Introduction to NLP
Project Outline

“Measure Text Fluency”

Team 27
Project No: 4
Team Name: $pecial ¢haracters

Team Members:

- Thota Gokul Vamsi, 2019111009
- Sai Akarsh C, 2019111017

Problem Statement

Text fluency is an estimate corresponding to the quality, readability, accuracy and comprehensibility of a piece of text. Any such textual information generated by an automated source is prone to errors of language understanding, and require rectification.

Measuring text fluency is an important task to address this issue, as it describes if the required information is conveyed in a desirable format, from a given piece of text. This requires taking into account various criteria such as grammar, style, choice of words etc. and hence is a relatively challenging task.

We address this problem as a multi-class classification problem, where a piece of text is classified either as Not Fluent, Neutral or Fluent. We also use some methods which take manual references and annotations into account, for measuring fluency.

Related Work

Liu et al. [1] have explored an approach to measure text fluency of an MT system, and have taken into account the n-gram features for scoring a sentence based on its fluency. It was also found that there was a high correlation between such features and fluency, thus achieving decent results even in absence of manual references for given text. Kann et al. [2] have proposed a measure called syntactic log-odds ratio (SLOR), which acts as a normalized LM scoring to estimate fluency. They have also incorporated wordpiece models to reduce vocabulary size and improve unknown-words handling.

Dusek et al. [3] have tried a supervised-learning approach using LSTMs to predict the quality of text. Lin et al. [5] have attempted using metrics such as ROUGE-L (longest common subsequence length between a sentence and its manual reference), and ROUGE-S (Skip-Bigram matches between a sentence and its manual reference) for evaluating quality of an MT system.

Implementation Scope and Plan

For the baseline, we intend to implement an n-gram language model (all n-gram probabilities of given sentence, perplexity as features), and then perform a 3-class classification using Logistic Regression (3 classes are Not Fluent, Neutral, Fluent). We further intend to experiment on it by calculating scores for each sentence considering above n-gram features (tweaking it such that it uses only

frequent or rare n-grams, as done in [1]) and compute its correlation (via Pearson correlation) with the annotated labels.

We also intend to experiment by performing classification using standard ML architectures such as Support Vector Machines, Decision Trees, Random Forest, K-nearest neighbours, by utilizing textual features. We would further add features such as POS n-grams overlap (between text and reference), frequency of rarest few n-grams in sentence, grammaticality check using libraries such as NLTK. We would compute the correlation scores and analyze the performances in both these cases.

We would implement computation of metrics such as ROUGE-L, ROUGE-S, SLOR (discussed briefly above), n-gram overlap (between text and reference) and perplexity, on experimenting with various Language Models utilizing RNN, LSTM, transformer architecture. We compute the Pearson correlation to analyze the performance of various metrics and their association with text fluency.

Eventually, we would implement a simple application which takes a piece of text as input and predicts its corresponding fluency label in real-time. For this, we would be using the best architecture (or combination of multiple architectures) based on above experimentations.

Datasets

We would primarily be using the below compression dataset provided by Microsoft. It contains single sentences and two-sentence paragraphs from the Open American National Corpus (OANC). We would be utilizing the English Gigawords corpus and CNN stories data for training of LMs as and when necessary.

- Intelligent Editing - Microsoft Research
- Gigaword Dataset - NLP Hub - Metatext
- Process-Data-of-CNN-DailyMail

Timeline

- *Feb 16th - 24th:* Baseline model implementation (n-gram, LR), experiments with scoring system as done in [1], correlation score computation
- *Feb 25th - Mar 3rd:* Experimentation with supervised approaches - SVM, DT, RF, KNN
- *Mar 4th - 11th:* Experimentation by adding multiple features - POS n-grams, grammaticality, frequency of rare n-grams in sentence
- *Mar 12th - 15th:* Detailed analysis of above experiments - interim report
- *Mar 16th - Apr 1st:* Implementing ROUGE-S, ROUGE-L, SLOR, n-gram overlap for RNN and LSTM LMs
- *Apr 2nd - 9th:* Implementing above metrics for transformer LM
- *Apr 10th - 15th:* Implementing application utilizing best approach / ensemble for fluency prediction
- *Apr 16th - 21st:* Detailed analysis of all experiments and results - final report

Interim Deliverables

- Interim report
- Code for baseline, all 3 experiments after that (described in timeline)
- Model checkpoints (wherever necessary)

Final Deliverables

- Final report
- Entire code for baseline, all experimentations, implementation of metrics and their functioning with various LMs - with README
- Relevant model checkpoints
- Application for real-time fluency prediction
- Final presentation

References

1. Automatic evaluation of sentence fluency
2. Sentence-Level Fluency Evaluation: References Help, But Can Be Spared!
3. Referenceless Quality Estimation for Natural Language Generation
4. Predicting Grammaticality on an Ordinal Scale
5. Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statistics
6. BLEU: a Method for Automatic Evaluation of Machine Translation
7. GLEU: Automatic Evaluation of Sentence-Level Fluency
8. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics | Proceedings of the second international conference on Human Language Technology Research
9. A Comparative Evaluation of Deep and Shallow Approaches to the Automatic Detection of Common Grammatical Errors
10. Sentence-Level Grammatical Error Identification as Sequence-to-Sequence Correction
11. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation