CAPSTONE REPORT

Becoming Jane by Deep Learning

Yoko Harada

October 27, 2016
Udaciy Machine Learning

INTRODUCTION

Jane? Yes, Jane. Her name is Jane Austen, a famous British novelist in 18th century. The title, "Becoming Jane" is borrowed from her biographical movie released in 2007 (http://www.imdb.com/title/tt0416508/). The next question would be why "becoming." The experiment here is whether a deep learning can produce or mimic sentences as if Jane wrote. In a catchy phrase, it would be "AI can become Jane?"

There are many areas the deep learning is good at. So called natural language processing (NLP) is among them. In NLP, the data are not independent points. For example, the word "becoming" has some relations to previous or later words, like "becoming Jane." Because of this nature, the AI should learn from a sequence of data (words). In such a case, the model called Recurrent Neural Networks or RNN is typically used. This model is designed to learn the data based on what have learned so far. As for the simple example, "becoming Jane," to learn the word, "Jane," the model uses what have learned by the word, "becoming." In the next phase of learning, the word comes next to "Jane" will be learned based on the knowledge acquired by "Jane".

Once RNN learns enough, it will have a word list with probabilities given some particular word. With this word lists, the model can predict a next word to next word based on the probabilities. When a prediction continues, say, 200 times, the sentence(s) of 200 words will be produced. This is what this experiment is trying.

DATA

Since "Becoming Jane" is the goal, Jane Austen's well-known two novels are used to train the model. One is "Pride and Prejudice"(http://www.fullbooks.com/Pride-and-Prejudice.html) and another is "Sense and Sensibility" (http://www.fullbooks.com/Sense-and-Sensibility-by-Jane-Austen.html). Whole two novels are concatenated into a single text file. The total size is 1.3 MB.

Once the data is loaded, sentences are split by a space to create a list of words. During this process, all line feed (\n) characters are replaced by a space. The original input data is formatted to have almost the same line length as in below. In this input data, line feeds don't contribute to predict the sequence of words. For this reason, the line feeds were eliminated.

> Elizabeth Bennet had been obliged, by the scarcity of gentlemen,
> to sit down for two dances; and during part of that time,
> Mr. Darcy had been standing near enough for her to hear a
> conversation between him and Mr. Bingley, who came from the
> dance for a few minutes, to press his friend to join it.

After the list of words is created, a sorted word index list and dictionary of word/frequency pair will be created. These are the feed data for the training.

EXPECTED RESULTS

1. Sequence of words

   When the sequence of words are generated after the training, those should look like natural sentences. Also, hopefully, those looks like an excerpt from Jane Austen's novel.

   The sequence of words will be generated by a sequential predictions starting from a seed word. In this experiment, the trained model's variables are saved in a file and can be restored for later predictions. Once the training finishes, the part of generating sequence of words in the notebook runs many times without running training code block again.

2. Word2Vec

   Another expected result is a two-dimensional visualization of vocabulary, known as word2vec. Similar words should reside in a close area on two-dimensional map. For example, modal verbs such as might, would or can are considered as similar and should be put together in a particular place on the map. This two-dimensional representation depicts an internal state when the training finishes.

3. Numerical Performance

   Alongside to intuitively understandable results, a training performance will be reported during the training. This performance is called a perplexity, or average per-word perplexity. The value should decrease as training goes.

These are overview of expected results. The details will be explained later.

ALGORITHMS

- Recurrent Neural Networks (RNNs)

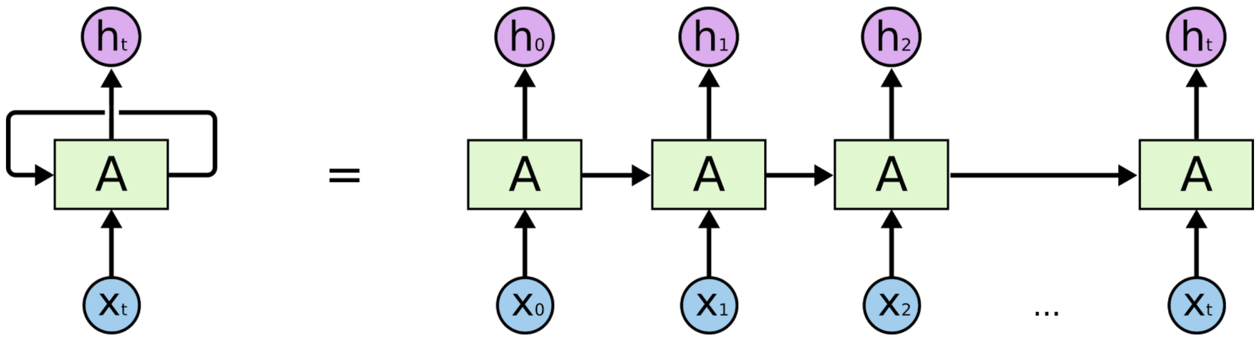
As mentioned earlier, the Recurrent Neural Networks (RNNs) is the algorithm used to train and predict Jane Austen's sentences. Like other neural network models, basically, RNN calculates:

