Project

Final Project: NLP Task Using Three Models

Objective: Apply three different modeling approaches to solve an NLP task of your choice. This project involves going through the entire pipeline of solving an NLP problem, from dataset selection, preprocessing, and tokenization, to training and evaluating models.

Steps to be Completed:

1. **Choose an NLP Task:**
    - Select an NLP task to do (e.g., sequence-to-sequence tasks like machine translation or summarization).
    - Choose a dataset suitable for your task from resources like Kaggle.
2. **Model Selection:**
    - Implement three different models:
        - **Traditional Model:** For example, RNN OR LSTM
        - **Transformers Model**.
        - **Pretrained Transformer Model:** Use BERT, GPT-2, or GPT-3 from the Hugging Face Transformers library.
3. **Data Preprocessing:**
    - **Load the Dataset:** Import the dataset and explore its structure.
    - **Clean the Text:** Remove special characters, handle missing values, and perform any other necessary cleaning steps.
    - **Prepare the Data:** Ensure the data is ready for the chosen NLP task (e.g., format conversion, normalization).
4. **Tokenization:**
    - Tokenize the text data using appropriate tokenizers for each model.
    - For the pretrained model, use the tokenizer provided by Hugging Face Transformers.
5. **Word Embedding:**

    - Traditional models (e.g., RNN/LSTM) often require word embeddings (e.g., Word2Vec, GloVe). In contrast, transformer models (e.g., BERT) generate contextualized embeddings during training.

6. **Dataset Splitting:**

- Split the dataset into training, validation, and test sets.

7. **Model Training:**
   - Train each model on the training dataset.
   - Implement techniques such as early stopping and learning rate decay where applicable.

     Difference between Training Transformers from Scratch and Fine-Tuning Pretrained Models:

   - Training from Scratch: This involves training a transformer model on a large corpus and requires substantial computational resources and time. It is generally used when a large dataset specific to the task is available.
   - Fine-Tuning Pretrained Models: Pretrained models have already been trained on massive corpora (e.g., books, Wikipedia) and can be fine-tuned with much less data to specialize for a given task. This is generally faster and more efficient for most NLP applications.
   -
8. **Model Testing:**
   - Evaluate each model on the test dataset.
   - Use relevant metrics for your task (e.g., accuracy, F1 score).
9. **Comparison and Evaluation:**
   - Compare the performance of the three models.
   - Discuss which model performed best and why.
   - Provide insights into the strengths and weaknesses of each model.

Notebook Structure:

1. **Import Libraries:**
   - Import necessary libraries and modules.
2. **Load the Dataset:**
   - Read and explore the dataset.
3. **Data Preprocessing:**
   - Perform text cleaning and preparation.
4. **Tokenization:**
   - Tokenize the text data.
5. **Add visualizations if possible to enhance the analysis.**
6. **Model Training:**
   - Train each model.
7. **Model Evaluation:**

    - Evaluate each model and compare their performance.

Report Requirements:

- **Introduction:**
    - Explain the NLP task and why you chose the dataset.
- **Preprocessing:**
    - Discuss whether preprocessing was needed and what steps were taken.
- **Model Selection:**
    - Specify the chosen models and the rationale behind each choice.
- **Fine-Tuning:**
    - Describe the changes made during model fine-tuning.
- **Results:**
    - Compare the results of the three models and discuss any differences.

Deadline: 23rd December