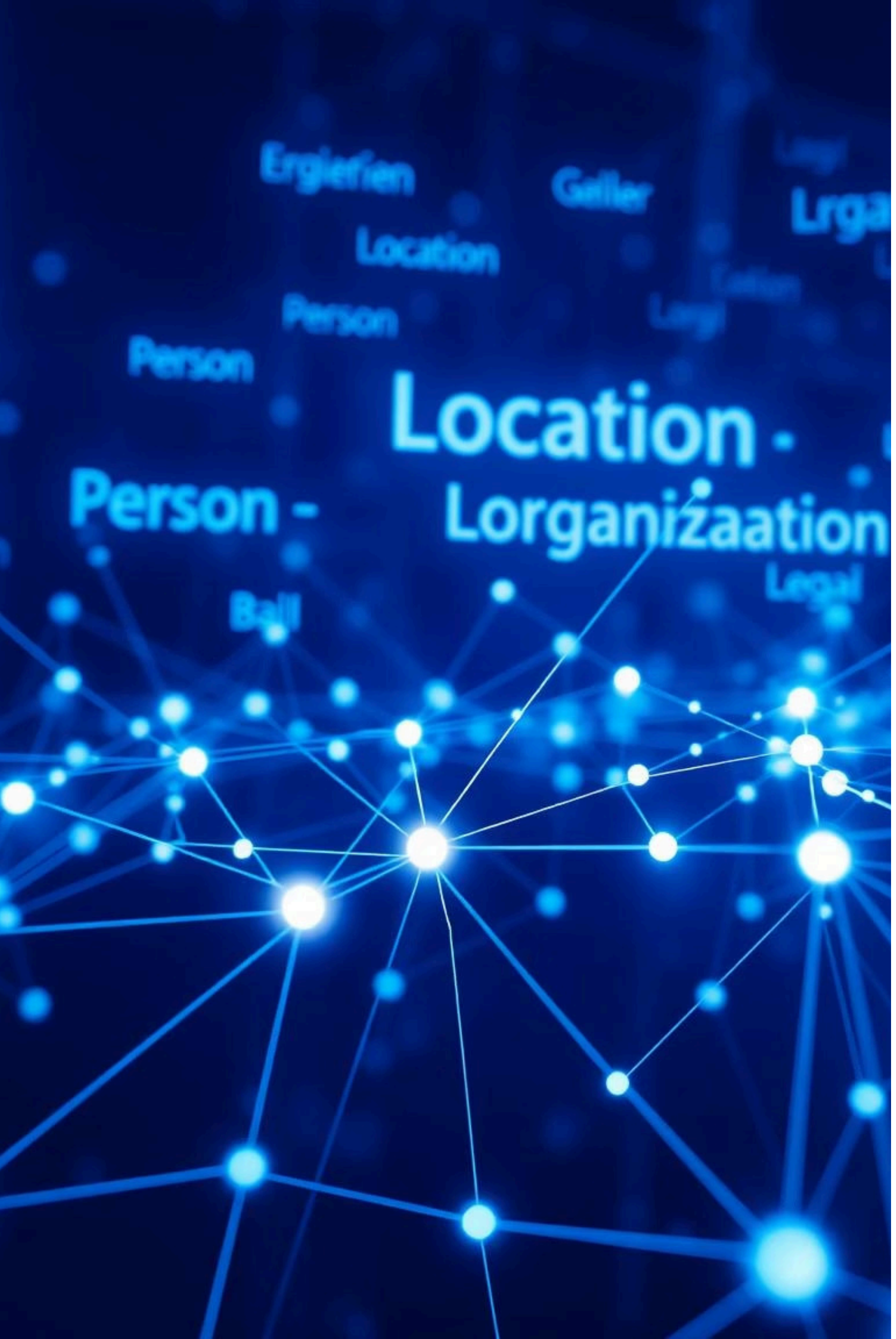


Named Entity Recognition (NER) Using BiLSTM

Objective: Automatically identify and classify named entities within text to enable more precise and meaningful text analysis.

