Problem Statement

Collect a parallel corpus of English - Hindi , or other near similar Indian languages.Determined by the quality sentence pairs using statistical machine translation (Moses, phrase-based)
Use BLEU score, METEOR, TER and WER measures to access the quality of MT

submit your results with quality scores, one per line, corresponding to the sentence pairs.

Introduction

GIZA++ steps:

Installation

Parallel Corpus

2.Create parallel corpus with tokenized space
3. Align words

Align Words

Navigate to the `GIZA++-v2` directory and run the following which will generate the vocabulary (`.vcb`) and sentences files (.snt):

```
./plain2snt.out ../parallel-n/IITB.en-hi.en ../parallel-n/IITB.en-hi.hi
```

extract the co-occurrences by running the following:

```
./snt2cooc.out ../parallel-n/IITB.en-hi.en.vcb ../parallel-n/IITB.en-hi.hi.vcb
../parallel-n/IITB.en-hi.en_IITB.en-hi.hi.snt > ../parallel-n/corp.cooc
```

Run GIZA++

./GIZA++ -S ../parallel-n/IITB.en-hi.en.vcb -T ../parallel-n/IITB.en-hi.hi.vcb -C ../parallel-n/IITB.en-hi.en_IITB.en-hi.hi.snt -CoocurrenceFile ../parallel-n/corp.cooc -outputpath ../parallel-n/

Output

`2022-10-10.023859.ubuntu.A3.final`: contains sentence pairs and word alignments from the source language (Hindi) into the target language (English) using Viterbi Alignment, which is the most probable alignment (the one that maximizes the alignment probability). One particular sentence pair of this file looks like the following:

# Sentence pair (301) source length 2 target length 3 alignment score : 2.37923e-05
कोई विवरण नहीं
NULL ({ }) No ({ 1 3 }) description ({ 2 })

# Sentence pair (355) source length 5 target length 5 alignment score : 4.10065e-08
एक अंतर्क्रियात्मक पाइथन पहुंचनीयता अन्वेषक
NULL ({ }) An ({ 1 }) interactive ({ 2 }) Python ({ 3 }) accessibility ({ 4 }) explorer ({ 5 })

The first line shows the length (number of words) of the source (Irish) and target (English) sentences, along with the Viterbi alignment score.

The second line is the target sentence, and the third line is the source sentence annotated with alignment information. Each source word is annotated with the set of indices of target words that are aligned to that source word. Note that in IBM Models assume that one target word is aligned at most one source word.

- `2020-04-01.030523.sina.actual.ti.final`: This is the final inverse T-tables (lexical translation probability) trained by the model. Lexical translation probability `t(e|f)` is the probability that word `f` in the source language (Irish) is translated to word `e` in the target language (Target). Since this is the inverse T-tables, it contains `t(f|e)`. The following is a sample of the file:

# file names of the TM tables

# Notes:

# 1. TTable and InversTTable are expected to use word IDs not

#   strings (Giza produces both, whereby the *.actual.* files

#   use strings and are THE WRONG CHOICE.

# 2. FZeroWords, on the other hand, is a simple list of strings

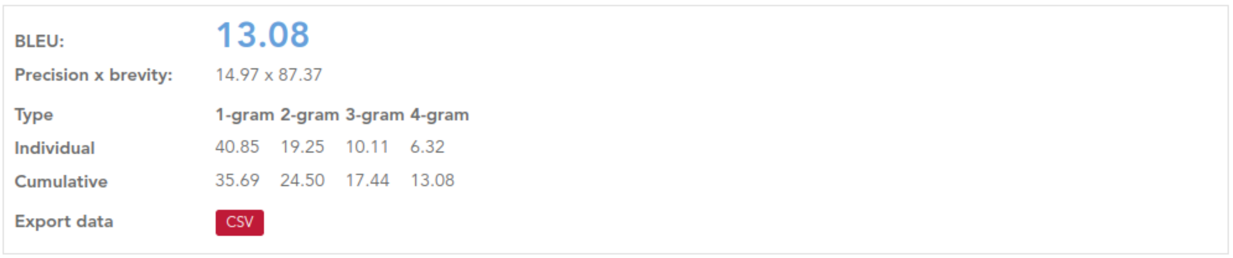
#   with one word per line. This file is typically edited

