SPEECH TECHNOLOGIES: EVALUATION OF SRILM

Lucas Rodés

lucas.rodes@alu-etsetb.upc.edu
BarcelonaTech - Universitat Politcnica de Catalunya (UPC)

ABSTRACT

SRILM toolkit is used to build language models from a given train corpus. These resulting models might be applied to test corpus in order to evaluate the performance of the generated model. It has many applications in different fields, such as machine translation, speech recognition, statistical tagging and segmentation etc. With this purpose I have evaluated some functionalities of this toolkit.

1. INTRODUCTION

This document presents the results of different language models (LMs) applied on a given set of test corpus. To evaluate the performance of the models, I will be using SRILM [1] on OS X.

This document is organized as follows: Section 2 enunciates the basics of Language Modeling. However, the interested user might address to [2, 3] for further details. Section 3 describes the workspace that has been used throughout this exercise. Section 4 presents the obtained results. Finally, Section 5 ends this document with the conclusions.

For the plots, Rstudio has been used. The respective markdown document is enclosed in the deliver.

2. LANGUAGE MODELING OVERVIEW

In speech recognition, we can identify three main steps:

- Acoustic Analysis: The speech recordings are processed and transformed into a set of feature vectors $\mathbf{x}_1^m = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ which might fully characterize a sentence (e.g. using MFC coefficients).
- Acoustic Model: It aims to find the likelihood function of a sequence of N words $\mathbf{w}_1^n = (w_1, \dots, w_n\}$ given any the set of feature vectors, i.e. $\Pr(\mathbf{x}_1^m|\mathbf{w}_1^L)$.
- Language Model: The prior probability distribution of word sequences is found, i.e. $\Pr(\mathbf{w}_1^n)$.

Once all previous parameters have been obtained, the speech recognition problem can be formulated as:

$$\mathbf{w}_{recog} = \text{argmax}_{\mathbf{w}}\big\{\Pr(\mathbf{w}|\mathbf{x}_1^m)\big\} = \text{argmax}_{\mathbf{w}}\big\{\Pr(\mathbf{w})\Pr(\mathbf{x}_1^m|\mathbf{w})\big\}. \tag{1}$$

where $\mathbf{w}$ and $\mathbf{w}_{recog}$ are column vectors of size n. Particularly, $\mathbf{w}_{recog}$ denotes the recognized word sequence according to the acoustic and language models.

2.1. N-gram Models

Using the chain rule, we obtain that $\Pr(w_1^n) = \Pr(w_1) \cdot \Pr(w_2|w_1) \cdots \Pr(w_n|w_1, w_{n-1})$. However this expression can't be modeled due to obvious memory and computational reasons. Hence, N-grams [2, p. 189] set a bound on the amount of previous context. In general, an N-gram model assumes that only the $N-1$ words have an impact:

$$\Pr(\mathbf{w}_1^n) = \prod_{k=1}^{n} \Pr(w_k|w_{k-(N-1)}, \dots, w_{n-1}). \tag{2}$$

Typical used N-grams are the Unigram, Bigram and Trigram, where $N = 1$, 2 and 3 respectively. The simplest N-gram is the Zerogram, which basically consists on setting equal probabilities to all the words according to the alphabet size W, i.e. $\Pr(w) = 1/W$ for all words w.

2.2. Estimating N-gram Probabilities

The probabilities of N-gram models can be obtained by counting and normalizing. The probability of a word w given its history h is given by

$$\Pr(w|h) = \frac{n(h,w)}{n(h)}. \tag{3}$$

Equation (3) basically estimates the probability by dividing the observed frequency of a particular sequence $n(h,w)$ by the observed frequency of the history $n(h)$. This is widely used and is known as the technique of Maximum Likelihood Estimation (MLE).

2.2.1. Generalization and unseen events

The model seen so far only works well for word prediction if the test corpus just looks like the training corpus, which is not the case in real practices. Thus, robust models that can generalize are required. Sequences appearing in the test but not in the training are named *unseen events*. If the model is not generalised, the probability unseen events is equal to zero, since the term $n(h, w)$ in (3) is zero.

Using smoothing methods we can increase the robustness of the LM by adjusting the estimation of the sequences probabilities, in particular for unseen events. This is achieved by flattening the distribution of events (sequences), i.e. increasing the probability of low-probable events and decreasing the probability of high-probable events.

Some of the smoothing methods are *Add one Smoothing* [2, p. 205], *Linear Discounting* [3, p. 216], *Absolute Discounting* [3, p. 216] and *Turing-Good Smoothing* [2, p. 212].

