USING CONTRASTIVE LEARNING AND LARGE LANGUAGE MODELS ON BIOMEDICAL INFORMATION EXTRACTION

by

Buğrahan Şahin

B.S., Computer Engineering, Boğaziçi University, 2016

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2024

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor, Professor Arzucan Özgür, without whom writing this thesis would not be possible. Thanks for your guidance and support. Thanks for being one of the most successful and humble academicians I have ever known. You are one of the most outstanding examples for us to follow.

I would like to thank Berke Kavak for sharing his work with me, for helping me progress, and for always being supportive and encouraging. He is definitely one of the best people I have worked with.

My wife, Gözde, always had my back. Thanks, love, for your endless support, for all the happiness, and your eyes.

Thanks, Mila, my daughter. Right now, you are minus 18 weeks of age, with 480 grams in the last measurement. Thank you for all the sweet kicks I could feel when the world was dark.

I want to thank my grandfather İsmail, from whom I learned what strength is, and my grandmother Hacer, from whom I learned what a good heart is.

I want to thank my mother, Yıldız; I owe her more than I can list. I want to thank my father, Abdurrahman, for teaching me how to ride a motorcycle.

ABSTRACT

USING CONTRASTIVE LEARNING AND LARGE LANGUAGE MODELS ON BIOMEDICAL INFORMATION EXTRACTION

As the volume of human knowledge grows at an extraordinary rate, it is becoming progressively challenging to fully comprehend and utilize the vast array of information available, even within a specialized domain. The field of Biomedicine exemplifies this, with thousands of research papers published each year contributing new insights about species, diseases, and chemicals. Such growth has created an overwhelming need for effective Information Retrieval systems, coupled with Natural Language Processing techniques, to enable the extraction and use of relevant data at a scale beyond human capacity. Effective Information Retrieval in Biomedicine demands an understanding of published research at a granular level, which involves solving several interconnected challenges. For each Biomedical abstract, it is necessary to recognize entities, normalize these entities with standardized identifiers, and extract relationships between different types of entities. These tasks facilitate a structured representation of knowledge and make complex Biomedical information more accessible for scientific and clinical applications.

In this study, we investigate and enhance the performance of an existing normalization tool, BioNEN [1], by incorporating Contrastive Learning techniques. We explore the integration of dictionaries, Contrastive Learning, and Large Language Models (LLMs) to improve entity recognition, and we examine the use of LLMs for extracting relationships between entities. The result is a streamlined tool designed to identify and normalize entities in Biomedical abstracts and to effectively extract relationships, thereby enabling advancement of Biomedical Information Retrieval systems.

ÖZET

KARŞILAŞTIRMALI ÖĞRENME VE GENİŞ DİL MODELLERİNİN BİYOMEDİKAL BİLGİ ÇIKARILMASINDA KULLANILMASI

İnsan bilgisinin hacmi olağanüstü bir hızla arttıkça, özel bir alanda bile mevcut çok çeşitli bilgileri tam olarak anlamak ve kullanmak giderek zorlaşmaktadır. Biyotıp alanı, her yıl yayınlanan ve türler, hastalıklar ve kimyasallar hakkında yeni anlayışlara katkıda bulunan binlerce araştırma makalesiyle buna örnek teşkil etmektedir. Bu büyüme, ilgili verilerin insan kapasitesinin ötesinde bir ölçekte çıkarılmasını ve kullanılmasını mümkün kılmak için Doğal Dil İşleme teknikleriyle birleştirilmiş etkili Bilgi Erişim sistemlerine yönelik büyük bir ihtiyaç yaratmıştır. Biyotıpta Etkili Bilgi Erişimi, birbiriyle bağlantılı birçok zorluğun çözülmesini içeren, yayınlanmış araştırmaların ayrıntılı düzeyde anlaşılmasını gerektirmektedir. Her Biyomedikal özet için varlıkları tanımak, bu varlıkları standartlaştırılmış tanımlayıcılarla normalleştirmek ve farklı varlık türleri arasındaki ilişkileri çıkarmak gerekmektedir. Bu görevlerin tamamlanması, bilginin yapılandırılmış bir temsilini kolaylaştırmakta ve karmaşık Biyomedikal bilgileri bilimsel ve klinik uygulamalar için daha erişilebilir hale getirmektedir.

Biz bu çalışmada, mevcut bir normalleştirme aracı olan BioNEN [1]'nin performansını Karşılaştırmalı Öğrenme tekniklerini dahil ederek araştırıyor ve geliştiriyoruz. Varlık tanımayı geliştirmek için sözlüklerin, Karşılaştırmalı Öğrenmenin ve Büyük Dil Modellerinin (LLM'ler) entegrasyonunu araştırıyoruz ve varlıklar arasındaki ilişkileri çıkarmak için LLM'lerin kullanımını inceliyoruz. Sonuç olarak, Biyomedikal özetlerdeki varlıkları tanımlamak ve normalleştirmek ve ilişkileri etkili bir şekilde çıkarmak, böylece Biyomedikal Bilgi Erişim sistemlerinin kapasitesinin geliştirilmesini sağlamak için tasarlanmış geliştirilmiş bir aracı sizlere sunuyoruz.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathrm{d}(\mathrm{x}_{i1}, x_{i2})$	Euclidean distance between embeddings x_{i1} and x_{i2}
m	Margin for dissimilar pairs
N	Total number of samples
p	Softmax probability for a pair
scaled_similarity	Rescaled similarity to avoid numerical instability
similarity(x_1, x_2)	Cosine similarity between embeddings x_1 and x_2
T	Temperature scaling factor
y_i	Label (1 for similar pairs, 0 for dissimilar pairs)
$\mathcal{L}$	Average loss across all samples
$\mathcal{L}_{\text{Euclidean-based}}$	Euclidean-based contrastive loss
$\mathcal{L}_{\text{InfoNCE}}$	InfoNCE loss
$\in$	Mathematical symbol for representing set membership
$\sum$	Mathematical symbol for representing sum of all entities

LIST OF ACRONYMS/ABBREVIATIONS

AI	Artificial Intelligence
API	Application Programming Interface
BART	Bidirectional and Auto-regressive Transformers
BC5CDR	BioCreative V Chemical Disease Relation
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
CRF	Conditional Random Fields
CSR	Contrastive Similarity Recognition
DBScan	Density Based Spatial Clustering of Applications with Noise
GPT	Generative Pretrained Transformer
ID	Identifier
InfoNCE	Information Noise-Contrastive Estimation
JBLPBA	Joint Workshop on Natuarl Language Processing in Biomedicine and its Applications
LLM	Large Language Models
LSTM	Long Short-Term Memory
MeSH	Medical Subject Headings
MNLI	Multi-Genre Natural Language Inference
NCBI	National Center for Biotechnology Information
NEN	Named Entity Normalization
NER	Named Entity Recognition
NLM	National Library of Medicine
NLP	Natural Language Processing
NN	Neural Networks
NT-Xent	Normalized Temperature-scaled Cross-Entropy Loss
PubMed	Public Medicine Database
RE	Relation Extraction

RF	Relevance Filter
SimCLR	Simple Framework for Contrastive Learning
SR4GN	Scientific Research for Gene Normalization
SupCon	Supervised Contrastive Learning
UNESCO	United Nations Educational, Scientific, and Cultural Organization

1. INTRODUCTION

Making use of large-scale data is a very challenging task. The size of available data has already reached incomprehensible amounts. The total digital data is projected to reach 175 zettabytes by 2025 [2]. UNESCO reported that only in 2021, at least 2.7 million books were published [3]. Healthcare data is estimated to surpass 2.3 zettabytes by 2025 [4].

