Slot Tagging of Natural Language Utterances

Soren Larsen
UCSC Silicon Valley Extension
Santa Clara, California
snlarsen@ucsc.edu

Abstract

This report details the implementation and evaluation of a slot-tagging model for natural language utterances using a BiLSTM model with attention. The project involved tokenization, vocabulary creation, and integration of GloVe embeddings for contextual understanding. Model performance was evaluated using F1 metrics from both sklearn and seqeval, offering insights into both token-level predictions and sequence context accuracy.

1 Introduction

The goal of this project is to solve the problem of slot tagging in natural language utterances, a critical task in natural language understanding (NLU) systems. The task is framed as a supervised sequence labeling problem, where the input is a natural language utterance and the output is a sequence of slot tags in the Inside-Outside-Beginning (IOB) format. For example, given the input sentence "Show me movies directed by Christopher Nolan," the desired output might be "O O B-movie O O B-director I-director." This project aims to develop a model that can learn these mappings effectively.

The provided dataset contains a collection of natural language utterances paired with their corresponding IOB slot tags. Descriptive statistics of the dataset include the number of unique tokens, slot tags, and distribution of utterance lengths. The input consists of tokenized sentences, and the output is a sequence of IOB slot tags. Table 1 provides a few examples from the dataset.

Input Utterance	IOB Slot Tags
"Show me movies directed by Christopher Nolan"	O O B-movie O O B-director I-director
"What are the top-rated comedies?"	O O O B-genre
"Play music by Adele"	O B-action O B-artist

Table 1: Example input-output pairs from the dataset.

2 Data Preparation

The data preparation process involved several key steps to ensure that the input was properly formatted for the model and evaluation metrics:

- **Tokenization**: Utterances were split into words to create tokenized inputs for the model. This step is essential to represent each word in the utterance as a distinct element in the sequence, allowing for more accurate modeling of the relationships between words.

- **Vocabulary Creation**: Separate vocabularies were constructed for tokens (words) and slot tags. Special tokens for padding (<PAD>) and unknown words (<UNK>) were assigned the first indices (0 and 1, respectively). This ensures consistent handling of out-of-vocabulary words and alignment of sequences during batch processing. All other tokens and tags were mapped to unique indices starting from the next available index, determined by the size of the vocabulary. This mapping enables efficient lookup during training and inference.

- **Data Splitting**: The data was divided into training and validation sets using an 80-20 split to enable model evaluation on unseen data while retaining enough examples for robust model training.

- **Custom Tag Conversion**: Tags in the B_ and I_ format were converted to IOB2-compliant B- and I- formats for compatibility with the seqeval evaluation metrics. This conversion was necessary because seqeval expects tags to follow the IOB2 standard. However, it is important to note that while this conversion was used for evaluation, the output file generated by the model retains the original underscores (B_ and I_) to align with the provided dataset's format.

- **Sequence Length Alignment**: To ensure consistency, the length of the tag predictions was constrained to match the length of the corresponding tokens. This step was crucial for maintaining alignment between the token sequences and their associated slot tags, avoiding any discrepancies during training and evaluation.

3 Embedding Methods

Word embeddings provide a dense vector representation of words, capturing semantic similarities and enabling the model to learn context-rich features. For this project, GloVe (Global Vectors for Word Representation) embeddings were chosen due to their widespread usage and effectiveness in representing word semantics. Specifically, the `glove-wiki-gigaword-100` embeddings with a 100-dimensional vector space were initially used. The choice of GloVe embeddings was motivated by their ability to capture global word co-occurrence statistics, resulting in dense vector representations that retain semantic and syntactic relationships.