Team Members

- **Mervat Mohammed Hassan** – ID: 20210892
- **Roa'a waleed Ahmed** – ID: 20210345
- **Nada Salah Ahmed** – ID: 20210998
- **Mirna Sherif Zayed** – ID: 20210981
- **Mohamed Ismail Hassan** – ID: 20210741
- **Mina Nadi Farag** - ID: 20210986

Project Tools & Libraries

Python

Primary programming language for development

TensorFlow & Keras

Deep learning frameworks for model building

NumPy & Pandas

Data manipulation and preprocessing libraries

Plotly & Matplotlib

Libraries for data visualization and analysis

Scikit-learn

For preprocessing and model evaluation

Livelossplot & TensorBoard

Libraries for tracking model performance

Dataset Overview

Dataset Files

- Train File (train.txt)
- Validation File (val.txt)
- Test File (test.txt)

Dataset Format

Each line in the dataset contains a word along with its associated tag. Sentences are separated by special tokens like . and $#$. For example:

```
John  B-PER
Doe   I-PER
went  O
to    O
Cairo B-LOC
```

Where:

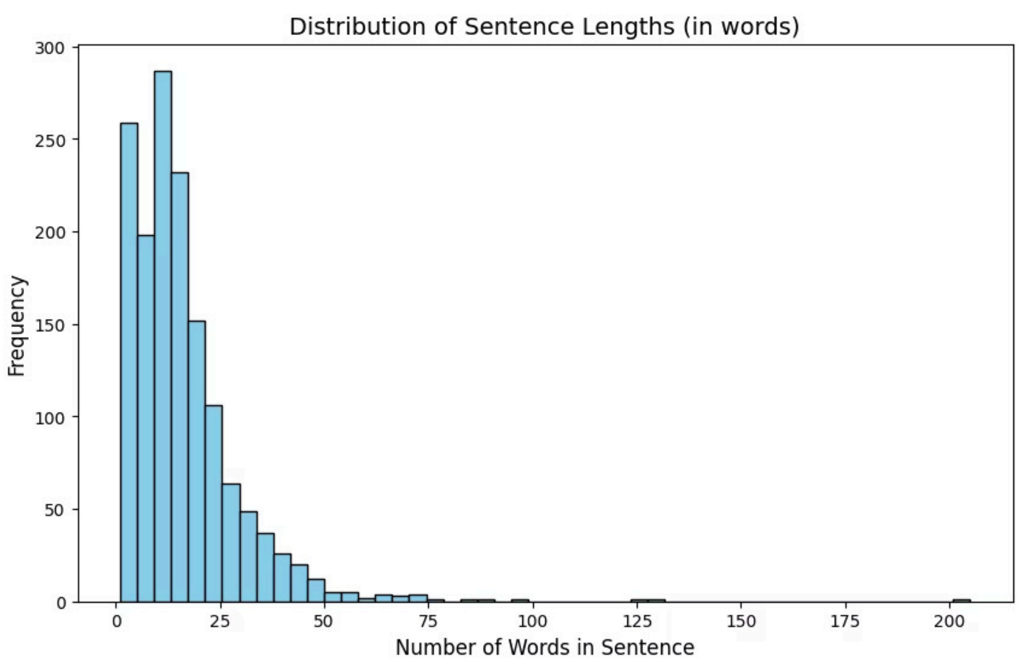
- **B-PER** stands for Beginning of Person Entity
- **I-PER** stands for Inside of Person Entity
- **O** stands for Other (non-entity word)
- **B-LOC** stands for Beginning of Location Entity

Dataset Characteristics

Sentence Length Distribution

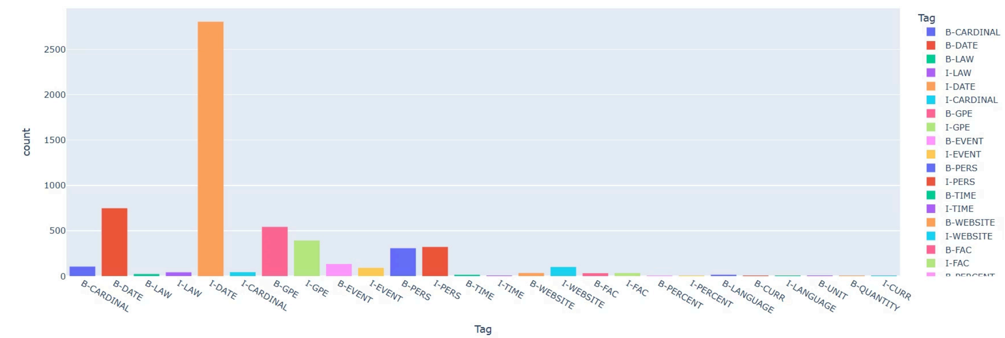
Diverse range of sentence lengths (25-75 words)

