Question 1: Predicting Customer Churn using Neural Networks

Objective: The objective of this assignment is to develop a neural-network model that can predict customer churn for a telecom company. You will use a dataset containing customer information, and whether or not the customer churned.

Tasks:

1. Download the dataset from Kaggle https://www.kaggle.com/datasets/blastchar/telco-customer-churn
2. Load the dataset and perform exploratory data analysis.
3. Pre-process the data by converting categorical variables to numerical variables, and scaling the data.
4. Build a neural-network model using PyTorch (or other library of your choice) to predict customer churn.
5. Split the data set in train (80%) and test (20%) sets. Train the model using the training set.
6. Evaluate the model using the testing set and report the accuracy, precision, recall, and F1 score.
7. Experiment with different hyperparameters (layer count and neurons per layer) and architectures to improve the performance of the model.
8. Write a report summarizing your findings, including the best performing model and the factors that contributed to its success.

Summary report:

The objective was to predict customer churn using a dataset from a telecommunications company. The dataset contained both numerical and categorical variables, with 'Churn' as the target variable. The data was initially loaded to understand and visualize the structure. There was a total of 7043 entries and 21 variables. The dataset did not have any missing values.

The categorical variables were encoded using the LabelEncoder from the sklearn library, while the numerical variables were standardized to ensure their scales were consistent. The 'TotalCharges' column was converted to a numerical type which was an object type.

Two models were built, one with a single hidden layer and the second with two hidden layers. Both models were trained over 100 epochs with binary cross-entropy loss as the loss function and Adam as the optimizer.

The first model achieved an accuracy of 73.31%, precision of 53.75%, recall of 73.31% and F1 score of 62.02%. The same performance was observed in the second model despite the additional hidden layer.

The standardization of numerical data, handling of missing values, and encoding of categorical variables were crucial pre-processing steps. However, the models' performance did not improve with additional hidden layers, which suggests that further tuning or a different approach must be taken.

Question 2: Spam Classification using Recurrent Neural Networks

Objective: The objective of this assignment is to develop an RNN model for spam classification. You will use a dataset containing SMS messages labeled as spam or ham (not spam). You will use this data to train an RNN model to classify messages as spam or not spam.

Tasks:

1. Download the spam classification dataset from Kaggle. https://www.kaggle.com/datasets/team-ai/spam-text-message-classification
2. Load and pre-process the data by converting the message into sequences of tokens, padding the sequences to a fixed length, and splitting the data into training and testing sets.
3. Build an RNN model using libraries such as Keras or PyTorch to classify messages as spam or ham.
4. Train the model using the training set and evaluate the model using the testing set.
5. Experiment with different hyperparameters and architectures to improve the performance of the model.
6. Write a report summarizing your findings, including the best performing model and the factors that contributed to its success.

Summary report:

Spam Detection Model Performance Summary:

In the above problem, I have used a sequential model with an embedding layer, LSTM, and a dense layer for the spam detection. The training data was tokenized and padded to ensure consistency and a consistent input shape.

The data was then split into a training set (80%) and a test set (20%). The initial model had an accuracy rate of 97.49% on the test set. The high accuracy is attributed to the LSTM's ability to understand and learn the pattern and context in the message's dataset.

It was further experimented with different learning rates (0.001, 0.01, 0.1) while keeping the model architecture constant.

The model with a learning rate of 0.01 performed best, with a test accuracy of 98.12%, slightly better than the initial model. And as for the other two models having different learning rates, they achieved similar accuracies (97.76%).

The high accuracies of these models is due to the combination of embedding and LSTM layers which effectively captured both the word representation and sequence context in text data.

We could conclude that even a minor change/adjustment to the learning rate can lead to performance improvements. Furthermore, steps like more hyperparameter tuning, deploying other complex architectures, or adding regularization methods could again help in enhancing the performance.

Question 3: Image Classification using Transfer Learning

Objective: The objective of this assignment is to develop an image classification model using transfer learning. You will find a pretrain a neural network model, and then use transfer learning to classify logos of popular food chains.

Tasks:

1. Download the dataset from: https://www.kaggle.com/datasets/kmkarakaya/logos-bk-kfc-mcdonald-starbucks-subway-none
2. Preprocess the data by resizing the images and splitting them into training (80%) and validation sets (20%).
3. Load a pre-trained model, such as VGG-16 or ResNet, using Keras or PyTorch. (or a library of your choice)
4. Replace the last fully connected layer of the pre-trained model with a new fully connected layer with the appropriate number of output nodes for the new dataset.
5. Freeze the weights of the pre-trained layers and train only the new fully connected layer using the training set.
6. Evaluate the performance of the model using the validation set.
7. Fine-tune the entire model by unfreezing some of the pre-trained layers and training them along with the new fully connected layer using the new dataset
8. Again, evaluate the performance of the model using the validation set.

Write a report summarizing your findings, including the best performing model and the factors that contributed to its success.

Summary report:

Image Classification using Transfer Learning

The main goal here was the image classification – one of the computer vision problems. The dataset comprised of various logos - logos of 6 food industries and was divided into six distinct classes.

This was a multi-class classification problem which leveraged transfer learning, a technique that make use of pre-trained models. I opted for the VGG16 model, a popular pre-trained model in deep learning due to its high performance on the ImageNet dataset.

I have imported VGG16 from Keras and have set the 'include_top' parameter to 'False', this excludes the fully connected layers at the top of the network, which are often task specific. To make my model more apt, I have added a new fully connected layer and an output layer to my

existing model. The output layer has six neurons, exactly corresponding to the six classes in the dataset.

Initially, the VGG16 layers were frozen, and the newly added layers were trained keeping it that way. This approach yielded a validation accuracy of around 87.82%. Later, by unfreezing the weights of the VGG16 layers, I fine-tuned the existing model. I first ran the model with a higher learning rate, this caused drastic updates to the weights, thereby nullifying entire learning and gave out a very poor accuracy of around 60%. To mitigate this, I reduced the learning rate further to 0.0001. Post this fine-tune, the model's validation accuracy increased significantly to 93.33%. This confirmed the importance of the learning rates and how they influence the model and how it alters the performance of the model.

Overall, this gave me valuable insights into the role of transfer learning, fine-tuning and image classification tasks.