Natural Language Processing - INF210
Exercise 4 of part 1

Gketsopoulos Petros | $F3321804$
Gikas Athanasios | $F3321808$
Euagelou Iordanis | $F3321801$
Kougia Vasiliki | $F3321805$

March 3, 2019

1

For the implementation of the models, we used a subset of Europarl's English version corpus [1]. We split the dataset into sentences, which were preprocessed by removing punctuation, converting all strings to lowercase and performing word tokenization using the NLTK library [2]. From the set of sentences, we used a 70% as the training set, 20% as the validation set and the remaining 10% as the test set. The words that appear less than 10 times in the training subset were removed from all three subsets and replaced by the *UNK* token. That is, from a total of 32.104.821 tokens, 205.290 were replaced with the *UNK* token.

All sentences in the training subset were augmented with the appropriate **start**, **end** tokens according to the n-gram model. We finally merged them into a single sequence of tokens in which n-gram tuples were constructed and the appropriate counts were computed and exported to a separate compressed file.

2

The joint probabilities for a given language sequence w_1^n of a bi-gram and a tri-gram are given by :

$$P(w_1^n) \approx P(w_1|\text{*start1*})P(w_2|w_1)...P(\text{*end*}|w_n)$$
$$P(w_1^n) \approx P(w_1|\text{*start2*}, \text{*start1*})P(w_2|w_1, \text{*start2*})...P(\text{*end*}|w_n, w_{n-1})$$

In which the individual probabilities in the product, account for the frequency of each n-gram in the training subset and they are equal to :

$$P(w_2|w_1) = \frac{C(w_1, w_2)}{C(w_1)}$$
$$P(w_2|w_1, \text{*start2*}) = \frac{C(\text{*start2*}, w_1, w_2)}{C(\text{*start2*}, w_1)}$$

Given the aforementioned equations, we split the sentence in the query into n-gram tuples according to the selected model and compute the log probability by applying *Laplace* smoothing. That is :

$$\log(P(w_1^n)) \approx \log(P(w_1|\text{*start1*})) + \log(P(w_2|w_1)) + ... + \log(P(\text{*end*}|w_n))$$
$$\log(P(w_1^n)) \approx \log(P(w_1|\text{*start2*}, \text{*start1*})) + \log(P(w_2|w_1, \text{*start2*})) + ... + \log(P(\text{*end*}|w_n, w_{n-1}))$$

And by applying smoothing the individual probabilities are rephrased as :

$$P(w_2|w_1) = \frac{C(w_1, w_2) + 1}{C(w_1) + |V|}$$
$$P(w_2|w_1, \text{*start2*}) = \frac{C(\text{*start2*}, w_1, w_2) + 1}{C(\text{*start2*}, w_1) + |V|}$$

Where we set $|V|$ parameter equal to the number of distinct tokens appearing in the model's vocabulary including the special tokens **start**, **end**.

In the following tables, we demonstrate two randomly selected sentences from the unseen test subset, that we earlier constructed. The table contents, are wrapped with the log probability for each n-gram depending on the model, followed by a random permutation of the same sentence in the table's counterpart.

1. **Sentence** : ['i', 'think', 'that', 'the', 'honourable', 'member', 'raises', 'an', 'important', 'point']
   Random permutation : ['member', 'the', 'think', 'that', 'raises', 'i', 'important', 'an', 'honourable', 'point']

bi-gram	**log-prob**		**bi-gram**	**log-prob**
<*start1*, i>	3.19		<*start1*, member>	8.42
<i, think>	4.37		<member, the>	11.42
<think, that>	2.48		<the, think>	17.71
<that, the>	2.63		<think, that>	2.48
<the, honourable>	10.44		<that, raises>	14.15
<honourable, member>	4.78		<raises, i>	13.52
<member, raises>	13.96		<i, important>	18.03
<raises, an>	10.93		<important, an>	12.38
<an, important>	4.24		<an, honourable>	11.11
<important, point>	7.12		<honourable, point>	14.62
<point, *end*>	5.05		<point, *end*>	5.05
final prob	69.23		**final prob**	152.84
tri-gram	**log-prob**		**tri-gram**	**log-prob**
<*start1*, *start2*, i>	3.19		<*start1*, *start2*, member>	8.42
<*start2*, i, think>	4.34		<*start2*, member, the>	14.64
<i, think, that>	2.80		<member, the, think>	14.48
<think, that, the>	4.26		<the, think, that>	13.48
<that, the, honourable >	10.60		<think, that, raises>	14.89
<the, honourable, member>	4.80		<that, raises, i>	14.48
<honourable, member, raises>	12.21		<raises, i, important>	14.48
<member, raises, an>	13.48		<i, important, an>	14.48
<raises, an, important>	12.89		<important, an, honourable>	14.48
<an, important, point>	7.94		<an, honourable, point>	14.48
<important, point, *end* >	8.38		<honourable, point, *end* >	14.48
final prob	84.95		**final prob**	152.84