2.2.2. Backoff

Typically a Full Trigram model with Backing-Off is used, which is based on a hierarchical model. That is, if there are no examples of a particular trigram $\{w_{i-2},\, w_{i-1},\, w_i\}$, its probability is estimated using the bigram $\{w_{i-1},\, w_i\}$. Similarly, if this bigram is not found, the unigram $\{w_i\}$ is used. Then, the probability of a given word is computed as:

- $\Pr(w_i|w_{i-2}, w_{i-1}) = \frac{n(\{w_{i-2}, w_{i-1}\}, w_i)}{n(\{w_{i-2}, w_{i-1}\})}$, if the trigram $\{w_{i-2},\, w_{i-1},\, w_i\}$ is seen.
- otherwise:
  - $\Pr(w_i|w_{i-2}, w_{i-1}) = n(w_{i-1}, w_i)/n(\{w_{i-1}\})$, if the bigram $\{w_{i-1},\, w_i\}$ is seen.
  - otherwise:
    - $\Pr(w_i|w_{i-2}, w_{i-1} = n(w_i)/n$, if the unigram w_i is seen.
    - otherwise:
      - $\Pr(w_i) = 1/W$ (zerogram).

Note that weighting parameters have to be introduced so that $\sum_{w_i} \Pr(w_i|w_{i-2}, w_{i-1}) = 1$. Typically used techniques are Absolute Discounting or Turing-Good.

2.3. Evaluation and perplexity

We train the parameters of our model on a *training corpus*. Once the LM is generated, we can apply it to a *test corpus* to evaluate its the performance data that has not been used in the language modeling. An evaluation metric tells us how well our model does on the test set. Particularly, this metric is known as the *Perplexity*.

2.3.1. Perplexity

Denoted as PP_q, it is defined as the n-geometric mean of the inverse probability q of a test corpus according to a language model, where n is - again - the number of words of the test corpus. Given a test sentence $\mathbf{w}_1^n$, its perplexity is computed as

$$PP_q(\mathbf{w}_1^n) = q(w_1, \ldots, w_n)^{-1/n} = \sqrt[n]{\prod_{k=1}^{n} \frac{1}{\mathrm{q}(w_k|w_{k-(N-1)}, \ldots, w_{n-1})}}. \quad (4)$$

Minimizing the perplexity value is equivalent as maximizing the probability of the word sequence. Hence, the objective is to obtain low perplexity values for all the test sets. Perplexity gives a notion of the average possible words at any position of the test text, always according to the model q.

2.3.2. Logprob

Denoted as LP, it gives a notion of the average digit uncertainty of the test set according to the model q.

$$LP_q(\mathbf{w}_1^n) = \log PP_q(\mathbf{w}_1^n) = -\frac{1}{n} \log\{q(w_1, \ldots, w_n)\} \quad (5)$$

The lower is the logprob, the lower is the digit uncertainty and hence better results will be obtained using the language model q.

3. EXPERIMENTAL WORKSPACE

For this exercise it is required:

- Test Corpus: The English part of the test set of the WMT2015 SMT evaluation will be used [4]. There are 5 language sets in total: French (FR), Czech (CS), German (DE), Finnish (FI) and Russian (RU).
- Training Corpus: Three training corpus from the English monolingual corpus in [5] have been selected:
  - `News Commentary`: Commentaries extracted from news reports (55 MB).
  - `Europarl v7/v8`: Extracted from the proceedings of the European Parliament [6] (324 MB).
  - `News Crawl, articles from 2007`: Extracted article text from various online news publications (until 2007) (474 MB).
  - `News Crawl, articles from 2010`: Extracted article text from various online news publications (until 2010) (827 MB)).

- Toolkit: As aforementioned, the SRILM toolkit [1] will be used.

Note that the downloaded set of test corpus are in `SGML` format, which contains several nomenclature that is not interesting for our purposes. Thus, a prior step need to be done in order to obtain the plain text versions of all the test corpus. In this regard, I did some research and finally decided to post a question in *Stackoverflow.com* [7] and used the perl script *strip_ sgml.pl* posted by user *bart*.

Once all the required files are downloaded and SRILM has been downloaded from [8] and correctly installed, I created the directory `Exercise` with the following folders:

`Train`: Training corpus used in this exercise.

`LM`: Generated Language Models out of each of the training corpus.

`Test`: Test corpus used in this exercise.