The pre-trained GloVe vectors were integrated into the model to enrich token embeddings, allowing the BiLSTM to leverage external knowledge learned from large corpora. This enhancement proved useful for capturing contextual nuances that would be challenging to learn solely from the project-specific dataset. As an example, words like "movie" and "film" may appear in different utterances with similar contexts; GloVe embeddings ensure these words have similar vector representations, thereby facilitating accurate slot tagging. To explore potential improvements, the embedding dimension was increased to 200 by using `glove-wiki-gigaword-200` embeddings. However, this resulted in only a marginal improvement in the F1 score, changing by approximately 0.01. The minimal performance gain, coupled with increased computational complexity, led to the decision to retain the 100-dimensional embeddings for subsequent experiments. This outcome highlights that increasing embedding dimensionality does not always translate to significant performance gains and can introduce diminishing returns depending on the dataset and model architecture.

4 Models

In this project, two primary models were explored for the task of slot tagging: an LSTM-based model and a BiLSTM model with an attention mechanism. Both models were chosen due to their capabilities in capturing sequential and contextual dependencies in natural language, which are essential for accurate slot tagging.

4.1 Model Evaluation Preface

In evaluating the performance of the models described in this paper, accuracy was computed by comparing the predicted outputs on a provided test dataset to a hidden ground truth, treated as a "black box." This meant that while accuracy and performance metrics were derived from these comparisons, the exact slot tags and labels within the test data were not visible during model training or evaluation. This approach was designed to maintain unbiased model assessments but also limited the opportunity to inspect specific incorrect classifications directly. Consequently, the reported metrics primarily reflect how well the model predictions aligned with the hidden ground truth without offering granular insight into individual errors. The selection of models submitted for evaluation was based on those demonstrating high F1 scores and low loss values during training and validation phases.

4.2 LSTM Model

Long Short-Term Memory (LSTM) networks were specifically designed to address the vanishing and exploding gradient problems that occur during training of traditional Recurrent Neural Networks (RNNs) by using memory cells and gates to control the flow of information. This capability allows LSTMs to capture long-term dependencies, making them particularly well-suited for modeling sequential data, such as language tasks. As outlined in (2), LSTM-based models have demonstrated strong performance in various sequential tasks, including large vocabulary speech recogni-

tion. By maintaining relevant context across longer sequences, LSTMs can effectively capture dependencies between words in an utterance, which is critical for accurate slot tagging.

4.3 BiLSTM with Attention

While LSTMs are effective at capturing past dependencies, a Bidirectional LSTM (BiLSTM) enhances this capability by processing sequences in both forward and backward directions. This allows the model to access both past and future context, providing a richer representation of the input sequence. The use of BiLSTMs is particularly beneficial for tasks where context on both sides of a token can influence slot tagging decisions. The BiLSTM model's architecture in this project builds on the advantages of bidirectionality by incorporating an attention mechanism to selectively focus on important tokens within the sequence.

Attention mechanisms have proven effective in various sequence-to-sequence tasks by dynamically weighting the importance of tokens based on their relevance to the task at hand. Liu and Lane (1) demonstrated that attention mechanisms can significantly enhance the performance of RNN-based models for joint intent detection and slot filling by focusing on relevant words in a given utterance. By applying attention, our BiLSTM model can more effectively identify important tokens and contextual cues, leading to improved slot tagging accuracy.

The BiLSTM with attention model in this project includes a pre-trained embedding layer, a bidirectional LSTM network, an attention layer that computes importance scores for each token, and a fully connected output layer. This combination allows the model to selectively emphasize relevant tokens while considering both past and future context, resulting in more accurate predictions.

4.4 LSTM Model

The Long Short-Term Memory (LSTM) network is a type of recurrent neural network (RNN) designed to capture long-term dependencies in sequential data. Unlike standard RNNs, LSTMs use memory cells and gating mechanisms to manage the flow of information across time steps, addressing issues such as the vanishing gradient problem.

- **LSTM Layer**: The model consists of a single-layer LSTM that processes input tokens sequentially. The LSTM cell architecture includes an input gate, forget gate, and output gate, which collaboratively control the retention and updating of relevant contextual information for each time step.
- **Fully Connected Layer**: The output for each token from the LSTM layer is passed through a fully connected layer, mapping it to the predefined set of slot tags, thereby generating predictions for each word in the input sequence.

