PRESENTING A COMPARATIVE ANALYSIS FOR GRAMMAR ERROR DETECTION

Group 2: Shreyas Nikam, Raj Doshi, Sravani Namburu

https://github.com/Rajdoshi99/Grammar-Error-Detection

Problem Statement:

Grammatical Error Detection is something that is employed in everyday life. People have begun to use various grammatical tools and chrome extensions like Grammarly to do their activities more conveniently and with more simplicity. This project will undoubtedly target customers who wish to know whether or not their language is grammatically correct. Although such initiatives have been established, they have automatically received a poor score and have not been compared with alternative embedding approaches. As of 2022, we have many embedding techniques and algorithms that can assist us to recognize whether phrases are grammatically accurate in the best way possible. We intend to construct numerous models with different embeddings to compare how well that specific model with that particular embedding could do in solving this challenge.

Dataset:

We will be using a dataset provided by Kaggle at the following link:
https://www.kaggle.com/datasets/s105022206/grammer-error

The dataset contains 303662 datapoints for training and 38295 datapoints for testing. The dataset consists of correct and incorrect sentences, which have to be further preprocessed and mapped to 0 and 1 accordingly..

Approach to Solution:

a. **Methodology:**

Referring to [1] as our **base paper**, we plan to compare the performance of different models for the task of grammar error detection. It involves a number of subtasks such as spelling error detection, word choice error, punctuation error, prepositional errors, syntactic and semantic errors.

We propose to deal with these subtasks at a logical level and compare its performance against neural networks. For example, the task of spelling error detection can be accomplished by a simple dictionary lookup. Word choice errors can be detected using **n-gram models**. We plan to use the outputs of these subtasks as features for algorithms like **Logistic Regression and Naive Bayes** and compare its performance against neural networks such as **LSTM, RNN and bidirectional models** which perform these subtasks automatically.

Embeddings need to be passed to these neural networks for them to perform precisely. For this task, we aim to compare the performance of different embeddings such as **CountVectorizer, TFIDF-Vectorizer, Glove, and Fastext** and present the results of the same in our study.

We **hypothesize** that bidirectional models would work better, since they would be able to capture the contextual awareness of words for detecting these errors.

b. **Assessment:**

Owing to the contributions of [3], we have decided to use $F_{0.5}$ **score** which combines precision and recall, giving more (twice) weightage to precision than recall, since we believe that accuracy is more important than coverage for this task. Alternative metrics: **F1 score, homogeneity, completeness, v-measure and rand index.**

c. **Ablation study:**

We aim to provide a comparative analysis of the results from the abovementioned models, alongwith the effect of the different embedding techniques. In addition to this, we will present a study of different architectures that perform well. Finally, we plan to compare the results of our techniques against state of the art [2], pretrained models to evaluate our approach.

Timeline

a. Week 9: Clean and pre-process our data and split between training and testing
b. Week 10: Implementing various embedding techniques and featuring the output into different models like Logistic Regression and Naive Bayes.
c. Week 11: Following up with week 10, LSTM, RNN and bidirectional models are implemented.
d. Week 12: Evaluating the models using internal evaluation metrics like F1 score, homogeneity, completeness, v-measure and rand index.
e. Week 13: Write the report.

Responsibilities:

a. Shreyas: Logistic Regression and Bidirectional LSTM
b. Sravani: LSTM and RNN
c. Raj: Naive Bayes and Bidirectional RNN
d. Together: Preprocessing the data, evaluation and final report

Related Work:

Paper [1] presents a comparative analysis of the different neural network models used for error detection. We hypothesize that LSTM and RNN models would work better than simple classifiers for the described task.

Paper [2] presents the state-of-the-art Neural Verification Network for Grammar error correction with multiple hypothesis. We plan to compare the results of our approach against this pre-trained model for better understanding of the standing of our approach.

Paper [3] describes the drawbacks of existing simple metric evaluation methods and works towards building a new metric system for evaluation of grammatical error detection and correction.

References:

[1] Marek Rei, Helen Yannakoudakis: Compositional Sequence Labeling Models for Error Detection in Learner Writing. CoRR abs/1607.06153 (2016)

[2] Zhenghao Liu, Xiaoyuan Yi, Maosong Sun, Liner Yang, and Tat-Seng Chua. 2021. Neural Quality Estimation with Multiple Hypotheses for Grammatical Error Correction. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5441–5452, Online. Association for Computational Linguistics.

[3] Mariano Felice and Ted Briscoe. 2015. Towards a standard evaluation method for grammatical error detection and correction. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 578–587, Denver, Colorado. Association for Computational Linguistics.