Rerank

Jordan Matelsky

March 31, 2016

1 Introduction

1.1 Runner.py

In order to improve the reliability and reproducibility of the testing scheme that I use, I wrote a runner (runner.py) to make testing as simple as running a single line from the terminal.

1.2 Testing

Test a Python file by running `./runner [filename] [optional csv args]`. For instance, to test the baseline that was provided, type the following into the terminal: `./runner rerank-baseline`. To test the provided oracle, run `./runner oracle`.

This prints two lines: a tuple of results, in the order of $(oracle, tested, time)$, and a timestamp of how long the test took to run (not including the oracle).

2 Improvement Strategy

To improve over the baseline reranker, I followed several methods to improve upon the low baseline set by the BLEU computation on `rerank`.

2.1 Non-ASCII Penalty

This function, implemented in `Modes.DEFAULT_CHECK_ASCII`, deduces a penalty from the score of a hypothesis every time a word is encountered for which an ASCII-decode is impossible. Naturally, this fails in languages that use the Roman alphabet, but for this example (Cyrillic alphabet), I am able to improve BLEU using a penalty of 20 per non-ASCII (and thus, presumably non-translated) word.

2.1.1 Advantages

This prevents sentences that have many untranslated words from being considered ‘best’. It also seems to generally *improve* BLEU — in my preliminary tests, I did not see cases where this reduced efficacy of the overall ranking.

This is also a very quick check to perform, and it likely serves as a good “first-line of defense” when considering hypotheses.

2.1.2 Disadvantages

This does not necessarily always give the best hypothesis. Consider a case like the following, where # indicates an ASCII failure:

He is a conference last Thursday.

# went to the conference last Thursday.

The second sentence is clearly superior in terms of conveying meaning, but the first sentence would be rated ‘better’ by this algorithm alone. Thus, it is likely that we need to add the insight from other algorithms rather than run this one alone.

2.1.3 Results

This algorithm alone performs only marginally better than the standalone baseline. While the baseline scores 27.35, ASCII-checking scores ≈ 0.003 higher, 27.54.

When the `CHECK_ASCII` algorithm is run on its own, it performs slightly worse than baseline, at 26.3.

2.2 Word Count

Next, I implemented a word-count feature, as this had been widely regarded as a useful measure in previous work [1, 2]. Because BLEU scores fundamentally correlate with sentence-length, it is important that we consider this when scoring sentences in order to prevent extreme-length sentences from being weighted improperly.

2.2.1 Advantages

Again, this is a very quick implementation, but it is, with regards to its speed, disproportionately (very!) useful. (See *Results* for numerical results.)

2.2.2 Disadvantages

This approach naturally fails in cases where sentence-length differs dramatically. Consider languages such as German where compound-words reduce the length of a sentence in word-count, but maintain meaning. BLEU is a useful metric for sentence structure in some languages, but a more general algorithm should be used for a general translation scoring mechanism.

2.2.3 Results

When run alone, word count is (not unexpectedly) useless, scoring well below baseline. However, when run in conjunction with the default scoring system (sum), it performs notably better than baseline, scoring 28.7 (baseline = 27.35).

When all three thus-far explored algorithms are run together (Default baseline, ASCII, and word-count), BLEU-score reaches 28.8.

2.3 2-Gram Popularity

My final contribution to the dev-score as submitted via the leaderboard was adding a basic 2-gram implementation that counts 2-grams *across the hypothesis file*. This is an interesting approach because it does not require a language model or any other external information; all it uses is the already-provided list of hypothesis for *all* sentences.

This provides interesting information: Namely, if a sentence is largely comprised of 2-grams that are seen nowhere else in the corpus, then it is likely that this is a poor translation. Conversely, a sentence whose n-grams are common across the corpus is likely well-formulated.

2.3.1 Advantages

Even running *all* of the aforementioned implementations, the runner is still quite quick, and so no efficiency is fundamentally lost by adding this feature.

Additionally, accounting for this feature improves the score of the overall computation, as explained in the next section.

2.3.2 Disadvantages

In the worst-case, a hypothesis corpus contains a high frequency of recurrent but inaccurate 2-gram (for instance, a 2-gram that is not translated) — in this case, any sentence that contains that 'bad' 2-gram would be positively weighted. However, in a "best-attempt" corpus such as the provided data, this algorithm should not encounter this weakness.

2.3.3 Results

When run in conjunction with all of the above implementations, BLEU-score is 28.87 (baseline = 27.35).

3 Running

To run any of the implementations mentioned above, use `rerank-2grams`. Comment out or de-comment any combination of lines 106-109 to explore behavior with various sub-components of this submission.

```
105|  score = score_hypothesis(hyp, feats, weights, fns=[
106|      (Modes.DEFAULT, {}),
107|      (Modes.CHECK_ASCII, {}),
108|      (Modes.WORD_COUNT, {'word_count_weight': 1.24}),
109|      (Modes.TWOGRAM, {'gram_weight': 5}),
110|   ])
```

To further modify, the syntax of my `score_hypothesis` function takes a hypothesis, features, weights, and then `fns`, which is a list of tuples, each of which is of the format `(function ref, named arguments)`.

References

[1] UPenn Student (Name not provided). *CIS526 HW4 Report*. University of Pennsylvania, 2015. `http://www.seas.upenn.edu/ cis526/reports/hw4/828463.pdf`

[2] Hopkins, Mark, and Jonathan May. *"Tuning as ranking."* Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2011.