Requires specialized preprocessing and modeling to handle variability in input size.



Named Entity Distribution

Broad spectrum of entity types (PERSON, ORGANIZATION, LOCATION, etc.)



Diverse entity distribution provides a robust dataset for training a high-quality NER model.

Made with GAMMA

Preprocessing Steps for NER

1

Load and Clean Data

Data files loaded into Pandas DataFrames using pd.read_csv(). Whitespace delimiter and bad lines skipped to avoid errors.

2

Sentence Separation and Tag Handling

Dataset processed to handle sentence separation and multiple tags. Each sentence given a unique number, and words split into individual tag parts.

3

Tokenization and Padding

Words converted to integer indices, tags converted to indices. Padding applied to ensure uniform sequence lengths.

4

Data Splitting

Dataset split into 80% training and 20% testing using train_test_split from sklearn.

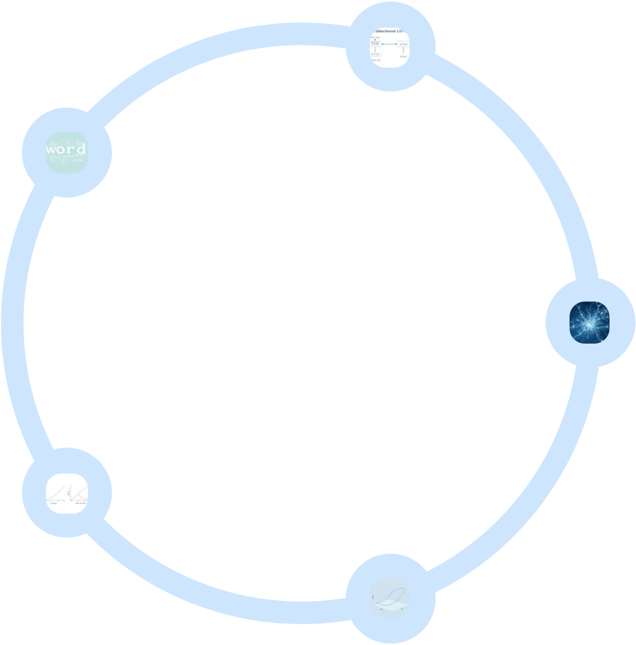
Model Architecture Overview

Embedding Layer

Transforms words into dense vector representations.

Evaluation Metrics

Measured by accuracy and ROC-AUC scores to assess performance.



Bidirectional LSTM Layer

Captures past and future context for better sequence understanding.

Dropout Layers

Spatial and recurrent dropout reduce overfitting with a rate of 0.1.

