Methodology Description for Named Entity Recognition (NER) Model

1. Introduction

This document details the methodology, achievements, system specifications, methods, and results of a Named Entity Recognition (NER) model developed and evaluated using the Spacy library. The process includes data preprocessing, model training, feature engineering, and evaluation. Additionally, it outlines the hyperparameters used and any specific implementation details.

2. Achievements

The project successfully developed and evaluated an NER model capable of identifying entities with a high level of accuracy. The model was trained and tested on a structured dataset, achieving the following results:

- F1-Score: 86.3%
- Accuracy: 90%

3. Data Preprocessing

Data preprocessing ensures that the input data is in a suitable format for training and evaluating the NER model.

3.1. Data Loading

The dataset is loaded from Huggingface. Each file contains columns: id, tokens, and ner_tags.

- id: Unique identifier for each sentence or data instance.
- tokens: List of words in the sentence.
- ner_tags: Corresponding NER tags for each token.

3.2. Data Cleaning

- Removing Null Values: Ensured there were no missing values in the dataset.
- Consistency Checks: Verified that each token had a corresponding NER tag, ensuring data integrity.

3.3. Data Splitting

The dataset was split into training and test sets. The training set was used to train the model, while the test set was reserved for evaluating its performance.

4. Feature Engineering

4.1. Tokenization

The text was already tokenized, as provided in the tokens column. Each token was treated as a feature for the NER model.

4.2. Embeddings

Pre-trained word embeddings called tok2vec were used to represent tokens in a high-dimensional vector space, capturing semantic similarities between words. A transformer has been used directly as a Tok2Vec layer.

5. Model Training

The model used for NER is implemented using the spacy library.

5.1. Model Architecture

The architecture of the NER model is based on the spacy NER component, which uses a transition-based approach to predict entity boundaries and labels. pipeline:

- Transformer: used huggingface model named **microsoft/deberta-v3-base**
- Ner: For tagging and tok2vec has been used in this component

5.2. Hyperparameters

- **Batch Size**: 32
- **Epochs**: 20
- **Learning Rate**: 0.001
- **Dropout Rate**: 0.5 (the rate at which neurons are randomly set to zero during training to prevent overfitting).

5.3. Training Procedure

- **Initialize Model**: Loaded the pre-trained spacy model and added a custom NER component(specific tags added).
- **Fine-tune model**: Used the spacy training loop to update the model (microsoft/deberta-v3-base) weights based on the training data. The

optimizer was configured with the specified learning rate and other hyperparameters.

- **Evaluation**: After each epoch, the model's performance was evaluated on a validation set to monitor overfitting and adjust training as needed.

6. Evaluation

6.1. Metrics

- F1-Score: The weighted average of Precision and Recall.
- Precision
- Recall
- Accuracy

6.2. Results

- F1-Score: 86.3%
- Accuracy: 90%

7. Implementation Details

7.1. Libraries and Tools

- **Python**: The programming language used.
- **Spacy**: For model implementation and training.
- **Pandas**: For data manipulation and preprocessing.

7.2. Hardware and Runtime

The model was developed and trained on Google Colab, leveraging its powerful computational resources. The system specifications are as follows:

- System: Google Colab
- GPU: NVIDIA T4
- System RAM: 12.7 GB
- GPU RAM: 15 GB

8. Conclusion

The NER model implemented using spacy provides an efficient and effective approach for named entity recognition tasks. The preprocessing steps ensured data quality, while the use of pre-trained embeddings enhanced the model's performance. The evaluation metrics indicate the model's robustness, with an F1-Score of 78% and an accuracy of 90%.