Today, Large Language Models tackle this task effectively. LLMs are trained on wide-scope data, and they solve problems from coding and math to recognizing entities and writing poems. ChatGPT [5] is integrated into GitHub Copilot, Office 365, and even the famous language learning application Duolingo. Gemini [6] is integrated with Google Workspace and provides features like Smart Compose and Smart Reply. Med-Palm2 is being tested in assisting doctors. LLaMA [7] is available as an open-source model, and it is being used by startups worldwide in different business areas. On the other end of the spectrum, specialized Machine Learning approaches, like Neural Network models, are making their way into the products we use every day. Tesla Autopilot relies on specialized Neural Networks, Google Photos uses advanced ML algorithms, Apple Face ID uses 3D Convolutional Neural Networks, and Netflix uses a combination of multiple ML approaches to recommend our favorite movies.

In the Biomedical domain, there exists a rich history of developing specialized sets of tools. It is only natural to enhance these tools with the help of LLMs, and we, in fact, see many researchers explore LLM capabilities in the Biomedical domain as of 2024. PubMedGPT built on [8] is developed to summarize Biomedical literature. BioGPT [9] specializes in answering questions for the Biomedical domain; MedPaLM [6] is developed to help with medical diagnosis and decision-making. Similar to these examples, we plan to bring specialized Biomedical approaches together with LLMs.

Our grand aim is to build an easy-to-use annotator that will recognize and normalize entities and find relations between found entities. To the best of our knowledge, such a tool does not exist.

In order to build such an effective product, we need to solve all three main problems in Biomedical text annotation, Entity Recognition, Entity Normalization, and Relation Extraction, and combine them into a single tool. This is a very challenging task, as each of these problems face significant linguistic challenges.

Entity Recognition is natural to humans. When we read a text, we understand the subject, verb, and sentence structure all at once automatically. When we see "human", "Human", "humans", or "mankind", we understand it all refers to the same entity. For computers, the same can not be said, as exact matches would conclude that none of the entities above is the same as another entity.

Normalization carries all the same challenges between found entities and available dictionaries that map those entities to their respective IDs. For reference, the NCBI taxonomy database [10] contains taxonomy IDs for the species. As humans, we would need to find the entity we recognized in this dictionary to retrieve its ID. While we are not limited to exact matches, finding the IDs would take time, and we would eventually abandon them. Computers have the same issue with exact matches; they need to tackle issues with synonymy, paraphrasing, abbreviation, capitalized letters, and plural singular forms, and they should be able to find the matching entity in the dictionary we are searching the ID in.

Relation Extraction is even more complicated, as grammar, language-specific challenges, negations, and sometimes even sarcasm need to be handled. LLMs' are applied to a broad spectrum of tasks, but what they do best, subjectively, is that they understand the given prompts, which means that they understand the entities we talk about and relations in between. Therefore, we foresaw that using LLMs to extract relations between recognized and normalized entities would produce a useful tool.

To solve such linguistic and semantic problems inherent to Recognition, Normalization and Relation Extraction tasks, we continued to build upon an existing high-accuracy normalization tool named BioNEN [1]. BioNEN has already shown significant performance in addressing the challenges of Normalization. However, we carried the accuracy of this tool even further using Contrastive Learning. We also used BioNEN to recognize entities that had not been explored before. We enriched the resulting Recognition tool by introducing LLM capabilities and compared the performance of LLMs with the inlined methods coming from the normalization tool. We based our LLM Relation Extraction on already annotated abstracts, with the aim of combining this method with our Recognition and Normalization tools. In the end, we combined it all in our easy-to-use annotator.

1.1. BioNEN

BioNEN [1] is an Entity Normalization tool developed by combining clustering and text similarly on embeddings. Three different similarity methods are included in the study and can be configurable in the tool produced:

- Jaccard Similarity [11] which is defined as

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}. \tag{1.1}$$

- Levenshtein Similarity [12] which is defined as

$$S_{Levenshtein}(A,B) = 1 - \frac{d(A,B)}{\max(|A|,|B|)}. \tag{1.2}$$

- Jaro-Winkler Similarity [13] which is defined as

$$S_{JW} = S_{Jaro} + (l \cdot p \cdot (1 - S_{Jaro}))\,. \tag{1.3}$$

For the purposes of similarity calculations, four different embedding methods were explored in the study:

- Term Frequency - Inverse Document Frequency [14] which is defined as

$$TF(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}. \tag{1.4}$$

- Word2Vec [15]
- Transformers [16]
- Bidirectional Encoder Representations from Transformers (BERT) [17]

Moreover, it can be similarly configured in the tool. To tackle the clustering problem, three different similarities:

- DBScan [18]
- Cosine Similarity which is defined as

$$\text{Cosine Similarity}(A,B) = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}. \tag{1.5}$$

- Jaro-Winkler Similarity [13]

Are employed, and the toolset is tested on the following datasets:

- NCBI-Disease [19]
- BC5CDR-Disease [20]
- SYMPTEMIST [21]

As a result of this work, BioNEN has been implemented and made generally available. As we build on BioNEN, we will use the specialized disease dictionary produced for BioNEN, as well as NCBI-Disease and BC5CDR-Disease datasets.

1.2. Outline

We followed a four-step plan:

- 1: Enhance BioNEN [1] framework with Contrastive Learning for disease Entity Normalization.
- 2: Apply methods like dictionary and Contrastive Learning, used in BioNEN, to recognize entities.
- 3: Explore capabilities of LLM on Entity Recognition, compare to dictionary, and Contrastive Learning.
- 4: Explore the capabilities of LLM on extracting relations for already tagged abstracts.

With the aim of following outcomes:

- 1: To improve the accuracy of BioNEN even further.
- 2: To have a working tool for Entity Recognition and perform recall/precision analysis.
- 3: To have a working tool employing LLMs for Relation Extraction on annotated abstracts.

Finally, our ultimate aim was to combine all of these in a streamlined tool to process a given abstract, produce a list of recognized and normalized entities, and find the interactions between pairs.

After working on three different tasks, NER, NEN, and RE, separately, we constructed our three-tool research sets and combined them into a single tool. A tool that processes the given abstract recognizes different types of entities, matches them to their respective IDs, and extracts interactions between these found entities.