$$y = Wx + b$$

where, x: input, W: weight, b: bias, y: output. Something RNN is different from others is it reuses the output. The training process goes like this:

$$y_0 = Wx_0 + b$$
$$y_1 = W(y_0,\ x_1) + b$$
$$y_2 = W(y_1,\ x_2) + b$$

The picture below is describes the RNN learning process. The leftmost network shows all recurrent processes in a single one, so it has a loop. This single network expression is equivalent to the right side. The second, third, … networks take a new input data and the output from previous training, which is a recurrent connection. This is why the model is called Recurrent Neural Networks. The programming model is the left one, while, what's going on is described on the right side.

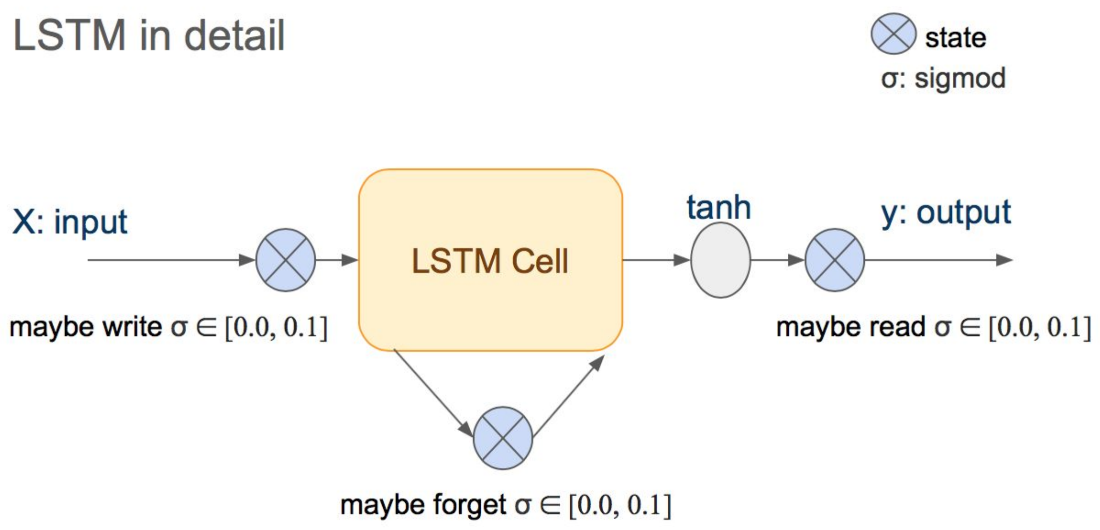


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

- Problem of RNN Model

For the natural language processing, the sequence is the matter. As explained earlier, "becoming Jane" makes sense because the word, "Jane," comes after "becoming." Often, a sentence makes sense in more than three words. It may be five words or even longer for people to figure out what it means. In this point of view, memorizing words learned so far works well to learn the next word effectively. However, how many should be memorized is to be considered. "Pride and Prejudice" has about 121500 words, “Sense and Sensibility” has 118700. To learn a word in the middle of the novel, all previous 60000 words don’t need. Additionally, too many previous words would be noisy and lower the prediction accuracy. To solve this problem, the idea of “forget it” has been introduced to RNN.