`Results`: Text files containing the results of the performance of each of the training corpus for all the test corpus.

`Srilm`: Srilm Toolkit (libraries etc.).

`LaTeX`: The LaTeX report project (i.e. this document).

`References`: Papers and references used for this exercise.

To generate a language model `sample.lm` from a training file `sample.train` we have to execute:

```
$ ngram-count -text sample.train -lm sample.lm
```

This will generate a file `sample.lm` containing the probabilities $q(\cdot)$ of all the uni-grams, bi-grams an tri-grams of the training corpus `sample.train`. Note that this is a standard LM (i.e. trigram with Good-Turing discounting and Katz backoff for smoothing). Afterwards, a test corpus can be evaluated with the generated language modeling by calling:

```
$ ngram -lm sample.lm -ppl sample.test
```

This command will outcome six main parameters of the test file `sample.test`:

- sentences: Number of sentences of the `sample.test`.
- words: Number of words of the `sample.test`.
- OOVs: Out-of-Vocabulary words of the test corpus, i.e. words that appear in the `sample.test` but do not appear in the `sample.train`.
- logprob: Log probability of `sample.test` according to `sample.train`.
- ppl: Perplexity of `sample.test` according to `sample.train`.
- ppl1: Perplexity (ignoring white spaces, i.e. <s/>) of `sample.test` according to `sample.train`.

The option `-debug` can be added at the end of this command in order to set the debugging output level [9].

3.1. Discrepancies: Theory and SRILM expressions

At this point, it is necessary to remark how SRILM defines the Perplexity and the Logprob [10], since it is slightly different from (4) and (5). Let the super-index ^{s} denote expressions used by SRILM.

On the one hand, the logprob is defined as

$$LP_q^s(\mathbf{w}_1^n) = \log\{q(w_1, w_2, \ldots, w_n)\}, \tag{6}$$

which is a $-n$ scaled version of the logprob defined in (5). Note that in this case, n is defined as

$$n = n_{tot} + n_s - n_{oov}, \tag{7}$$

where n_{tot} denotes the total amount of words, n_s denotes the number of sentences (i.e. </s>) and n_{oov} stands for the number of OOV words.

On the other hand, the perplexity is defined in base 10 as

$$PP_q^s(\mathbf{w}_1^n) = 10^{-\frac{1}{n}LP_q^s(\mathbf{w}_1^n)} = q(w_1, \ldots, w_n)^{-1/n} \tag{8}$$

which is exactly the same expression as in (4).

4. RESULTS

Finally, in this section the results for the different training and test corpus will be displayed. Remember that the aim of a good and robust language model is to minimize the perplexity for a wide range of test corpus.

In Fig. 1 and Fig. 2 we can observe the number of words of the five test corpus. As noted in [4], the text for all the test sets is drawn from news articles except for the French-English set, where it has been drawn from user-generated comments on the news articles. This might be the explanation of why the sample associated with the French-English set is not aligned with the rest. Particularly, it seems that it contains shorter sentences than the rest of test sets.

Furthermore, Fig. 3 and Fig. 4 illustrate the amount of words and sentences of the three training corpus. Note that the training corpus `News Crawl 2010` is the more extensive corpus and `News Commentary` the less one.

Once we are aware of the data that we have, we can evaluate the three generated LMs onto the five test corpus. Fig. 5 shows the relation between the Perplexity and the OOV words or all test-corpus combinations. Further details are given in

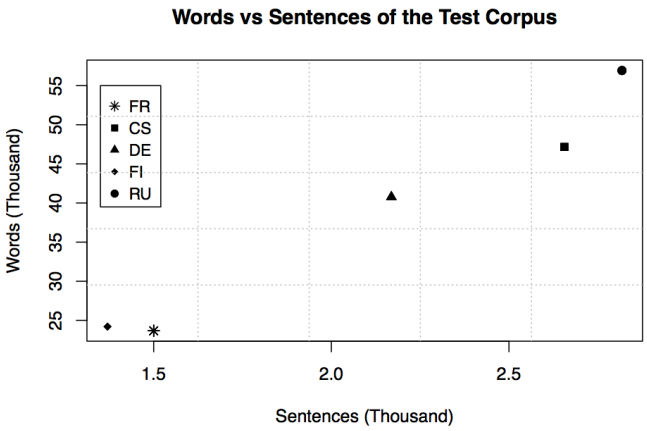

Fig. 1. (words, sentences) for the three test corpus

Test file	Words	Sentences
FR	23,668	1,500
CS	47,160	2,656
DE	40,771	2,169
FI	24,204	1,370
RU	56,934	2,818

Fig. 2. Table containing words and sentences of all test corpus.