The power of our tool is that our normalization is handled by a high-performing normalization tool, and our recognition and Relation Extraction capabilities will automatically increase over time as LLMs progress.

2. BACKGROUND

2.1. Named Entity Recognition

Recognition of entities is finding subtexts within a given text, in our case, the abstract of a scientific paper, that defines a known entity, such as "human".

In Biomedical data, there are many classes of entities, including diseases, drugs, chemicals, cell types, species, mutations, phenotypes, and so on. For species, there are 1.2 million entities scientifically described and categorized. The number of diseases is in the order of tens of thousands. The number of treatments is in the hundreds. As a result, Entity Recognition in the Biomedical domain is a classification problem with millions of class labels.

Entity Recognition is one of the critical problems to solve in order to make sense of any text. A broad spectrum of studies solve NER without specializing in entity type to the domain. [22] used bidirectional LSTM-CNN, where CNN is used for character-level recognition of the entity, and LSTM is used to understand the context related to the entity. [23] used contextual string embeddings to capture the effect of context. [17] introduced the BERT model and showed how it can be fine-tuned for the NER task. In the Biomedical domain, [24] specialized BERT for Biomedical text. [25] provided different models that can solve NER and multiple other NLP tasks. [26] focused on using LSTM networks to recognize chemical entities. [27] combined three different model types: CNN, BiLSTM, and CRF.

Tools we produce in this study apply to all types of entities, given the dictionary and training dataset for the type, but following [1], we developed them on diseases, and we have thousands of diseases to identify, meaning we have thousands of possible class labels.

2.1.1. Recognition Example

For illustration purposes, we recognized the entities in the study titled *"Biochemical effects of Solidago virgaurea extract on experimental cardiotoxicity"* [28]. Several chemicals and diseases were identified.

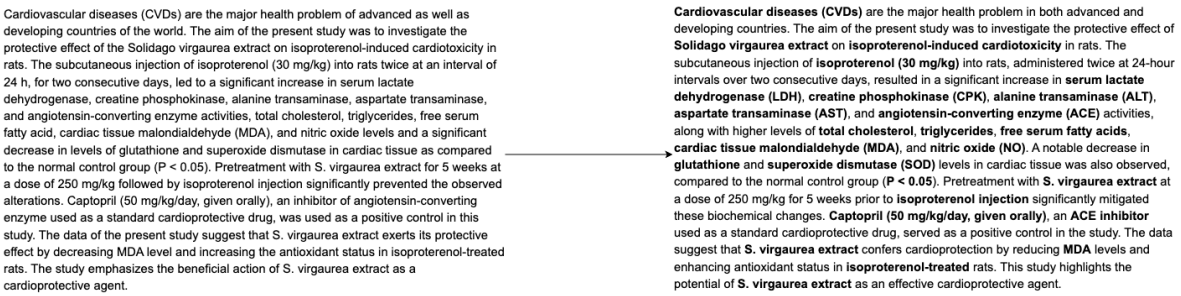


Figure 2.1. Named Entity Recognition visualized.

2.1.1.1. Chemicals.

- Isoproterenol (ISO)
- Cholesterol
- Triglyceride
- Malondialdehyde (MDA)
- Nitric oxide (NO)
- Glutathione
- Superoxide dismutase (SOD)
- Captopril

2.1.1.2. Diseases.

- Cardiovascular diseases (CVDs)
- Cardiotoxicity

2.2. Named Entity Normalization

Entity Normalization is converting text from entity names to their respective IDs from scientific literature; in our study, following [1], we map diseases to MeSH IDs.

There are thousands of labels difference between exactly matching the entities for normalization and including approximations, acronyms, and synonyms. With exact matching, normalization is a yes or no problem; if the entity is found in the dictionary for available scientific IDs, we retrieve it; if not, we fail to normalize. When we do not constrain to exact matching, just like Entity Recognition, we must solve a classification problem with millions of class labels. We are scoped down to diseases for the datasets we used in Entity Normalization, and we have thousands of possible labels.

Many researchers have tried to solve the normalization problem with various methods. BioNEN [1] combined dictionary-based methods with similarity-based methods to improve performance. [29] combined dictionary-based learning with Neural Networks. [30] remodeled the normalization as a ranking task to tackle the problem. [31] used dense vector representations, modeled the problem as a context-entity problem, and trained a model to retrieve an entity for the given novel context. [32] used self-attention networks and redirected attention to focus on the most defining parts of the abstract to match predefined IDs.

In this study, we plan to build on BioNEN [1], a combination of dictionary-based and similarity-based methods, and combine it with Contrastive Learning. Since this tool is unique, this contribution is also a unique application of Contrastive Learning.

2.2.1. Normalization Example

Following up the same example, normalizing the entities found in [28], we match the entities to their scientific identifiers.

2.2.1.1. Normalized Chemicals.

- Isoproterenol (ISO) - [CID 3779]
- Cholesterol - [CID 5997]
- Triglyceride - [CID 5460048]
- Malondialdehyde (MDA) - [CID 10964]
- Nitric oxide (NO) - [CID 145068]
- Glutathione - [CID 124886]
- Superoxide dismutase (SOD) - [CID 133082383]
- Captopril - [CID 44093]

2.2.1.2. Normalized Diseases.

- Cardiovascular diseases (CVDs) - [MESH:D002318]
- Cardiotoxicity - [MESH:D066126]

2.3. Large Language Models

Large Language Models are trained on all sorts of available written material, mimic human interactions, and have the capability of solving a comprehensive set of problems. Today, popular LLM models, ChatGPT [5], Gemini [6], LLaMA [7] are making their way into products we use every day.

LLMs are mostly based on transformer-based neural network architectures, which divide input into parts and understand their relationships by making use of their sequencing information. The difference between LLMs and previously used NNs is that they contain hundreds of billions of parameters and are trained on datasets that contain billions of words. Because the training dataset also includes academic papers, LLMs can answer scientific questions, such as, “What diseases exist in the following text?”.

In the future, LLMs can be used directly to solve all sorts of problems, but today, they are at the beginning of their journey, and for the most part, they are used in support of other applications. Complicated queries and tasks often can not be solved properly by the LLM, and they need to be divided, simplified, and structured in order to receive meaningful results.

2.3.1. Transformer Architecture

Transformer architecture [16] introduced self-attention mechanisms that process sequential inputs, capturing the structure of the sentences and the relation between different words. Furthermore, the attention mechanism is multiplied to work in parallel to grasp different sides of the relationships in the text, and they are called multi-head attention.

In order to produce an output, fully connected networks are applied after the multi-head attention layer and are used to perform transformations. Transformer models are used in an encoding-decoding fashion, where models trained as encoders produce the representations of the inputs, and decoders produce outputs for a given representation.

In a chatbot model, like ChatGPT [5], the queries we make are the representations for which the outputs are generated. In other words, we are the encoders.