- Long Short Term Memory Networks (LSTMs)


LSTM in detail
state
σ: sigmod
X: input
LSTM Cell
tanh
y: output
maybe write σ ∈ [0.0, 0.1]
maybe read σ ∈ [0.0, 0.1]
maybe forget σ ∈ [0.0, 0.1]

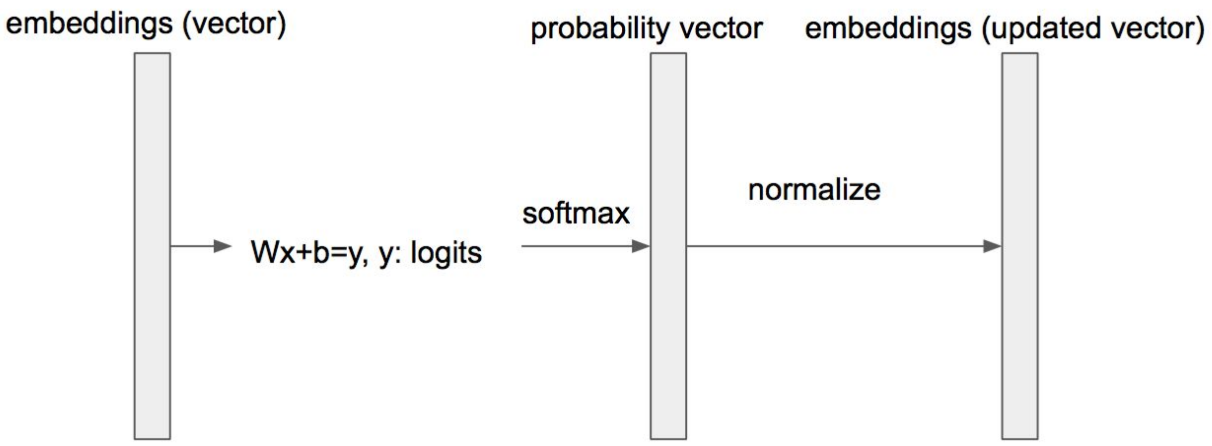
The picture above describes the details of LSTM (based on the lecture, Udacity Deep Learning). Parameters are controlled by a sigmoid function at the three points; input, forget, and output. The tanh (hyperbolic tangent) before output is there to keep output values in the range [-1.0, 1.0]. When maybe write/forget/read is 1.0, parameters will be passed through as those are. When it is 0.0, nothing will be passed. The sigmoid function is expected to produce the value between them. This architecture solves the problem cause by memorizing all words equally.

Since the code uses TensorFlow’s LSTM cell, it doesn’t control input/forget/output parameters explicitly. However, the feature is implemented within.

- Word2vec

In NLP, the process called word2vec is important to measure how much those two words are close. The closeness or distance is calculated by cosine distance. As mentioned in the Expected Results section, all words can be plotted on the two dimensional space based on similarity. The word2vec is the process to learn similarity. The word similarity is expressed by a vector which is called embeddings. During RNN training, inputs are always looked up from this embeddings.

Roughly, the word2vec works as in the picture below:



The leftmost is an embeddings vector, which is an output from LSTM cell. At the next step, logits will be calculated. Following step creates probability vector using a softmax function. In the end, the normalized probability vector will be the embeddings to feed to RNN for the next training iteration.

Softmax function: $S(y_i) = e^{y_i} / \sum_j e^{y_i}$

RESULTS

1. Sequence of Words

   For comparison, let's see generated sentences after only one epoch has finished, then, after 15 epoch has done. Both, the seed word is Elizabeth, and has 200 words. The word sampling is done by weighted pick. It is hard to compare numerically; however, apparently, the second one looks natural. For the comparison, the word sampling by argmax is showed on the third.

   - only one epoch, word sampling by weighted pick

   Elizabeth looked; married.--Well, a cottage, so; which do walked for by their handsomest was joining whose strangers brilliancy she dare wish from the favour they and her expected the youths Hertfordshire." however, on every way and the concerned, complaint till particulars,--and and were towards the great happy." which hope, attaching, and by the disclaim till the second sent. were not Charlotte the half.--Five prudently and Mrs. John and haughty. of them, which while the borne." much There was income the different shocking! the few concealed the contrary, was the room. from the hour walked thousand the Imprudence the few son trust; out of the embarrassment or "Ten perhaps, the hope are at the 'For well! the fellow every holding "but and I talked was so exactly." for the marriage to listen, the wish blast; But I was did How the until himself "HER she was distracted." "I must favours I should never add I should liked him him heard I should pride--for through the anyhow from Building, if and the choose," arrival; expressions to him. the Dashwood was believed but "believe like the park, but she affecting said its mother. the little best life--" But Marianne is not give the possible

