ANLP Assignment 1

Chunchuan Lyn s1544871
Yue Yu s1563228

October 19, 2015

1 Perplexity of the Test Case

$$\begin{aligned} PP_M(\vec{w}) &= 2^{H_M(\vec{w})} \\ &= 2^{-\frac{1}{n}\log_2 P_M(\vec{w})} \\ &= 2^{\log_2 P_M(\vec{w})^{-\frac{1}{n}}} \\ &= P_M(\vec{w})^{-\frac{1}{n}} \\ &\approx \prod_{i=1}^{n} P(w_n|w_{n-1}, w_{n-2})^{-\frac{1}{n}} \\ &= (0.2 * 0.7 * 0.6 * 0.25 * 0.5 * 0.1)^{-\frac{1}{6}} \\ &\approx 3.1367 \end{aligned}$$

p.s.: w_{-1} and w_0 refer to the first two '[' characters of each sentence.

2 Line Preprocessing

```
#function turns input into required format
def preprocess_line(line):
    #remove non-necessary characters,
    #and turn string to lowercase
    p = re.compile('[^\w\s,.]')
    line = re.sub(p,'',line.lower())
    #replace \n by ]
    line = re.sub('\n',']',line)
    #turn numbers into 0
```

```
    line = re.sub('[0-9]','0',line)
    #add begining and end [[
    return '[['+line
```

By preprocessing input in this fashion, we essentially assumed that there are no interconnection between lines. All the lines are preprocessed into line units. During language model building, we will not compute $P(]|[[)$ nor $P([|*[)$. This probability will be equal to one, if we treat the whole text as one unit. We consider this as an artifact of ngram model instead of the true underlying language model.
As a consequence, we will sample line by line independently in task 4. Also, we will exclude those probability in computing perplexity.

3 Language Model

3.1 Estimation of Probabilities

In principle, we first assumed trigram approximation of the underlying probability of language.

$$\begin{aligned} P(\vec{w}) &= P(w_1 \dots w_n) \\ &= P(w_n|w_{n-1}, w_{n-2}, \dots w_1)P(w_{n-1}|, w_{n-2}, \dots w_1) \dots P(w_1) \\ &\approx \prod_{i=1}^{n} P(w_n|w_{n-1}, w_{n-2}) \end{aligned}$$

Then, we used maximum likelihood method to estimate the conditional probability required by trigram model. That is we counted the number of occurrences of trigram, and divide it by number of occurrences of the first twogram as condition.

$$P(w_n|w_{n-1}, w_{n-2}) = \frac{C(w_{n-2}, w_{n-1}, w_n)}{C(w_{n-2}, w_{n-1})}$$

In addition, to smooth our model, we used 0.1 smooth.

$$P(w_n|w_{n-1}, w_{n-2}) = \frac{C(w_{n-2}, w_{n-1}, w_n) + smooth = 0.1}{C(w_{n-2}, w_{n-1}) + (smooth * ntypes = 0.1 * 31)}$$

To start the language model, we inserted '[[' at the beginning of the lines and ']' at the end of lines. However, we did not compute $P(]|[[)$ nor $P([|*[)$, and lines are independent as mentioned in 2.

3.2 Data Structure

We used dictionary of dictionary to store all the conditional probability. Basically, we have conditionProbs as a dictionary of dictionary of float. This could

be used to retrieve conditional probabilities given condition. The conditional probabilities retrieved are stored within a dictionary of float. To store data in files, we simply used json.

3.3 Conditional Probabilities Discussion

3.3.1 Conditional Probabilities for th

$P(|th) = 0.0540999451425$
$P(,|th) = 0.00159347979415$
$P(.|th) = 0.00185470598992$
$P(0|th) = 2.61226195763e-05$
$P(]|th) = 2.61226195763e-05$
$P(a|th) = 0.125675922782$
$P(b|th) = 2.61226195763e-05$
$P(c|th) = 0.000287348815339$
$P(d|th) = 0.00107102740263$
$P(e|th) = 0.659883493117$
$P(f|th) = 2.61226195763e-05$
$P(g|th) = 2.61226195763e-05$
$P(h|th) = 2.61226195763e-05$
$P(i|th) = 0.121496303649$
$P(j|th) = 2.61226195763e-05$
$P(k|th) = 2.61226195763e-05$
$P(l|th) = 0.000548575011102$
$P(m|th) = 2.61226195763e-05$
$P(n|th) = 2.61226195763e-05$
$P(o|th) = 0.018311956323$
$P(p|th) = 2.61226195763e-05$
$P(q|th) = 2.61226195763e-05$
$P(r|th) = 0.00577309892636$
$P(s|th) = 0.00263838457721$
$P(t|th) = 2.61226195763e-05$
$P(u|th) = 0.00394451555602$
$P(v|th) = 2.61226195763e-05$
$P(w|th) = 0.000548575011102$
$P(x|th) = 2.61226195763e-05$
$P(y|th) = 0.00185470598992$
$P(z|th) = 2.61226195763e-05$

