Trabajo final de grado

GRADO DE INGENIERÍA INFORMÁTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

A comparison of machine learning methods for biomedical named entity recognition

Autor: Enrique Chueca Negre

Director:	Dr. Daniel Ortiz Martínez
Realitzat a:	Departament de Matemàtiques i Informàtica
Barcelona,	21 de febrer de 2026

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my thesis advisor, Daniel Ortiz Martinez, for his invaluable guidance, continuous support, and patience throughout the development of this project. His expertise in Natural Language Processing and Machine Learning and insightful feedback were instrumental in shaping this work.

Special thanks go to my family for their unconditional love and encouragement. Their unwavering belief in me has been a constant source of motivation throughout my academic journey.

I am also grateful to my friends and colleagues who offered their support, shared their knowledge, and made this experience more enjoyable.

Abstract

This thesis presents a comparative analysis of machine learning methods for Named Entity Recognition (NER) in biomedical text. Three approaches were implemented and evaluated on the JNLPBA dataset: a statistical baseline using Conditional Random Fields (CRF), and two Transformer-based models (BERT and BioBERT).

Results demonstrate that domain-specific pre-training significantly improves performance, with BioBERT achieving the highest Micro F1-Score of 0.7422, compared to 0.7195 for BERT and 0.6575 for CRF. Error analysis reveals that boundary detection and semantic similarity between entity types (CELL_TYPE vs. CELL_LINE, DNA vs. PROTEIN) are the main sources of errors across all models.

The findings confirm the importance of transfer learning and domain adaptation for specialized NLP tasks, while also highlighting the continued challenges in biomedical entity recognition.

Keywords: Named Entity Recognition, Biomedical NLP, BERT, BioBERT, Conditional Random Fields, Deep Learning, JNLPBA, Transfer Learning

Resumen

Este trabajo de fin de grado presenta un análisis comparativo de métodos de aprendizaje automático para el Reconocimiento de Entidades Nombradas (NER) en texto biomédico. Se implementaron y evaluaron tres enfoques utilizando el dataset JNLPBA: un modelo estadístico basado en Campos Aleatorios Condicionales (CRF) como línea base, y dos modelos basados en Transformers (BERT y BioBERT).

Los resultados demuestran que el pre-entrenamiento específico de dominio mejora significativamente el rendimiento. BioBERT alcanzó la puntuación F1 Micro más alta (0,7422), en comparación con 0,7195 de BERT y 0,6575 del CRF. El análisis de errores revela que la detección de límites de entidades y la similitud semántica entre tipos de entidades (CELL_TYPE vs. CELL_LINE, DNA vs. PROTEIN) son las principales fuentes de error en todos los modelos.

Los hallazgos confirman la importancia del aprendizaje por transferencia y la adaptación de dominio para tareas especializadas de PLN, al tiempo que destacan los desafíos persistentes en el reconocimiento de entidades biomédicas.

Palabras clave: Reconocimiento de Entidades Nombradas, PLN Biomédico, BERT, BioBERT, Campos Aleatorios Condicionales, Aprendizaje Profundo, JNLPBA, Aprendizaje por Transferencia

Resum

Aquest treball de fi de grau presenta una anàlisi comparativa de mètodes d'aprenentatge automàtic per al Reconeixement d'Entitats Anomenades (NER) en text biomèdic. S'han implementat i avaluat tres enfocaments utilitzant el dataset JNLPBA: un model estadístic basat en Camps Aleatoris Condicionals (CRF) com a línia base, i dos models basats en Transformers (BERT i BioBERT).

Els resultats demostren que el pre-entrenament específic de domini millora significativament el rendiment. BioBERT va assolir la puntuació F1 Micro més alta (0,7422), en comparació amb 0,7195 de BERT i 0,6575 del CRF. L'anàlisi d'errors revela que la detecció de límits d'entitats i la similitud semàntica entre tipus d'entitats (CELL_TYPE vs. CELL_LINE, DNA vs. PROTEIN) són les principals fonts d'error en tots els models.

Les troballes confirmen la importància de l'aprenentatge per transferència i l'adaptació de domini per a tasques especialitzades de PLN, alhora que destaquen els reptes persistents en el reconeixement d'entitats biomèdiques.

Paraules clau: Reconeixement d'Entitats Anomenades, PLN Biomèdic, BERT, BioBERT, Camps Aleatoris Condicionals, Aprenentatge Profund, JNLPBA, Aprenentatge per Transferència

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces the problem of Named Entity Recognition (NER) in the biomedical domain, outlines the motivation for this work, and presents the main objectives of this thesis. We begin by describing the specific challenges of biomedical text processing and then define the scope and goals of our comparative study.

1.1 Contextualization of the Problem

The exponential growth of biomedical literature presents a significant challenge for researchers and healthcare professionals. PubMed, the primary repository for biomedical publications, contains over 35 million citations, with thousands of new articles added daily. Manually extracting relevant information from this vast corpus is impractical, creating an urgent need for automated text mining tools.

Named Entity Recognition (NER) serves as a foundational step in biomedical text mining pipelines. By automatically identifying mentions of genes, proteins, diseases, and other biological entities, NER systems enable downstream applications such as:

- **Knowledge Base Construction:** Automatically populating databases like UniProt or DrugBank with entity relationships extracted from literature.
- **Drug Discovery:** Identifying potential drug-target interactions from scientific publications, accelerating the pharmaceutical research pipeline.
- **Clinical Decision Support:** Extracting relevant entities from electronic health records (EHRs) to assist clinicians in diagnosis and treatment planning.
- **Literature Curation:** Assisting biocurators in annotating databases with information from new publications.

The transition from rule-based systems to machine learning, and more recently to deep learning approaches, has dramatically improved NER performance. However, the biomedical domain presents unique challenges that general-purpose NLP models struggle to address, motivating this comparative study.

1.2 Motivation

1.2.1 Academic Motivation

From a research perspective, understanding the capabilities and limitations of different machine learning paradigms for biomedical NER is crucial for advancing the field. While transformer-based models have achieved state-of-the-art results on many NLP benchmarks, their performance advantage over classical methods varies significantly across domains and tasks. This thesis aims to quantify this advantage in the specific context of biomedical entity recognition.

1.2.2 Personal Motivation

From a personal standpoint, the intersection of natural language processing and biomedicine represents a fascinating application of artificial intelligence with direct real-world impact. The release of BERT in 2018 and its subsequent domain-specific variants sparked my interest in understanding how pre-training and transfer learning affect model performance on specialized tasks. This thesis provided an opportunity to explore these questions hands-on.

1.3 Problem Statement

General-purpose NLP models often fail when applied to the biomedical domain due to the specific challenges of scientific text: complex nomenclature (e.g., "p53", "IL-2"), polysemy, and abbreviations. This thesis addresses the need for specialized models that can handle these complexities effectively.

1.4 Objectives

The main objective of this thesis is to analyze and compare the performance of different Machine Learning approaches for Biomedical NER. Specifically, the goals of this work are:

1. Implement a statistical baseline using Conditional Random Fields (CRF).
2. Implement state-of-the-art Deep Learning models: BERT (general-domain) and BioBERT (domain-specific).
3. Evaluate and compare these models using the JNLPBA dataset under strict metrics (Exact Match F1-Score).
4. Analyze the impact of domain-specific pre-training on performance.

1.5 Project Planning

1.5.1 Timeline

The development of this thesis followed a structured timeline over approximately 4 months: Table 1.1 summarizes the main phases and their duration.

Table 1.1: Project timeline

Phase	**Duration**	**Tasks**
Literature Review	3 weeks	Research NER methods, study BERT architecture
Data Preparation	2 weeks	Dataset analysis, preprocessing pipeline
Implementation	5 weeks	Implement CRF, BERT, BioBERT pipelines
Experimentation	3 weeks	Train models, collect results
Analysis	2 weeks	Analyze results, create visualizations
Writing	3 weeks	Write thesis document

1.5.2 Computational Resources

The experiments were conducted using personal hardware with the following specifications:

- **CPU:** Intel(R) Core(TM) i5-14400F (2.50 GHz)
- **GPU:** NVIDIA GeForce RTX 5060 TI (8GB VRAM)
- **RAM:** 32GB DDR4
- **Storage:** 1TB NVMe SSD

1.6 Document Structure

The remainder of this thesis is organized as follows:

- **Chapter 2 (State of the Art):** Reviews the theoretical foundations of NER, from classical statistical methods to modern transformer architectures, with a focus on biomedical applications.
- **Chapter 3 (Materials and Methods):** Describes the experimental setup, including the dataset, preprocessing pipeline, evaluation metrics, and software tools used.
- **Chapter 4 (Implementation):** Details the technical implementation of the three models, including code architecture and training procedures.
- **Chapter 5 (Results and Discussion):** Presents the experimental results with quantitative metrics and qualitative analysis, followed by a comparative discussion.
- **Chapter 6 (Conclusions and Future Work):** Summarizes the main findings and outlines directions for future research.

Chapter 2

State of the Art

This chapter provides a comprehensive review of the theoretical foundations and existing approaches for Named Entity Recognition. This chapter first introduces the fundamentals of NER and its specific application to biomedical text. Then, the evolution of NER methods is examined, from classical statistical approaches to modern deep learning architectures.

2.1 Fundamentals of NER

Named Entity Recognition (NER), also known as entity identification or entity extraction, is a fundamental subtask of Information Extraction (IE) within the field of Natural Language Processing (NLP). Formally, NER can be defined as the process of locating and classifying atomic elements in text into predefined categories such as names of persons, organizations, locations, expressions of times, quantities, monetary values, and percentages.

The primary objective of NER is to transform unstructured text data into structured information that can be processed by machines. Given a sequence of tokens $W = \{w_1, w_2, ..., w_n\}$, the goal of an NER system is to output a sequence of tags $Y = \{y_1, y_2, ..., y_n\}$, where each y_i corresponds to the entity type of the token w_i.

2.1.1 Standard Entity Types

While entity categories can vary depending on the domain (e.g., genes or proteins in bioinformatics), the general purpose NER systems typically follow standard schemas defined by benchmark datasets such as CoNLL-2003 [7]. The most common entity types include:

- **PER (Person):** Names of people (e.g., *"Elon Musk"*).
- **ORG (Organization):** Companies, institutions, and government bodies (e.g., *"Tesla"*, *"United Nations"*).
- **LOC (Location):** Cities, countries, mountains, and other geographical entities (e.g., *"Barcelona"*, *"Mount Everest"*).
- **MISC (Miscellaneous):** Other entities that do not fit into the specific categories above (e.g., nationalities, events).

2.1.2 Biomedical Named Entity Recognition

Biomedical NER (BioNER) is a specialized form of NER that focuses on identifying entities relevant to the life sciences domain. Unlike general-domain NER, which typically recognizes persons, organizations, and locations, BioNER targets domain-specific entities such as:

- **Genes and Proteins:** e.g., "p53", "insulin", "NF-kappa B"
- **Diseases:** e.g., "diabetes", "Alzheimer′s disease"
- **Chemicals and Drugs:** e.g., "aspirin", "metformin"
- **Cell Types and Cell Lines:** e.g., "T cells", "HeLa cells"
- **DNA and RNA:** e.g., "IL-2 gene", "mRNA"

BioNER presents unique challenges compared to general NER:

1. **Complex Nomenclature:** Biomedical entities often have alphanumeric names (e.g., "IL-2", "CD4+"), Greek letters (e.g., "TNF-α"), and special characters that confuse standard tokenizers.
2. **Ambiguity:** The same surface form can refer to different entity types depending on context. For example, "p53" can refer to the p53 protein or the TP53 gene.
3. **Nested Entities:** Biomedical text often contains nested entities, such as "IL-2 receptor α chain", where "IL-2" is a protein within a larger protein complex name.
4. **Evolving Terminology:** New genes, proteins, and diseases are constantly being discovered, requiring models that can generalize to unseen entities.

These challenges motivate the need for specialized approaches, which are explored in the following sections.

2.1.3 The NER Pipeline

The process of recognizing entities generally follows a standard pipeline which transforms raw text into classified entities. This pipeline parallels the structure described in general NLP literature and has been successfully applied in end-to-end biomedical text mining platforms such as PharmKE [33], which integrates NER with relation extraction and knowledge graph construction for pharmaceutical texts.

Figure 2.1 illustrates the general NER pipeline used in this work.

