DSE 3159 DEEP LEARNING LAB

WEEK 2

Exer 1:

Using the Body Fat dataset, design a Neural Network to predict body fat. Accurate measurement of body fat is inconvenient/costly and it is desirable to have easy methods of predicting Body Fat.

The attributes are :

1. Density determined from underwater weighing
2. Percent body fat from Siri's (1956) equation
3. Age (years)
4. Weight (lbs)
5. Height (inches)
6. Neck circumference (cm)
7. Chest circumference (cm)
8. Abdomen 2 circumference (cm)
9. Hip circumference (cm)
10. Thigh circumference (cm)
11. Knee circumference (cm)
12. Ankle circumference (cm)
13. Biceps (extended) circumference (cm)
14. Forearm circumference (cm)
15. Wrist circumference (cm)

1. Perform experiments using (70,15,15) split and tabulate the performance in terms of RMSE for the following Hyper parameters :

    a. Number of Hidden Layers and Number of Units per Layer

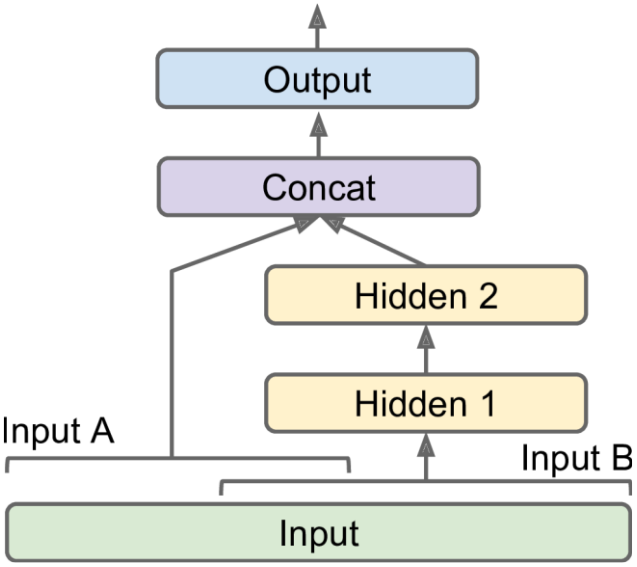
Number of Hidden Layers	Number of Units
1	128, 0 ,0
2	128, 64, 0
3	128, 64, 32

    b. Epochs (10,20,30,40)
    c. Activation function (Sigmoid /RELU)
    d. Without Regularization, with Regularization (L1/L2)
    e. Learning rate (1, 0.3, 0.1, 0.01,0.03,0.001,0.0001,0.00001)

2. Visualize the training and validation loss against the epochs and comment on optimal hyperparameters.

WEEK 5

1. Read documentation about The Keras functional API is a way to create models. https://www.tensorflow.org/guide/keras/functional
2. Using the Scikit-Learn's fetch_california_housing() function to download the California Housing Problem data.
3. Using the sequential API build a regression MLP (to make predications. Model can have 1 hidden layer with 30 units. Visualize the MSE for 20 epochs. Comment on overall accuracy.
4. Using functional API build the following Wide & Deep Neural Network with the following architecture.



Let Hidden layers contain 30 units each with ReLU activation. Pass features 0 to 4 in the wide path and features 2 to 7 into the deep path. Visualize the MSE for 20 epochs. Comment on overall accuracy.

WEEK 6:

EXERCISE 1:

https://keras.io/examples/structured_data/wide_deep_cross_networks/

Use the Covertype dataset from the UCI Machine Learning Repository. The task is to predict forest cover type from cartographic variables.

Create two representations of the input features: sparse and dense:

1. In the **sparse** representation, the categorical features are encoded with one-hot encoding using the `CategoryEncoding` layer. This representation can be useful for the model to *memorize* particular feature values to make certain predictions.

2. In the **dense** representation, the categorical features are encoded with low-dimensional embeddings using the `Embedding` layer. This representation helps the model to *generalize* well to unseen feature combinations.

Split the data into training (85%) and validation (15%) sets.

Model 1 :

Create a multi-layer feed-forward network, where the categorical features are one-hot encoded.

learning_rate = 0.001

dropout_rate = 0.1

batch_size = 265

num_epochs = 50

hidden_units = [32, 32]

Model 2:

Create a Wide & Deep model where the wide part of the model a linear model, while the deep part of the model is a multi-layer feed-forward network. Use the sparse representation of the input features in the wide part of the model and the dense representation of the input features for the deep part of the model.

Model 3:

Create a Deep & Cross model. The deep part of this model is the same as the deep part created in the previous experiment. The key idea of the cross part is to apply explicit feature crossing in an efficient way, where the degree of cross features grows with layer depth.

Compare the loss and accuracy of the three models.

EXERCISE 2:

Use the google stock price dataset available in Kaggle.

https://www.kaggle.com/datasets/medharawat/google-stock-price

Using a training set of 50 time steps, build a Simple RNN model vs a LSTM model, both with 4 layers. Compare their accuracy using an mean square error.

Plot the actual vs predicted values using the test data for the year 2017.

WEEK 7

Using the following data set:

https://www.kaggle.com/code/cemalefetezcan/imdb-review-sentiment-classification

divide the data into train/validation data sets, build 2 models to perform movie review sentiment analysis

Model 1:

1. Perform required text pre-processing – lowering text, removing URLs, punctuation , stop words and correct spelling .
2. Perform tokenization and lemmatization on cleaned data .
3. Visualize the most frequent words and bigrams
4. Visualize the practical words that represent positive and negative sentiment in the dataset.
5. Create an embedding layer and build a 15 layer LSTM and a 20 layer BidRNN for predicting the sentiment.
6. Build your own test dataset with 10 movie reviews and tabulate accuracy.