GRUNER: GRU-based Named Entity Recognition Model

Team: Dev Patel
Hasant Asalam
Ravi Pandey

Date: 05/08/2025

University of Nebraska Omaha

Introduction

Introduction

- Named Entity Recognition (NER) is the task of identifying and classifying entities in unstructured text.
- Entities can be locations, organizations, etc. In biomedical domain, entities can be genes, proteins, or diseases.
- This project focuses on biomedical NER using the **CRAFT dataset**, a collection of full-text biomedical articles annotated with domain-specific concepts.
- The final output is a trained GRU-based NER model, based on BioBERT embeddings.

Objective

Objective

- Generate IOB-tagged data by parsing Gene Ontology annotations from the CRAFT dataset's XML files.
- Develop a GRU-based Named Entity Recognition (NER) model that utilizes BioBERT embeddings for contextual understanding of biomedical text.
- Train and validate the model on the prepared IOB data to predict tags for word tokens.
- Experiment with hyperparameters (e.g., GRU hidden size, learning rate) to optimize model performance.
- Evaluate the model using key NER metrics: Precision, Recall, and F1 Score.

Methodology

Methodology

- Dataset
  - CRAFT GO annotations
- Data Preprocessing Pipeline
  - Word-Tag tuple
- Model Architecture and Configuration
  - A GRU-NER model
- Training Strategy and Enhancements
  - Trained model
- Evaluation and Performance Insights

Dataset

- The Colorado Richly Annotated Full-Text (CRAFT) Corpus
- A richly annotated biomedical corpus.
- Contains full-text biomedical journal articles with annotations for molecular function, biological process, and cellular component.

```
<annotation>
    <mention id="GO_BP_2016_02_16_Instance_28421" />
    <annotator id="GO_BP_2016_02_16_Instance_10000">Mike Bada,
    University of Colorado Anschutz Medical Campus</annotator>
    <span start="2556" end="2568" />
    <span start="2581" end="2590" />
    <spannedText>formation of ... complexes</spannedText>
  </annotation>
Note: Here, the "..." indicates a discontinuous annotation.
```

Data Preprocessing Pipeline

- Annotation Extraction
  - For each ontology XML file, we parse every element and collect its spans in a list of 4-tuples (start, end,spannedText, nextStart)
- Tokenization and Alignment
  - Used spaCy for sentence and token segmentation.
  - For each token, we computed its character-based start and end positions and match these against our span tuples.
  - When a discontinuous annotation is encountered, the words between it's tagged components are dropped from the final result.

Data Preprocessing Pipeline

- Overlap Resolution
  - When two annotations overlap in their character offsets, we discard the smaller span (shorter character length) to avoid conflicting labels.
- Output
  - For each input file, we produce a corresponding output file containing (word, tag) tuples.

```
def create_iob_tags_discontinuous(
    text: str,
    spans: List[Tuple[int,int,str,int]]
) -> List[List[Tuple[str,str]]]:
    """
    text: full document
    spans: list of (start, end, spanned_text, next_start)

    - Drops any span fully contained in a larger one.
    - For discontinuous spans (next_start > 0), skips tokens in [end, next_start).
    - Emits B at the start of each new annotation, I for inside, O otherwise.
    """
```

Data Preprocessing Pipeline

- Sample Output

```
IOB output for 15550985:
[['A', 'O'], ['Chemoattractant', 'O'], ['Role', 'O'], ['for', 'O'], ['NT-3', 'O'], ['in', 'O'], ['Proprioceptive', 'B'], ['Axon', 'B'], ['Guidance', 'I'],
[['Deletion', 'O'], ['of', 'O'], ['the', 'O'], ['proapoptotic', 'O'], ['gene', 'O'], ['Bax', 'O'], ['in', 'O'], ['NT-3', 'O'], ['knockout', 'O'], ['mice',
[['TrkC', 'O'], ['-', 'O'], ['positive', 'O'], ['peripheral', 'O'], ['and', 'O'], ['central', 'O'], ['axons', 'B'], ['from', 'O'], ['dorsal', 'O'], ['root'
[['Peripherally', 'O'], [',', 'O'], ['muscle', 'O'], ['spindles', 'O'], ['are', 'O'], ['absent', 'O'], ['and', 'O'], ['TrkC', 'O'], ['-', 'O'], ['positive'
[['Centrally', 'O'], [',', 'O'], ['proprioceptive', 'B'], ['axons', 'B'], ['branch', 'O'], ['in', 'O'], ['ectopic', 'O'], ['regions', 'O'], ['of', 'O'], ['
[['In', 'O'], ['vitro', 'O'], ['assays', 'O'], ['reveal', 'O'], ['chemoattractant', 'O'], ['effects', 'O'], ['of', 'O'], ['NT-3', 'O'], ['on', 'O'], ['dors
[['Our', 'O'], ['results', 'O'], ['show', 'O'], ['that', 'O'], ['survival', 'O'], ['factor', 'O'], ['NT-3', 'O'], ['acts', 'O'], ['as', 'O'], ['a', 'O'], [
[['Introduction', 'O'], ['Neurotrophin-3', 'O'], ['(', 'O'], ['NT-3', 'O'], [')', 'O'], ['is', 'O'], ['a', 'O'], ['key', 'O'], ['requirement', 'O'], ['for'
[['Mice', 'O'], ['deficient', 'O'], ['in', 'O'], ['NT-3', 'O'], [',', 'O'], ['its', 'O'], ['tyrosine', 'O'], ['kinase', 'O'], ['receptor', 'O'], [',', 'O']
[['Klein', 'O'], ['et', 'O'], ['al', 'O'], ['.', 'O'], ['1994', 'O'], [';', 'O'], ['Tessarollo', 'O'], ['et', 'O'], ['al', 'O'], ['.', 'O'], ['1994', 'O'],
Processing 97 files...

100%|██████████| 97/97 [01:49<00:00,  1.13s/it]
Done!
```

