Dariia Dragunova
6 December 2022
CIS 4526

ML Final: Systems Description

Features Description

For this project, I tried to identify features that would help tell if two sentences were similar or not. Most of the features considered the structural aspect of sentences such as their length, number of identical words, number of n-gram overlaps and more. By looking at sentences in the training and development datasets, I tried to identify differences between the sentences and how I could find those differences in the algorithm. Some things that stood out were the capitalized words and numbers that appeared in a lot of sentences. Below is the list of features that were used for my model along with their descriptions.

1. Word Count Difference
    a. This feature compares the number of words in each sentence and returns the difference
2. Word Overlap
    a. This feature compares the words in each sentence and returns the number of words that are the same in both sentences
2. Word Union
    a. This feature counts the number of unique words in two sentences
3. 3 Gram Overlap
    a. This features calculates three gram overlaps in each sentence and returns the number of 3-gram-overlaps that are the same in both sentences
4. 3 Gram Union
    a. This feature counts the number of unique 3-grams in both sentences
5. 2 Gram Overlap
    a. This features calculates three gram overlaps in each sentence and returns the number of 2-gram-overlaps that are the same in both sentences
6. 2 Gram Union
    a. This feature counts the number of unique 2-grams in both sentences
7. Numbers Overlap
    a. This features counts occurrence of numbers in a sentence and returns the difference between sentences
8. Capitalized Words Overlap
    a. This features counts occurrence of numbers in a sentence and returns the difference between sentences
9. Cosine Similarity Score
    a. This feature calculates the cosine similarity score for two sentences

10. <u>BLEU-4, BLEU-3, BLEU-2</u>
    a. These features calculate a number between zero and one that measures the similarity of the machine-translated text to a set of high quality reference translations
11. <u>NIST-2, NIST-3</u>
    a. This feature calculates n-gram co-occurrence statistics
12. <u>METEOR</u>
    a. This feature scores machine translation hypotheses by aligning them to one or more reference translations
13. <u>Levenshtein Distance</u>
    a. This feature is a measure of the similarity between two strings by counting the distance in the number of deletions, insertions, or substitutions required to transform string 1 into string 2.

Data Preprocessing

1. Identifying bad lines
    a. When opening the files, I had trouble processing some lines because of bad formatting. Some lines had strings in the column for 'gold_label' as well as extra quotation marks in the sentences themselves.
2. Working with sentences
    b. Processing the strings was a very important aspect of this project. All the features dealt with identifying structural components of sentences, thus it was important to perform the following operations on sentences.
        i. Lowercase all words in a sentence
        ii. Tokenize the words
        iii. Remove the stopwords
        iv. Lemmatize the words (reduce the different forms of a word to one single form)
3. StandardScaler()
    a. After getting all the feature vectors, one of the important steps was to use StandardScaler for standardization of the data because Multi-layer Perceptrons are sensitive to feature scaling

Libraries

1. NLTK
    a. NLTK or Natural Language Toolkit is a Python suite of libraries and programs for symbolic and statistical natural language processing in English
    b. I used this library for data preprocessing and feature extraction. Useful functions helped me tokenize, lemmatize and perform other manipulations with sentences.
2. Pandas
    a. Pandas is a Python library written for data manipulation and analysis.

    b. Used Pandas to load the data, create data frames and perform manipulations on the data.
3. Sklearn
    a. Sklearn is a Python library that provides a selection of efficient tools for machine learning and statistical modeling.
    b. Used this library to create and experiment with different models.
4. Re
    a. Re is a module of regular expression matching operations.
    b. Used Re library to dissect sentences and find useful features in them such as capital letters and numbers.
5. Math
    a. Math is a module that provides access to the mathematical functions defined by the C standard.
    b. Used math to perform basic mathematical operations as well as matrix manipulations.
6. Seaborn and Matplotlib
    a. Used these two libraries to visualize the data
7. Numpy
    a. Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices
    b. In this project, it was used in order to find the best value for MLP parameter 'alpha' using 10.0 ** -np.arange(1, 10)

Algorithms

I tested four different models for the this task.

1. <u>MLP - Multi-Layer Perceptron</u>
    a. Parameters
        i. Tried different parameters using GridSearch()
        ii. `{'solver': ['lbfgs'], 'alpha': 10.0 ** -np.arange(1, 10), 'hidden_layer_sizes':np.arange(10, 17), 'random_state':[0,1,2,3,4,5,6,7,8,9]}`
        iii. Best parameters were:
            1. Solver: ‘lbfgs’
            2. Alpha: 0.1
            3. Hidden Layer Sizes: 13
            4. Random State: 4
    b. Activation function
        i. I used Logistic activation function as it gave me the best score
    c. Results

        i. Using this MLP algorithm with said parameters I was able to get an F1 score of 0.87 trained on train set and tested on dev.

Experience and Lessons

a. Working with a large data set
    i. I have never worked with a dataset this large. It was a new experience for me and taught me a lot about data pre-processing. It also showed me how long it may take to run a model on larger data sets. For some of the models, training and testing would take up to 40 minutes.
b. Working with Neural Networks
    i. Before starting this project, I had a slight idea of what neural networks were. I thought I understood how they work and wouldn't have trouble implementing it. I started off by trying to use PyTorch to create a model and got confused very fast. I read multiple articles and documentation materials to understand it better and still wasn't sure what I was doing. With the sklearn library implementing MLP was way easier and gave me understanding of important parameters that can be fine-tuned for the model.
c. Learning new features
    i. After reviewing the midterm projects, I learned about a lot of useful metrics that could be used to measure differences between sentences. NIST score and METEOR were biggest revelations. As I was researching more, I found a bunch of other cool features that could help separate sentences apart.

Resources

- https://scikit-learn.org/stable/modules/neural_networks_supervised.html
- https://www.kaggle.com/code/hatone/mlpclassifier-with-gridsearchcv/script
- https://cla2019.github.io/paraphrases_scikit_numpy.pdf
- https://towardsdatascience.com/top-10-binary-classification-algorithms-a-beginners-guide-feeacbd7a3e2