**Report: Named Entity Recognition (NER) with BERT**

**1. Data Preprocessing and Feature Engineering**

Steps Taken:

1.Dataset Loading:

The CoNLL-2003 dataset was loaded using the datasets library from Hugging Face. This dataset contains sentences annotated with named entities like PER, ORG, LOC, etc.

2.Tokenization and Label Alignment:

* The input text was tokenized using the BERT tokenizer (bert-base-cased).
* Labels were aligned with the tokenized input to ensure that each token corresponds to the correct entity label. Special tokens (e.g., [CLS], [SEP]) and subword tokens were assigned the label -100 to ignore them during training.

3.Padding and Truncation:

Input sequences were padded or truncated to a fixed length to ensure uniform input size for the model.

4.Feature Engineering:

The tokenized input and aligned labels were converted into a format suitable for training the BERT model.

**2. Model Selection and Optimization**

Model Selection:

* The BERT model (bert-base-cased) was chosen for its strong performance on token classification tasks like NER.
* The model was fine-tuned on the CoNLL-2003 dataset using the transformers library.

Optimization Approach:

1. Hyperparameter Tuning:

Learning rate: 2e-5

Batch size: 16

Number of epochs: 3

Weight decay: 0.01

2.Evaluation Metrics:

Precision, recall, F1-score, and accuracy were used to evaluate the model's performance.

3.Training:

The model was trained using the Trainer API from Hugging Face, which handles batching, gradient updates, and evaluation.

**3. Deployment Strategy**

Flask API:

* A Flask API was created to serve the fine-tuned BERT model.
* The API has a /predict endpoint that accepts JSON input and returns recognized entities.

Basic Authentication:

* Basic authentication was implemented to secure the API. Users must provide a username and password to access the endpoint.

Dockerization:

* The application was containerized using Docker for easy deployment and scalability.

**4. API Usage Guide**

**Using Postman:**

1. Set the request method to POST.

2. Enter the URL: http://localhost:5000/predict.

3. Go to the Authorization tab, select Basic Auth, and enter:

Username: admin

Password: password

4. Go to the Body tab, select raw, and choose JSON.

5. Enter the JSON input:

{

"text": "John works at Google in New York."

}

6. Send the request

**GitHub Repository**

The complete code, including the Flask API, Dockerfile, and Jupyter Notebook for training and evaluation, is available in the following GitHub repository:

**NER with BERT - GitHub Repository**

**Conclusion**

This project demonstrates how to preprocess data, fine-tune a BERT model for NER, and deploy it as a Flask API with basic authentication. The Dockerized application can be easily deployed to production environments. The API provides a simple and secure way to extract named entities from text.