Data Preprocessing Pipeline

- Training and validation split
  - Data is randomly split 80% for training and 20% for validation before training the model.
- Data normalization or filtering
  - We filter out the dataset based on I and B tags in any sentence.
  - Two filters are used
    - Four number of tags
    - Eight number of tags

Model Architecture and Configuration

- Bidirectional GRU–based sequence labeling model.
- Architecture:
  - Embedding layer:
    - Maps each token to a dense vector
  - Bidirectional GRU Layer:
    - Hidden size set by hyperparameter.
    - Captures forward and backward contextual dependencies.

Model Architecture and Configuration

- Architecture:
  - Activation:
    - Two variants used
      - Adam-activated GRU: standard GRU update with Adam optimizer
      - ReLU-activated GRU: applies a ReLU after the GRU outputs to improve gradient flow and stability
  - Linear Output Layer
    - Projects the bidirectional GRU hidden states to the IOB tag space, producing token-level classification logits.

Model Architecture and Configuration

- Hyperparameter Settings:
  - Hidden dimensions: 64, 128, 256, 512, 1024, 2048, 4096
  - Learning rates: 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.09
  - Batch size: 32, 64, 128
  - Number of epochs: 1000, 2000, 3000
  - Optimizers:
    - Standard Adam
    - Adam
    - SDG

Training Strategy and Enhancements

- Grid Search:
  - Initially trained with a lower number of hidden dimensions (64 to 512)
  - Increased from 1024 to 4096 hidden dimensions to get better performance scores.
  - Learning rate 0.09 added for final training runs.
- Grid search was done to identify the optimal combination of hyperparameters
- Goal was to enhance the performance of the model on unseen data.

Training Strategy and Enhancements

- Data Enhancements:
  - Improved normalization and padding of input sequences.
  - Accounted for discontinuous and overlapping annotations. This showed a decreased F1 score but increased recall.
- Training Monitoring
  - Tracked loss, precision, recall, and F1 score at each configuration
  - Evaluated effect of ReLU on model convergence.

Results

Evaluation Approach

- **Precision**
  - Proportion of correctly predicted entities out of all predicted entities.
- **Recall**
  - Proportion of correctly predicted entities out of all actual entities.
- **F1 Score**
  - Harmonic mean of precision and recall - balances false positives and false negatives.
- **Two Evaluation Schemes**
  - **Fine-Grained Evaluation**:
    - Treats **B (Beginning), I (Inside), O (Outside)** as distinct classes.
    - Evaluates the model's ability to classify exact entity positions.
  - **Binary Grouped Evaluation**:
    - Groups **B & I as IB**, and evaluates against **O**.
    - Emphasizes detection of entity vs. non-entity rather than exact boundaries.