2.3.1.1. Prompt Engineering. In order to improve the performance of applications that rely on chat bot models, we need to optimize our queries, to better represent the task we need the model to perform. This is called prompt optimization, and it is creating a new area of study.

2.3.1.2. Temperature Optimization. Similar to all Neural Networks, LLMs involve a tradeoff between consistency and variety. Low temperatures should be used for tasks

that require consistency, and high temperatures should be used for tasks that require creativity and imagination.

If unsure, temperature optimization should be performed to determine the range of temperatures where the desired metrics on outputs are maximized.

2.3.2. Relation Extraction With LLM

Relational Extraction is finding pairs of entities within the given text that have a predefined relationship between them. LLMs' strengths come from their understanding of the relation of consecutive words, word groups, and sentences, which gives them the ability to understand and reply to the given prompt. Therefore, we claim that Relationship Extraction is inherent to LLMs and that they can be used for Relational Extraction tasks in the Biomedical domain.

2.3.2.1. Relation Extraction Example. Following the previous example [28], we see for rats, multiple chemicals have relations with Cardiotoxicity:

- Isoproterenol (ISO) [CID: 3779] - Cardiotoxicity [MESH: D066126]
- Glutathione [CID: 124886] - Cardiotoxicity [MESH: D066126]
- Cholesterol [CID: 5997] - Cardiotoxicity [MESH: D066126]

2.4. How Biomedical Corpora is Built

LLMs require training data on the scale of billions of words. In all ML and AI solutions, training data is the keystone for the success of the tool produced. In the Biomedical domain, producing such data is quite costly.

As we looked into BioRED [33] employed significant human resources to build up a corpus, which we plan to use in extracting relations with LLMs.

When we look into how some of the other corpora are generated, we see some, such as NLM-Gene [34], and AIMed [35], are constructed with manual annotation, some build upon available data, making use of existing tools as in GNormPlus [36]. Some are collected from existing annotated datasets as in JBLPBA [37].

NLM-Gene [34] is a manually annotated corpus developed by the US National Library of Medicine, which has 550 PubMed abstracts containing many gene mentions, multiple species, and multiple entity types. In manual annotation, six professionals with an average of 20 years of experience worked together.

In contrast, GNormPlus [36] is a tool and dataset produced without human annotators. It was built upon datasets from community tasks BioCreative 1[[38] and BioCreative 2 [39], and combined it with previously produced tools, SR4GN [40] for normalization, used with SimConcept [41], and Ab3P which is an abbreviation detector. For tokenization, tmChem [42] and tmVar [43] are used, and recognition is solved as a labeling task using CRF [44].

JBLPBA [37] is a shared task that produced a corpus by querying Medline with "human," "transcription factors," "blood cells," and an additional newly annotated 404 abstracts sampled from the unpublished set of GENIA corpus.

AIMed [35] consists of manually annotated 1000 Medline abstracts. It is constructed and used in a study [35] that compares different machine learning approaches in a large dataset, covering both NER and RE tasks.

2.5. Contrastive Learning

Contrastive Learning is a self-supervised neural network that absorbs similar data points as positive pairs and different data points as negative pairs and creates a mapping that will transform similar data points closer and different data points further in the embedding space.

The power of Contrastive Learning comes from the fact that it transforms a large-scale multi-class classification problem into a binary classification of yes or no. The difference is that the same amount of training data, while it may not be enough to solve a large-scale multi-class classification problem, might be sufficient to solve a binary classification problem.

Contrastive Learning has been researched in various Biomedical NLP tasks and proven to improve or provide high performance. [45] used Contrastive Learning for improving performance of Relation Extraction, [46] for clinical event detection, [47] for protein sequence embeddings. [48] used Contrastive Learning to predict future biosignals using past biosignals.

In Entity Recognition, if we apply Contrastive Learning to (text, entity) pairs, we would have a way of checking if the text is closer to the tested entity, based on which we can determine a threshold and classify the text as an entity. We can also map all possible entities and see which is closest to the text.

2.5.1. Embedding Space

Embedding space is what defines our similarity, and our similarity definition constructs the embedding space. The second part of the logic is more straightforward to imagine. Suppose we have our data points as brands and define the similarity as producing the same product in our embedding space. In that case, cosmetics brands will be closer to each other when compared to the distance between a cosmetic brand and a technology brand. After the training phase, our embedding space, in return, can be used to assess the similarity between brands, and we do not need to know the products produced by brands to cluster them into similar groups or classify them as similar or dissimilar.

2.5.2. Defining Inputs

Inputs to the Contrastive Learning should be pairs of data points where we know if they are similar or dissimilar. Pairs, which should be close to each other in the embedding space, are defined as positive pairs, meaning they are similar data points. Pairs, which should be further from each other, are defined as negative pairs, meaning dissimilar data points.

2.5.3. Contrastive Loss

Loss functions, or with their second name, cost functions, are mathematical representations of the difference between the results for training data and model outputs for the inputs in training data.

Differences between problems, model architectures, and characteristics of training/test datasets result in a difference in performance for different loss functions.

There are many established Contrastive Loss functions which are proven to perform well on Contrastive Learning tasks:

- Euclidean-based loss [49]
- InfoNCE loss [50]
- Triplet loss [51]
- NT-Xent loss [52]
- SupCon loss [53]

For our study, we used Euclidean-based loss and InfoNCE loss.

2.5.3.1. Euclidean-based Loss. For N number of samples and two embeddings x_1 and x_2 produced by the Contrastive Model, in the embedding space, when we define the positive-negative label as y and leave m as the minimum unit distance between dissim-

ilar pairs, the Euclidean-based loss function is calculated as

$$\mathcal{L}_{\text{Euclidean-based}} = \frac{1}{N}\sum_{i=1}^{N}\left[(1-y_i)\cdot d^2(x_1^i, x_2^i) + y_i \cdot \max(0, m - d(x_1^i, x_2^i))^2\right]. \tag{2.1}$$

2.5.3.2. InfoNCE Loss. InfoNCE loss calculation requires multiple steps to calculate; we first calculate the cosine similarities as

$$\text{similarity}(x_1, x_2) = \frac{x_1 \cdot x_2}{\|x_1\|\|x_2\|T}, \tag{2.2}$$

where T is temperature, we used a default temperature of 0.5 in this study. Rescaling for numerical instability as

$$\text{scaled\_similarity} = \text{similarity} - \max(\text{similarity}). \tag{2.3}$$

Calculating softmax probabilities as

$$p = \frac{\exp(\text{scaled\_similarity})}{\sum \exp(\text{scaled\_similarity})}. \tag{2.4}$$

Loss is calculated as

$$\mathcal{L}_{\text{InfoNCE}} = -\log\left(\sum(y \cdot p)\right). \tag{2.5}$$

The average loss is calculated as

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_{\text{InfoNCE}}^i. \tag{2.6}$$

3. DATASETS

We have used three datasets in this thesis. We used NCBI Disease [10] and BC5CDR [20] datasets for Named Entity Recognition and Named Entity Normalization tasks, and we used BioRED [33] for the Relation Extraction task. Below, you can find a comparison of the datasets:

Table 3.1. Comparison of datasets.