#   manually. Hoeever, this one listed here is generated by GIZA

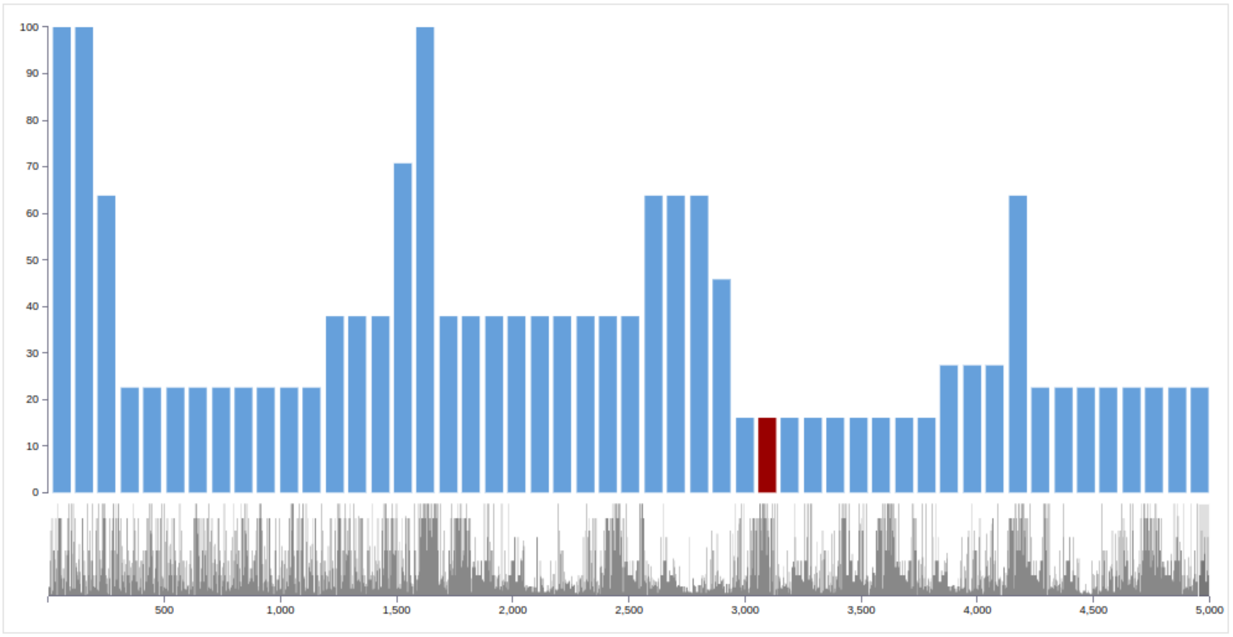
- TTable = 2022-10-10.023859.ubuntu.t3.final
- InverseTTable = 2022-10-10.023859.ubuntu.ti.final
- NTable = 2022-10-10.023859.ubuntu.n3.final
- D3Table = 2022-10-10.023859.ubuntu.d3.final
- D4Table = 2022-10-10.023859.ubuntu.D4.final
- PZero = 2022-10-10.023859.ubuntu.p0_3.final
- Source.vcb = ../parallel-n/IITB.en-hi.en.vcb
- Target.vcb = ../parallel-n/IITB.en-hi.hi.vcb
- Source.classes = ../parallel-n/IITB.en-hi.en.vcb.classes
- Target.classes = ../parallel-n/IITB.en-hi.hi.vcb.classes
- FZeroWords   = 2022-10-10.023859.ubuntu.fe0_3.final

Evaluation

BLEU Score



BLEU:	13.08			
Precision x brevity:	14.97 x 87.37			
Type	1-gram	2-gram	3-gram	4-gram
Individual	40.85	19.25	10.11	6.32
Cumulative	35.69	24.50	17.44	13.08
Export data	CSV			



WER

Word error rate (WER), one of the first automatic evaluation metrics applied to statistical machine translation, is borrowed from speech recognition and takes word order into account. It employs the Levenshtein distance, which is defined as the minimum number of editing steps insertions, deletions, and substitutions needed to match two sequences.

$$\text{WER} = \frac{\text{substitutions} + \text{insertions} + \text{deletions}}{\text{reference-length}}$$

Used python library : https://pypi.org/project/jiwer/

Code:https://gist.githubusercontent.com/ymoslem/f1783b566b3a17b4107a34198daee6a6/raw/298a701d1ea945b3ab5d9fc4044e6c4eecca5af9/sentence-wer.py

METEOR

METEOR incorporates the use of stemming and synonyms by first matching the surface forms of the words, and then backing off to stems and finally semantic classes.METEOR, incorporates a stronger emphasis on recall.

Used Library: NLTK
Code:https://gist.githubusercontent.com/ymoslem/5174469f88d9f1fb1660121a663bb87f/raw/8a98fe6670b1c6833bdb0e503fa50ead09f10ebe/sentence-meteor.py

TER

Translation Error Rate, One disadvantage of the Levenshtein distance is that mismatches in word order require the deletion and re-insertion of the misplaced words. We may remedy this by adding an editing step that allows the movement of word sequences from one part of the output to another. This is something a human post-editor would do with the cut-and-paste function of a word processor.Evaluation metrics based on such editing steps have been proposed, including translation error rate (TER).

Code: https://github.com/jhclark/tercom