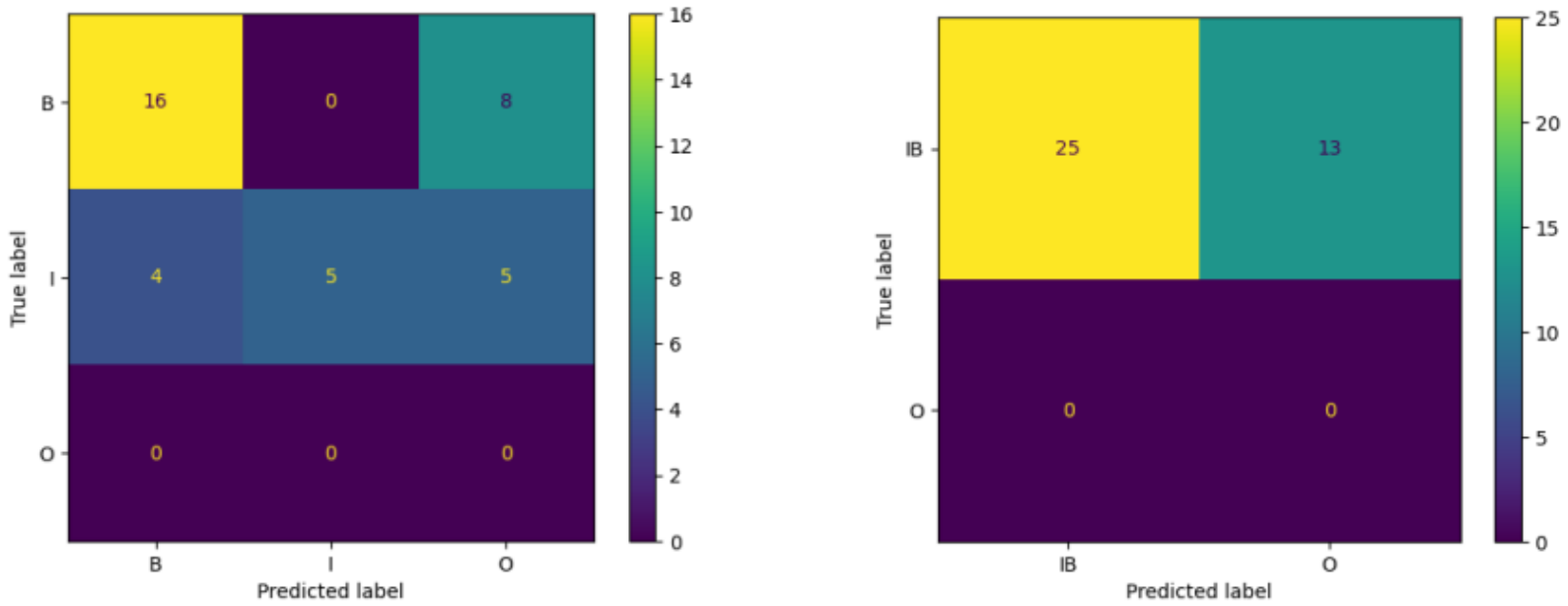
Evaluation Approach

- First, we consider all 3 classes I,O and B.
- Excluding ‘O’ tags gives better view of actual NER quality.
- ‘O’ vs ‘Non-O’ entity classes used for metrics calculation
- Highlights challenges in identifying multi-token entities.

Results Overview

- **Best Performing Configuration**
  - **Hidden Dimensions**: 4096
  - **Learning Rate**: 0.09
  - **Optimizer**: Adam
  - **Epochs**: 3000
  - **F1 Score**:
    - ~0.418 (All Classes),
    - ~0.397 (Grouped IB vs. O)

