CS 224n Programming Assignment 2

Rebecca Weiss

1 Introduction

2 Methods

2.1 Models

2.1.1 Superficial Word Aligner

This model extends the BaselineWordAligner class, which is very simple. In the BaselineWordAligner class, alignment is calculated along the diagonal; if the target sentence is longer than the source sentence, the alignment vector returns -1 for each target word beyond the source sentence. Additionally, the train() method takes pairs of sentences, iterates through each pair, and counts all possible mappings between source and target sentence pairs.

The SuperficialWordAligner class goes a little bit further than that. I based my implementation off of the suggested alignment probability, expressed formally as:

$$\frac{P(f,e)}{P(f)P(e)} \tag{1}$$

I calculated $P_{MLE}(F)$ and $P_{MLE}(E)$, using the getWordProbability() method from the EmpiricalUnigramLanguageModel class (modifying the LanguageModel and EmpiricalUnigramLanguageModel classes so that getWordProbability() accepted a string). For $P(f,e)$, I modified the train() method such that

2.1.2 IBM Model 1

2.1.3 IBM Model 2

2.2 EM Algorithm

2.3 Decoding

2.4 Backing off

3 Results

3.1 Model evaluation metrics

3.2 Model performance

test	BaselineWordAligner	SuperficialWordAligner	Model1WordAligner	Model2WordAligner
Recall	0.225854383358	0.159732540861		
Precision	0.365896980461	0.395578365574		
AER	0.68649249583	0.74935321868		

validate	BaselineWordAligner	SuperficialWordAligner	Model1WordAligner	Model2WordAligner
Recall	0.33489096573	0.02366863905		
Precision	0.19822485207	0.31460674157		
AER	0.71224489795	0.91569086651		

3.3 Decoding performance

3.4 Observations

4 Error analysis