Measure Text Fluency

Final Report

Team Name: Finals
`
Team number: 12
Ruchitha Jujjuru: 2020101093
Yalaka Surya Teja Reddy: 2020101042
Immadisetty Josh ujwal: 2020102018

1. Statistical Approach to Evaluation of Sentence Fluency

1.1 Amelioration 1: evaluating sentence fluency using the product of n-grams' Conditional Probabilities

$$M(W_1W_2...W_m) = [\sum_{i=1}^{m} \log(P(W_i \mid N - Gram))] / m$$

Here, M is the fluency score of the sentence. P (wi / N-Gram) to denote the Conditional Probability of different n-grams in a sentence

1.2 Amelioration 2: Evaluating sentence fluency by discriminatingly treating strange and familiar n-grams

```
$M(W_1W_2...W_m) = 1/m$
for $i = 1, ...m$
  if $(P(W_i \mid N - Gram) \geq ValForGood)$
    $M(W_1W_2..W_m) = M(W_1W_2..W_m) / P(W_i \mid N - Gram)$
elseif $(P(W_i \mid N - Gram) \leq ValForBad)$
    $M(W_1W_2..W_m) = M(W_1W_2..W_m) \times P(W_i \mid N - Gram)$
endfor
```

In the above algorithm, ValForGood and ValForBad are the two constants used to select those good and bad n-grams. Considering in different training corpus, the CP of good n-grams and bad n-grams may be different, we choose the values of

ValForGood and ValForBad by the following steps. First, we sort descendingly all the CP estimated by the training corpus; Second, under the assumption that the first forty percent of the sorted CP corresponds to those good n-grams, the lowest CP of the first forty percent is assigned to ValForGood; Finally, we consider the last twenty percent of the sorted CP as corresponding to those bad n-grams and then the highest CP of the last twenty percent is assigned to ValForBad.

We have implemented the above two approaches to calculate fluency scores using the N-Gram model trained on the gigaWord corpus.

Summary	A1_score	A2_score
prescriptions elusive for curbing microsoft	-1.015306760687785	174.3800609910745
chinese foreign minister meets with secretary-general of algerian ministry of foreign affairs	-0.6539564915788506	676.094968747561
expansion plan for ##nd street y riles neighbors	-0.7915526763150786	273.9489212342299 4

There is much lesser variation in A1_scores compared to A2_scores.But in most cases, these scores correlate to each other.

2. Syntactic log-odds ratio

Slor is defined as syntactic log-odds ratio (SLOR), a normalized language model score, as a metric for referenceless fluency evaluation of natural language generation output at the sentence level.

For a sentence, it can be calculated as

$$\mathrm{SLOR}(S) = \frac{1}{|S|}(\ln(p_M(S)) - \ln(p_u(S)))$$

The $P_m(S)$ can be calculated via the output of the Language model used, and $P_u(S)$ is the unigram probability of a sentence.

The reason for normalizing with unigram probabilities is that we do not want rare words to bring the score down. We need to divide by sentence length so that we do not prefer shorter sentences over longer ones.

For example, take the two sentences

1. He is a citizen of France.

2. He is a citizen of Tuvalu.

Both are fluent, and we should not penalize them because Tuvalu is less frequent than France, hence normalized by unigram probabilities.

Language Models:

We have used the Gigaword Corpus, which uses the new data, with headlines as a summary of the story, a dataset for compression to train an LSTM. The embeddings have been trained by word2vec with the given description of headlines on the train dataset and have been chosen as a size of 300.

The LSTm has hidden layers of 512 neurons and 2 stacks. The output is a value of all the words that can appear next, each given a value.

The probability of a sentence has been calculated by the last fully connected layer on top of LSTM, whose values are sent through softmax and hence can be used as probabilities. Normalized by the unigram probability (since fluency should not be low if infrequent words are used) we get SLOR our first measure for fluency proposed by the paper “Sentence Level Fluency Evaluation”

After MId :

WPSLOR

With the vocab size very large, the model itself increases in size, increasing training times. We need to reduce this vocab size for more efficient process of training and evaluations.