Dataset	Articles	Entities	Relations
BioRED	600	genes/proteins, species, diseases, chemicals, variants, cell lines	8 relation types
NCBI Disease	793	790 diseases, 6892 disease mentions	none
BC5CDR	1,500	4,409 chemicals, 5,818 diseases	3,116 chemical-disease relations

3.1. BioRED

BioRED [33], self-defined as a "rich Biomedical Relation Extraction dataset", is a document-level bioinformatics corpus with six entities and eight relation types that can exist between those entities, spanning 600 PubMed abstracts, separating novel and known findings.

The entity types listed in BioRED are gene(protein), species, disease, variant, chemical, and cell line. If we are interested in genes, associated species, and gene-gene relations, gene/protein implies the species. Four relation types apply to gene-gene relations: positive correlation, negative correlation, association, and binding.

Significant human resources are employed to construct BioRED. First, a set of articles was collected from existing datasets. Annotation guidelines are defined, and annotators are familiarized with the TeamTat [54] and the task at hand. Existing entity annotations are carried over, and using PubTator [55], initial entity annotations are created. For relation annotations, only annotations from BC5CDR [20] are carried over,

and the rest are only annotated by human annotators. All the articles are annotated by three annotators and reviewed by another senior annotator; additionally, to assess the novelty of Relation Extractions, each relation is inspected by two biologists.

3.2. NCBI-Disease Dataset

NCBI-Disease [10] is a disease dataset annotated by fourteen annotators; for each document, two annotators are randomly paired. The dataset is constructed with a three-phase annotation and checked for corpus-wide consistency.

With 790 unique disease and 6892 disease mentions found in 793 PubMed abstracts, NCBI disease dataset is divided into training, development, and test and is publicly available.

3.3. BC5DR-Disease Dataset

BC5DR-Disase [20] dataset contains 5818 diseases and 4409 chemicals and has 3116 chemical-disease interactions, in a total of 1500 PubMed articles. This dataset is manually annotated by experts and has become a keystone benchmark for studies that research NER and RE tasks.

Though it contains chemicals and chemical-disease interactions, we only use the disease information from this dataset, similar to what is done in [1].

4. METHODOLOGY

4.1. Improving Normalization Accuracy Using Contrastive Learning

Using Contrastive Learning, we produced an outcome that showed how similar or dissimilar the embeddings of our entities are. This allowed us to find the most similar mention for which the ID is known, boosting the accuracy of BioNEN up to 86%.

Contrastive Learning requires defining positive pairs that match and negative pairs that do not match. We have used the pairs of mentions and mentions found by dictionary similarity as our training data, defining a pair as similar when their IDs are the same. To calculate accuracy, we separated half of all the pairs we had as test data.

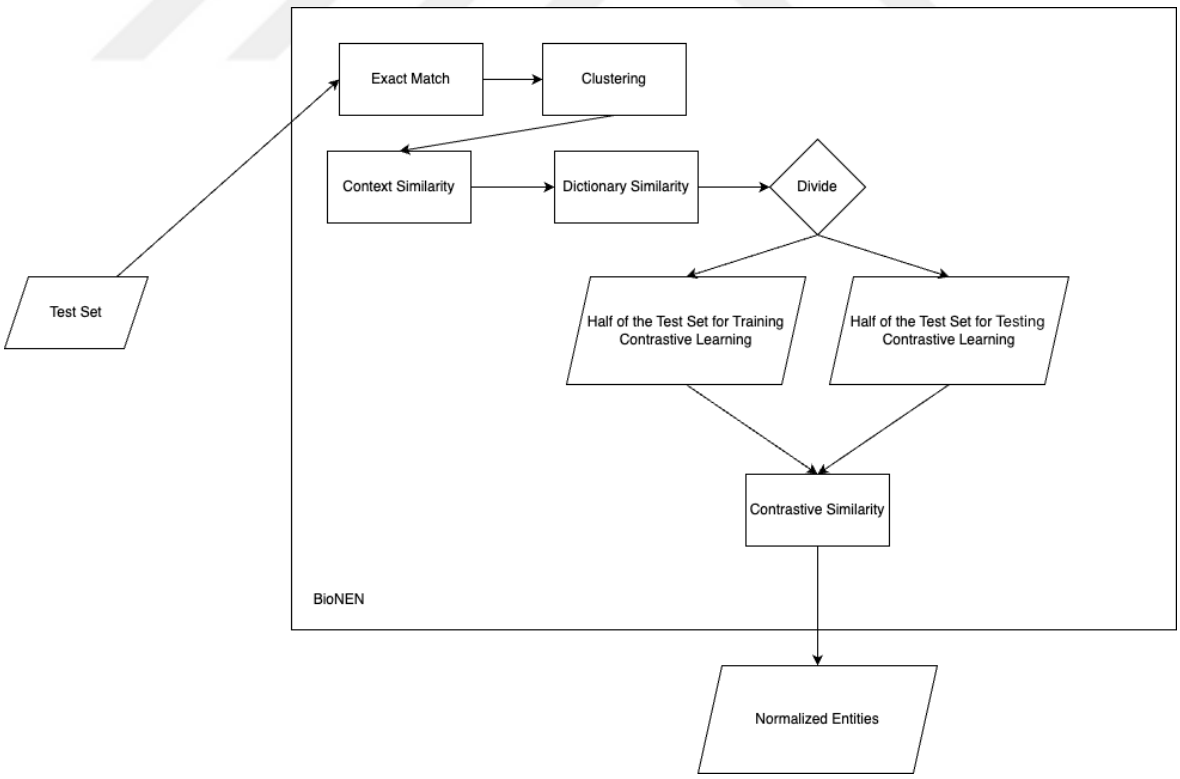


Figure 4.1. Workflow of BioNEN with Contrastive Similarity extension.

We used BERT Embeddings of the mentions as our data representation. As a network, we used a dense neural network consisting of three layers:

- Initial fully connected linear layer.
- Rectified linear unit, which introduces non-linearity.
- Final fully connected linear layer.

As we mentioned earlier, to maximize the similarity between positive pairs and minimize the similarity between negative pairs, Contrastive Learning requires defining a Contrastive Loss function. We have experimented with the following two loss functions:

- Euclidean Distance-based loss function, which is a commonly used Contrastive Loss function [49]
- InfoNCE loss, which is a Cosine Similarity based loss function widely used in frameworks like SimCLR [50]

4.2. Recognizing Entities with BioNEN and LLMs

BioNEN gives us a highly accurate Entity Normalization tool supporting approximate matches. This approximate match support allows us to convert it into a NER tool. The idea is the following:

- Parse abstracts to create tokens, or consecutive sets of tokens, as entity candidates.
- Check for exact matches using BioNEN; if there is one, we have found our entity, and we have also normalized it at the same time.
- If there is no exact match, try to find a match with a high similarity score using the Contrastive Model.