Key Experimentation and Results The main experimental effort focused on handling excess predictions generated by the LSTM model. This involved the manipulation and trimming of predictions to ensure alignment with input token lengths. By refining the predicted sequences to match the length of the input utterances, substantial improvements were observed in model accuracy.

Performance Improvements The trimming and manipulation approach boosted the testing accuracy of the LSTM model from 23.79% to 55.81%. Correspondingly, the F1 scores improved from 0.89 to 0.92, demonstrating the impact of these adjustments on model performance. This highlights the importance of ensuring that the output sequence length matches the input sequence length, as discrepancies can lead to significant drops in accuracy and F1 scores.

Training Details The LSTM model was trained using the following configuration:

- **Loss Function**: Cross-entropy loss, suitable for multi-class classification tasks.
- **Optimizer**: Adam optimizer, chosen for its adaptive learning rate capabilities, which enhance convergence speed.
- **Hyperparameters**:
  - Learning Rate: 0.001
  - Hidden Dimension: 128
  - Number of LSTM Layers: 1
  - Batch Size: 32
  - Dropout Rate: No dropout was applied for this model.

During training, padding tokens were ignored in the loss computation to prevent them from skewing the gradient updates. The model's weights were updated iteratively using mini-batches, and performance on validation data was monitored to prevent overfitting.

4.5 BiLSTM with Attention Model

To capture richer contextual information, we extended the LSTM to a Bidirectional LSTM (BiLSTM) architecture. BiLSTMs process input sequences in both forward and backward directions, enabling the model to consider both past and future context simultaneously. This bidirectional approach enhances slot tagging performance by providing a holistic view of the input (1).

Initial BiLSTM Experimentation The first iteration of the BiLSTM model retained a simple structure, aiming to assess the potential gains from bidirectional context without additional complexity:

- **Model Configuration**:
  - **No Embedding Layer**: Unlike subsequent iterations that incorporated embeddings, this initial BiLSTM model directly processed tokenized inputs without any embedding transformation.
  - **Single BiLSTM Layer**: The architecture consisted of a single layer of BiLSTM cells, preserving the sequential nature of the input data while enhancing context capture through bidirectional processing.
  - **Hyperparameters**: Key hyperparameters, such as learning rate, batch size, and optimizer configuration, were kept consistent with those used for the previous LSTM model. This included a learning rate of 0.001, a hidden dimension of 128, a batch size of 32, and the use of the Adam optimizer.
  - **No Dropout**: No regularization through dropout was applied, maintaining a similar configuration to the baseline LSTM model for a fair comparison.
- **Performance Comparison**:
  - **Accuracy Improvement**: The BiLSTM model exhibited a marked increase in testing accuracy, improving from the LSTM baseline of 55.81% to 68.16%. This demonstrates the power of incorporating bidirectional context, allowing the model to better understand the relationships between tokens across the input sequence.
  - **F1 Score Increase**: The F1 score similarly improved, rising from 0.92 to 0.94. This reflects greater precision and recall, suggesting the BiLSTM's increased capacity for capturing both token-level predictions and overall sequence structure.
  - **Loss Reduction**: One of the most telling metrics was the reduction in loss. The training loss dropped from 0.132 (LSTM) to 0.053, while the validation loss decreased from 0.281 to 0.221. This significant improvement in loss values indicates that the BiLSTM model was better at fitting the training data while maintaining improved generalization on the validation set. This result underscores the effectiveness of bidirectional processing for capturing richer contextual dependencies in slot tagging tasks.

This initial BiLSTM experiment highlighted the potential for leveraging bidirectional LSTM networks to improve slot tagging performance significantly, even without the inclusion of embeddings or additional architectural enhancements. Further experiments aimed to build on these gains through more advanced configurations, as discussed in the following sections.