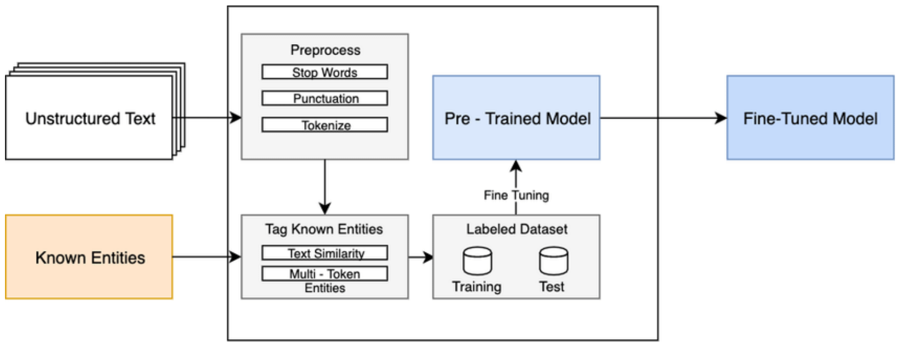

Figure 2.1: General NER pipeline: from raw text to classified entities.

1. **Tokenization:** The input text is decomposed into smaller units called tokens (words or sub-words).
2. **Contextual Representation:** The model analyzes the context of each token. In modern Deep Learning approaches, this is achieved through embeddings (e.g., Word2Vec, BERT embeddings).
3. **Tagging/Decoding:** The system assigns a specific label to each token. The choice of tagging scheme significantly impacts performance. Two widely used schemes are:
   - **BIO (Begin, Inside, Outside):** The standard scheme used in this work, where tags mark the beginning of an entity, the inside tokens, and non-entity tokens.
   - **BIOES (Begin, Inside, Outside, End, Single):** A more expressive extension that explicitly marks the end of an entity and single-token entities. While it provides more boundary information, standard BIO is often sufficient for BERT-based models.

2.2 Main NER Approaches

2.2.1 Classic Approaches: HMMs and CRFs

Before the advent of Deep Learning, statistical sequence labeling models were the standard for NER tasks. These approaches model the dependencies between neighboring tags to predict the most probable sequence of labels for a given sentence.

2.2.1.1 Hidden Markov Models (HMMs)

Hidden Markov Models are **generative models** that assume the underlying sequence of tags (hidden states) generates the observed sequence of words. An HMM models the joint probability $P(X, Y)$ of the input sequence X and the label sequence Y.

The model relies on two strong independence assumptions:

1. The tag y_i depends only on the previous tag y_{i-1} (Markov assumption).
2. The word x_i depends only on the current tag y_i (Emission probability).

While HMMs are computationally efficient, their strict independence assumptions limit their ability to capture overlapping features or long-range context, which are crucial for recognizing complex entities in biomedical text.

2.2.1.2 Conditional Random Fields (CRFs)

To overcome the limitations of HMMs, Lafferty et al. introduced the so-called conditional random fields [2]. Figure 2.2 illustrates the structure of a linear-chain CRF.

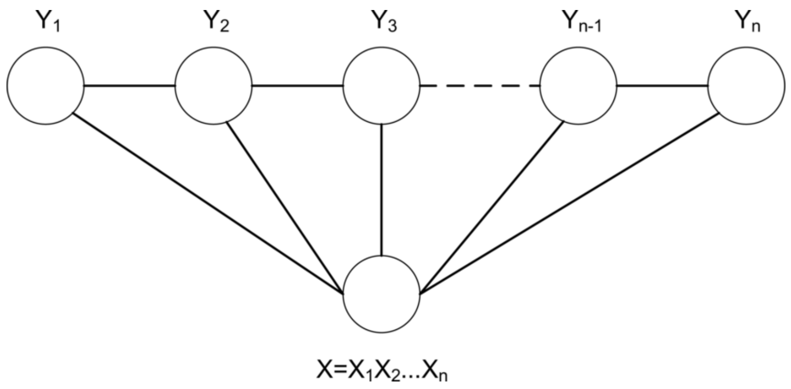

Figure 2.2: Structure of a Linear-Chain CRF. The nodes y_i represent the hidden label sequence (e.g., BIO tags), and x_i represent the observed input sequence (words). The edges model the dependencies between adjacent labels and between labels and observations. Source: Adapted from ResearchGate [2].

Unlike HMMs, CRFs are **discriminative models** that directly model the conditional probability $P(Y|X)$.

The probability of a label sequence Y given an observation sequence X is defined as:

$$P(Y|X) = \frac{1}{Z(X)} \exp\left(\sum_{i=1}^{n}\sum_{k}\lambda_k f_k(y_{i-1}, y_i, X, i)\right) \tag{2.1}$$

Where:

- f_k are **feature functions** (e.g., "is the current word capitalized?", "does it end in -ase?").
- λ_k are the learned weights for each feature.
- $Z(X)$ is a normalization factor.

Advantages over HMMs: The primary advantage of CRFs is their ability to incorporate arbitrary, non-independent features of the input without modeling their distribution. This makes CRFs heavily dependent on **feature engineering**, where domain experts must manually define rules (prefixes, suffixes, POS tags) to help the model distinguish entities. This characteristic serves as the baseline for this thesis, representing the "pre-Deep Learning" era of NLP.

2.2.1.3 Word Embeddings in Biomedical NLP

Pre-trained word embeddings significantly improved BioNER before transformers:

- **Word2Vec** [8]: Static word vectors.
- **BioWordVec**: Trained on PubMed + MeSH.
- **ELMo** [9]: Context-dependent embeddings.

2.2.2 Deep Learning Approaches

Modern Deep Learning approaches have replaced the need for manual feature engineering. Instead, these models automatically learn feature representations from raw text using dense vector embeddings [1].

2.2.2.1 Recurrent Neural Networks (RNNs) and BiLSTM-CRF

Before the dominance of Transformers, Recurrent Neural Networks (RNNs), specifically Long Short-Term Memory (LSTM) networks, were the state-of-the-art for sequence labeling. Unlike HMMs, RNNs can capture long-range dependencies in text preventing the "vanishing gradient" problem.

The most influential architecture in this era was the **BiLSTM-CRF** model [3]. This architecture combines:

- **BiLSTM Encoder:** Reads the sentence in both directions (forward and backward) to capture full context for each word.
- **CRF Decoder:** Applies valid sequence constraints (e.g., ensuring an `I-Protein` tag follows a `B-Protein` tag) on top of the neural predictions.

2.2.2.2 Transformer-based Models (BERT)

The introduction of the Transformer architecture [4] marked a paradigm shift in NER. Unlike Recurrent Neural Networks (RNNs) that process text sequentially, Transformers utilize a **Self-Attention mechanism**.Figure 2.3 shows the original Transformer architecture. This allows the model to weigh the importance of different words in a sentence simultaneously, effectively capturing long-range dependencies regardless of the sequence length.

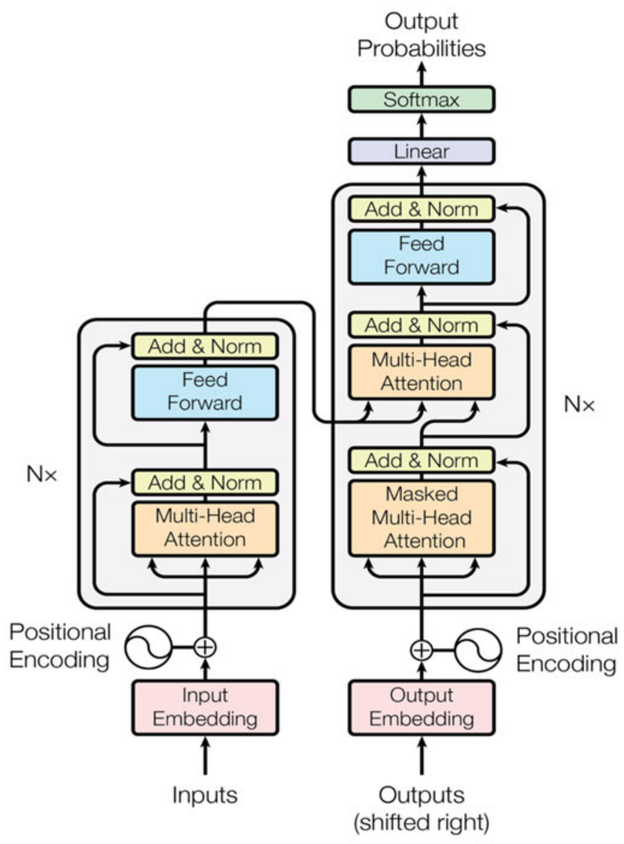

Figure 2.3: The Transformer model architecture, illustrating the Multi-Head Attention mechanism. Source: Vaswani et al. [4].

Models like BERT (Bidirectional Encoder Representations from Transformers) [5] are pre-trained on massive general corpora.

2.2.2.3 Domain Adaptation (BioBERT)

For specialized fields, domain adaptation is crucial. **BioBERT** [6] adapts the BERT architecture by further pre-training on biomedical corpora (PubMed, PMC), enabling the system to disambiguate complex medical terminology (e.g., genes vs. proteins) significantly better than general-purpose models.

2.2.3 Evolution of Biomedical NER

The history of BioNER can be divided into three main eras:

1. **Rule-based systems (1990s):** Dictionary lookup and pattern matching.
2. **Statistical ML (2000-2015):** CRFs dominated, as evidenced by the JNLPBA 2004 shared task [12], where the winning systems achieved F1-scores around 72%.

3. **Deep Learning era (2015-present):** From BiLSTM-CRF to Transformer-based models.

Table 2.1 summarizes the progression of results on the JNLPBA dataset over the years, and Figure 2.4 illustrates the general NER pipeline used in this work.

Table 2.1: Previous results on the JNLPBA dataset

Model	**Year**	**F1-Score**
Zhou & Su (HMM + SVM)	2004	72.55
Settles (CRF + features)	2004	69.80
Habibi et al. (BiLSTM-CRF)	2017	73.17
BioBERT	2020	77.49
PubMedBERT	2021	79.10

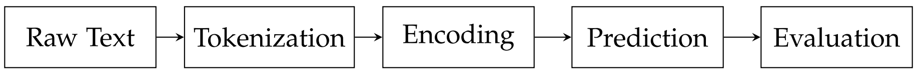

Figure 2.4: General NER pipeline used in this work

Chapter 3

Materials and Methods

This chapter describes the experimental setup used in this study. The three NER systems selected for comparison are presented, along with the dataset used for evaluation, the preprocessing steps applied to the data, and the metrics employed to assess model performance.

3.1 Selected NER Systems

To provide a comprehensive analysis, three distinct systems were selected representing the evolution of NLP:

- **CRF (Conditional Random Fields):** Selected as the statistical baseline. It relies on manual feature engineering and represents the classical approach to sequence labeling.
- **BERT (bert-base-cased):** Selected to represent the transformer revolution. It uses a general-domain vocabulary and demonstrates the power of transfer learning.
- **BioBERT (biobert-v1.1):** Selected as the domain-specific contender. It shares the architecture of BERT but has been pre-trained on biomedical corpora (PubMed abstracts and PMC full-text articles), making it theoretically superior for this task.

3.2 Dataset

For this study, the **JNLPBA** dataset was used (Joint Workshop on Natural Language Processing in Biomedicine and its Applications) 2004 shared task dataset [12], accessed through the Hugging Face Datasets library (`commanderstrife/jnlpba`).

This dataset is derived from the **GENIA corpus** and consists of MEDLINE abstracts annotated with five bio-entity types. Table 3.1 summarizes the dataset statistics.

Table 3.1: JNLPBA dataset statistics (Hugging Face: `commanderstrife/jnlpba`). Note: Validation and Test splits are identical in this version.

Statistic	**Training**	**Validation**	**Test**
Samples	37,094	7,714	7,714
Entity Distribution (test-validation split)			
PROTEIN	–	–	10,134
CELL_TYPE	–	–	3,842
DNA	–	–	2,112
CELL_LINE	–	–	1,000
RNA	–	–	236
Total Entities	–	–	17,324

As shown in Table 3.1, the Hugging Face version provides three pre-defined splits with identical sizes for validation and test sets (7,714 samples each). The test set contains a total of 17,324 annotated entities distributed across five categories:

- **PROTEIN** (58.5%): Names of proteins, including enzymes, receptors, and transcription factors (e.g., "IL-2", "NF-kappa B").
- **CELL_TYPE** (22.2%): Types of cells defined by their function or lineage (e.g., "T cells", "macrophages").
- **DNA** (12.2%): References to genetic material, including genes and genetic sequences (e.g., "p53 gene", "promoter region").
- **CELL_LINE** (5.8%): Specific cultured cell populations used in research (e.g., "HeLa", "Jurkat cells").
- **RNA** (1.4%): References to RNA molecules (e.g., "mRNA", "siRNA").