On the high level, the proposed approach looks like this:

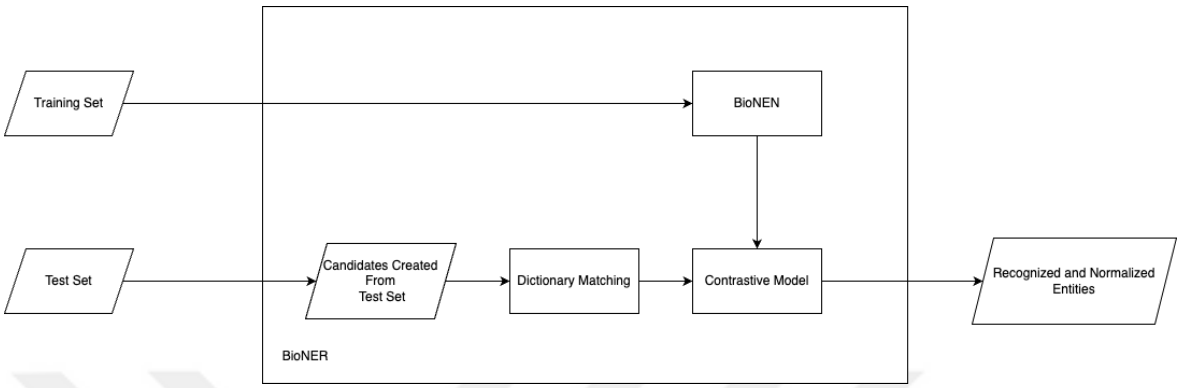


Figure 4.2. Workflow of BioNER.

The challenge with the proposed approach is that we are introducing a new lever to control the accuracy of Entity Recognition, which is the threshold for the similarity score. Using this score, we will decide which pairs can be considered a match. As a result, to do Entity Recognition, we have to:

- Restructure the BioNEN as a NER tool.
- Optimize the matching threshold of Contrastive Similarity.

4.2.1. Assessing the Performance of Named Entity Recognition with Normalization

4.2.1.1. Definitions. To ensure clarity, we define the key terms and variables used throughout the calculations:

- D: The set of all keys in df_dictionary_test, where each key represents an abstract in the test set.
- mentions$_k$: The set of mentions in df_dictionary_test[k]['mentions'] for a given key k.
- found_entities$_k$: The set of recognized entities in entities[k] for a given key k.
- mentions_ids$_k$: The set of (mention, ID) pairs in df_dictionary_test[k] for a given

key k.

- entities_found_correct$_k$: The subset of found_entities$_k$ that are also in mentions$_k$.
- entity_ids$_k$: The set of IDs in found_entities[k].
- existing_ids$_k$: The set of IDs in df_dictionary_test[k]['id'].
- matching_ids$_k$: The intersection of entity_ids$_k$ and existing_ids$_k$.

4.2.1.2. Recall Calculation. The recall calculation measures the proportion of mentions that were correctly recognized.

Total Number of Mentions is calculated as

$$\text{num\_all\_mentions} = \sum_{k \in D} |\text{mentions}_k|. \tag{4.1}$$

Total Number of Found Mentions is calculated as

$$\text{num\_all\_mentions\_found} = \sum_{k \in D} |\text{mentions\_found}_k|, \tag{4.2}$$

where mentions_found for any given k is calculated as

$$\text{mentions\_found}_k = \{m \in \text{mentions}_k \mid m \in \text{found\_entities}_k\}. \tag{4.3}$$

Recall is calculated as

$$\text{Recall} = \left(\frac{\text{num\_all\_mentions\_found}}{\text{num\_all\_mentions}} \right) \times 100. \tag{4.4}$$

4.2.1.3. Recall Calculation with IDs. The recall calculation with IDs measures the proportion of mention-ID pairs correctly recognized.

Total Number of Mentions is calculated as

$$\text{num\_all\_mentions} = \sum_{k \in D} |\text{mentions\_ids}_k|. \tag{4.5}$$

Total Number of Found Mentions is calculated as

$$\text{num\_all\_mentions\_found} = \sum_{k \in D} |\text{mentions\_found}_k|, \tag{4.6}$$

where mentions found for any given k is calculated as

$$\text{mentions\_found}_k = \{(m, \text{id}) \in \text{mentions\_ids}_k \mid \text{id} \in \text{found\_entities}_k\}. \tag{4.7}$$

Recall by ID is calculated as

$$\text{Recall by ID} = \left(\frac{\text{num\_all\_mentions\_found}}{\text{num\_all\_mentions}} \right) \times 100. \tag{4.8}$$

4.2.1.4. Precision Calculation. The precision calculation measures the proportion of recognized entities that are correct.

Total Number of Found Entities is calculated as

$$\text{num\_all\_entities\_found} = \sum_{k \in D} |\text{found\_entities}_k|. \tag{4.9}$$

Total Number of Correctly Found Entities is calculated as

$$\text{num\_all\_entities\_found\_correct} = \sum_{k \in D} |\text{entities\_found\_correct}_k|, \tag{4.10}$$

where entities found correct for any given k is calculated as

$$\text{entities\_found\_correct}_k = \{e \in \text{found\_entities}_k \mid e \in \text{mentions}_k\}. \tag{4.11}$$

Precision is calculated as

$$\text{Precision} = \left(\frac{\text{num\_all\_entities\_found\_correct}}{\text{num\_all\_entities\_found}}\right) \times 100. \tag{4.12}$$

4.2.1.5. Precision Calculation with IDs. The precision calculation with IDs measures the proportion of recognized entities with correct IDs.

Total Number of Found Entities is calculated as

$$\text{num\_all\_entities\_found} = \sum_{k \in D} |\text{found\_entities}_k|. \tag{4.13}$$

Total Number of Correctly Found Entities is calculated as

$$\text{num\_all\_entities\_found\_correct} = \sum_{k \in D} |\text{entities\_found\_correct}_k|, \tag{4.14}$$

where entities found correct for any given k is calculated as

$$\text{entities\_found\_correct}_k = \{(e, \text{id}) \in \text{found\_entities}_k \mid \text{id} \in \text{matching\_ids}_k\}. \tag{4.15}$$

Precision by ID is calculated as

$$\text{Precision by ID} = \left(\frac{\text{num\_all\_entities\_found\_correct}}{\text{num\_all\_entities\_found}}\right) \times 100. \tag{4.16}$$

4.2.2. Restructuring and Extending BioNEN

Previously, we normalized all the data received with all approaches except Contrastive Similarity. To calculate contrastive similarity accuracy, we randomly divided our test dataset into two and calculated the accuracies of all other methods on this divided dataset.