   - 15 epochs, word sampling by weighted pick

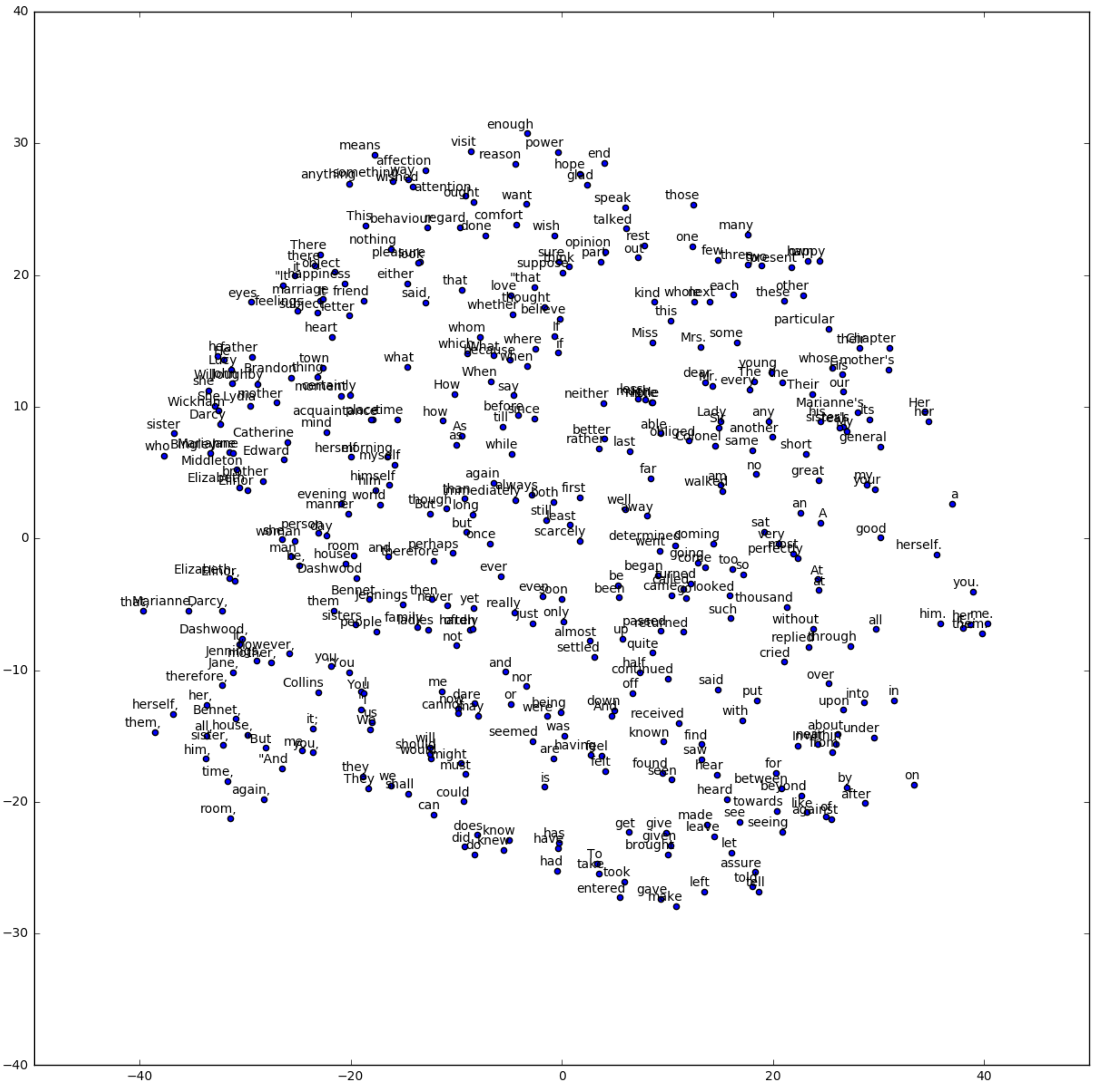
Elizabeth would never have received, prevented her resolution was impatient to her while she believed that the whole equally justly lives. Mrs. Jennings, however, was particularly softened in one of the brilliant From her brother's and making a letter of remains to being informed that his two hours at vain. Do not oblige me for your father, I can then believe, though I am sure it will hardly tempt you to say, to Frosts on dining over it, I beg it but her aunt has interrupted this declaration; but that any period cannot be estimated too easily added to each other. I was less married; she opened before her father almost with a family of course or unforeseen from favourable, she would allow me the same; and I shall answer myself with the assurance which her sister explaining it. You will not be honest enough to accept what I recollect consequently after all; for on Lucy, of hearing that you could not continue the smallest preferment. I am the selfish part of his having ever ceased to wait out trifle from me. "Now I was most diverted. "Yes, I am sure being quite necessary that he should soon married the same,