The dataset exhibits significant class imbalance, with PROTEIN representing over half of all entities while RNA accounts for only 1.4%. This imbalance poses challenges for model training and evaluation, particularly for minority classes.

3.2.1 Class Imbalance Analysis

A critical characteristic of the JNLPBA dataset is the extreme class imbalance at the token level. While the previous statistics showed entity-level distribution, Table 3.2 reveals the full extent of the imbalance by showing the distribution of BIO tags in the training set.

Table 3.2: Token-level distribution of all BIO tags in the JNLPBA training set, showing the dominance of the "O" (Outside) tag with 77.75% of all tokens.

Tag	Count	Percentage
O (Outside)	765,926	77.75%
B-PROTEIN	60,538	6.15%
I-PROTEIN	49,426	5.02%
I-CELL_TYPE	23,085	2.34%
B-CELL_TYPE	17,484	1.77%
I-DNA	17,207	1.75%
B-DNA	13,395	1.36%
I-CELL_LINE	8,119	0.82%
B-CELL_LINE	6,718	0.68%
I-RNA	1,674	0.17%
B-RNA	1,530	0.16%
Total	985,102	100.00%

Key Observations:

1. **Dominance of "O" tags:** The "Outside" tag represents 77.75% of all tokens. This means that in a typical sentence, approximately 3 out of every 4 tokens do not belong to any biomedical entity.

2. **Extreme imbalance ratio:** The ratio between the most frequent tag (O) and the least frequent (B-RNA) exceeds **500:1**. Even among entity tags, B-PROTEIN outnumbers B-RNA by approximately **40:1**.

3. **Implications for evaluation:** This extreme imbalance is precisely why the `seqeval` library excludes "O" tags from metric calculations. A naive model that predicts "O" for every token would achieve 77.75% token-level accuracy while being completely useless for entity extraction.

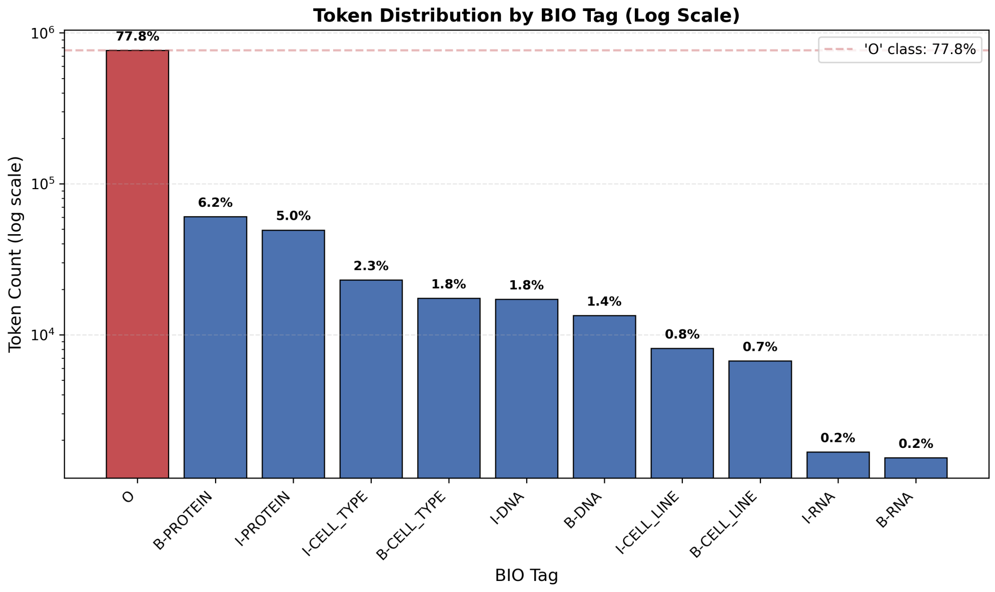

Figure 3.1: Token-level distribution of all BIO tags in the JNLPBA training set (logarithmic scale). The "O" (Outside) tag dominates with 77.75% of all tokens, illustrating the extreme class imbalance characteristic of NER datasets.

Figure 3.1 visualizes this imbalance, clearly showing why standard accuracy metrics are inappropriate for NER evaluation. Figure 3.2 shows the distribution among entity types only (excluding "O" tags).

This analysis reinforces the importance of using entity-level F1-Score (via `seqeval`) rather than token-level accuracy as the primary evaluation metric, as discussed in Section 3.4.

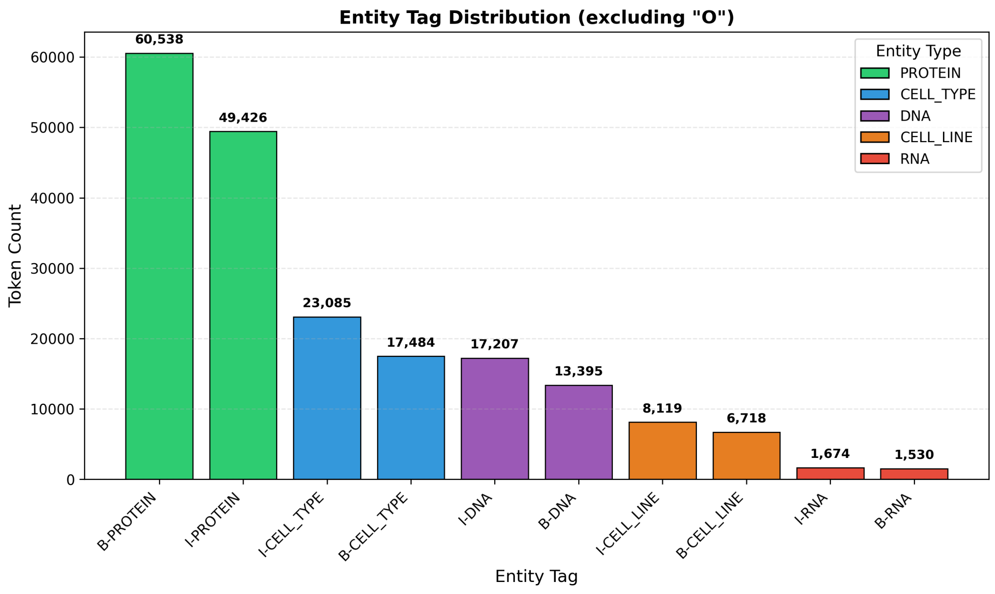

Figure 3.2: Entity-level distribution excluding "O" tags. PROTEIN entities dominate with 58.5% of annotated entities, while RNA represents only 1.4%, highlighting the inter-class imbalance among biomedical entity types.

3.2.2 Dataset Split

The JNLPBA dataset was used through Hugging Face (`commanderstrife/jnlpba`). It provides three labeled splits: training (37,094 samples), validation (7,714 samples), and test (7,714 samples).

Critical observation (validation = test): Split independence was verified during data exploration, revealing that the Hugging Face `validation` and `test` splits are identical in this dataset version. Therefore, the dataset does not provide an independent validation set.

Implication: Since both splits contain identical data, they are collectively referred to as the *test-validation split* throughout this document. All reported metrics correspond to this single evaluation set.

3.3 Preprocessing Pipeline

Data preprocessing is crucial for model performance. The pipeline differs slightly between models:

- **For CRF:** The text was tokenized into words and extracted features manually for each token, including: current word suffix (last 2-3 characters), capitalization flags, digit presence, and context window (previous and next words).
- **For BERT/BioBERT:** The text was processed using a **WordPiece tokenizer**. This splits complex medical terms into sub-words (e.g., "immunoglobulin" $\rightarrow$ "immuno", "##globu-

lin"), effectively handling the out-of-vocabulary problem. Labels were aligned to the first sub-word token to maintain valid BIO tagging.

3.4 Selected Evaluation Metrics

3.4.1 Metric Definitions

To assess the performance of the NER systems,Standard metrics were used defined in the literature: Precision, Recall, and F1-Score. Unlike simple binary classification, evaluating NER systems requires a strict definition of what constitutes a correct prediction.

For entity-level evaluation, We used `seqeval` library [13], a Python framework for sequence labeling evaluation that implements the CoNLL evaluation protocol. This library computes precision, recall, and F1-Score at the entity level, requiring exact boundary and type matching for a prediction to be considered correct.

3.4.2 Specific Label Treatment in NER Evaluation

A critical aspect of NER evaluation that distinguishes it from standard classification tasks is the treatment of the "Outside" (O) label. In the BIO tagging scheme, the vast majority of tokens in a typical sentence are tagged as "O" (not part of any entity). For example, in the sentence:

*"The **IL-2** protein regulates **T cell** activation."*

Only 3 out of 7 tokens belong to entities ("IL-2" $\rightarrow$ B-PROTEIN, "T" $\rightarrow$ B-CELL_TYPE, "cell" $\rightarrow$ I-CELL_TYPE), while the remaining 4 tokens ("The", "protein", "regulates", "activation") are tagged as "O".

Why "O" Labels Are Excluded from Metrics:

The CoNLL evaluation protocol, implemented in the `seqeval` library, deliberately excludes "O" tags from precision, recall, and F1-score calculations. This design choice is motivated by two factors:

1. **Class Imbalance:** In typical biomedical text, 70-90% of tokens are "O". Including them would artificially inflate accuracy metrics, as a model that predicts "O" for everything would achieve high accuracy but be useless for entity extraction.
2. **Task Relevance:** The goal of NER is to *find* entities, not to correctly label non-entities. A model's utility is measured by its ability to identify the entities that exist, not by its ability to recognize ordinary words.

Entity-Level Collapse:

Furthermore, the evaluation collapses consecutive B- and I- tags into complete entities. For example:

Token:	"interleukin"	"-"	"2"	"receptor"
Tag:	B-PROTEIN	I-PROTEIN	I-PROTEIN	I-PROTEIN

These four tokens are evaluated as a **single entity** "interleukin-2 receptor". A prediction is considered correct only if:

- All token boundaries match exactly (start and end positions).
- The entity type is correct (PROTEIN in this case).

This strict evaluation ensures that reported metrics reflect the system's practical utility for downstream applications like knowledge base population.

3.4.3 Definition of Positives and Negatives in NER

Following the **Exact Match** criterion used by the `seqeval` library (standard in CoNLL evaluations), We define the classification outcomes as follows:

- **True Positive (TP):** The system predicts an entity that matches the ground truth in both **boundaries** (start and end offsets) and **type**.
- **False Positive (FP):** The system predicts an entity that does not exist in the ground truth, or predicts an entity with incorrect boundaries or type.
- **False Negative (FN):** The system fails to predict an entity that exists in the ground truth.

3.4.4 Core Metrics

Based on these definitions, the metrics are calculated as follows:

- **Precision (P):** Measures the "trustworthiness" of the system. It answers the question: *Of all the entities the model predicted, how many were actually correct?*

$$P = \frac{TP}{TP + FP} \tag{3.1}$$

- **Recall (R):** Measures the "coverage" or "sensitivity" of the system. It answers the question: *Of all the entities that really exist in the text, how many did the model find?*

$$R = \frac{TP}{TP + FN} \tag{3.2}$$

- **F1-Score:** The harmonic mean of Precision and Recall. This metric is particularly important in NER because datasets are often unbalanced (many 'O' tags vs few entities). The harmonic mean penalizes extreme values more than the arithmetic mean, ensuring that a model must have both good Precision and good Recall to achieve a high F1-Score.

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{3.3}$$

3.4.5 Averaging Strategies

Since NER is a multi-class problem (e.g., DNA, Protein, RNA), We must aggregate the scores of individual classes to obtain a global metric.The following averages are reported:

1. **Micro Average:** Aggregates the total TP, FP, and FN over all classes before calculating metrics. It gives equal weight to every instance, so it is heavily influenced by the most frequent classes (e.g., Protein).
2. **Macro Average:** Calculates metrics for each class independently and then takes the unweighted mean. It treats all classes as equally important, regardless of their frequency.
3. **Weighted Average:** Calculates metrics for each class and averages them weighting by the number of true instances (support). In this thesis it is reported as a secondary descriptive statistic.

