Deep Learning: Assignment 3

Implementation of CNN, RNN, LSTM, and GRU

This assignment involves the following tasks:

- Image classification using CNN
- Sentiment Analysis using RNN, LSTM, and GRU

Submit the executed code in Jupyter notebook. You can write your observations and results using the heading and markdown cells in Jupyter. If you have memory or GPU constraints, you can use Google colab or Kaggle. The links to the libraries and resources required are added in Moodle. If the training takes lot of time, then train in intervals by saving and restoring the models from checkpoints. You can learn about saving and restoring the models from the documentation of Tensorflow or Keras.

1. Image Classification using CNN:
   The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The task is to construct a CNN for CIFAR10 classification.

   (a) Fetch the training and test datasets for CIFAR10 using the built in functions in Tensorflow or Keras. Follow the links in Moodle, to know how to download the datasets.

   (b) Create a validation set of 10000 images from the training set.

   (c) Train the following CNN architectures as follows:

   (d) **Model1:**

   - The network comprises of 2 convolutional layers and 2 max pooling layers and 1 fully connected hidden layer.
   - Number of kernels in the first convlayer =32 (5x5 filters)
   - Number of kernels in the 2nd conv layer =64 (5x5 filters)
   - Max pooling of size 2x2
   - Non-Linearity used in all hidden layers is ReLU
   - The output layer is softmax
   - Number of units in the fully connected hidden layer1 = 64
   - Number of units in the output layer =10
     The network architecture would be as given below:
     Input → conv1 (32 filters (5x5)) → Maxpool (2x2) → conv2 (64 filters (5x5)) → Maxpool (2x2) → Flatten → FCL1(64) → Softmax output layer(10)

   (e) **Model2:**

   - The network comprises of 2 convolutional layers and 2 max pooling layers and 1 fully connected hidden layer.

- Number of kernels in the first convlayer =32 (5x5 filters)
- Number of kernels in the 2nd conv layer =64 (5x5 filters)
- Max pooling of size 3x3
- Non-Linearity used in all hidden layers is ReLU
- The output layer is softmax
- Number of units in the fully connected hidden layer1 = 64
- Number of units in the output layer =10
  The network architecture would be as given below:
  Input → conv1 (32 filters (5x5)) → Maxpool (3x3) → conv2 (64 filters (5x5)) → Maxpool (3x3) → Flatten → FCL1(64) → Softmax output layer(10)

(f) **Model3:**

- The network comprises of 2 convolutional layers and 2 max pooling layers and 1 fully connected hidden layer.
- Number of filter kernels in the first convlayer =32 (3x3 filters)
- Number of filter kernel in the 2nd conv layer =64 (3x3 filters)
- Max pooling of size 2x2
- Non-Linearity used in all hidden layers is ReLU
- The output layer is softmax
- Number of units in the fully connected hidden layer1 = 64
- Number of units in the output layer =10
  The network architecture would be as given below:
  Input → conv1 (32 filters (3x3)) → Maxpool (2x2) → conv2 (64 filters (3x3)) → Maxpool (2x2) → Flatten → FCL1(64) → Softmax output layer(10)

(g) **Model4:**

- The network comprises of 3 convolutional layers and 2 max pooling layers and 1 fully connected hidden layer.
- Number of kernels in the first convlayer =32 (3x3 filters)
- Number of kernels in the 2nd conv layer =64 (3x3 filters)
- Number of kernels in the 3rd conv layer =128 (3x3 filters)
- Max pooling of size 2x2
- Non-Linearity used in all hidden layers is ReLU
- The output layer is softmax
- Number of units in the fully connected hidden layer1 = 64
- Number of units in the output layer =10
  The network architecture would be as given below:
  Input → conv1 (32 filters (3x3)) → Maxpool (2x2) → conv2 (64 filters (3x3)) → Maxpool (2x2) → conv3 (128 filters (3x3)) → Maxpool (2x2) → Flatten → FCL1(64) → Softmax output layer(10)

(h) Which the best model among model1 to model4? Why?

(i) **Model5:**

- Choose the architecture of best model among model1, model2, model3, and model4. Add strides of 2 to all the convolution layers. If it is not possible to add strides due to negative dimension, choose the next best model.

(j) **Model6:**

- Choose the architecture of best model of model1, model2,model3, and model4. Add "SAME" padding to all the convolution layers.