With the advent of new tokenizers that learn from the corpus and add the most frequently occurring one to the token sets, we can reduce the vocab size. Since we club sub-words, the morphemes are well characterized giving meaning to the tokens. And we also have meaning retained since words are clubbed together not just characters.

Wordpiece tokenizer helps us also to take care of rare words, thereby giving a robust way to decrease training time.

Pre-Trained Decoder Model :

Training the LSTMs was GPU intensive, which we had no access to, so we had to reduce training, so we used GPT.

We used the GPT2 decoder to predict the probabilities of given sentences and the unigram probabilities are obtained based on the count-based methods, these two are used to compute metrics like slor etc..

Human Annotated Dataset :

We correlated our metrics based on human-annotated scores which are obtained from the compressed dataset , The dataset comprises as follows :

Each line of our file will have information on source text and corresponding compressed versions.
Format of source text; SourceID \t Domain \t SourceText
Here sourceId consists of integers connected by _ based on the number of sentences in the source text, The range of domains is newswire, letters, journal, and non-fiction.

Format of compression: CompressedText \t JudgeId \t numRatings[\t Rating]^numRatings
Each rating can take values from 6 to 24 as follows :

6	Most important meaning Flawless language	(3 on meaning and 3 on grammar)
7	Most important meaning Minor errors	(3 on meaning and 2 on grammar)
9	Most important meaning Disfluent or incomprehensible	(3 on meaning and 1 on grammar)
11	Much meaning Flawless language	(2 on meaning and 3 on grammar)
12	Much meaning Minor errors	(2 on meaning and 2 on grammar)
14	Much meaning Disfluent or incomprehensible	(2 on meaning and 1 on grammar)
21	Little or none meaning Flawless language	(1 on meaning and 3 on grammar)
22	Little or no meaning Minor errors	(1 on meaning and 2 on grammar)
24	Little or none meaning Disfluent or incomprehensible	(1 on meaning and 1 on grammar)

Each compressed version of a sentence is evaluated in three different ways 1) grammar 2) meaning 3) Total score and the values in each way are the average of the ratings given by different people in each way.

Baseline Metrics

To understand how better/worse our metric is, we compare it with human-annotated values. We do this comparison by Pearson correlation.

We also compare how well this correlates to human judgement by comparing it with various other metrics like:
ROGUE - used to decide how good summarisation is. It has 3 variants, a bigram, a trigram and ROGUE-L - measures the similarity of two sentences based on their longest common subsequence

ROUGE-1 - measures the overlap of unigrams (single words) between the generated text and the reference text

ROUGE-2- measures the overlap of bigrams (sequences of two consecutive words) between the generated text and the reference text

NCE(negative cross entropy) : This metric calculates the log probability which is only normalized by sentence length. The score of a sentence S is calculated as

$$\mathrm{NCE}(S) = \frac{1}{|S|} \ln(p_M(S))$$

Perplexity : This metric corresponds to the exponential cross entropy ,This can be formulated as follows

$$\mathrm{PPL}(S) = \exp(-\mathrm{NCE}(S))$$

Results:

We first trained an LSTM on gigaword for next-word prediction, but the SLOR scores were very bad due to very less training data (more OOV tokens) and fewer iterations.

With WPSLOR we had lesser model size and training time(Wordpiece tokenizer gave less Vocabulary) but we could not take the entire data and load it into RAM since Disk Utilisation was too slow. We had to constrain the amount of data and the scores were abysmal, as expected since the next token prediction would need much more training Data. The Pearson correlation came out to be 0.02 for the fluency scores, which is as good as nothing, i.e. more fluent or less fluent was not known due to many OOV tokens.

With training as a constraint, we put our sights on pre-trained Models, chose GPT2 as a decoder, calculated the SLOR scores, and compared them with human annotation values
We used a GPT2 decoder to get the SLOR values and these values gave a Pearson correlation 0.12 slightly higher than the correlation with ROGUE-L (0.11)