2-Layer BiLSTM with Dropout Experimentation To further explore the capabilities of the BiLSTM architecture, a two-layer BiLSTM model was introduced with the inclusion of dropout regularization. The intent was to assess the impact of deeper network architecture and regularization on model performance:

- **Model Configuration**:
  - **Two BiLSTM Layers**: The architecture was extended to include two layers of BiLSTM cells, allowing for deeper processing and potentially enhanced contextual understanding of the input sequences.
  - **Dropout Regularization**: A dropout rate of 0.3 was applied to the model, introducing stochastic regularization and reducing overfitting risk by randomly deactivating a fraction of neurons during training.
  - **Hyperparameters**: Key hyperparameters, such as learning rate, batch size,

hidden dimension (128), and optimizer (Adam), were kept consistent with the previous BiLSTM iteration for comparative purposes.

- **Performance Comparison**:
  - **Accuracy Improvement**: The 2-layer BiLSTM model exhibited a further increase in testing accuracy compared to the single-layer BiLSTM. Accuracy rose to approximately 73.18%, demonstrating the benefits of deeper network architecture and dropout regularization.
  - **F1 Score Stability**: The F1 score remained relatively stable around 0.94 across different dropout rates, suggesting that while dropout improved generalization, it did not significantly alter token-level precision and recall.
  - **Minor Changes with Dropout Adjustment**: Adjusting the dropout rate to 0.1 resulted in a marginal accuracy decrease to 73.05%, indicating that regularization had a stabilizing effect on the model's performance.

This experiment demonstrates that the incorporation of multiple BiLSTM layers and dropout can enhance model performance by reducing overfitting and capturing richer contextual dependencies, though the impact on precision and recall as measured by the F1 score remained consistent.

Epoch	Training Loss	Validation Loss	F1 Score (sklearn)
1	1.459	0.976	0.665
2	0.848	0.705	0.762
3	0.619	0.531	0.842
4	0.473	0.414	0.874
5	0.362	0.341	0.899
6	0.289	0.294	0.916
7	0.230	0.276	0.916
8	0.192	0.254	0.926
9	0.157	0.244	0.935
10	0.123	0.237	0.936
11	0.120	0.225	0.939
12	0.095	0.205	0.942
13	0.085	0.198	0.949

Table 2: Performance metrics for the 2-layer BiLSTM model with dropout (0.3) across 13 epochs.

BiLSTM with Token-Level Attention Experimentation In this experiment, token-level attention was introduced to the BiLSTM model to explore whether selectively focusing on important tokens in the input sequence could enhance slot-tagging performance. Initially, a 2-layer BiLSTM architecture with a dropout rate of 0.1 was used. However, this configuration showed a slight decrease in performance compared to previous iterations. Consequently, further adjustments were made by reducing the model to a single layer and removing the dropout regularization to allow the model to learn more effectively without the constraints of regularization.

- **Model Configuration**:
  - **BiLSTM Architecture with Token-Level Attention**: The BiLSTM model was augmented with an attention mechanism that computed attention weights over tokens to emphasize key parts of the input sequence during slot tagging.
  - **2 Layers, Dropout 0.1 (Initial Attempt)**: The initial configuration had two layers of BiLSTM and applied a dropout rate of 0.1.
  - **Single Layer, No Dropout (Adjusted Configuration)**: To improve performance, the model was simplified to a single BiLSTM layer with no dropout.
  - **Hyperparameters**: The same hyperparameter values used in previous experiments were retained, including a learning rate of 0.001, a hidden dimension of 128, a batch size of 32, and the use of the Adam optimizer.
- **Performance Metrics**:
  - **Accuracy Drop**: The accuracy decreased to 72.68%, indicating a slight decline in the model's performance compared to previous BiLSTM iterations without attention.
  - **F1 Score Trends**: Detailed performance metrics for training loss, validation loss, and F1 scores across 34 epochs are presented in Table 3 (see Appendix).

Experimentation with Custom Cross-Entropy Loss and 0.1 Smoothing To further refine the model's performance, a custom cross-entropy loss

function with 0.1 smoothing was introduced. The purpose of this approach was to mitigate overconfidence in predictions by redistributing some probability mass from the maximum prediction, thereby promoting better generalization and potentially more stable learning dynamics.