Optimization & Loss

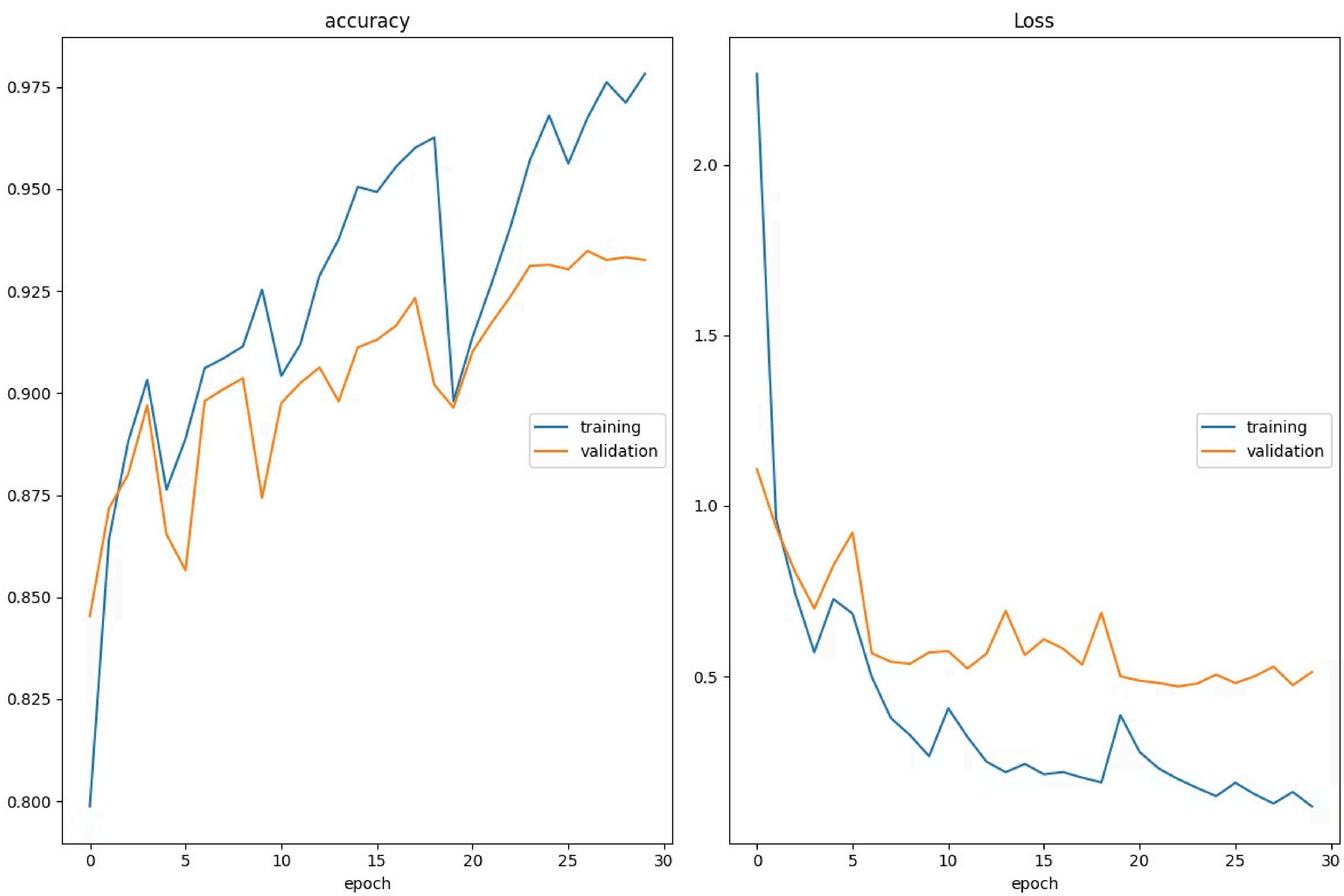
Uses Adam optimizer and sparse categorical crossentropy loss.

Training Configuration

- **Epochs:** 30 cycles for model convergence
- **Batch Size:** 32 samples per gradient update
- **Validation Split:** 20% of data reserved for validation
- **EarlyStopping:** Patience set to 7 to prevent overfitting
- **ModelCheckpoint:** Automatically saves the best performing model
- **Callbacks:** Includes LiveLossPlot and TensorBoard for monitoring progress