3.4.6 Choice of Primary Metric

For this study, We report the **Micro-averaged F1-Score** as the primary evaluation metric. This choice is motivated by several factors:

1. **Entity-level aggregation:** Micro-averaging treats each entity prediction equally, regardless of entity type. This aligns with the practical goal of correctly identifying as many entities as possible.
2. **Handling class imbalance:** In JNLPBA, PROTEIN entities represent 58.5% of all entities while RNA accounts for only 1.4%. Micro-averaging naturally weights performance by class frequency, reflecting the distribution a system would encounter in real biomedical text.
3. **Consistency with literature:** The original JNLPBA shared task [12] and subsequent publications on this dataset report micro-averaged F1-Score, enabling direct comparison with prior work.
4. **SeqEval default:** The `seqeval` library computes micro-averaged metrics by default, ensuring consistency across all model evaluations.

While macro-averaging would give equal importance to each entity type (beneficial for evaluating performance on rare classes like RNA), micro-averaging provides a more realistic measure of overall system utility for biomedical text mining applications.

3.5 Evaluation Procedure

A consistent evaluation protocol was followed for all models:

1. **Training:** Models were trained on the training split of JNLPBA.
2. **Test-validation split:** Since the Hugging Face `validation` and `test` splits are identical in `commanderstrife/jnlpba`, all results are reported on this single *test-validation split*.
3. **Metric:** All models are compared primarily using **micro-averaged entity-level F1** computed with `seqeval`.
4. **Overfitting:** Overfitting is analyzed using the gap between training performance and the test-validation split performance, and by inspecting the loss/F1 curves (noting that the curves correspond to this duplicated split).

3.6 Software and Hardware

This section provides a detailed description of the software tools, libraries, and services used in the development and execution of this project.

3.6.1 Programming Languages

3.6.1.1 Python

Python 3.10 was the primary programming language used for all experiments. Python's extensive ecosystem of machine learning libraries, particularly for NLP tasks, made it the natural choice. Key advantages include:

- Native support for scientific computing through NumPy and SciPy.
- Comprehensive deep learning frameworks (PyTorch, TensorFlow).
- NLP libraries (Hugging Face Transformers, spaCy, NLTK).
- Interactive development through Jupyter notebooks.

3.6.2 Deep Learning Libraries

3.6.2.1 PyTorch

PyTorch [14] served as the underlying deep learning framework. Version 2.0 was used, which includes significant performance improvements. Key components used include:

- `torch.nn`: Neural network modules and layers.
- `torch.optim`: Optimization algorithms (AdamW).
- `torch.cuda`: GPU acceleration utilities.
- `torch.utils.data`: Data loading and batching.

3.6.2.2 Hugging Face Transformers

The Transformers library [15] provided pre-trained models and tokenizers:

- `BertForTokenClassification`: BERT model with token classification head.
- `AutoTokenizer`: Automatic tokenizer selection based on model.
- `Trainer`: High-level training API with built-in logging.
- `TrainingArguments`: Configuration for training hyperparameters.

3.6.2.3 Hugging Face Datasets

The Datasets library was used to load and preprocess the JNLPBA dataset:

- Efficient data loading with memory mapping.
- Built-in caching to avoid redundant preprocessing.
- Easy integration with the Transformers `Trainer`.

3.6.3 Evaluation Libraries

3.6.3.1 seqeval

The seqeval library [13] implemented the entity-level evaluation protocol:

- `classification_report`: Detailed per-class metrics.
- `f1_score`: Overall F1-Score computation.
- Support for BIO, BIOES, and IOB tagging schemes.

3.6.3.2 scikit-learn

scikit-learn was used for:

- Confusion matrix computation and visualization.
- Hyperparameter optimization (RandomizedSearchCV for CRF).

3.6.4 Visualization

3.6.4.1 Matplotlib and Seaborn

Matplotlib and Seaborn were used for creating all figures in this thesis:

- Training loss and F1-Score curves.
- Confusion matrices with color mapping.
- Bar charts for model comparison.

3.6.5 Development Environment

3.6.5.1 Jupyter Notebooks

Jupyter notebooks were used for interactive development and experimentation. Each model (CRF, BERT, BioBERT) was implemented in a separate notebook to ensure modularity and reproducibility.

Chapter 4

Implementation

This chapter details the technical development of the three models implemented for this project. We begin describing the specific implementation choices for the CRF baseline and the Transformer-based models (BERT and BioBERT), including data loading, tokenization strategies, and the training procedure.

4.1 CRF Implementation

The CRF model was implemented using the `sklearn-crfsuite` library. The core component is the feature extraction function `word2features`, which transforms raw text into a dictionary of attributes.

- **Feature Engineering:** Features such as `word.isupper()`, `word[-3:]` (suffixes), and `word.isdigit()` were defined.
- **Training:** The model was trained using the L-BFGS algorithm with Elastic Net regularization (L1/L2) to prevent overfitting.
- **Hyperparameter Optimization:** A Randomized Search Cross-Validation was performed to find the optimal values for coefficients $c1$ and $c2$ using the `sklearn-crfsuite`.

4.1.1 Model Interpretability: CRF Transitions

A key advantage of CRFs over deep learning models is their interpretability. By analyzing the learned transition weights, we can understand the grammatical rules the model has internalized.

Table 4.1 presents the top transitions learned by the baseline model. As expected, the model assigns high positive weights to valid BIO transitions (e.g., `B-PROTEIN` $\rightarrow$ `I-PROTEIN`) and correctly penalizes invalid transitions (e.g., `O` $\rightarrow$ `I-DNA`), demonstrating that it successfully learned the structural constraints of the entity recognition task without explicit hard-coded rules.

Table 4.1: Top learned transition weights in the Baseline CRF model. High positive values indicate likely sequences, while the model learns to enforce BIO constraints.

From Label	**To Label**	**Weight**
B-PROTEIN	I-PROTEIN	6.84
B-DNA	I-DNA	6.12
I-RNA	I-RNA	5.95
B-CELL_TYPE	I-CELL_TYPE	5.43
O	O	4.15

4.1.2 CRF Extension with Advanced Features

In addition to the baseline features, an extended feature set was implemented to capture biomedical semantics and push the limits of statistical learning. This implementation includes:

- **Linguistic Context (POS Tags):** Integration of Part-of-Speech tags (e.g., Noun, Adjective) using NLTK to distinguish between descriptive terms and proper entity names.
- **Orthographic Triggers:** Boolean flags to detect alphanumeric patterns typical of gene symbols (e.g., *"p53"*, *"CD28"*) and hyphenated structures (e.g., *"IL-2"*).
- **Greek Nomenclature:** Specific detection of spelled-out Greek letters (e.g., *"alpha"*, *"kappa"*), which are strong indicators of protein subunits.
- **Lookahead Context:** Features analyzing the *next* word in the sequence to detect trigger terms (e.g., if the next word is *"receptor"*, the current word is likely a *PROTEIN*).

While the transition weights remained consistent with the baseline model (enforcing BIO constraints), the true value of the CRF Extension lies in its learned feature weights. By analyzing the model's parameters, we can verify if the manually engineered features were effectively utilized.

Table 4.2 displays the top weighted features contributing to entity classification. The model assigned high positive weights to domain-specific patterns such as the suffix `-ase` for proteins and the lookahead context `indicates_dna` (triggered by words like "gene"), validating the hypothesis that biomedical nomenclature follows predictable morphological and semantic patterns.

Table 4.2: Top feature weights learned by the CRF Extension model. The high values for suffixes and context triggers confirm the effectiveness of the advanced feature engineering.

Feature Type	Feature Name	Target Label	Weight
Suffix	`word[-3:]:ase`	B-PROTEIN	5.12
Context	`+1:word.indicates_dna`	B-DNA	4.87
Trigger	`word.is_greek`	I-PROTEIN	3.95
Morphology	`word.is_alphanumeric`	B-DNA	3.42
Suffix	`word[-2:]:in`	B-PROTEIN	3.15
POS Tag	`postag:NN`	B-CELL_TYPE	2.88

4.1.3 Metric Unification Strategy

A critical aspect of this implementation was ensuring a fair comparison between the statistical baseline and the deep learning models. Standard CRF implementations typically calculate accuracy at the token level or use permissive F1-scores that do not require exact boundary matching.

To resolve this, a conversion layer was implemented that transforms the CRF output (lists of tag strings) into the format required by the `seqeval` library [13]. Specifically:

1. The CRF produces predictions as lists of BIO tag strings (e.g., `['B-PROTEIN', 'I-PROTEIN', 'O', ...]`).
2. These predictions are passed directly to `seqeval`'s `classification_report()` and `f1_score()` functions, which compute entity-level metrics using the CoNLL evaluation protocol.
3. This ensures that the reported F1-Score for the CRF is calculated using the exact same "Entity-Level Strict Match" criteria as the Transformer models, guaranteeing scientific validity.

The following code snippet illustrates this integration:

```
from seqeval.metrics import classification_report, f1_score

# CRF predictions (list of lists of tag strings)
y_pred = crf.predict(X_test)
y_true = y_test

# Compute entity-level metrics (same as BERT evaluation)
micro_f1 = f1_score(y_true, y_pred, average='micro')
print(classification_report(y_true, y_pred))
```

This approach ensures that all three models (CRF, BERT, BioBERT) are evaluated under identical conditions, enabling a fair and scientifically valid comparison of their entity recognition capabilities.

4.2 Transformer Models (BERT & BioBERT) Implementation

The implementation for BERT and BioBERT follows a similar structure using the PyTorch framework.

4.2.1 Data Loading and Tokenization

We implemented a custom `Dataset` class inheriting from PyTorch. This class handles the tokenization using the pre-trained tokenizers and aligns the BIO tags with the sub-word tokens generated by WordPiece. A maximum sequence length of 128 tokens was used due to GPU memory constraints.

Handling Sub-word Tokenization: Since BERT uses WordPiece tokenization, a single medical term like "Hydrocortisone" might be split into ["Hydro", "##cor", "##tisone"]. To prevent metric inflation, a "Label Alignment" strategy was implemented:

- The first sub-word ("Hydro") retains the original entity tag (e.g., B-PROTEIN).
- Subsequent sub-words ("##cor", "##tisone") are assigned a special ignore index ID (typically -100).
- During loss calculation and metric evaluation, these ignored tokens are masked out, ensuring the model is evaluated only on meaningful entities.

Label Filtering for Evaluation: During the evaluation phase, additional filtering was applied to ensure accurate metric computation. Specifically, the following auxiliary labels were excluded from both the confusion matrix and the classification report:

- **'X' label:** Assigned to sub-word tokens generated by WordPiece. Since only the first sub-word of each word receives the actual entity label, subsequent sub-words marked as 'X' do not represent independent annotations and must be excluded.
- **'_' label:** An artifact that may appear due to dictionary misalignment during label conversion.
- **Index -100:** Used by PyTorch's CrossEntropyLoss to automatically ignore padding tokens ([PAD]), special tokens ([CLS], [SEP]), and sub-word continuations during training.

This filtering ensures that the reported metrics (Precision, Recall, F1-Score) reflect the model's actual performance on meaningful entity predictions, following standard practices in token-level NER evaluation with Transformer models [13].

4.2.2 Fine-tuning Loop

The `BertForTokenClassification` architecture was utilized. The training loop consists of:

1. **Forward Pass:** The input IDs and attention masks are fed into the model.
2. **Loss Calculation:** The Cross-Entropy Loss is calculated between the predicted logits and the true labels (ignoring padding tokens).

3. **Backward Pass:** Gradients are computed and weights are updated using the **AdamW optimizer** with a learning rate of $2e-5$.

This process was repeated for 5 epochs for both models to ensure convergence.

4.3 Experimental Settings

The experiments were conducted with consistent hyperparameters to ensure comparability:

- **Batch Size:** 16 (limited by GPU VRAM).
- **Learning Rate:** $2e-5$ (standard for BERT fine-tuning).
- **Epochs:** 5 epochs (fixed schedule).
- **Optimizer:** AdamW (Adam with Weight Decay Fix).
- **Seeds:** Random seeds were fixed to 42 for reproducibility.

Note: In this dataset version the validation and test splits are identical (referred to as the *test-validation split*). No independent validation set was available for checkpoint selection.

4.3.1 Code Availability

The complete source code for this project, including all Jupyter notebooks and reproducibility instructions, is publicly available on GitHub:

`https://github.com/Quikaragon/A-comparison-of-machine-learning-methods-for-biomedical-named-entity-recognition`

Chapter 5

Results and Discussion

This chapter presents the experimental results obtained from the three implemented models: CRF, BERT, and BioBERT. The training dynamics are first presented (loss curves and validation metrics), followed by the final evaluation results on the Hugging Face evaluation split (labeled as `test`). Performance is analyzed using the metrics defined in Chapter 3, and the implications of these findings are discussed.