- 15 epochs, word sampling by argmax

Elizabeth was forced to go to the house by a few days after their arrival, and in the morning of the last to the world. The two girls joined them in the house, and the two gentlemen were to be in town last than the present. Mr. Collins was not in a settled at the same time by the house, and in the first of the world. In the first place, he was not in a settled like himself, and in the first of the neighbourhood they were to be in town last morning, and when they were all the way of the house, and the two gentlemen were to be the happiest of the world. But the girls were in the habit of a different manner of the world, and the very great attentions in the evening was in the same opinion of the world; and the two girls were to be the happiest of the world. But the girls were in the habit of a different manner of the world, and the very great attentions in the neighbourhood was in no great humour to Mrs. Jennings, that in spite of the most much of its friends, and the whole

2. Word2vec

   After 15 epochs has finished, the words similarity is mapped as in the picture below. At the bottom center, slightly left side, we can see modal verbs such as can, could, must, might, or will are put together. From this result, the training successfully classified words.



3. Numerical Performance

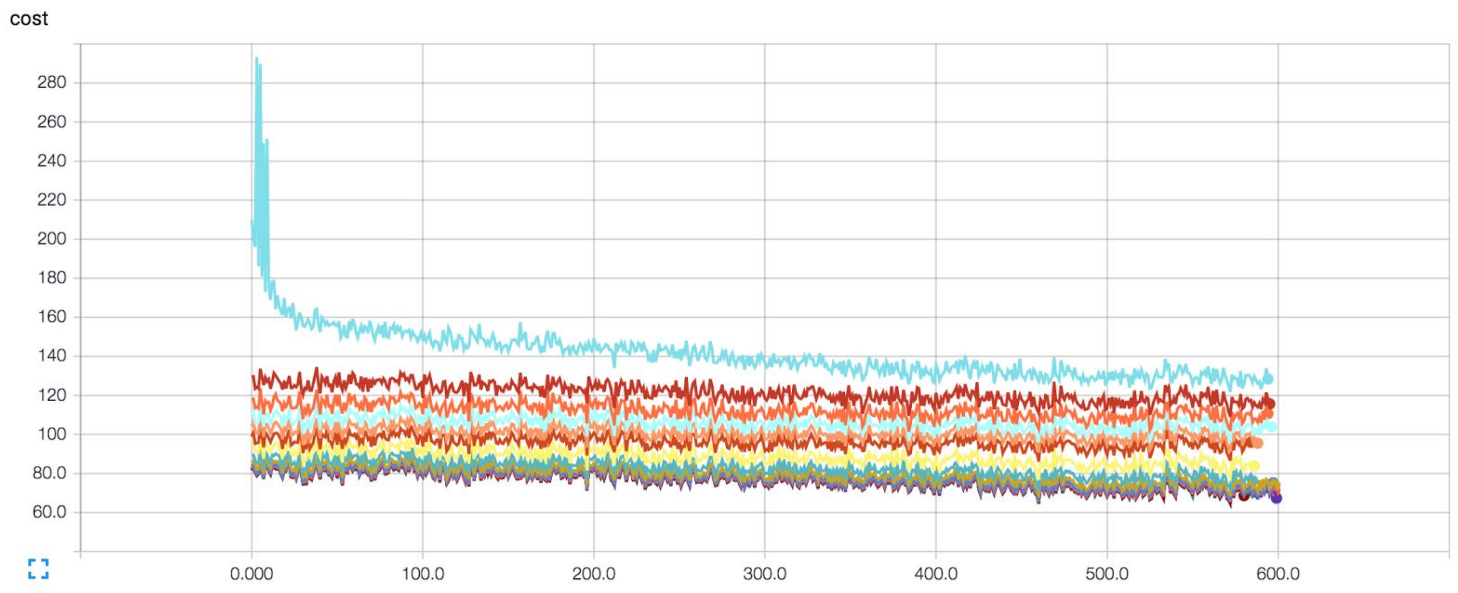
   a. Cost

This is a numerical results of training. Since the training tries to minimize a cost, it should decrease. The graph below shows how the cost has changed in each epoch. The graph has 15 curves in different colors. Each color corresponds to each epoch. The cost is an "average negative log probability of the target words," and calculate by the formula below:

