Assignment 3

CSE 143: Intro to Natural Language Processing

University of California, Santa Cruz

You may work in groups of up to four.

For this assignment, the primary goal is to get comfortable working with the Keras library to build neural network models, so we can do some neural NLP in practice.

The deliverables for this assignment will be your code, as well as a written report of your experiments and the findings from your programming. Most of your grade will depend on the quality of the report. You should submit it as a PDF. We recommend typesetting your scientific writing using LATEX. Some free tools that might help: Overleaf (online), TexLive (cross-platform), MacTex (Mac), and TexStudio (Windows).

Dataset

For this assignment, you are provided with a slightly modified version of the "Twenty Newsgroups" dataset (due to Tom Mitchell), a classic dataset meant for multi-way text classification. This corpus collects posts from various Usenet groups, early Internet forums in which people could argue angrily, before the advent of modern social media or Reddit.

The goal here is to take the text of a Usenet post and determine which "newsgroup" it was posted to. The newsgroups were forums dedicated to particular topics, ranging from technology to sports, politics and religion.

For scientific integrity, you ought to use the test data only once, just before you report all the final results. Otherwise, you will start overfitting on the test set indirectly. Please don't be tempted to run the same experiment more than once on the test data.

Getting set up

First of all, you'll need to install Keras and Tensorflow into your Python environment. On Ubuntu Linux with Python 3.12, you can make a virtualenv and (as of November 2024), simply run:

```
$ pip install keras
$ pip install tensorflow
```

Making Keras and Tensorflow run quickly on your machine may take some configuration, if you have a CUDA-enabled GPU, although it is possible to run on your CPU as well.

Programming: Text classification with RNNs

You are given starter code, in `assignment3.py`, which implements a simple text classifier that does multi-way classification over the 20 classes in the "Twenty Newsgroups" dataset. This classifier uses a neural network strategy that we did not discuss in class, convolutional layers.

Your goal, in this section, is to use Keras to build an RNN-based text classifier to solve the same problem.

Specifically, we want to:

- Embed the input text as a sequence of vectors
- Transform the sequence of embeddings into a single vector using a simple RNN
- Apply a feed-forward layer on that vector to obtain a label.

Though this seems like a detailed specification, you will still need to make several hyperparameter decisions to make this work. You could explore different hyperparameters if you like, or you could use these ones, recommended by Jeff.

- Choice of nonlinearity = tanh
- Word embedding dimension size = 16
- Hidden dimension size = 64
- Dropout rate = 0.5
- Choice of optimization method = adam
- Learning rate = 0.001
- Training batch size = 32
- Number of training epochs = 20

Programming: Text classification with LSTMs

The simple RNN suffers from two related problems:

- "Vanishing gradients" during learning make it hard to propagate errors into the distant past.
- State tends to change a lot on each iteration; the model "forgets" too much.

In practice, we often use more complex recurrent units, like long short-term memories (LSTMs) and the similar gated recurrent units (GRUs) Update your model to use LSTM units instead, and compare its performance to the simple-RNN-based one from the previous section. You can use the same hyperparameters.

Deliverables

You should turn in your code, which can be based on the included starter code if you like, with a README for how to run each of the variants you developed.

In the writeup, be sure to describe your models and experimental procedure. Provide some results on your validation and test sets, ideally with some tables or graphs.

Submission Instructions

Submit a zip file (A3.zip) on Canvas, containing the following:

- Code: You code should be implemented in Python 3, and needs to be runnable. Submit your code together with a neatly written README file explaining how to run your code with different settings. We assume that you always follow good practice of coding (commenting, structuring), and these factors are not central to your grade. If you have multiple files, provide a short description in the preamble of each file.
- Report: As noted above, your writeup should be in PDF. Include your teams names at the top of the report. Part of the training we aim to give you in this class includes practice with technical writing. Organize your report as neatly as possible, and articulate your thoughts as clearly as possible. We prefer quality over quantity. When discussing the experimental results, do not copy and paste the entire system output directly to the report. Instead, create tables and figures to organize the experimental results.

References

1. https://archive.ics.uci.edu/dataset/113/twenty+newsgroups