Note: In `commanderstrife/jnlpba`, the `validation` and `test` splits are identical. All results in this chapter are reported on this *test-validation split*.

5.1 Experimental Results

5.1.1 Statistical Baseline: CRF

The conditional random fields (CRF) model served as the statistical baseline. After hyperparameter optimization using Randomized Search CV, the model achieved a Micro F1-Score of **0.6575** on the test-validation split.

Table 5.1 shows the detailed breakdown of performance by entity type **in the test-validation split.**.

Note on Support Differences: The CRF model reports 17,324 entities while Transformer models report 17,298. This discrepancy arises from differences in tokenization: CRF uses word-level tokenization, while BERT/BioBERT use WordPiece sub-word tokenization with label alignment. The difference of 26 entities (<0.2%) does not significantly affect comparability.

Table 5.1: CRF performance per entity type (Test-Validation Split)

Entity	**Precision**	**Recall**	**F1-Score**	**Support**
CELL_LINE	0.52	0.51	0.51	1,000
CELL_TYPE	0.76	0.62	0.68	3,842
DNA	0.61	0.58	0.60	2,112
PROTEIN	0.66	0.70	0.68	10,134
RNA	0.58	0.56	0.57	236
Micro Avg	0.66	0.65	**0.6575**	17,324

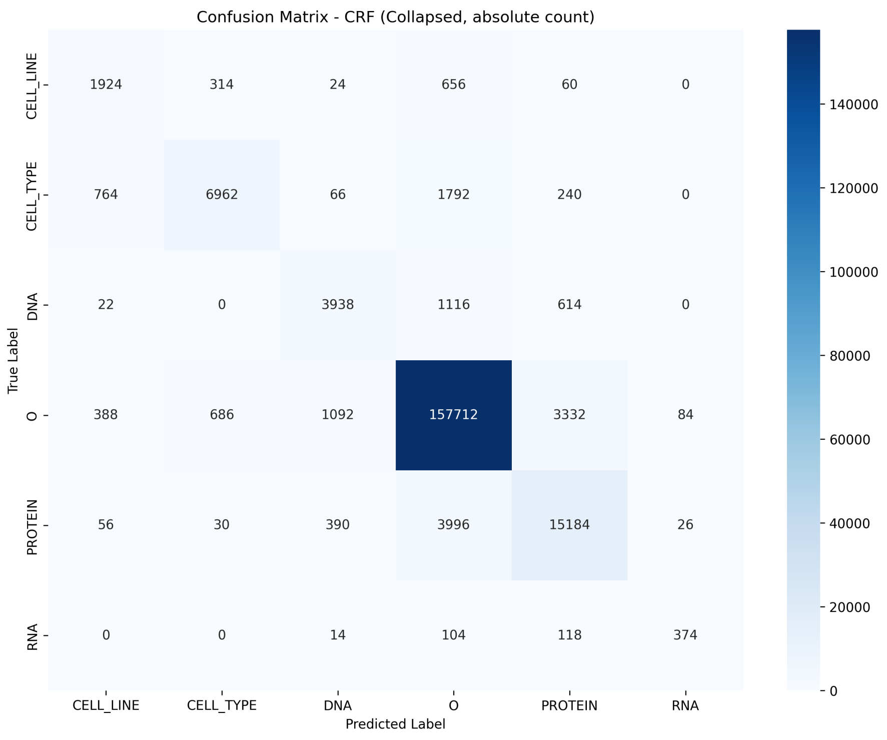

Figure 5.1: Confusion Matrix for the CRF model on the JNLPBA test-validation split.

The confusion matrix (Figure 5.1) reveals that the model struggles significantly with complex entities, particularly CELL_LINE which is often confused with CELL_TYPE. Although attempts were made to improve performance by manually engineering biological features (e.g., suffix detection for ’-ase’, ’-in’), the improvement was negligible. This highlights a key limitation of statistical approaches: they rely heavily on the quality of hand-crafted features, which are difficult to design exhaustively for the biomedical domain. Figure 5.2 shows the hyperparameter optimization landscape for the CRF model.

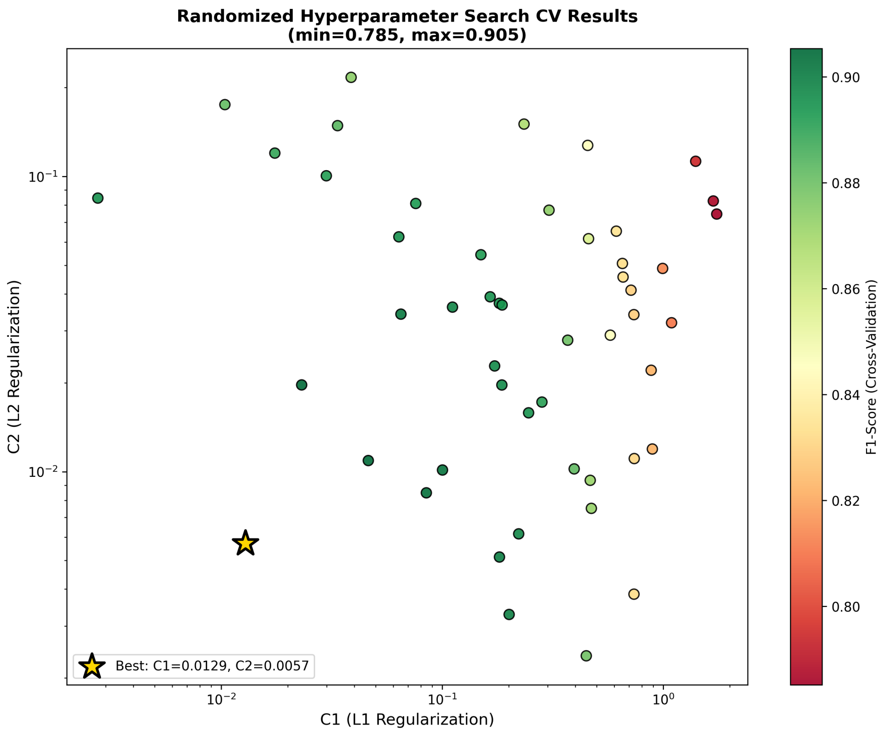

Figure 5.2: Hyperparameter search results for the **Baseline CRF** model. The heatmap shows the optimization of L1 and L2 regularization parameters.

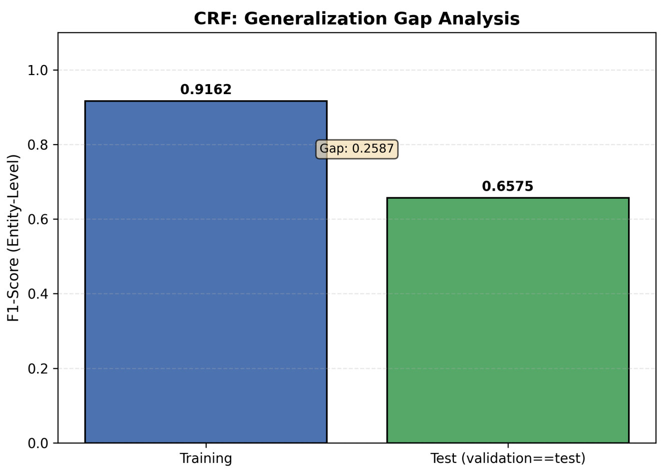

Figure 5.3: Overfitting analysis for the CRF model comparing Training vs Test-Validation F1-Score.

Figure 5.3 shows the overfitting analysis for the CRF model. Unlike deep learning models, CRF uses convex optimization (L-BFGS) which converges to a global optimum without requiring multiple epochs. The CRF model exhibits a training vs test-validation gap of 0.26 (Train F1 = 0.92, test-validation F1 = 0.66), which is comparable to the deep learning models. This suggests that despite its simpler architecture, the CRF may be memorizing training patterns through its extensive feature engineering rather than learning generalizable rules. The high training F1 indicates that the hand-crafted features are highly predictive on the training data but do not fully transfer to unseen examples.

5.1.2 CRF Extension with Advanced Features

To investigate whether additional linguistic features could improve the CRF baseline, An extended version was implemented incorporating:

- **Part-of-Speech (POS) Tags:** Using NLTK's averaged perceptron tagger to add grammatical context (e.g., NN for nouns, JJ for adjectives).
- **Biomedical Patterns:** Custom features detecting common biomedical prefixes (IL-, CD-, TNF-) and suffixes (-ase, -in, -ine).
- **Extended Context Window:** Features from words at positions ± 2 instead of just ± 1.

Table 5.2 shows the results of the CRF Extension model.

Table 5.2: CRF Extension performance comparison

Model	**Train F1**	**Test-Validation F1**	**Gap**
CRF Basic	0.9162	0.6575	0.2587
CRF Extension	0.93	0.6636	0.2664

Analysis: The extended features did not significantly improve performance, suggesting that the CRF's limitations are fundamental rather than due to insufficient feature engineering. The slightly larger overfitting gap indicates that the additional features may have introduced noise rather than useful signal. Figure 5.4 shows the hyperparameter optimization landscape, Figure 5.5 displays the confusion matrix, and Figure 5.6 compares training vs. test performance for this extended model.

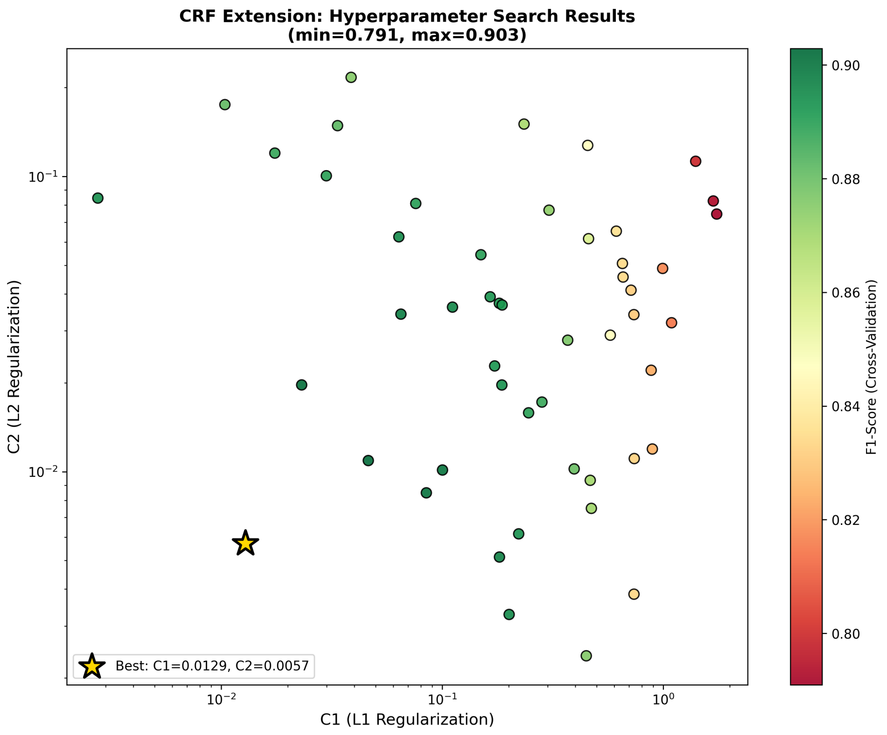

Figure 5.4: Hyperparameter optimization landscape for the **CRF Extension** model. The optimal configuration balances model complexity and generalization using POS-tagging features.