Fig. 6-9 (find in bold the sample associated with the test corpus `newstest2015-ende-src.en` or in short, DE).

First thing to notice is the fact that the LM generated by the training of `News Commentary` seems to generally produce better results than the one generated by `Europarl v7/v8`, although the training text `Europarl v7/v8` is far more extensive (i.e. contains more vocabulary). This might be due to the nature of the test corpus, which might be more similar to the training corpus extracted from the news than from the one extracted from the parliament. This makes total sense, since, as aforementioned, the test corpus are extracted from news articles, except french set. Due to this fact, we realize that the sample associated with the English-French set (FR) can be considered as an outlier. Notice that the FR samples, unlike the others, lie nearby for all four language models. This makes clear once again, that the generation of this test corpus was quite different than from the others. In fact, note that for this specific test corpus the LM generated by `Europarl v7/v8` produces better results than using the LM generated by `News Commentary`.

Besides this, we can conclude that the test corpus `News Craw 2010` is the one producing lower perplexity values.

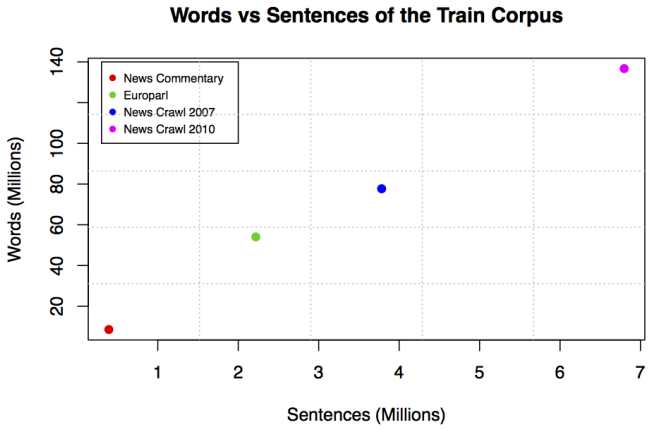

Fig. 3. (words, sentences) for the three train corpus

Test file	Words	Sentences
News Commentary	8,542,120	391,233
Europarl	54,044,600	2,218,200
News Crawl 2007	77,702,200	3,782,550
News Crawl 2010	136,710,000	6,797,220

Fig. 4. Table containing words and sentences of all training corpus.

4.1. Interpolation

A way to further improve the previous results can be obtained by carefully combining all four language models. This has been studied in the literature [11, 12, 13, 14]. In particular, SRILM has an implementation called *compute-best-mix* [15], which computes the optimal interpolation weights $[\lambda_1, \lambda_2, \lambda_3, \lambda_4]$ for the corresponding models. To verify the performance improvement of interpolation, we will only consider the file `newstest2015-ende-src.en`. Thus, since we have 4 models the command was:

```
$ compute-best-mix DE.results1 DE.results2
DE.results3 DE.results4
```

, where `DE.results1` is the output of the *ngram -debug 2 -ppl* run on the aforementioned file when using the first LM, i.e. `News Commentary`. This computation is iterative, stopping when the interpolation weights change by less than a given value (default 0.001).

After 25 iterations the execution stops and the obtained weights are

$$[\lambda_1, \lambda_2, \lambda_3, \lambda_4] = [0.0819, 0.0541647, 0.2533, 0.6006].$$

Note that although we found out in Fig. 5 that the models `News Commentary` and `Europarl v7/v8` produce

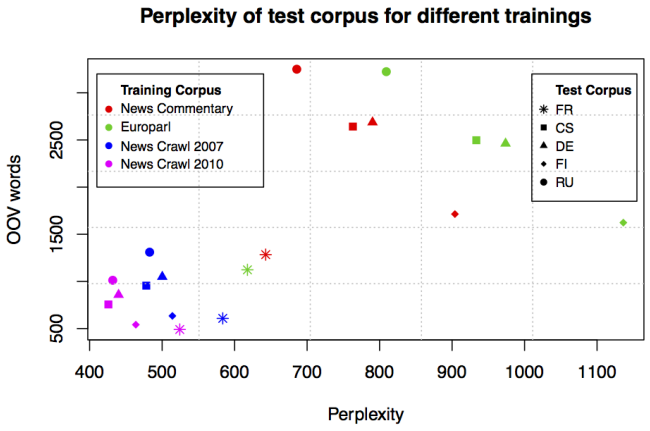

Fig. 5. Plot of the features OOV and Perplexity for all the considered test corpus and the three training corpus

