Assignment 2

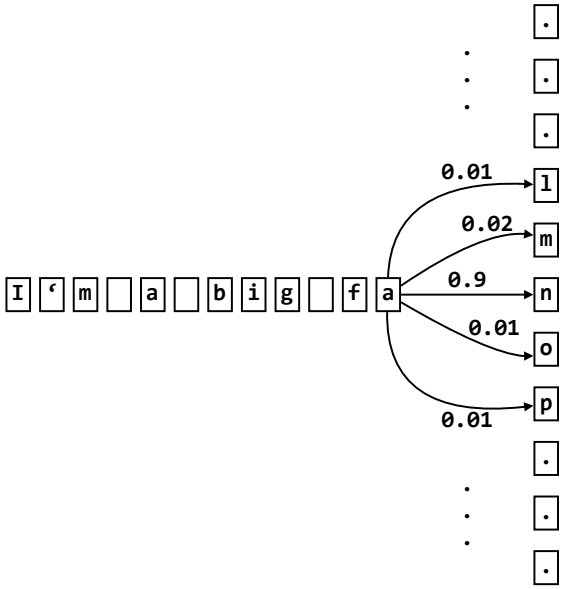
March 7, 2018

Due **April 4**

Introduction

We have discussed a some powerful building blocks so far. We engaged in a few thought experiments where we designed solutions for non-trivial problems. In this assignment, you will put all that to use. We are going to use convolutional neural networks to generate text. You can use our starter code (or roll out your own).

Our model in this assignment operates on a sequence of characters and predicts the next. See the figure below for a visualization of the model.



In the above figure, the sequence `I'm a big fa` is the input to the model and it produces a set of probabilities for all the characters. In this case, the character `n` has the highest likelihood - 0.9.

The highest probability sequence then is `I'm a big fan`. To predict the character that follows, we would use the character sequence `'m a big fan` as input and get probabilities for which character can follow next and so on. There are ways we can incorporate a longer sequence: just use a longer character sequence for the input (say 100 or 200 characters), for instance.

Instructions

We have provided some starter code for you in `training.py` available at this URL: https://deeplearningcourse.onai.com/files/training.py.

We are going to use a corpus of Nietzsche's works. The provided sample code downloads the data and sets up a simple model for you.

To run the code you will need Tensorflow, Keras and common scientific libraries like numpy and scipy.

The solution implemented is an MLP (i.e. a network with fully connected layers only). Your task is to replace this with a convolutional neural network. You can design your own architecture but we really recommend that you look for existing architectures and use them (and modify if necessary). **You are free to use code from any other source or models from any other existing literature. Do cite it in your submission.**

Our code will generate text for you at the end of each epoch. Most of your work will be in the function `build_model`.

The neural network provided does not learn anything useful and the text is barely better than gibberish. We will attempt to produce something nicer.

Answer the following questions about your model:

1. What architecture did you use? A summary of layers is fine. If you used an existing architecture in literature, then provide a citation.

2. Do you see any coherent text (you can decide what is coherent) or not? If you do see coherent text, how many epochs did it take to get to this point? You analysis could include, for example, at which stage you see words that look like English words, whether or not you see full well-formed sentences, etc..

3. Do you observe any relationship between the filter size and the quality of the generated text? To see how this works, you want to play with the kernel size hyperparameters in your model.

4. Try to play with the hyperparameters to see if you can improve the model. An example of the type of modification you can make is increasing the field of view of your network (at least of the layers close to the input) so your filters see more of the text at each stage. Report what you observe. Does this match your expectations? If not, do you think there is any reason things did not work out as expected?

For extra credit, try to use your convnet to generate text from a dataset of your choice—for example Shakespeare, restaurant names, or whatever you prefer. Share the output, the details of your dataset (a description or a URL are sufficient) and run through the above analysis for any model you train.

Administrative Details

- You can work in groups of 3 maximum. One submission for the group is sufficient. Please list all the group members.
- You can use any deep learning framework. While the starter code uses Keras, feel free to swap it out for PyTorch, Tensorflow, or whatever else you like.
- Print out and submit your answers and your code.
- If you use any external code, please cite it. This is allowed (encouraged even).
- **The assignment is due April 4 in class.**

Resources

- Keras documentation: https://keras.io/
- Keras convolutions: https://keras.io/layers/convolutional/
- Convolutional arithmetic: https://github.com/vdumoulin/conv_arithmetic