UNIVERSITY OF CAMBRIDGE

Devan Kuleindiren
Robinson College
`dk503`

Computer Science Tripos - Part II Project Progress Report

Language Modelling for Text Prediction

January 30, 2017

supervised by
Dr. Marek Rei & Dr. Ekaterina Shutova

with Director of Studies
Dr. Alastair Beresford

and Overseers
Dr. Markus Kuhn & Prof. Peter Sewell

Overall, my project is roughly on schedule. There is one bullet point on my timetable up to this point that I have not yet covered (experimentation with language models on error-prone text), but I don't foresee this being an issue because there are other parts of my timetable that I have completed ahead of time.

Work completed

- I have implemented n-gram language models with various smoothing techniques. These include add-1 smoothing [1], Katz smoothing [2], absolute discounting [3], Kneser-Ney smoothing [4] and modified Kneser-Ney smoothing [5], as set out in the proposal.
- Alongside the n-gram models, I have also implemented some recurrent neural network (RNN)-based language models. The differences in the RNN architectures arise in the way that the weighted inputs and hidden activations from the previous time step are wired up. I explored three such architectures: the first being the standard RNN[6], the second the gated recurrent unit (GRU) [7] and the third long short-term memory (LSTM) [8].
- I have built a generic benchmarking framework into which you can plug-in any of the aforementioned language models and generate a series of metrics. So far, these metrics include: perplexity, average keys saved[1], guessing entropy, physical memory usage and average inference time.
- In order to demonstrate the capability of such models, I have also constructed a fun command-line application that can predict your next word, finish your sentence or generate text.
- Tests have been written for all of the above.

Work to complete

As mentioned at the top of this page, I am yet to explore the performance of language models on error-prone text, and so I will be working on this over the next few weeks.

Apart from this, the only other work set out in my proposal is the extension of embedding a language model in a mobile keyboard on iOS. I have partially implemented this, but there are still some blocking issues, and I am still investigating whether TensorFlow can actually run inside an app extension[2] (the subtle difference between apps and app extensions is something I did not foresee at the start of this project). For now, this will be low priority, and I will only complete it if I establish that it's possible.

[1] The average number of characters per word that the language model would save you from typing as a result of it getting the correct next word in its top 3 predictions.

[2] https://developer.apple.com/app-extensions/

Challenges faced

One of the earliest challenges I faced was inferring exactly how some of the n-gram smoothing techniques should be written in code, because there were some implementation details omitted in the respective papers.

Another particularly challenging aspect of my project has been working with the C++ API for TensorFlow, which has very little documentation.

References

[1] W. E. Johnson, “Probability: The deductive and inductive problems,” *Mind*, vol. 41, no. 164, pp. 409–423, 1932.

[2] S. Katz, “Estimation of probabilities from sparse data for the language model component of a speech recognizer,” *IEEE transactions on acoustics, speech, and signal processing*, vol. 35, no. 3, pp. 400–401, 1987.

[3] H. Ney, U. Essen, and R. Kneser, “On structuring probabilistic dependences in stochastic language modelling,” *Computer Speech & Language*, vol. 8, no. 1, pp. 1–38, 1994.

[4] R. Kneser and H. Ney, “Improved backing-off for m-gram language modeling,” in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1, pp. 181–184, IEEE, 1995.

[5] S. F. Chen and J. Goodman, “An empirical study of smoothing techniques for language modeling,” *Comput. Speech Lang.*, vol. 13, pp. 359–394, Oct. 1999.

[6] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, “Recurrent neural network based language model.,” in *Interspeech*, vol. 2, p. 3, 2010.

[7] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[8] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.