphs)

I guess the benchmark model in this case would be a human being manually going through and tagging each job description which would take a few minutes per job posting. With just 4375 job postings, this would then take approximately 2 minutes x 4375 = 8750 minutes or 145.83 hours or 18.229 days x 8 hour work days. In a month there are approximately 22 working days. So this means that a human being would take almost a month to tag just 4375 job descriptions. I am sure that Indeed.com has many more job descriptions than this. Possibly in the millions of job postings. I'm not sure what algorithms they use right now, but even a simple model would be better then having a human tag all of the job postings manually.

Evaluation Metrics

(approx. 1-2 paragraphs)

The evaluation of the various models that I will try will be measured against the F1 score on a preselected set of 1446 of the test set descriptions. The F1 score is calculated as follows: F1 = 2 * (precision * recall) / (precision + recall) where precision = (# of True Positives) / ((# of True Positives) + (# of False Positives)) and recall = (# of True Positives) / ((# of True Positives) + (# of False Negatives))

Project Design

(approx. 1 page)

In this final section, summarize a theoretical workflow for approaching a solution given the problem. Provide thorough discussion for what strategies you may consider employing, what analysis of the data might be required before being used, or which algorithms will be considered for your implementation. The workflow and discussion that you provide should align with the qualities of the previous sections. Additionally, you are encouraged to include small visualizations, pseudocode, or diagrams to aid in describing the project design, but it is not required. The discussion should clearly outline your intended workflow of the capstone project.

Intended workflow at a high level:

1. Import Data from training file

Use pandas read_csv with the delimiter set to "/t" for tab.

1. Clean Data and Preprocess so that it is ready to be fed into the model

This is a supervised learning problem, so we have to get the labels and the data into a format that is useable by one of our supervised learning algorithms.

I would like to compare the deep learning models to the standard machine learning models to see which performs better, but the data needs to be in two different formats for these two different algorithms.

For the deep learning models the labels can be one hot encoded as a vector with shape (1,12) to be fed into a deep neural net or if I were to use a standard machine learning model, I would run a classification algorithm 12 different times for each of the labels as a 0 or 1.

1. Train the model

When training the model for the standard machine learning classification I plan to use the following models as possibilites:

MultinomialNB RidgeClassifier LinearSVC SGDClassifier Perceptron PassiveAggressiveClassifier BernoulliNB KNeighborsClassifier NearestCentroid RandomForestClassifier

For the deep learning algorithm, I plan to use an experimental approach that was put forth by this academic paper: https://arxiv.org/pdf/1502.01710.pdf

They used a specific Convolutional Neural Network or CNN to train a classifier of the individual letters as vectors.

1. Make Predictions using test set

5. Save the output to a .tsv file

Before submitting your proposal, ask yourself. . .

- Does the proposal you have written follow a well-organized structure similar to that of the project template?
- Is each section (particularly **Solution Statement** and **Project Design**) written in a clear, concise and specific fashion? Are there any ambiguous terms or phrases that need clarification?
- Would the intended audience of your project be able to understand your proposal?
- Have you properly proofread your proposal to assure there are minimal grammatical and spelling mistakes?
- Are all the resources used for this project correctly cited and referenced?