NLP 2 - Project 1

April 2, 2015

This project will help you familiarise yourself with word-based models. Word-based models remain at the core of today's SMT systems in the form of alignment models. You will implement the simplest (though still widely used) word-based models, namely, IBM model 1, a lexical translation model, together with some improvements to the difficulties mentioned in Moore (2004), and IBM model 2, which models an impoverished form of word alignments.

In summary, your task is to

- implement IBM model 1
- implement IBM model 2
- try to think up and implement improvements to the problems of IBM Model 1 as discussed in Moore (2004), or borrow from that paper,
- (Optional: come up with and implement your own further improvements of the alignment model)
- write a report where you compare the baseline models and the improvements. Your report should also present learning curves where applicable along with a discussion explaining aspects such as non-convexity, stability and convergence.
- the structure of the report and what is expected to be submitted is made available in the separate instruction made available to you by the TAs.
- Submit your code that outputs the alignments in NAACL format (see below). Also include an executable shell script that executes your code (`align.sh [ibmModel] [englishData] [dutchData]`) and writes the alignemnts to a file called `alignments.out`. The parameter ibmModel specifies which IBM model to use (1 or 2).

1 IBM model 1

1. implement IBM model 1 and its EM training (Brown et al., 1993);
2. plot the evolution of the log likelihood function as a function of the iteration;
3. obtain Viterbi alignments for every sentence pair in a test corpus and compute AER using a gold-standard provided by the assistant;

In your report, you should also consider the limitations of Model 1 as described by Moore (2004), and find examples in your output to illustrate these limitations (both the three 'structural' problems not addressed by Moore as well as the two 'non-structural' problems).

2 IBM model 2

1. extend your previous model by implementing a full IBM model 2 (Brown et al., 1993);
2. IBM 2 is non-convex, thus you will see that optimising the log-likelihood function is not as trivial as in the case of IBM model 1, particularly, convergence will depend on how you initialise the model parameters, you will try
   - uniform initialisation
   - random initialisation (try 3 different starting points)
   - initialise the lexical parameters using the output of a complete run of IBM model 1
3. plot the log-likelihood function as a function of the iteration for all these cases
4. compare to IBM model 1 in terms of AER in the test set

3 Improvements to IBM model 1 from Moore (2004)

Moore (2004) proposes to improve IBM model 1 by

1. smoothing counts for rare words
2. assigning more probability mass to NULL alignments
3. heuristic initialisation of lexical parameters

You should implement these three improvements and run your evaluations to measure the extent of their effect. If you compare the example sentences from the original model 1 to the alignments from the extended version, have they improved?

4 Data

You can find the data to test your model on on the deze server. That one you can access through ssh:
`ssh -p 4100 your.username@deze.science.uva.nl`
In /home/bart/project_1/ you'll find a 1000 sentence parallel corpus. The same corpus can also be found under Course Materials/Project 1 on blackboard. It is intended for development purposes. We will soon upload another corpus for testing.

5 Output format

In order to allow for a quick evaluation of your code, make sure to adhere to the NAACL format. Details on the format can be found here. Please omit NULL-alignments. The basic layout of the NAACL format for one alignment link then is:

```
sentenceNum foreignWordPosition englishWordPosition
```

6 Further extensions not included in Project I

These extensions are actually not part of the Project 1 but they provide you with some material and questions to dwell on related to word alignment models.

In case you really are interested and are keen on implementing some of these extensions, showing that you've also read some of this extra literature, you could earn yourself extra bonus points on top of Project 1.

Smoothing

The add-n smoothing used by Moore (2004) is relatively crude, and potentially a more sophisticated method might improve alignments. For instance, Riley and Gildea (2012) address the problem of sparse distributions for rare words by smoothing expected counts in the M-step using a technique called *variational Bayes*.

Symmetric alignments

IBM models 1 and 2 are asymmetric models, (Liang et al., 2006) introduce a simple yet effective way to make the parameters of source-to-target and target-to-source alignments agree effectively learning a symmetric model.

Over-parameterisation

IBM model 2 is simple and inference is efficient, which is particularly important given that parallel corpora might contain millions of sentences. However, estimation for IBM model 2 is somewhat crude due its over-parameterisation. The model explicitly conditions on sentence length (source and target) and on alignment point amounting to $O(n^4)$ parameters. Dyer et al. (2013) use a parametric alignment distribution which reduces the number of parameters to 2 (a null alignment probability and a precision parameter).

Non-convexity

The log-likelihood function under IBM model 2 is non-convex, thus unlike the case of IBM model 1, EM is not guaranteed to converge to a global optimum. That complicates parameter estimation as EM becomes sensitive to initialisation. Simion et al. (2013) relax IBM model 2 obtaining a closely related model for which the log-likelihood objective is convex.

Semantic similarity

The lexical parameters of IBM model 1 are unaware to the intuition that the semantic relationship between words should in principle help obtain better translation models. Songyot and Chiang (2014) learn word similarities and integrate those with IBM model 1 by marginalising over similar words. The authors choose to learn a probability distribution over similar words using a neural network. Interacting with a neural network toolkit is beyond scope, so for this task you may choose to use a simpler framework. For instance, you can employ ideas from distributed semantics without necessarily training a neural network.[1]

Inference

Inference for IBM models is typically done with the Viterbi algorithm, that is, one typically produces the alignment which has highest probability under the model given the sentence pair. Kumar and Byrne (2002) propose to use a minimum risk criterion. In minimum Bayes risk (MBR) decoding, one chooses the hypothesis (set of alignment points) which minimises the expected loss or **risk**. A loss measures how compatible a prediction is with a true response. If you choose this extension, you will implement MBR decoding and experiment with 3 kinds of loss functions: i) a loss inspired by alignment error rate (AER), ii) a loss defined in terms of automatic word-cluster distance, and iii) a loss defined in terms of paraphrases.

[1] Or the assistant could provide students with probability distributions $p(f'|f)$ and $p(e'|e)$ (obtained for instance from PPDB).

References

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

Dyer, C., Chahuneau, V., and Smith, N. A. (2013). A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Kumar, S. and Byrne, W. (2002). Minimum bayes-risk word alignments of bilingual texts. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 140–147. Association for Computational Linguistics.

Liang, P., Taskar, B., and Klein, D. (2006). Alignment by agreement. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 104–111, New York City, USA. Association for Computational Linguistics.

Moore, R. C. (2004). Improving ibm word alignment model 1. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 518–525, Barcelona, Spain.

Riley, D. and Gildea, D. (2012). Improving the ibm alignment models using variational bayes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 306–310, Jeju Island, Korea. Association for Computational Linguistics.

Simion, A., Collins, M., and Stein, C. (2013). A convex alternative to IBM model 2. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1574–1583, Seattle, Washington, USA. Association for Computational Linguistics.

Songyot, T. and Chiang, D. (2014). Improving word alignment using word similarity. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1840–1845, Doha, Qatar. Association for Computational Linguistics.