Transformers

MCDA5511 Assignment #3

Due: Mar 21 before midnight

Set Up

In this assignment you will implement a basic neural network and a basic transformer from scratch. The purpose is to understand how these models actually work under the hood, so that when you use high-level libraries you know what they are doing.

You will submit your answers in a Jupyter notebook (*.ipynb* file). Use markdown cells to create section titles, python cells for code, and markdown cells to answer written questions. Make sure that the layout of your code and written answers is easy to follow.

Homework Part 1: Backprop

Start by selecting a dataset for a binary classification task. Make sure to choose a dataset with balanced classes so that you don't have to deal with class imbalance issues.

(1) Start with a basic exploration of your data. What is the interpretation of the labels? What types of features are included?

(2) Split the data in train and test sets. (Note that you can do this with `train_test_split` from *sklearn*.) Use 80% train, 20% test.

(3) Fit a logistic regression model and compute accuracy on the test set. You can use the `sklearn.linear_model` and `sklearn.metrics` modules for this.

(4) Now, compare with a neural network (NN). However, instead of using *sklearn*, write out the code to train the NN from scratch. Use a network with a single layer with number of neurons equal to the number of features and sigmoid activation. Initialize the weights randomly, implement forward propagation to compute the predictions, and the backpropagation algorithm to update the weights using gradient descent. It may be helpful to include a learning rate parameter. Run the training process for long enough for the model to converge. Compute the accuracy - you may use `sklearn.metrics` for this. Compare the results with the logistic regression from the previous question.

(5) Imagine that you are hired as a data scientist and your boss asks you to implement a neural network to solve a classification problem, similar to what you did in the previous problem. Your boss is pleased with the results and wants to give a presentation about how the company is deploying cutting edge "self-learning" AI. She knows what linear regression is but doesn't know anything about neural networks. How do you explain to her what "self-learning" means in this case given what you know about the backprop algorithm?

Homework Part 2: Transformer

(1) Implement a transformer from scratch in PyTorch, i.e. without relying on high-level transformer-specific libraries such as HuggingFace's `Transformers`. Instead, use only the basic components available in the PyTorch framework, such as `torch.nn`. To keep things tractable, use fixed hyperparameters.

- Begin by populating a markdown cell with an overview of model components you will implement and a brief description of what they do, e.g. token and position embeddings, attention, feed forward network, etc.
- Next, implement each component as a class. Where possible, include a test to check that outputs returned for each method have the correct shape.

If you wish, you can restrict the implementation to an encoder only architecture and still receive full marks. Also, feel free to use a hyperparameter optimization package like Optuna, although this is not required.

(2) Train your transformer to reverse a sequence of five integers from 1 - 9. For example, if you input [3, 5, 2, 4, 1], the output should be [1, 4, 2, 5, 3]. Your code should include the following steps:
- Generate a synthetic data set of inputs and outputs.
- Tokenize the data.
- Create a training loop.
- Write a function to generate output from the trained model.

NOTE: If you wish, you may train the model to perform a more complex task.

(3) Draw a diagram showing the architecture you have implemented, including the dimensions of inputs/outputs at each step.

(4) Evaluate your model's performance quantitatively and write some commentary explaining your results. If you decided to build a model to do something more complex than reversing a sequence, you will need to give some thought to how you measure performance. Note that you will be graded on your implementation of the various transformer components, not on the model performance, so don't worry too much if the performance is poor.