Despite expectations for improved results, the performance changes were marginal. While the F1 score on the sklearn scale showed a slight improvement, increasing to 0.934, the overall loss metrics were not as favorable as anticipated. Specifically:

- **Loss Metrics**: The training loss increased to 0.364, and the validation loss rose to 0.432, both higher than previous configurations without smoothing.
- **Accuracy Impact**: When evaluated on the test dataset, the model exhibited minimal change in accuracy, recording a score of 72.24%. This small decline suggests that the smoothing technique had a limited effect on improving the model's overall performance for the slot-tagging task.

The slight increase in the F1 score, contrasted with higher loss values and a marginal dip in accuracy, indicates that the 0.1 smoothing approach introduced subtle regularization benefits but did not significantly alter the model's generalization capabilities. This experiment highlights the nuanced impact of smoothing on slot-tagging tasks, suggesting that while it may promote more balanced probability distributions, its effect on overall model performance can be minimal when applied in isolation.

BiLSTM with GloVe Embeddings and Learning Rate Tuning The next major iteration introduced GloVe embeddings to the BiLSTM model, aiming to enhance the semantic understanding of tokens. The GloVe embeddings provided a rich contextual representation that allowed the model to better capture relationships between words in the input sequences. The model also experimented with different learning rates to further optimize training dynamics.

- **Model Configuration**:
  - **GloVe Embeddings**: The GloVe embeddings used were pre-trained on the `glove-wiki-gigaword-100` dataset, providing a 100-dimensional embedding space. This allowed for leveraging semantic relationships pre-learned from a large corpus, enriching token representations.
  - **Hyperparameters and Architecture**: A BiLSTM model with one layer and no dropout was utilized to initially assess the impact of embeddings. Experiments with two different learning rates (0.001 and 0.01) were conducted to observe their influence on convergence and generalization.
- **Performance Metrics**:
  - **Accuracy Improvement**: The introduction of GloVe embeddings led to a further improvement in accuracy, reaching 74.18% on the test set. This increase highlights the utility of using pre-trained word vectors for contextual understanding in slot-tagging tasks.
  - **Loss and F1 Score Analysis**: The training and validation losses, as well as F1 scores, are detailed in the appendix (Table 4). Key observations include:
    - A learning rate of 0.01 yielded a final F1 score of 0.950 using sklearn's metric, compared to 0.946 for a learning rate of 0.001.
    - The training and validation losses for the 0.01 learning rate converged to 0.042 and 0.216 respectively, while the 0.001 learning rate achieved losses of 0.078 and 0.225.

This experiment demonstrated that pre-trained embeddings significantly boost model performance by providing contextual richness to the input representations. Additionally, careful tuning of the learning rate can lead to further gains in model effectiveness, as evidenced by the comparison between 0.001 and 0.01 learning rates.

Final BiLSTM Model with GloVe Embeddings and Optimized Hyperparameters The final iteration of the BiLSTM model incorporated GloVe embeddings to enhance contextual understanding and included optimized hyperparameters based on previous experimental findings. This model configuration aimed to maximize both accuracy and F1 score for slot tagging.

- **Model Configuration**:
  - **GloVe Embeddings**: The model utilized the `glove-wiki-gigaword-100` pre-trained embeddings, offering 100-dimensional dense representations to enrich the input token's semantic context.
  - **Architecture**: A single-layer BiLSTM model was used, coupled with token-level attention. No dropout regularization was applied in this configuration, aligning with observations that dropout tended to limit model learning in earlier experiments.
  - **Hyperparameters**: The model was trained with a learning rate of 0.001 using the Adam optimizer for efficient weight updates.
- **Performance Metrics and Observations**:
  - **Accuracy and F1 Score Improvement**: This final configuration achieved a test set accuracy of 75.37%. The sklearn F1 score reached 0.950, while the seqeval F1 score peaked at 0.839. The use of seqeval for F1 score evaluation was particularly valuable for this task because seqeval provides sequence-based evaluation metrics, capturing context-sensitive relationships within the token predictions. Unlike token-level metrics, which only evaluate individual predictions, seqeval considers the correctness of sequences as a whole. This is critical for slot tagging tasks, where maintaining the structure of predicted tags (e.g., beginning and inside tags) is just as important as their individual accuracy.
  - **Training and Validation Loss**: The model demonstrated effective convergence, as evidenced by the decreasing trend of training and validation losses. Final values for training and validation losses were 0.042 and 0.216, respectively.
  - **Detailed Epoch Results**: Detailed performance metrics, including epoch-wise training and validation losses along with F1 scores, are provided in the appendix (Table 5).

