BERT-Based Text Classification

Approach:

The approach taken in this project involves using a pre-trained BERT (Bidirectional Encoder Representations from Transformers) model for the task of text classification. The primary steps in this approach are:

1. **Data Preparation:**
    - Text data, specifically abstracts from research papers, are preprocessed to remove noise (like punctuation) and then tokenized using the BERT tokenizer. Labels for the predefined domains are encoded into numerical format using LabelEncoder.
2. **Model Setup:**
    - A BERT model, specifically fine-tuned for sequence classification, is loaded from the Hugging Face transformers library. The model is set up with an output layer configured for the 7 predefined domains.
3. **Training:**
    - The model is trained using a training loop that updates the model's parameters based on the input text data. The AdamW optimizer is used alongside a learning rate scheduler to ensure efficient training over multiple epochs. The model learns to map input text to one of the predefined categories.
4. **Evaluation:**
    - After training, the model is evaluated on a validation set. Predictions are made without calculating gradients to save computational resources. The model's performance is measured using metrics such as accuracy and the weighted F1 score.

Assumptions:

1. **Text Representation:**
    - The assumption is that BERT's pre-trained embeddings are sufficiently robust to capture the nuances of the abstract text, even though they are fine-tuned on specific tasks like text classification.
2. **Label Distribution:**
    - It is assumed that the labels (domains) are adequately represented in the training data and that the model will generalize well to the validation data based on this distribution.
3. **Model Generalization:**
    - The approach assumes that the BERT model, fine-tuned with a relatively small dataset, will still generalize well to unseen data in the validation set.
4. **Data Balance:**
    - It is assumed that the classes (predefined domains) are either balanced or that the model is robust enough to handle any class imbalance without extensive preprocessing or balancing techniques.

Future Scope:

- **Tuning the Mode**l: We could fine-tune the model’s settings to make it even more accurate.
- **Trying Other Models**: We might explore other similar models to see if they work better.
- **Adding More Data**: We could create more training examples, like by rephrasing sentences, to help the model learn better.
- **Balancing the Data:** If some categories don’t have enough examples, we could try balancing the data to improve the model’s accuracy.
- **Adapting to Other Text Types**: We might teach the model to handle different kinds of text or even different languages.
- **Deploying the Model**: Eventually, this model could be used in real-time to automatically classify new abstracts.
- **Explaining the Model’s Decisions**: We could work on making the model’s choices easier to understand, so users know why it classified an abstract in a certain way.