For the purposes of restructuring BioNEN, we first divided the BC5CDR dataset into two and refactored our tool to work and only train on the whole training data given, with no accuracy calculations being made, including Contrastive Similarity. All of the recognition is made in the second pass of the algorithm, using the Contrastive Learning model trained in the first pass on the given test dataset. Similar to small data tests, we have also produced a second small dataset.

After the training was completed, we read the data we had separated earlier and found exact matches using BioNEN. For candidate entities without exact matches, we employ context similarity. As a result, we now have a BioNEN for the NER task, which we call BioNER, to distinguish between two versions of BioNEN.

The interface of BioNER is extended from BioNEN and can be found in Chapter 6, where we provide its usage and examples similar to the preceding work.

We first calculated precision for BioNER with a tentative threshold of 0.9.

Table 4.1. Performance of BioNER with similarity threshold of 0.9.

Metric	Dictionary Recognition	Contrastive Similarity Recognition
Recall by entity	65.90 %	86.13%
Recall by id	84.40 %	26.01%
Precision by entity	10.93 %	0.57%
Prevision by id	26.65 %	3.88%

When calculating precision and recall by ID, we can see if another name recognizes the entity, but the IDs are the same. The drawback is that we rely on the accuracy of the normalization. A correctly recognized entity might also count as a false positive if it is normalized wrong by the BioNEN. Due to the high accuracy of the BioNEN, we are not worried about such false positives disturbing our optimizations. Therefore, we used these calculations as indicators for optimization purposes.

Using n-grams of the abstract creates far too many candidates, and some are quite obviously irrelevant. We decided to explore relevance filtering to eliminate such candidates and possibly improve the tool's performance.

When we inspect the performance of Contrastive Similarity, we see an increase only in the recall by the entity. Recall by id and precision calculations fall sharply. This means our threshold of 0.9 percent was too low, and we recognized many false positives, re-recognized some previously found candidates, and overridden them with an ID that did not match. One possible solution is to make dictionary recognition the absolute truth but let Contrastive Similarity override its found candidates only if the similarity score is increased. We explored this approach separately after the threshold optimization was completed.

4.2.3. Relevance Filtering of Entity Candidates

Using a 0.90% threshold, we have used "facebook/bart-large-mnli" to perform a zero-shot classification to eliminate most unrelated candidates and Entity Recognition on n-grams that are most likely to be a disease name.

BART [56] is a model that combines bidirectional encoders with autoregressive models. MNLI stands for fine-tuning on the MultiNLI dataset.

Table 4.2. Performance of BioNER with a similarity threshold of 0.9 and Relevance Filter.

Metric	Dictionary Recognition With RF	Contrastive Similarity Recognition With RF
Recall by entity	49.71 %	68.21%
Recall by id	71.68 %	25.43%
Precision by entity	16.37 %	1.19%
Precision by id	37.5 %	5.08%

The Relevance Filter resulted in scores we recall from both dictionary and Contrastive Learning dropping; however, precision from both increased. This means the Relevance Filter's threshold works as a lever to control the tradeoff between precision and recall.

We aim for high recall, intending to use the annotations as candidates for LLM to find relations. We continued our optimizations without using the Relevance Filter we introduced. Details on how to use the relevance filter can be found in Chapter 6.

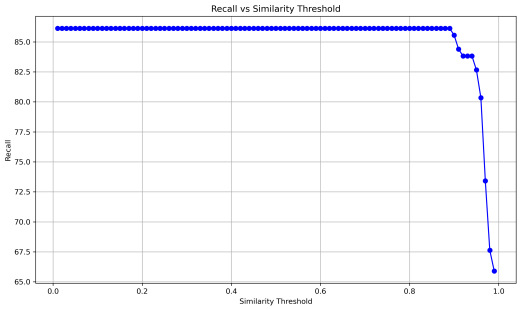
4.2.4. Optimization of the Matching Threshold of Contrastive Similarity

We have run BioNER for all thresholds from 0.1 to 0.99 and calculated recall, recall by ID, and precision, precision by ID, for all of them.

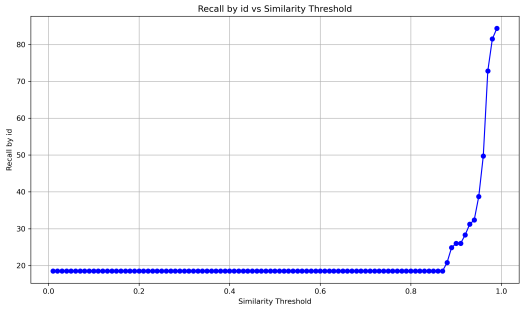
When we increase the threshold from as low as 1% to 99% recall falls from 86.13% to 65.90%. This means that with the strictest option, we tried 0.99, but we did not find any matches. If we just recognized all candidates, we could have a recall of 86.13% with our candidate set. When the threshold reaches 99%, all scores converge at a no-effect zone, where using Contrastive Similarity does not affect any calculations.

Like the Relevance Filter, Contrastive Similarity also gives us a recall vs. precision tradeoff. Reducing the similarity threshold will increase the false positive rate.

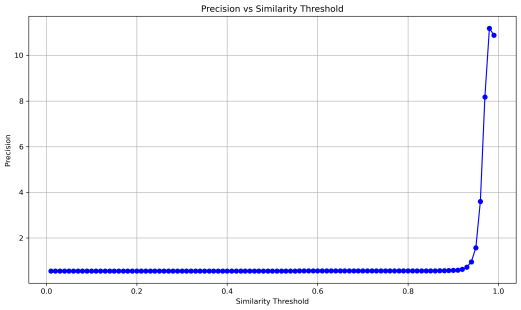
How many false positives LLMs can handle depends on their attention span, which has been increasing since they made their debut.



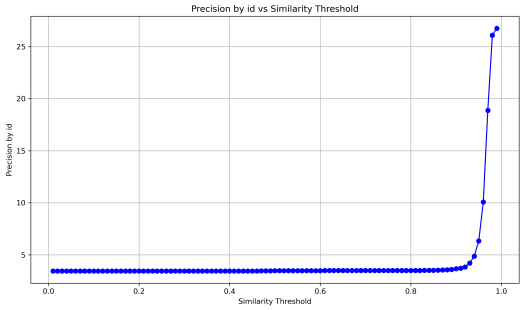
(a) Recall vs Similarity Threshold



(b) Recall by id vs Similarity Threshold



(c) Precision vs Similarity Threshold



(d) Precision by id vs Similarity Threshold

Figure 4.3. Performance metrics as a function of similarity threshold.

If we compare the performance of 0.96 and 0.97 contrastive similarities to dictionary similarity, we get:

Table 4.3. Performance of BioNER with similarity thresholds of 0.96, 0.97.