This final configuration highlighted the effectiveness of incorporating pre-trained embeddings and attention mechanisms in BiLSTM-based models for slot tagging. The significant gains in F1 scores demonstrate the model's ability to accurately capture and generalize complex token dependencies in natural language utterances.

5 Conclusion and Reflection

5.1 Key Findings and Contributions

In this project, we explored various models and configurations for slot tagging of natural language utterances, ranging from LSTM and BiLSTM architectures to BiLSTM models with attention mechanisms. The integration of GloVe embeddings significantly enhanced the semantic understanding of tokens, leading to substantial improvements in performance. Key findings include:

- BiLSTM models consistently outperformed LSTM models due to their ability to incorporate both past and future context, resulting in a more comprehensive understanding of token dependencies.
- Token-level attention mechanisms allowed the model to selectively focus on key parts of the input sequence, though fine-tuning was necessary to maximize performance gains.
- GloVe embeddings enriched token representations, contributing to a final accuracy of 75.37% and a high F1 score on both sklearn and seqeval metrics. The use of pre-trained embeddings proved essential for achieving high accuracy with limited data.
- Hyperparameter tuning, particularly for learning rate and dropout, played a crucial role in optimizing model performance while minimizing overfitting.

5.2 Limitations and Future Work

While this project demonstrated significant gains in slot tagging accuracy and F1 scores, certain limitations remain. The model's reliance on a limited dataset may hinder its generalization to broader contexts or domain-specific applications. Moreover, hyperparameter tuning was conducted manually and relied on empirical observations, leaving room for more systematic approaches.

Future work could include:

- **Automated Hyperparameter Optimization**: Implementing a grid search or other automated optimization techniques to systematically explore combinations of hyperparameters, such as learning rate, hidden dimensions, number of layers, and dropout rates. This approach could yield an even more robust and optimized model configuration.

- **Exploration of Transformer-Based Models**: Given their success in sequence-to-sequence tasks, models such as BERT or Transformer-based architectures could further enhance slot tagging performance through their ability to capture long-range dependencies and contextual nuances more effectively.

- **Data Augmentation and Expansion**: Expanding the dataset or leveraging data augmentation techniques could improve the model's generalization and adaptability to unseen data.

- **Domain Adaptation**: Applying transfer learning techniques to adapt the trained model to new domains or datasets with minimal additional training could further demonstrate its utility and versatility.

Overall, this project illustrates the challenges and opportunities inherent in slot tagging tasks, offering a strong foundation for continued research and development.

References

[1] Bing Liu and Ian Lane. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. *arXiv preprint arXiv:1609.01454*, 2016. https://arxiv.org/abs/1609.01454.

[2] Hasim Sak, Andrew Senior, and Francoise Beaufays. Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition. *arXiv preprint arXiv:1402.1128*, 2014. https://arxiv.org/abs/1402.1128.

A Appendix

Epoch	Training Loss	Validation Loss	F1 Score (sklearn)
1	2.290	1.351	0.590
2	1.167	1.121	0.590
3	1.012	1.013	0.590
4	0.942	0.948	0.591
5	0.865	0.978	0.591
6	0.808	0.972	0.591
7	0.782	0.962	0.591
8	0.810	0.936	0.591
9	0.757	0.845	0.608
10	0.722	0.811	0.668
11	0.687	0.771	0.711
12	0.640	0.729	0.751
13	0.595	0.669	0.777
14	0.549	0.625	0.789
15	0.509	0.591	0.816
16	0.479	0.566	0.836
17	0.453	0.542	0.852
18	0.428	0.512	0.865
19	0.404	0.474	0.877
20	0.382	0.443	0.885
21	0.359	0.414	0.888
22	0.340	0.396	0.890
23	0.319	0.378	0.892
24	0.299	0.362	0.894
25	0.279	0.346	0.896
26	0.260	0.330	0.898
27	0.243	0.315	0.900
28	0.225	0.299	0.902
29	0.208	0.284	0.904
30	0.192	0.270	0.906
31	0.177	0.258	0.908
32	0.163	0.246	0.910
33	0.150	0.234	0.912
34	0.137	0.223	0.914

Table 3: Performance metrics for the BiLSTM model with token-level attention, 2 layers, and dropout (0.1).