Test file	OOVs	logprob	Perplexity	Perplexity1
FR	1,284	-67,069	642.86	991.50
CS	2,642	-135,984	763.15	1,133.94
DE	**2,687**	**-116,644**	**790.26**	**1,155.60**
FI	1,714	-70,533.5	903.94	1,368.41
RU	3,250	-160,248	685.72	966.10

Fig. 6. Results obtained for test `News Commentary`

Test file	OOVs	logprob	Perplexity	Perplexity1
FR	1,123	-67,103.90	617.68	947.20
CS	2,497	-140,542	933.45	1,439.32
DE	**2,462**	**-120,967**	**973.79**	**1,437.70**
FI	1,623	-73,181.70	1,136.25	1,741.21
RU	3,224	-164,388	809.25	1,149.90

Fig. 7. Results obtained for test `Europarl v7/v8`

worse results, they contain valuable information that can be exploited in order to obtain an interpolated LM which has a better performance in terms of perplexity. With this set of weights, the perplexity obtained 412.75. A comparison between all the models is shown in Fig. 10. We can observe that by using interpolation we can achieve a 6.17% perplexity reduction.

5. CONCLUSIONS

With this exercise I have brought into practice several concepts learned through the classes of Speech Technologies. Furthermore, I have gained experience using a LM tool such as SRILM which is of great interest and might help me in the development of my final project for this subject.

Test file	OOVs	logprob	Perplexity	Perplexity1
FR	608	-67,936	583.61	883.19
CS	955	-130,926	478.15	681.71
DE	**1049**	**-113,069**	**500.18**	**702.28**
FI	634	-67,988.3	514.16	766.53
RU	1,310	-156,843	482.76	660.23

Fig. 8. Results obtained for test `News Crawl 2007`

Test file	OOVs	logprob	Perplexity	Perplexity1
FR	490	-67,113.80	524.30	786.29
CS	756	-128,992	425.86	602.22
DE	**858**	**-111,239**	**439.93**	**612.40**
FI	541	-66,743	463.69	661.59
RU	1,012	-154,799	431.85	586.31

Fig. 9. Results obtained for test `News Crawl 2010`

6. REFERENCES

[1] Andreas Stolcke et al. Srilm-an extensible language modeling toolkit. In *INTERSPEECH*, volume 2002, page 2002, 2002.

[2] Dan Jurafsky. *Speech & language processing*. Pearson Education India, 2000.

[3] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[4] Wmt2015 smt evaluation (test corpus). `http://www.statmt.org/wmt15`. Accessed: 2016-23-04.

[5] Wmt2016 translation task. `http://www.statmt.org/wmt16/translation-task.html`. Accessed: 2016-23-04.

[6] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86, 2005.

[7] Stackoverflow: Convert .sgm to .txt. `http://stackoverflow.com/questions/36827515/convert-sgm-to-txt`. Accessed: 2016-28-04.

[8] SRI Interntional: Downloading and building srilm. `http://www.speech.sri.com/projects/srilm/download.html`. Accessed: 2016-23-04.

[9] Srilm manual for ngram. `http://www.speech.sri.com/projects/srilm/manpages/ngram.1.html`. Accessed: 2016-28-04.

LM	Perplexity
News Commentary	790.26
Europarl v7/v8	973.79
News Crawl 2007	500.18
News Crawl 2010	439.93
Interpolation	412.75

Fig. 10. Perplexity results of `newstest2015-ende-src.en` for all LM

[10] Srilm-faq. `http://www.speech.sri.com/projects/srilm/manpages/srilm-faq.7.html`. Accessed: 2016-23-04.

[11] Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

[12] Cyril Allauzen and Michael Riley. Bayesian language model interpolation for mobile speech input. In *Interspeech*, pages 1429–1432. Citeseer, 2011.

[13] Alexander Gutkin. *Log-linear interpolation of language models*. PhD thesis, Citeseer, 2000.

[14] Bo-June Hsu. Generalized linear interpolation of language models. In *Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on*, pages 136–140. IEEE, 2007.

[15] Srilm manual for ppl-scripts. `http://www.speech.sri.com/projects/srilm/manpages/ppl-scripts.1.html`. Accessed: 2016-28-04.