Results Overview

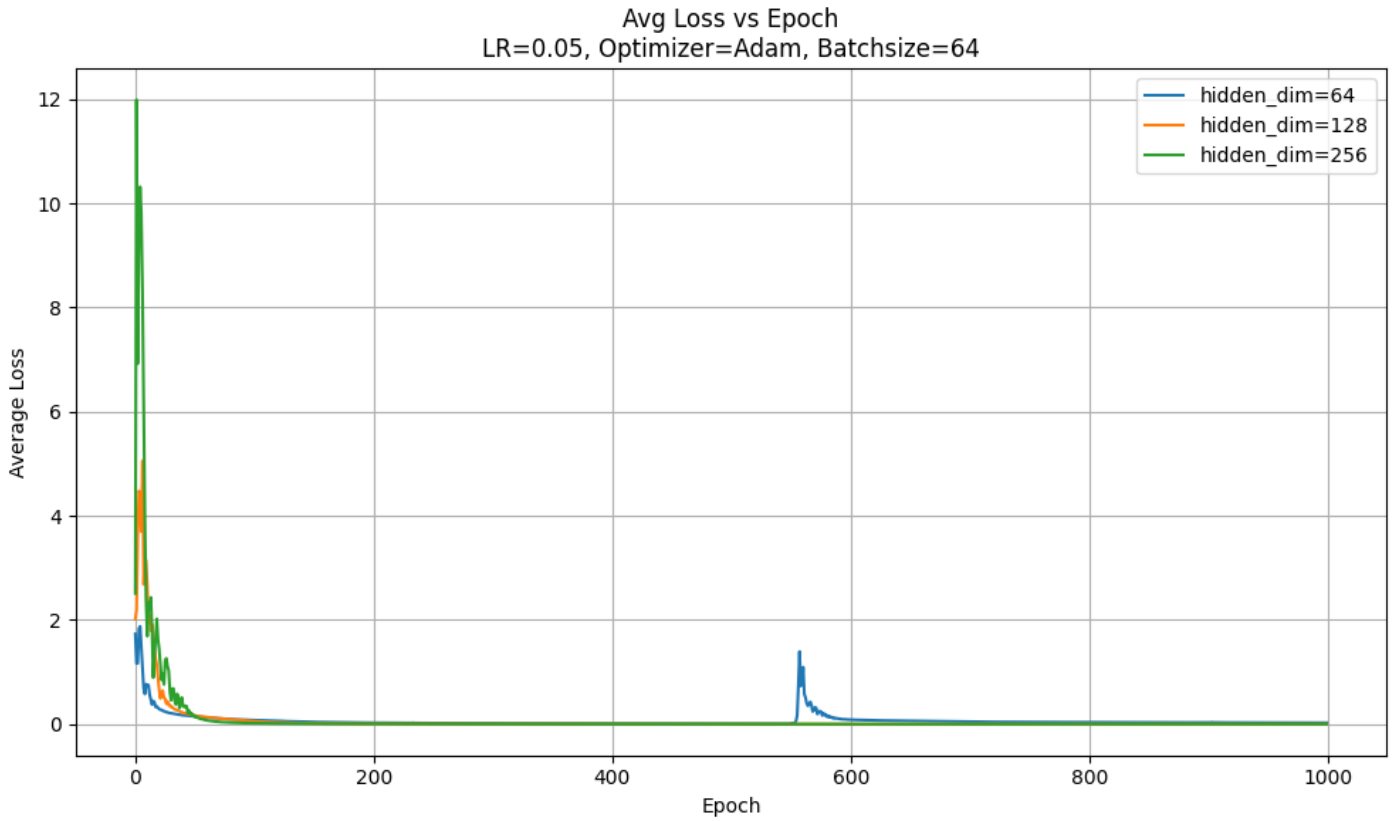


(a) I, O and B classes

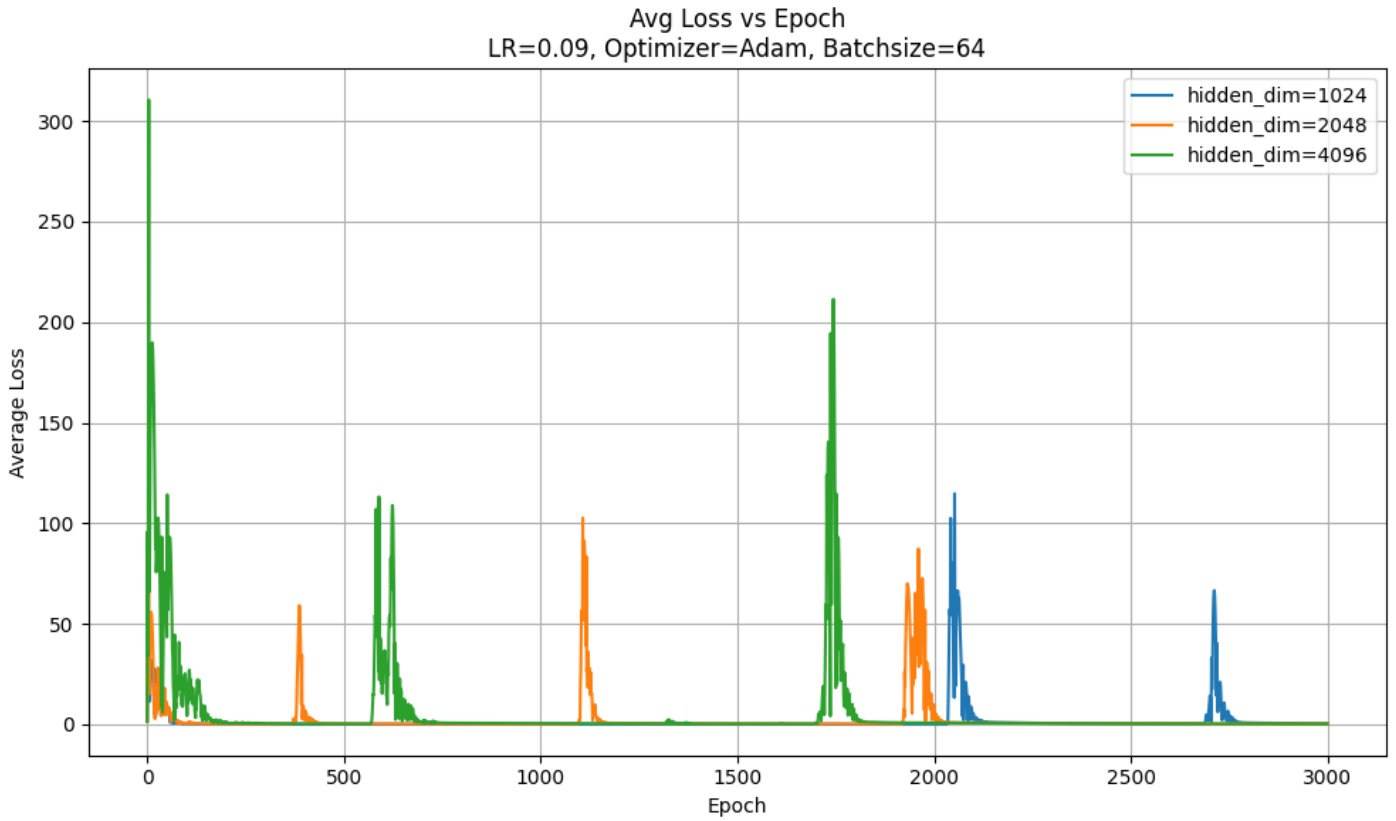
(b) O and Non-O (IB) classes

Figure 1: Contigency matricies for best performing model configuration

Results Overview



Results Overview



Results Overview

- Higher Hidden Dimensions -> Better Recall
  - Recall improved from **0.22 (1024)** to **0.34 (4096)**
  - Captures complex biomedical patterns more effectively
- Learning Rate Impact
  - Higher LR = faster convergence and better recall
  - LR = 0.09 gave best F1 but risked instability without ReLU
- Precision Stability
  - Precision stayed near 0.5 across most configs
  - F1 was primarily driven by improvements in recall

Evaluation and Performance Summary

- Evaluation Strategy
  - **Metrics**: Precision, Recall, F1 Score
  - **Schemes**:
    - *Multi-class*: Evaluates B, I, O separately
    - *Grouped*: B & I as one class vs O (entity vs non-entity)
- Best Model Configuration
  - Hidden Units: **4096**, Learning Rate: **0.09**
  - Optimizer: **Adam**, Epochs: **3000**
  - F1 Score: **0.418** (multi-class), **0.397** (grouped)
- Key Insights
  - Recall boosts drove most F1 gains; precision stayed ~0.5
  - ReLU activation stabilized high learning rate training
  - Data filtering had minimal impact; full dataset performed better

Precision-Recall Tradeoff

- Precision
  - Measures how many predicted entities were correct
  - Stayed stable across all configurations (~0.50)
- Recall
  - Measures how many actual entities were correctly identified
  - Improved significantly with:
    - Higher **hidden dimensions** (1024 → 4096)