2. **Sentence** : ['it', 'possesses', 'political', 'economic', 'and', 'diplomatic', 'leverage']
   Random permutation : ['economic', 'it', 'leverage', 'and', 'diplomatic', 'possesses','political']

bi-gram	**log-prob**		**bi-gram**	**log-prob**
<*start1*, it>	4.12		<*start1*, economic>	11.96
<it, possesses>	14.78		<economic, it>	14.57
<possesses, political>	13.48		<it, leverage>	17.78
<political, economic>	7.74		<leverage, and>	11.03
<economic, and>	3.15		<and, diplomatic>	13.14
<and, diplomatic>	13.14		<diplomatic, possesses >	14.54
<diplomatic, leverage>	14.54		<possesses, political>	13.48
<leverage, *end*>	10.90		<political, *end*>	9.46
final prob	81.90		**final prob**	106.00
tri-gram	**log-prob**		**tri-gram**	**log-prob**
<*start1*, *start2*, it>	4.12		< *start1*, *start2*, economic >	11.96
< *start2*, it, possesses>	15.22		< *start2*, economic, it >	14.49
< it, possesses, political>	14.48		< economic, it, leverage >	14.48
<possesses, political, economic>	13.48		< it, leverage, and >	14.48
<political, economic, and>	7.09		< leverage, and, diplomatic >	14.48
<economic, and, diplomatic>	11.47		< and, diplomatic, possesses >	14.48
< and, diplomatic, leverage >	14.48		< diplomatic, possesses, political>	14.48
< diplomatic, leverage, *end* >	14.48		< possesses, political, *end* >	14.48
final prob	94.86		**final prob**	113.37

We observe that words, which are commonly used together have the highest probabilities, e.g. bi-gram: <think, that> and tri-gram: <i, think, that> in the first sentence. Overall, comparing the probabilities between the correct and the random sentences, we see that the random bi-grams and tri-grams have smaller probabilities since they are more unlikely to occur. We also observe that the bi-grams have overall larger probabilities than the tri-grams. This can be caused by small corpus' sizes combined with the fact that *Laplace* smoothing gives rather simple probability estimates that can favor bi-gram probabilities, while tri-gram probabilities are harder to estimate.

3

We compute the cross-entropy and perplexity of our models by applying the definitions given from Jurafsky and Martin [3], that is :

$$H(W) = -\frac{1}{N}\log P(w_1, w_2, ..., w_n)$$
$$\textit{perplexity(W)} = 2^{H(W)}$$

Where W refers to the sequence of words, $P(w_1, w_2, ..., w_n)$ is the joint probability that the model will generate given the sequence and N the number of words that are present in the sequence.

Given the aforementioned definitions, we augment the sentences of our test subset according to the model in use and merge them into a single sequence of tokens. For the computation of these values, we exclude every probability of the form $P(*start* | ...)$ and we apply *Laplace* smoothing. Our results are demonstrated in the following table.

	cross entropy	perplexity
bi-gram	8.2	294.5
tri-gram	10.52	1469.07

4

We linearly interpolate between the bi-gram and the tri-gram model using the validation subset and the following formula to compute each one of the probabilities :

$$P(w_n|w_{n-1}, w_{n-2}) = \lambda P(w_n|w_{n-1}, w_{n-2}) + (1 - \lambda)P(w_n|w_{n-1}), 0 \leq \lambda \leq 1$$

We mention that the hyper-parameter λ remains the same for every individual probability of the chosen model. Thus, we progressively estimate the joint probability by first assigning the full weight to the bi-gram model and we interpolate towards the tri-gram with a step of 0.2. We measure the performance of each model by calculating the cross-entropy and we finally decide the assignment of λ by picking the minimum. By employing the previous procedure, the validation subset results to :

$$\begin{aligned} H(W) &= 8.24 \\ perplexity(W) &= 303.81 \\ \lambda &= 0.0 \end{aligned}$$

Which indicates that the bi-gram model has a dominant information value in our corpus. As a result, every query in the test subset will be handled with the bi-gram distribution and the tri-gram will be completely discarded. We believe that the model does not perform better since the corpus is not well defined for a tri-gram model and the *Laplacian* smoother increases this unstable behavior by overestimating the rare cases of an ngram. Since we observed that the bi-gram model always performs better than the tri-gram, it is only natural that a linear interpolation between the two is meaningless. On the other hand, a mixed behaviour would have made the interpolation a more realistic solution.

References

[1] "Corpus source." `http://www.statmt.org/europarl/`.

[2] "Nltk page." `https://www.nltk.org/#`.

[3] D. Jurafsky and J. H. Martin, *Speech and Language Processing (2Nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2009.