As one can see $P(e|th) = 0.659883493117$, which corresponds to 'the' has a high frequency in English. Small none zero probabilities are the effect of smoothing.

3.3.2 Conditional Probabilities for an

Conditional probability for an $P(|an) = 0.173350506411$
$P(,|an) = 0.00181488203267$

$P(.|an) = 5.8544581699e-05$
$P(0|an) = 5.8544581699e-05$
$P(]|an) = 5.8544581699e-05$
$P(a|an) = 0.0164510274574$
$P(b|an) = 5.8544581699e-05$
$P(c|an) = 0.0597740179146$
$P(d|an) = 0.485978572683$
$P(e|an) = 0.00357121948364$
$P(f|an) = 5.8544581699e-05$
$P(g|an) = 0.0281599437972$
$P(h|an) = 5.8544581699e-05$
$P(i|an) = 0.0146946900064$
$P(j|an) = 5.8544581699e-05$
$P(k|an) = 0.0211345939933$
$P(l|an) = 5.8544581699e-05$
$P(m|an) = 5.8544581699e-05$
$P(n|an) = 0.0275744979802$
$P(o|an) = 0.00649844856858$
$P(p|an) = 5.8544581699e-05$
$P(q|an) = 5.8544581699e-05$
$P(r|an) = 5.8544581699e-05$
$P(s|an) = 0.0679702593525$
$P(t|an) = 0.0568467888297$
$P(u|an) = 0.00415666530063$
$P(v|an) = 5.8544581699e-05$
$P(w|an) = 0.000643990398689$
$P(x|an) = 0.000643990398689$
$P(y|an) = 0.0299162812482$
$P(z|an) = 5.8544581699e-05$

We were expecting $P(d|an)$ to be high, since 'and' should be a fairly common words. Meanwhile, we also find $P(|an)$ to be quite high. Apparently, this finding could be attributed to 'an' as a word.

4 Random Output

We generated those sentences line by line. We count the end line line as one character, which is simply newline in the text. We do not count the beginning of sentence '[[' as character.

4.1 English

this tommis, polight. thissidealre to be the st of the of th of adve ing youren exes ancempte fermorissive ouse hatergento the nes iniume itioncent hat suctill of yount elikinithaves, a regall con pable this fork mis our of con a sculdinesse

to ber so the withe re cand ress. thes ishich fund the

4.2 German

zu son, dium vorden unge den. esinen, dansjhr kom binsen gefortscherstignang vern so verunnes wir genn geng, ancesen nissicht, te aufter frerrnahneurder klandeut. daspesen her bitteichterich abeste regie wichen vommit die in gieden und, der pro wirkur ungsannenhnemalem den dierr ischerdnurogung

4.3 Spanish

ra car los imite rabarionspunas, ime pro. pida cue estacintear pejo de troya a al ms se lo re cohes la cuerogra poreadqedo de dentracinincerion que recondistreas regasos flusionadad ectura conestamo quientarrentortacacias de dr entos imablesustropeciascumin de cel pal te. aanten su en nue lta y

5 Perplexity & Language Identification

Model / Language	Unigram	Bigram	Trigram
English	19.4157	11.5054	9.2210
German	21.3752	21.4868	30.1338
Spanish	22.4430	24.7187	29.5903

Table 1: Perplexity of Testing Dataset

The results given in Table 1 suggest that the testing dataset shows the lowest perplexity under the trigram model of English. Therefore, the testing sample is most likely to be identified as English texts. In addition, as the training model become simpler (from trigram to unigram) the margins among the results under different language models become smaller. Consequently it will be harder to tell which language the testing sample belongs to although, in this case, the performance of the testing dataset under English language model is still better than the results under other language models.

6 Extension

We dis two extensions: first, our code can compute from uni gram to ngram by a change of parameter n; second based on this, we have implemented deleted interpolation algorithm.

References

Smith, J. M. and Jones, A. B. (2012). *Chemistry*. Publisher, 7th edition.