$$loss = -\frac{1}{N}\sum_{i=1}^{N} ln(p_{target_i})$$

$$cost = \frac{loss}{batch\ size}$$

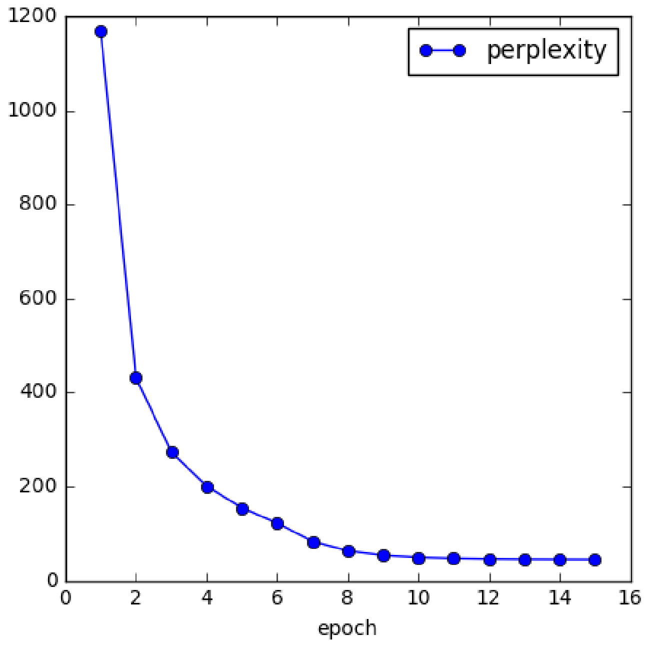
The words are processed by a batch size rather than a single word, the cost is divided by the batch size. Looking at the graph, we can see the cost is decreasing. This means the training went well.



b. Perplexity

A perplexity (average per-word perplexity) is also a numerical measurement. The perplexity basically sees the same parameter as the cost. As for a performance measurement, the perplexity might be common. The formula is:

$$perplexity = e^{loss}$$



CONCLUSION

By training, RNN could generate somehow readable sentences. However, it is still far from Jane Austen's novels. In the field, many trials of this sort have been reported, for example, "Andrej Karpathy blog: The Unreasonable Effectiveness of Recurrent Neural Networks" (http://karpathy.github.io/2015/05/21/rnn-effectiveness/) . The author attempted to generate Shakespeare, Wikipedia, Linux source code and others. The results are amazingly excellent. The biggest difference is the input data size. According to the blog post, those are Shakespeare: 4.4 MB, Wikipedia: 96MB, and Linux source code: 474 MB. Compared to that, the data size of two Jane Austen novels is only 1.3 MB. Probably, more data would make the sequence of words more natural.

However, a computational resource is limited for a personal project. In the blog post, it mentioned, "several as-large-as-fits-on-my-GPU 3-layer LSTMs over a period of a few days." For deep learning, a good computational environment might be necessary to get excellent results from a vast amount of data.

If the training was done using all Jane Austen's novels, the deep learning would have produced sentences as if she wrote.

REFERENCES

1. Udacity, Deep Learning, https://www.udacity.com/course/deep-learning--ud730
2. TensorFlow Tutorial, Recurrent Neural Networks, https://www.tensorflow.org/versions/r0.11/tutorials/recurrent/index.html#recurrent-neural-networks
3. The Unreasonable Effectiveness of Recurrent Neural Networks, http://karpathy.github.io/2015/05/21/rnn-effectiveness/
4. colah's blog: Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/
5. word-rnn-tensorflow, https://github.com/hunkim/word-rnn-tensorflow
6. YouTube, Sirajology, https://www.youtube.com/channel/UCWN3xxRkmTPmbKwht9FuE5A