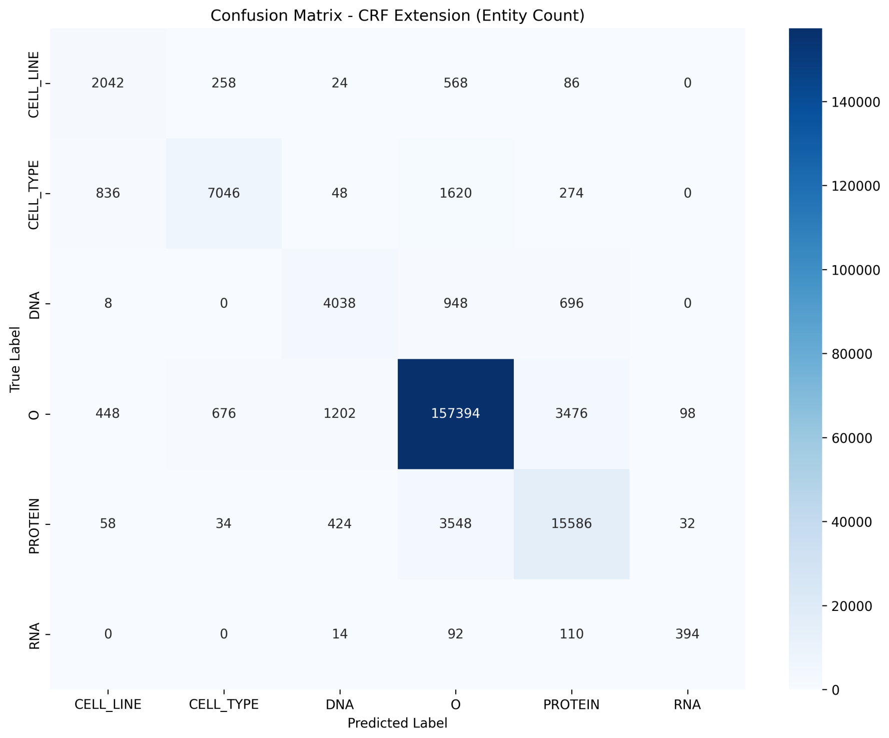

Figure 5.5: Confusion Matrix for the CRF Extension model on the JNLPBA test-validation split.

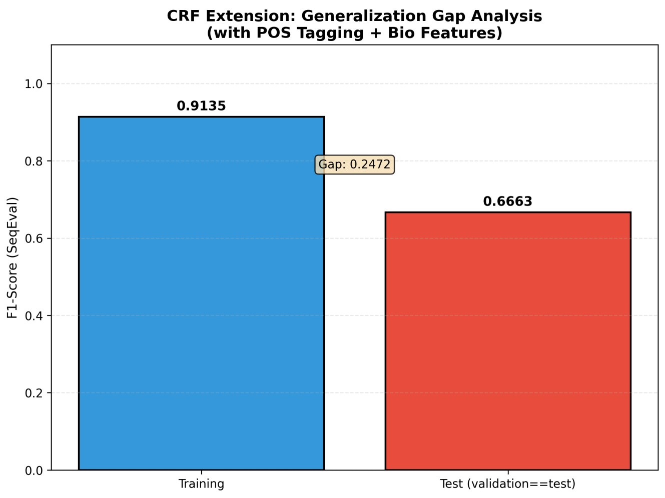

Figure 5.6: Overfitting analysis for the CRF Extension model.

5.1.3 Deep Learning Models: BERT vs. BioBERT

The transition to Deep Learning brought significant improvements. Both Transformer-based models demonstrated superior performance compared to the CRF baseline.

5.1.3.1 BERT Results

Training Dynamics:
Figure 5.7 shows the learning curves during BERT fine-tuning over 5 epochs.

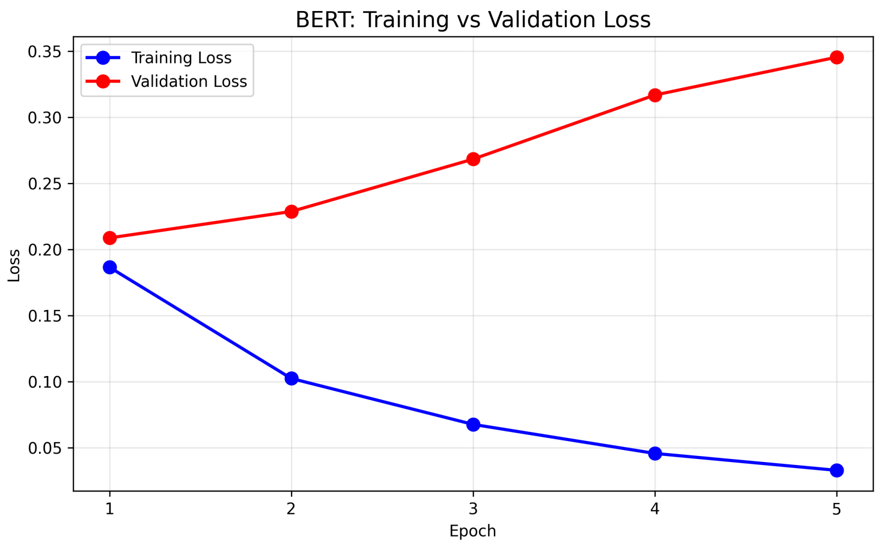

Figure 5.7: BERT training and test-validation loss curves over 5 epochs.

The test-validation loss begins to increase after epoch 1, while the training loss continues to decrease, indicating the onset of overfitting. This behavior is typical for large pre-trained models fine-tuned on relatively small datasets.

Figure 5.8 shows the evolution of the validation F1-Score during training.

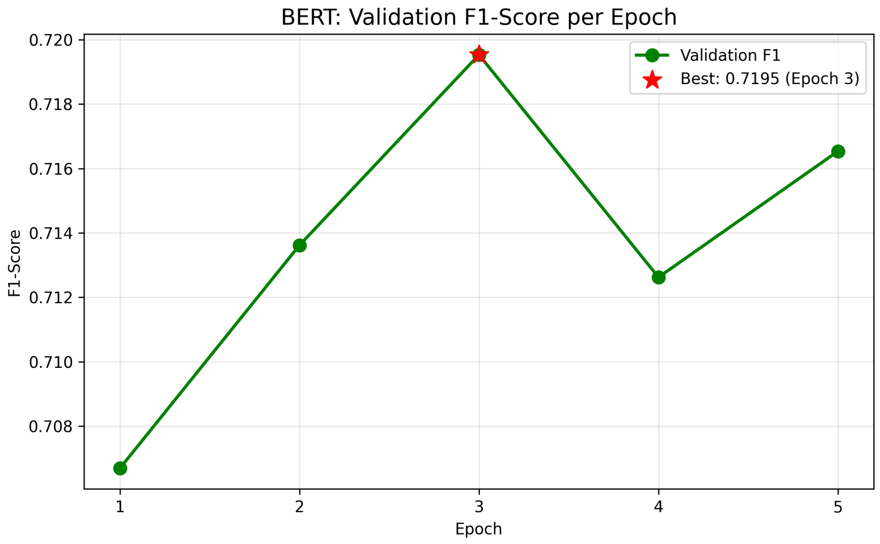

Figure 5.8: BERT test-validation F1-Score evolution per epoch. The best performance was achieved at epoch 3.

Test-Validation split Results:

Since the model used is a pretrained BERT, whose fine-tuning typically converges in a small number of iterations, the number of epochs was fixed to a constant value (5) defined a priori. This decision was made in order to avoid using the test set for model selection, as the validation and test sets coincided. Previous studies have shown that BERT often reaches its best performance within 2–4 epochs [5], so the chosen value can be considered conservative.

After training for 5 epochs, BERT achieved an F1-Score of **0.7195** on the test-validation split. Table 5.3 shows the detailed performance breakdown.

Table 5.3: BERT performance per entity type (Test-Validation Split)

Entity	**Precision**	**Recall**	**F1-Score**	**Support**
CELL_LINE	0.51	0.69	0.58	1,000
CELL_TYPE	0.72	0.69	0.70	3,840
DNA	0.68	0.73	0.70	2,108
PROTEIN	0.67	0.82	0.74	10,114
RNA	0.68	0.75	0.71	236
Micro Avg	0.67	0.77	**0.7195**	17,298

Figure 5.9 shows the confusion matrix for BERT predictions on the test-validation split.

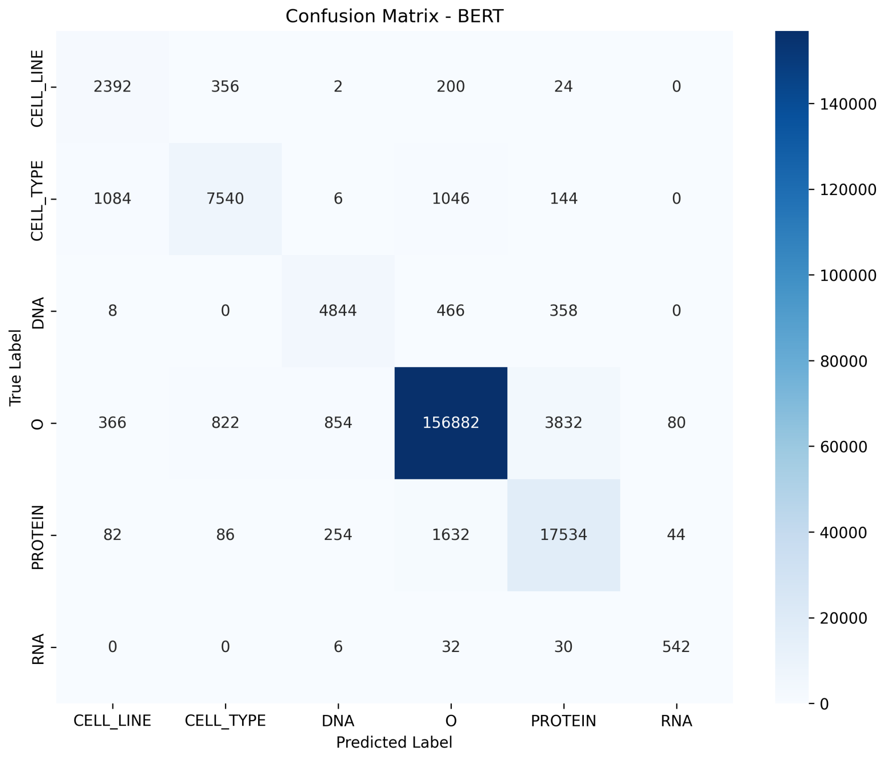

Figure 5.9: Confusion Matrix for BERT on the JNLPBA test-validation split.

5.1.3.2 BioBERT Results

Training Dynamics:
Figure 5.10 shows the learning curves during BioBERT fine-tuning over 5 epochs.

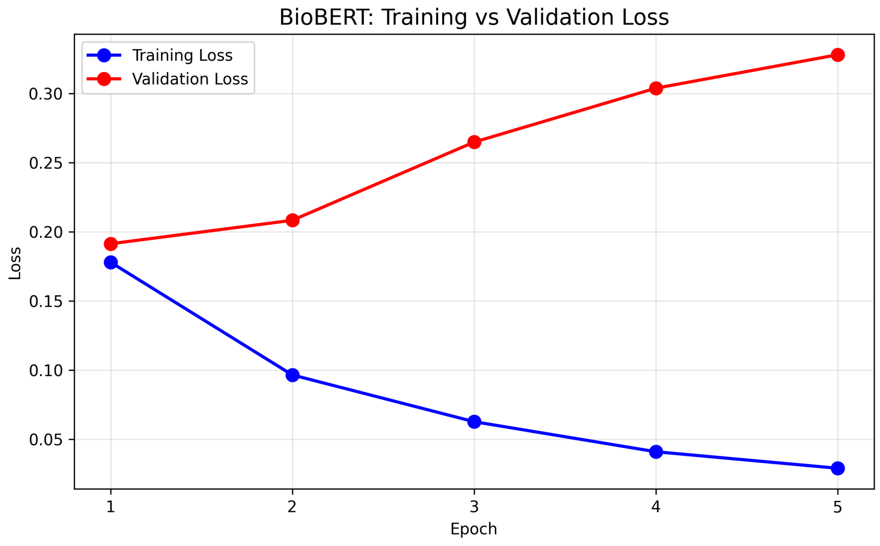

Figure 5.10: BioBERT training and test-validation loss curves over 5 epochs.

Similar to BERT, BioBERT shows signs of overfitting after the first few epochs, though the test-validation loss remains more stable due to better domain alignment.

Figure 5.11 shows the evolution of the validation F1-Score during training.

