Write-Up

Approach:

In this project, we built a text classification model using BERT (Bidirectional Encoder Representations from Transformers) to categorize emails into predefined categories. The model is fine-tuned for the task of classifying email texts into categories such as products or services (e.g., Product A, Product B, Product C). The approach involves several stages, including data preprocessing, model selection, training, and prediction.

Preprocessing Steps

1. **Dataset**: We used a sample dataset consisting of email texts and their corresponding categories. Each email in the dataset corresponds to a particular product or service. This dataset is stored in a Pandas DataFrame.
2. **Tokenization**: To feed the email texts into the BERT model, we tokenize the text using the pre-trained BERT tokenizer (bert-base-uncased). Tokenization converts the text into numerical tokens (input IDs), making it compatible with the BERT model.
3. **Label Encoding**: The email categories (e.g., "Product A", "Product B") are converted into numerical labels using LabelEncoder from sklearn. This step is essential for supervised learning as the model needs numerical values for training.
4. **Data Splitting**: The dataset is split into training and validation sets using train_test_split from sklearn, with 80% of the data used for training and 20% for validation.
5. **Custom Dataset Class**: A custom PyTorch dataset class (EmailDataset) is defined, which takes tokenized email texts and their labels as inputs. This class is used to create DataLoader objects for efficient batching and shuffling during training.

Model Selection

For this classification task, we used a pre-trained BERT model (bert-base-uncased) for sequence classification. BERT is a transformer-based model known for its strong performance in various NLP tasks due to its pre-training on a large corpus of text. We fine-tuned BERT on our specific dataset using the BertForSequenceClassification model from the Hugging Face Transformers library.

Key configurations:

- **Pre-trained BERT Model**: bert-base-uncased
- **Output Labels**: 4 (corresponding to the categories: Product A, Product B, Product C, Product A/Product B)
- **Optimizer**: AdamW optimizer with a learning rate of 2e-5
- **Loss Function**: Cross-entropy loss, appropriate for multi-class classification tasks

Training and Evaluation

- **Training**: The model was trained for 10 epochs with a batch size of 8. In each epoch, we performed forward and backward passes on the training data, computed the loss, and updated the model

weights using the AdamW optimizer. The model's performance was evaluated using accuracy, which was calculated by comparing the predicted class with the actual class for each email in the batch.

- **Validation**: After every training epoch, we evaluated the model on the validation set using the same steps as the training phase. The learning rate was adjusted using the ReduceLROnPlateau scheduler to prevent overfitting and to improve the model's performance over time.
- **Metrics**: The primary evaluation metric used was **accuracy**, which indicates the percentage of correctly predicted email categories. The model was also evaluated using a loss function, where a lower loss indicates better performance.

Prediction and User Input

Once the model was trained and evaluated, we implemented a function to make predictions on new email text. The user can input email text via the console, and the model will classify the text into one of the predefined categories. Here's how the prediction function works:

1. The email text is tokenized using the BERT tokenizer.
2. The model processes the tokenized text and generates a probability distribution over the categories.
3. The category with the highest probability is selected as the prediction.

Final Model and Save/Load Mechanism

Once the model was trained and evaluated, we saved both the fine-tuned model and tokenizer using the save_pretrained method. This allows the model and tokenizer to be reloaded for future inference tasks.

This ensures that the trained model can be reused without needing to retrain it each time.

Conclusion

In conclusion, we developed a BERT-based email classification model capable of categorizing emails into predefined categories based on their content. The model performs well on the validation data, and it is ready to classify new emails with high accuracy. The approach can be extended to classify more complex text data or to handle different categories for other applications.