    - Higher **learning rates** (up to 0.09)
    - **ReLU activation** aiding gradient flow
- Trade-off Observed
  - Boosting recall led to F1 improvements
  - But increasing recall often came at the cost of training stability
  - Precision stayed flat → Recall was the main driver of model gains
- Insight
  - Careful balancing of model complexity and learning rate is key to optimizing recall without sacrificing precision

Conclusion

Conclusion

- This project developed a GRU-based NER model tailored for biomedical text using the CRAFT corpus.
- BioBERT embeddings enriched token representations, improving semantic understanding of domain-specific entities.
- A thorough grid search over hyperparameters revealed that:
  - 4096 hidden units, learning rate of 0.09, and Adam optimizer gave the best results.

Conclusion

- Evaluation using multi-class and grouped tagging showed that:
  - Recall improvements, aided by ReLU activation, were key to maximizing F1 Score.
- While filtering helped address class imbalance, using the full dataset yielded better performance overall.
- The study shows that domain-adapted embeddings + GRU architecture can effectively tackle biomedical NER, with further gains possible via architecture tuning and domain-specific optimization.

References

- Cohen, K. B., Verspoor, K., Fort, K., Funk, C., Bada, M., Palmer, M., & Hunter, L. E. (2017). The Colorado richly annotated full text (CRAFT) corpus: multi-model annotation in the biomedical domain. *In Handbook of Linguistic Annotation* (pp. 1379-1394). Springer, Dordrecht.
  - CRAFT Corpus GitHub Repository

Thank You