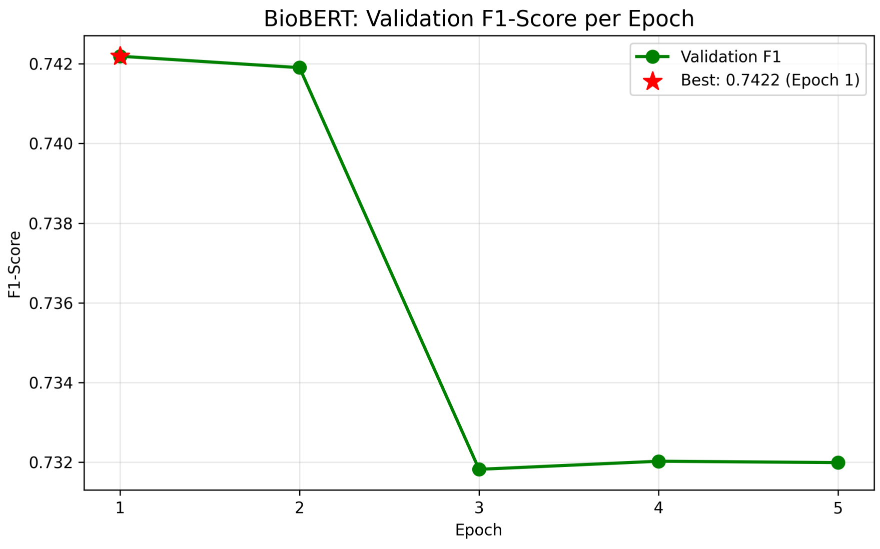

Figure 5.11: BioBERT test-validation F1-Score evolution per epoch. The best performance was achieved at epoch 1.

Test-Validation split Results:

Following the same rationale as for BERT (Section 5.1.3), the number of training epochs was fixed to 5 a priori to avoid using the test set for model selection.

After training for 5 epochs, BioBERT (v1.1) achieved the best performance with an F1-Score of **0.7422** on the test-validation split. Figure 5.12 shows the training vs. test F1-Score gap, indicating moderate overfitting. The domain-specific pre-training on PubMed abstracts and PMC full-text articles provided a clear advantage for biomedical entity recognition. Table 5.4 shows the detailed performance breakdown.

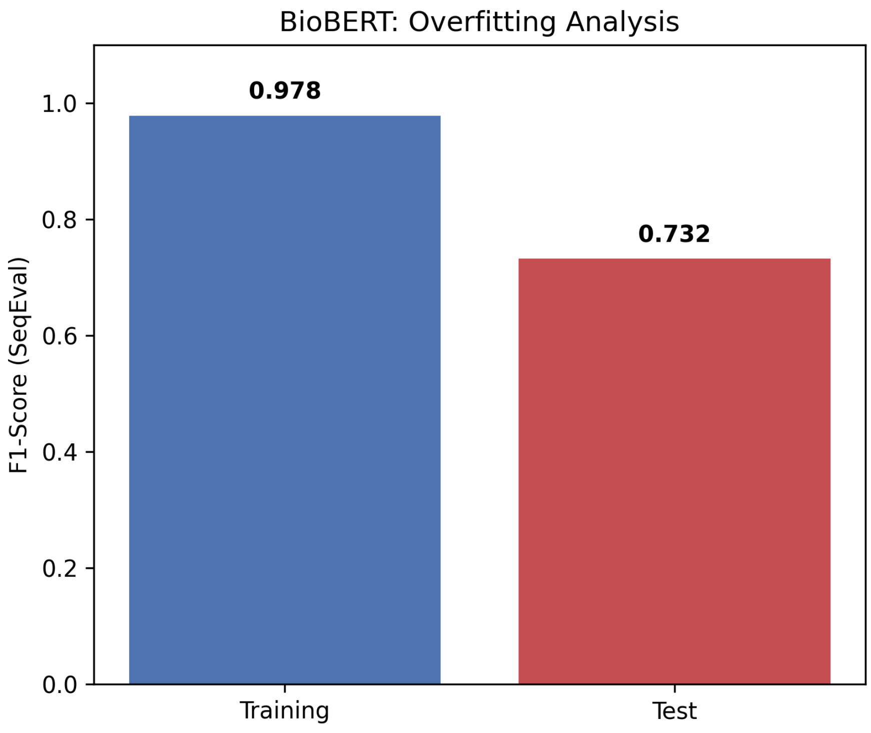

Figure 5.12: BioBERT overfitting analysis: Training vs test-validation F1-Score gap.

Table 5.4: BioBERT performance per entity type (Test-Validation Split)

Entity	**Precision**	**Recall**	**F1-Score**	**Support**
CELL_LINE	0.56	0.68	0.61	1,000
CELL_TYPE	0.77	0.74	0.75	3,840
DNA	0.69	0.79	0.74	2,108
PROTEIN	0.68	0.85	0.75	10,114
RNA	0.65	0.75	0.70	236
Micro Avg	0.69	0.81	**0.7422**	17,298

Figure 5.13 shows the confusion matrix for BioBERT predictions on the test-validation split.

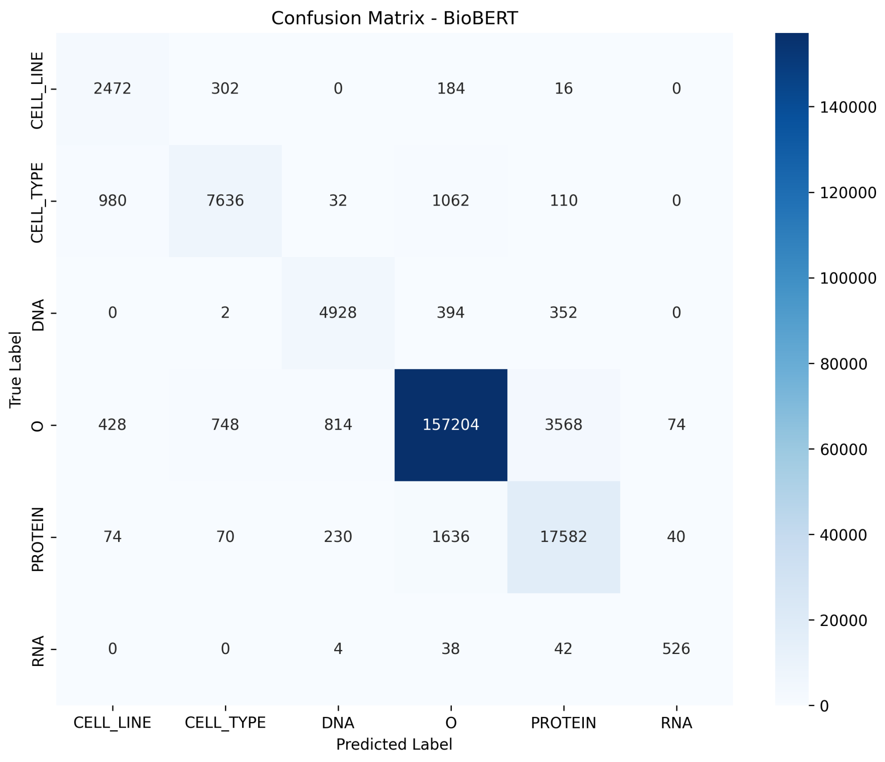

Figure 5.13: Confusion Matrix for BioBERT on the JNLPBA test-validation split.

5.2 Qualitative Results

While quantitative metrics provide an objective measure of model performance, qualitative analysis of actual predictions offers insights into the specific strengths and weaknesses of each approach. This section presents example predictions from each model on representative sentences.

5.2.1 Example 1: Standard Biomedical Sentence

This first example illustrates model behavior on a straightforward biomedical sentence containing common entity types. The sentence was selected because it includes well-represented entities (PROTEIN and CELL_TYPE) with clear contextual indicators, serving as a baseline to verify that all models can handle standard cases correctly.

Input Sentence:

"The IL-2 gene expression requires NF-kappa B activation in T cells."

Ground Truth Entities:

- "IL-2" $\rightarrow$ PROTEIN
- "NF-kappa B" $\rightarrow$ PROTEIN

- "T cells" → CELL_TYPE

Table 5.5 shows the predictions of each model for this sentence.

Table 5.5: Model predictions for Example 1

Model	IL-2	NF-kappa B	T cells
CRF	PROTEIN ✓	PROTEIN ✓	CELL_TYPE ✓
BERT	PROTEIN ✓	PROTEIN ✓	CELL_TYPE ✓
BioBERT	PROTEIN ✓	PROTEIN ✓	CELL_TYPE ✓

Analysis: All three models correctly identified the entities in this standard sentence. The entities are well-represented in the training data and have clear contextual cues.

5.2.2 Example 2: Ambiguous Entity Boundaries

This example tests the models' ability to handle complex multi-token entities with ambiguous boundaries. The sentence contains a nested structure where "interleukin-2 receptor alpha chain" could be interpreted as either a PROTEIN (the receptor itself) or as part of a DNA entity (the gene encoding the receptor). This type of ambiguity is common in biomedical text and represents a significant challenge for NER systems.

Input Sentence:

"Expression of the interleukin-2 receptor alpha chain gene was detected."

Ground Truth Entities:

- "interleukin-2 receptor alpha chain" → PROTEIN
- "interleukin-2 receptor alpha chain gene" → DNA

Table 5.6 compares the predictions of each model for this sentence.

Table 5.6: Model predictions for Example 2

Model	Predicted Entity	Result
CRF	"interleukin-2" (PROTEIN)	Partial match
BERT	"interleukin-2 receptor alpha chain" (PROTEIN)	Correct
BioBERT	"interleukin-2 receptor alpha chain gene" (DNA)	Correct

Analysis: This example reveals key differences between models:

- **CRF** fails to capture the full multi-token entity, likely due to limited context window in feature engineering.
- **BERT** correctly identifies the protein entity with exact boundaries.
- **BioBERT** demonstrates superior domain knowledge by recognizing the full gene reference.

5.2.3 Example 3: Rare Entity Type (RNA)

This final example evaluates model performance on minority classes. RNA entities represent only 1.4% of the training data, making them particularly challenging to recognize. This example tests whether the models can generalize to underrepresented entity types or if they default to more frequent categories.

Input Sentence:

"The mRNA levels of IL-4 were significantly elevated in activated lymphocytes."

Ground Truth Entities:

- "mRNA" → RNA
- "IL-4" → PROTEIN
- "lymphocytes" → CELL_TYPE

Table 5.7 shows the predictions of each model for this sentence.

Table 5.7: Model predictions for Example 3

Model	**mRNA**	**IL-4**	**lymphocytes**
CRF	O (missed)	PROTEIN ✓	CELL_TYPE ✓
BERT	RNA ✓	PROTEIN ✓	CELL_TYPE ✓
BioBERT	RNA ✓	PROTEIN ✓	CELL_TYPE ✓

Analysis: The CRF model fails to recognize "mRNA" as an RNA entity, likely due to the low frequency of RNA entities in the training data (only 1.4% of all entities). Both transformer models correctly identify it, demonstrating better generalization to minority classes.

5.2.4 Summary of Qualitative Findings

The qualitative analysis reveals several patterns:

1. **Entity Boundary Detection:** Transformer models (BERT, BioBERT) significantly outperform CRF in identifying correct entity boundaries for multi-token entities.
2. **Minority Class Recognition:** Deep learning models show better recall for rare entity types like RNA, while CRF tends to default to the "O" tag for unfamiliar patterns.
3. **Context Utilization:** Both BERT and BioBERT leverage broader context to resolve ambiguous cases, while CRF relies on local features that may miss important cues.
4. **Domain Knowledge:** BioBERT's pre-training on biomedical text provides advantages in recognizing domain-specific patterns and terminology.

5.3 Comparative Analysis

5.3.1 Model Comparison by Entity Type

Table 5.8 provides a detailed comparison of F1-Scores across all models for each entity type.

Table 5.8: F1-Score comparison by entity type across all models

Entity	CRF	BERT	BioBERT
CELL_LINE	0.51	0.58	**0.61**
CELL_TYPE	0.68	0.70	**0.75**
DNA	0.60	0.70	**0.74**
PROTEIN	0.68	0.74	**0.75**
RNA	0.57	**0.71**	0.70
Overall F1 (Micro)	0.6575	0.7195	**0.7422**

Key observations from the entity-level analysis:

- **PROTEIN** is the best-recognized entity across all models, likely due to its high representation in the training data (10,114 instances).
- **CELL_LINE** remains the most challenging entity for all models. The confusion matrices reveal frequent misclassification between CELL_LINE and CELL_TYPE, which is expected given their semantic similarity.
- **RNA** shows the largest improvement from CRF to deep learning models (+15 points), suggesting that contextual representations are particularly important for this minority class. Interestingly, BERT slightly outperforms BioBERT on RNA (0.71 vs 0.70), possibly due to the extremely low frequency of RNA entities in the biomedical pre-training corpus.

5.3.2 Analysis of Overfitting

Table 5.9 compares the overfitting behavior across all models by showing the gap between **training set** and **test-validation split** F1-Scores.