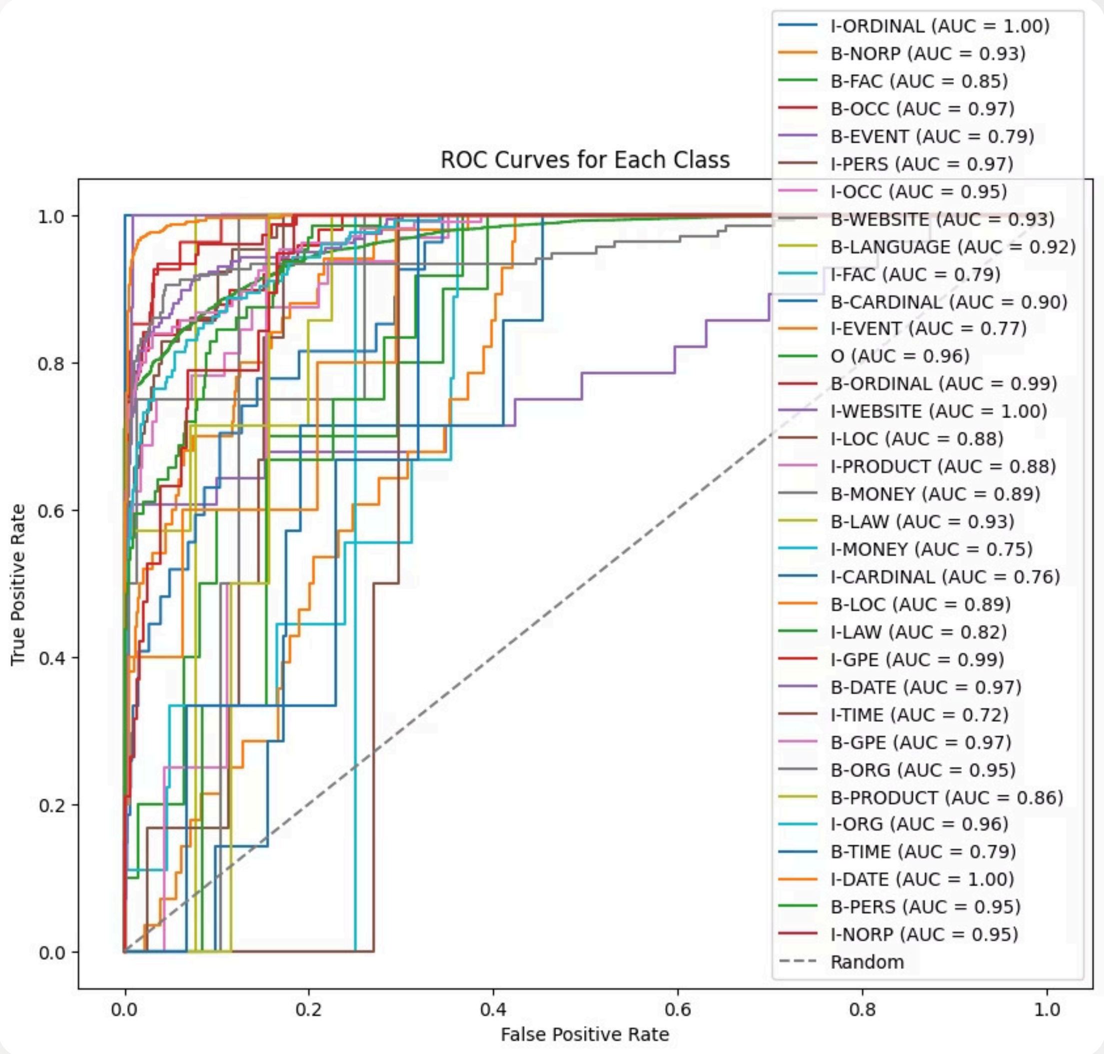
Training Performance Metrics

	Training	Test Accuracy
Accuracy	97.5%	93%
Loss	0.18	0.48 (overfitting after epoch 20)
Optimizer	Adam optimizer	
Loss Function	Sparse categorical crossentropy	
Batch Size	32	
EarlyStopping	Patience set to 3 epochs to prevent overfitting	



ROC-AUC Curve Performance

- The ROC-AUC scores provide insights into the model's performance across different classes, allowing for better understanding of its predictive capabilities.



Limitations of the BiLSTM Model

Data Imbalance: Bias towards frequent entities like Person

Model Complexity: Struggles with long dependencies and overfitting

Arabic Language: Morphological and syntactic challenges

Entity Boundaries: Hard to detect boundaries in long or nested entities

Next Steps & Improvements

Enhance Model

Explore advanced architectures and hyperparameter tuning

Expand Dataset

Include more entity types and larger annotated data

Deploy & Monitor

Integrate model into applications and track real-world performance