Epoch	Training Loss	Validation Loss	F1 Score (sklearn)
1	2.116	1.249	0.590
2	1.092	1.029	0.590
3	0.968	0.959	0.590
4	0.902	0.915	0.590
5	0.864	0.882	0.591
6	0.813	0.844	0.591
7	0.778	0.806	0.618
8	0.746	0.774	0.618
9	0.710	0.738	0.751
10	0.656	0.688	0.778
11	0.606	0.641	0.810
12	0.565	0.602	0.834
13	0.525	0.562	0.856
14	0.485	0.521	0.878
15	0.455	0.487	0.890
16	0.425	0.459	0.896
17	0.400	0.432	0.900
18	0.376	0.406	0.908
19	0.356	0.385	0.912
20	0.338	0.367	0.916
21	0.321	0.351	0.920
22	0.305	0.334	0.922
23	0.289	0.319	0.926
24	0.275	0.304	0.930
25	0.260	0.289	0.934
26	0.245	0.275	0.936
27	0.230	0.261	0.938
28	0.216	0.247	0.940
29	0.202	0.234	0.942
30	0.189	0.222	0.944
31	0.177	0.210	0.946
32	0.166	0.198	0.946
33	0.156	0.187	0.948
34	0.147	0.177	0.948
35	0.138	0.167	0.950

Table 4: Performance metrics for the BiLSTM model with GloVe embeddings using a learning rate of 0.01 across 35 epochs.

Epoch	Training Loss	Validation Loss	Sklearn F1	Seqeval F1
1	2.116	1.249	0.590	0.000
2	1.092	1.029	0.590	0.000
3	0.968	0.959	0.590	0.000
4	0.907	0.913	0.590	0.000
5	0.868	0.881	0.591	0.000
6	0.834	0.866	0.591	0.000
7	0.807	0.837	0.591	0.000
8	0.776	0.802	0.601	0.000
9	0.748	0.780	0.618	0.003
10	0.713	0.733	0.712	0.170
11	0.656	0.686	0.758	0.311
12	0.586	0.594	0.807	0.331
13	0.515	0.542	0.820	0.366
14	0.465	0.521	0.818	0.357
15	0.429	0.488	0.833	0.395
16	0.403	0.458	0.838	0.403
17	0.365	0.409	0.866	0.480
18	0.315	0.367	0.897	0.631
19	0.266	0.340	0.900	0.645
20	0.231	0.312	0.912	0.672
21	0.205	0.299	0.918	0.704
22	0.179	0.285	0.926	0.737
23	0.160	0.269	0.931	0.749
24	0.142	0.259	0.931	0.745
25	0.126	0.246	0.937	0.768
26	0.112	0.247	0.939	0.777
27	0.103	0.237	0.943	0.800
28	0.091	0.234	0.944	0.807
29	0.084	0.227	0.945	0.808
30	0.078	0.225	0.946	0.816
31	0.071	0.225	0.948	0.824
32	0.066	0.225	0.944	0.810
33	0.062	0.222	0.945	0.811
34	0.058	0.218	0.947	0.822
35	0.056	0.223	0.946	0.816
36	0.052	0.220	0.946	0.823
37	0.048	0.216	0.947	0.820
38	0.046	0.218	0.949	0.828
39	0.043	0.217	0.950	0.833
40	0.042	0.216	0.950	0.839

Table 5: Performance metrics for the final BiLSTM model with GloVe embeddings and optimized hyperparameters.