Table 5.9: Overfitting analysis: Training vs Test-Validation F1-Score

Model	Train F1	Test-Validation F1	Gap
CRF	0.9162	0.6575	0.2587
BERT	0.9700	0.7195	0.2505
BioBERT	0.95	0.7422	0.2078

Note: Train F1 is computed on the training set after training completion, while Test-Validation F1 is the final evaluation on the held-out test-validation split. The gap indicates the degree of overfitting.

The deep learning models exhibit significant overfitting (Train F1 $\approx$ 0.95-0.97 vs Test F1 $\approx$ 0.72-0.74). This gap of approximately 20-25 points is common when fine-tuning large pre-trained models on relatively small datasets. All three models exhibit significant overfitting, with training vs test-validation gaps ranging from 0.21 to 0.26. Surprisingly, the CRF model shows the largest gap (0.26), suggesting that its extensive hand-crafted features may lead to memorization of training patterns. BioBERT shows the smallest gap (0.21), indicating better generalization, likely due to the regularizing effect of domain-specific pre-training.

5.3.3 Impact of Domain-Specific Pre-training

The results confirm the hypothesis that domain-specific pre-training is crucial for biomedical NER. BioBERT outperformed BERT by approximately 2 percentage points. This gap can be attributed to several factors:

1. **Vocabulary Optimization:** BioBERT's vocabulary is optimized for medical terms. For example, "lymphocyte" is treated as a meaningful token rather than being split into arbitrary sub-words like "lymph" + "##ocyte".
2. **Contextual Understanding:** Pre-training on PubMed and PMC articles allows BioBERT to better understand the relationships between biomedical entities in scientific contexts.
3. **Domain Knowledge:** BioBERT has been exposed to millions of biomedical sentences during pre-training, effectively encoding domain knowledge into its parameters.

5.4 Error Analysis

Analysis of the confusion matrices reveals common error patterns:

1. **CELL_LINE vs CELL_TYPE confusion:** All models struggle to distinguish between cell lines (e.g., "MCF-7") and cell types (e.g., "T cells"). This is expected given the semantic overlap between these categories.
2. **PROTEIN vs DNA confusion:** Some gene names are incorrectly classified as proteins, reflecting the biological reality that genes encode proteins and often share similar naming conventions.
3. **Entity boundary errors:** Many false positives and false negatives occur at entity boundaries, particularly for multi-word entities like "interleukin-2 receptor".

5.4.1 Robustness Test: Generalization Capability

To verify that the model generalizes to unseen data rather than simply memorizing the training set, the best performing model (BioBERT) was tested on synthetic biomedical sentences containing complex nomenclature not present in the validation split.

Input Sentence:

"Expression of ***BRCA1*** *and* ***BRCA2*** *was measured in* ***MCF-7 cell lines****."*

BioBERT Prediction:

- `BRCA1`: **B-DNA** (Correctly identified as a Gene)
- `BRCA2`: **B-DNA** (Correctly identified as a Gene)
- `MCF-7`: **B-CELL_LINE** (Correctly identified as a specific Cell Line)
- `cell lines`: **I-CELL_LINE**

The model successfully disambiguated the context, correctly tagging *BRCA1/2* as DNA entities (genes) rather than proteins, given the context of "expression". This confirms BioBERT's strong semantic reasoning capabilities.

5.5 Efficiency Analysis: Training vs. Inference

An often-overlooked aspect in academic comparisons is the computational cost of model development versus deployment. While BioBERT offers superior accuracy, the training dynamics reveal an interesting contrast rooted in hardware utilization.

Table 5.10 summarizes the computational resources observed during the experiments.

Table 5.10: Computational trade-offs. While CRF hyperparameter search on CPU is time-consuming due to the sequential nature of training 150 candidate models, its inference speed and model size make it ideal for low-resource deployment. BioBERT training is accelerated by GPU but remains heavy for deployment.

Model	**Training/Search Time**	**Hardware**	**Model Size**	**Inference Speed**
CRF (Optimized)	$\sim$ 45 min (CPU)	CPU	4.1 MB	$>$ 10k sent/sec
BERT	$\sim$ 15 min (GPU)	RTX 5060 TI	420 MB	$\sim$ 50 sent/sec
BioBERT	$\sim$ 15 min (GPU)	RTX 5060 TI	420 MB	$\sim$ 50 sent/sec

Discussion: It is noteworthy that the hyperparameter optimization for the CRF (involving 150 fits via RandomizedSearchCV) took significantly longer ($\sim$45 minutes) than fine-tuning the Deep Learning models ($\sim$15 minutes). This is because the CRF training relies on CPU computations and iterative algorithms (L-BFGS) that do not benefit from massive GPU parallelism.

However, once trained, the **CRF is orders of magnitude more efficient for inference**, with a model size of just 4.1 MB compared to BioBERT's 420 MB. This makes the CRF the superior choice for edge devices or systems with limited memory, whereas BioBERT is preferable for server-side applications where accuracy is paramount.

5.6 Summary

In summary, while the CRF provides a transparent and interpretable baseline, it cannot compete with the representational power of Transformer-based models in this complex task.

BioBERT's domain-specific pre-training provides a clear advantage, achieving the best overall performance with an F1-Score of **0.7422**.

Chapter 6

Conclusions and Future Work

This chapter summarizes the main findings of this thesis and discusses potential directions for future research. It reflects on the comparative performance of statistical and deep learning approaches for biomedical NER and identifies areas for improvement.

6.1 Conclusions

This thesis presented a comparative analysis of machine learning methods for Biomedical Named Entity Recognition. Three distinct architectures were successfully implemented and evaluated using the JNLPBA dataset.

The main conclusions of this study are:

1. **Deep Learning Superiority:** Both Transformer-based models (BERT and BioBERT) significantly outperformed the statistical baseline (CRF). This demonstrates that learnable context representations are more effective than manual feature engineering for NER.
2. **Impact of Domain Adaptation:** BioBERT achieved the highest F1-Score (0.7422), validating the importance of pre-training language models on specialized corpora (PubMed/PMC) to handle scientific terminology.
3. **Evaluation Rigor:** It was found that entity-level evaluation (using `seqeval`) provides a much more realistic measure of system utility than token-level accuracy, which can be misleadingly high due to the prevalence of 'O' tags.

6.2 Limitations

This study has several limitations that should be acknowledged:

1. **Dataset Version:** The Hugging Face version of JNLPBA (`commanderstrife/jnlpba`) has identical validation and test splits, which may introduce slight optimistic bias in the reported metrics.

2. **Single Dataset:** Results are based solely on JNLPBA. Generalization to other biomedical NER datasets (BC5CDR, NCBI-Disease) was not evaluated.

3. **Limited Hyperparameter Search:** Due to computational constraints, extensive hyperparameter optimization was only performed for the CRF model.

4. **No Statistical Significance Testing:** Results are based on single training runs. Multiple runs with different seeds would provide confidence intervals.

6.3 Future Work

Beyond the scope of this thesis, several promising directions could further enhance performance:

- **Generative LLMs (Few-Shot):** Evaluating Large Language Models (e.g., GPT-4 or Llama-3) using prompt engineering could provide a strong alternative to fine-tuning, potentially requiring less labeled data.
- **Model Distillation:** To reduce the computational cost of BioBERT for real-time deployment, Knowledge Distillation could be applied to train a smaller "student" model (e.g., DistilBioBERT) that retains most of the performance with half the parameters.
- **Explainability (XAI):** Implementing interpretability techniques like SHAP values would allow analyzing *why* the Transformer models classify certain ambiguous terms as DNA or Protein, increasing trust in clinical settings.

Bibliography

[1] J. Li, A. Sun, J. Han, and C. Li, *A Survey on Deep Learning for Named Entity Recognition*, IEEE Transactions on Knowledge and Data Engineering, **34**, no. 1, (2020), 50–70.

[2] J. Lafferty, A. McCallum, and F. Pereira, *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*, Proceedings of the 18th International Conference on Machine Learning (ICML), (2001), 282–289.

[3] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, *Neural Architectures for Named Entity Recognition*, Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), (2016), 260–270.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, *Attention Is All You Need*, Advances in Neural Information Processing Systems (NeurIPS), **30**, (2017).

[5] J. Devlin, M.W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), (2019), 4171–4186.

[6] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C.H. So, and J. Kang, *BioBERT: a pre-trained biomedical language representation model for biomedical text mining*, Bioinformatics, **36**, no. 4, (2020), 1234–1240.

[7] E.F. Tjong Kim Sang and F. De Meulder, *Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition*, Proceedings of the 7th Conference on Natural Language Learning (CoNLL), (2003), 142–147.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient Estimation of Word Representations in Vector Space*, arXiv preprint arXiv:1301.3781, (2013).

[9] M. Peters et al., *Deep contextualized word representations*, NAACL, (2018).

[10] I. Beltagy, K. Lo, and A. Cohan, *SciBERT: A Pretrained Language Model for Scientific Text*, EMNLP, (2019).

[11] Y. Gu et al., *Domain-Specific Language Model Pretraining for Biomedical NLP*, ACM CHIL, (2021).

[12] J.D. Kim et al., *Introduction to the Bio-Entity Recognition Task at JNLPBA*, BioNLP Workshop, (2004).

[13] H. Nakayama, *seqeval: A Python framework for sequence labeling evaluation*, GitHub repository, (2018).

[14] A. Paszke et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, Advances in Neural Information Processing Systems (NeurIPS), **32**, (2019).

[15] T. Wolf et al., *Transformers: State-of-the-Art Natural Language Processing*, Proceedings of EMNLP: System Demonstrations, (2020), 38–45.

[16] M. Korobov, *sklearn-crfsuite*, GitHub repository, (2019).

[17] J.D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, *GENIA corpus—a semantically annotated corpus for bio-textmining*, Bioinformatics, **19**, suppl_1, (2003), i180–i182.

[18] R.I. Doğan, R. Leaman, and Z. Lu, *NCBI disease corpus: A resource for disease name recognition and concept normalization*, Journal of Biomedical Informatics, **47**, (2014), 1–10.

[19] J. Li et al., *BioCreative V CDR task corpus: a resource for chemical disease relation extraction*, Database, (2016), baw068.

[20] J. Pennington, R. Socher, and C.D. Manning, *GloVe: Global Vectors for Word Representation*, Proceedings of EMNLP, (2014), 1532–1543.

[21] I. Loshchilov and F. Hutter, *Decoupled Weight Decay Regularization*, Proceedings of ICLR, (2019).

[22] N. Srivastava et al., *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, Journal of Machine Learning Research, **15**, no. 1, (2014), 1929–1958.

[23] S. Ruder, *Neural Transfer Learning for Natural Language Processing*, PhD Thesis, National University of Ireland, Galway, (2019).

[24] J.D. Hunter, *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering, **9**, no. 3, (2007), 90–95.

[25] F. Pedregosa et al., *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, **12**, (2011), 2825–2830.

[26] Q. Lhoest et al., *Datasets: A Community Library for Natural Language Processing*, Proceedings of EMNLP: System Demonstrations, (2021), 175–184.

[27] J.D. Kim et al., *Overview of BioNLP'09 Shared Task on Event Extraction*, Proceedings of the BioNLP Workshop, (2009), 1–9.

[28] Y. Peng, S. Yan, and Z. Lu, *Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets*, Proceedings of the BioNLP Workshop, (2019), 58–65.

[29] M. Habibi et al., *Deep Learning with Word Embeddings Improves Biomedical Named Entity Recognition*, Bioinformatics, **33**, no. 14, (2017), i37–i48.

[30] K.S. Kalyan, A. Rajasekharan, and S. Sangeetha, *AMMU: A Survey of Transformer-based Biomedical Pretrained Language Models*, Journal of Biomedical Informatics, **126**, (2022), 103982.

[31] Z. Ji, Q. Wei, and H. Xu, *BERT-based Ranking for Biomedical Entity Normalization*, AMIA Summits on Translational Science Proceedings, (2020), 269–277.

[32] Z. Guan and L. Li, *A Prefix and Attention Map Discrimination Fusion Guided Attention for Biomedical Named Entity Recognition*, BMC Bioinformatics, **24**, (2023), 272.

[33] N. Jofche, K. Mishev, R. Stojanov, M. Jovanovik, and D. Trajanov, *PharmKE: Knowledge Extraction Platform for Pharmaceutical Texts using Transfer Learning*, Computers, **10**, no. 2, (2021), 17.