(k) **Model7:**

- Choose the architecture of best model of model1, model2,model3, and model4. Add "SAME" padding to all the convolution layers with stride2. If it is not possible to add strides due to negative dimension, choose the next best model.

(l) Which the best model among model1 to model7? Why?

(m) **Model8:**

- Choose the architecture of the best model among model1 to model7. Use tanh activation function in the hidden layers instead of ReLU

(n) **Model 9:**

- Choose the architecture of the best model among model1 to model7. Use sigmoid activation function in the hidden layers instead of ReLU

(o) For each model trained, plot the loss vs iteration curve and accuracy vs iteration curve for training data.

(p) Tabulate the following for model 1 to model 9. You can create the table by hand and then upload the screenshot of the table in the jupyter notebook.

- Total number of trainable parameters.
- Training time
- Training accuracy
- Validation accuracy
- Test accuracy

2. Sentiment analysis on IMDB movie review dataset:
The dataset comprises of 50,000 movie reviews taken from IMDb. The training and test data sets are uploaded in moodle. The datasets are in the form of text files. Every line in the files is a movie review. The training set and test set comprises of 25000 movie reviews each. For both training and test data sets, the first 12500 reviews are positive and the rest of the reviews are negative.

(a) Preprocess the dataset:
Remove punctuation and $< br >$ tag. Convert all the characters to lower case.

(b) Create the target vector for training and test data. The first 12500 values of target vector is 1 and the rest of the values are zero. The target vector is same for training and test data.

(c) Create a validation set which is 20% of the training set.

(d) Train the following models.

(e) **Model 1:**

- Integer encode the reviews
- To make the length of all the reviews equal, pad the reviews to maximum length of 200. You can use pad_sequences API in Tensorflow or Keras.
- It is possible to learn the vector embedding of words using the embedding layer in Tensorflow or Keras. Follow the links in Moodle to learn about vector embeddings.
- The network consists of an embedding layer, 1 RNN layer and output layer.

- The embedding size of embedding layer = 128
- Number of nodes in RNN layers = 200
- Non-linearity used in all RNN layers is tanh.
- The output layer is sigmoid.
- The network architecture is as follows:
  Input → Embeddding Layer (embedding size=128) → RNN1(200 nodes) → Sigmoid output

(f) **Model 2:**

- Integer encode the reviews
- The network consists of an embedding layer, 1 LSTM layer and output layer.
- The embedding size of embedding layer = 128
- Number of nodes in LSTM layers = 200
- Non-linearity used in all LSTM layers is tanh.
- The output layer is sigmoid.
- The network architecture is as follows:
  Input → Embeddding Layer (embedding size=128) → LSTM1(200 nodes) → Sigmoid output

(g) **Model 3:**

- Integer encode the reviews
- The network consists of an embedding layer, 1 GRU layer and output layer.
- The embedding size of embedding layer = 128
- Number of nodes in GRU layers = 200
- Non-linearity used in all GRU layers is ReLU.
- The output layer is sigmoid.
- The network architecture is as follows:
  Input → Embeddding Layer (embedding size=128) → GRU1(200 nodes) → Sigmoid output

(h) Which is the best model among model1 to model3 - RNN, LSTM or GRU? Why?

(i) **Model 4:**

- Integer encode the reviews
- The network consists of an embedding layer, 2 (GRU/LSTM/RNN based of the best performing model in model1 to model3) layers and output layer.
- All other configurations are same as model1.

(j) **Model 5:**

- Integer encode the reviews
- The network consists of an embedding layer, 3 (GRU/LSTM/RNN based of the best performing model in model1 to model3) layers and output layer.
- All other configurations are same as model1.

(k) **Model 6:**

- Use word2vec to create word embeddings of length 128
- Create vector representation of reviews from word embedding
- Choose the best performing model among model1 to model5. Remove the embedding layer in the best model and give the vector embeddings obtained by word2vec as input.

- The network architecture is as follows:
  Input(Vector embeddings obtained using word2vec) → RNN/LSTM/GRU layers (the number of layers and nodes same as the best model among model 1 to model5) → Sigmoid output layer

(l) Plot the loss vs iteration and accuracy vs iteration curve for training data for all the models.

(m) Tabulate the training, testing and validation accuracy for all the models.