ANLP Assignment 1: Language Modeling

Akash Srivastava (s1362249) and Greta Kaufeld (s1355318)

October 10, 2013

Introduction

This report will cover our implementation of three trigram language models using Python 2.7. In order to ensure comparable results for the English, German and Spanish training data, we lowercased the data and removed all non-alphabetic characters (keeping spaces, commas and periods), thus levelling the vocabulary size of all three languages to V=29. Since we were curious about which smoothing algorithm would perform best on our data, we implemented three different language models, using Laplace-Smoothing, Good-Turing Discounting and Modified Kneser-Ney-Smoothing, respectively. We found that Good-Turing Discounting using Back-Off gave us the lowest perplexity scores for the test data, so all the examples given in this report will be in respect to our Good-Turing language model. The implementations of language models using Laplace- and Modified Kneser-Ney-Smoothing can be found in the appendix.

Excerpt of the language model for English

insert an excerpt of the language model for English, displaying all n-grams and their probability with the two-letter history *t h*.

Random Output

insert 300 characters of random output for each of the three languages:

English

insert 300 characters of random English output

Spanish

insert 300 characters of random Spanish output

German

insert 300 characters of random German output

Perplexity

	Perplexity
English	
Spanish	
German	

Table 1: *Perplexity scores for trigram language models with English, Spanish and German training data*

Discussion

Do you need to test the test set on all three language models or is the score from a single language model sufficient?

The test set will have to be tested on all three language models. Since perplexity is an "intrinsic evaluation metric" (Jurafsky and Martin 2009, p. 129), it will only be meaningful in a relative setting: A "low" perplexity score for any given language model can only be interpreted if it is seen in comparison to the perplexity scores of other language models (or an altered version of the original one).

Would a unigram or bigram model work as well?

Unigram and bigram models would not perform as well as a trigram (or higher-order) language model. An ideal language model would take as much history as possible into consideration, but since that is not practical, a Markov assumption is being made and only the most recent history is taken into account. The language model that takes into consideration the largest history should be the one that performs best on the test data.

Table 2 shows the perplexity scores for English unigram, bigram and trigram models: As expected, the trigram language model results in the lowest perplexity score. The bigram model shows a slightly higher perplexity score, while the unigram model proves to not be a very good fit to the data.

Perplexity		
Unigram	Bigram	Trigram

Table 2: *Perplexity scores for English unigram, bigram and trigram language models*

Do the language models show anything about similarity of languages?

Since our language model deals with *letter* n-grams, the only comparison that we can make between the three different languages will be concerning the way that letters can be combined. Based on our perplexity scores, we can say that German and Spanish are more similar to each other than to English - but only in the way they combine letters. In order to properly compare the three languages, disciplines such as phonolgy, morphology, or syntax would need to be taken into consideration.

Bibliography

Jurafsky, Daniel and James H. Martin: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Second Edition. Prentice-Hall.