Metric	Dictionary Recognition	Contrastive Similarity Recognition 0.96	Contrastive Similarity Recognition 0.97
Recall by entity	65.90 %	80.35%	73.41%
Recall by id	84.40 %	49.71%	72.83%
Precision by entity	10.93 %	3.60%	8.17%
Precision by id	26.65 %	10.07%	18.87%

For both thresholds, we see an increase in the recall by the entity and a decrease in the recall by ID. If we want to protect the precision of dictionary recognition, we

should go with a more conservative threshold of 0.97. For the purposes of ID overrides optimization, we will be using 0.96, as we expect the optimization to address the drop in recall by ID, and we require higher recall.

4.2.5. Optimization of ID Overrides

As described earlier, for the threshold of 0.96, we have fixed IDs found by the exact matches, and we let Contrastive Similarity override the normalization for its own recognized mentions only if the similarity score is higher than the previous match.

Table 4.4. Performance of BioNER with a similarity threshold of 0.96 with and without ID Override optimization.

Metric	Dictionary Recognition	Contrastive Similarity Recognition	ID Overrides Optimized
Recall by entity	65.90 %	80.35%	80.35%
Recall by id	84.40 %	49.71%	86.13%
Precision by entity	10.93 %	3.60%	3.52%
Prevision by id	26.65 %	10.07%	11.75%

As we expected, when we optimized the ID overrides, recall by ID rose over dictionary recognition. We also see a slight improvement in precision by id.

4.2.6. Using LLM to Recognize Entities

We introduced LLM in three ways to BioNER:

- Process entity candidates created by n-grams to identify diseases and normalize by the dictionary.
- Process the abstract directly to identify diseases and normalize by the dictionary.
- Recognize entities by Contrastive Learning, filter by LLM.

Processing entity candidates to identify diseases, produced a list of recognized diseases, where diseases that identified multiple times, existed multiple times for each index, like rest of the methods.

Processing abstracts produced a set of diseases, which we could ask LLM to repeat by the index. However, complicating the query lowers the performance of existing tasks; in our case, this would reduce the precision and recall of the LLM recognition. To compare, we have calculated recall and precision based on the sets of found entities and sets of entities in the test set.

We observed LLM had high precision, but low recall, and Contrastive Similarity had low precision, but high recall, we combined them to LLM filtering of Contrastive Similarity results.

4.2.7. LLM Entity Recognition Prompts

Improve performance of zero-shot use of LLMs requires two optimizations:

- Prompt optimization.
- Temperature optimization.

We did not do these optimizations for the Relation Extraction. We used simple prompts to describe our task, and we used the default temperature settings.

We used LLMs for two different operations and used them in three methods. We asked LLM to identify each given term and whether it is a disease or not, and we asked for the diseases in the given abstracts.

4.2.7.1. System Prompts. The system prompts define the behavior of the model and inform the model on who the model is and what it should do.

We used two system prompts, one for each of the two operations:

```
You are a skilled assistant in identifying if each given term is a disease or
not.
```

Figure 4.4. System prompt for identifying each term with disease or not.

```
You are a skilled assistant in identifying disease names in a given
abstract.
```

Figure 4.5. System prompt for finding disease names in a given abstract.

4.2.7.2. User Prompts. User prompts are our queries to the LLM, which carry the necessary information for the LLM to carry out its task and reply to us.

We used two user prompts, one for each of the two operations:

```
Please indicate whether each of the following terms exactly matches the
name of a known disease.
Respond with 'yes' or 'no' next to each term.
Format the response strictly as 'disease_name - yes'
or 'disease_name - no' on each line. All terms:\n\n{candidates_list_str}
Return the result:
```

Figure 4.6. User prompt for identifying each term with disease or not.

```
Extract all disease names mentioned in the following
abstract:\n\n{abstract}\n\n
Please return only the disease names, with each name on a new line.
Do not include any extra text or explanations, just the disease names,
exactly as they appear in the abstract.
```

Figure 4.7. User prompt for finding disease names in a given abstract.

4.3. LLMs as a RE Tool

Large Language Models are applicable to many natural language problems without fine-tuning; they are quite capable of manipulating text, extracting logic from the given prompts, and replying accordingly. For this reason, we aim to use them for the Relation Extraction. When we have an abstract, that is annotated and normalized,

our task is simplified, because we know in the given text, what are our entities, and their respective IDs.

We used BioRED dataset for experimenting with LLM's performance on this task, because it already had annotated entities of different types, and their extracted relations.

An annotated paper [57] in BioRED looks like:

```
1848636|t|Debrisoquine phenotype and the pharmacokinetics and beta-2
receptor pharmacodynamics of metoprolol and its enantiomers.
1848636|a|The metabolism of the cardioselective beta-blocker
metoprolol is under genetic control of the debrisoquine/sparteine type.
The two metabolic phenotypes, extensive (EM) and poor metabolizers
(PM), show different stereoselective metabolism, resulting in apparently
higher beta-1 adrenoceptor antagonistic potency of racemic metoprolol in
EMs. We investigated if the latter also applies to the beta-2 ....
1848636 0 12 Debrisoquine ChemicalEntity D003647
1848636 52 67 beta-2 receptor GeneOrGeneProduct 154
1848636 88 98 metoprolol ChemicalEntity D008790
1848636 171 181 metoprolol ChemicalEntity D008790
...
1848636 Negative_Correlation 153 D008790 No
1848636 Positive_Correlation D013726 D007008 No
1848636 Association D008790 D013034 No
1848636 Association D008790 D003647 No
1848636 Positive_Correlation D008790 D013726 Novel
...
```

Figure 4.8. An annotated abstract from BioRED dataset with PubTator format.

We first tried to use ChatGPT4-o without simplifying the task; we just gave the abstract and annotated entries in PubTator format and asked for relations to be extracted. However, we could not achieve any meaningful success this way.

Second, we tried to give examples from the development set and asked the LLM to extract relations. Sadly, we still could not receive any meaningful success. Recall, accuracy, and precision were all either 0 or close to 0.

We, therefore, simplified the task by using two LLM queries to extract relations and produce meaningful results.

4.3.1. Pseudo Code for LLM Relation Extraction on BioRED

```
FOR each document in the test data
   abstract = extract_abstract(document)
   entity_pairs_string = CONCATENATE entity_text and entity_id pairs
from extract_entity_id_pairs(document)

   first_prompt = "Replace entities with their IDs in the abstract: " +
abstract + " " + entity_pairs_string
   modified_abstract = call_chatgpt(first_prompt)

   second_prompt = "Identify relation pairs (id1, id2) in the text: " +
modified_abstract
   estimated_relation_pairs = call_chatgpt(second_prompt)

   precision, recall, accuracy = calculate_metrics(relation_id_pairs,
estimated_relation_paxirs)

   ADD precision, recall, and accuracy to respective lists

average_precision = MEAN of precisions list
average_recall = MEAN of recalls list
average_accuracy = MEAN of accuracies list
```

Figure 4.9. Pseudo Code for LLM Relation Extraction on BioRED.

4.3.2. LLM Relation Extraction Prompts

4.3.2.1. System Prompts. We used two system prompts, one for each of the two calls in an iteration of our code:

```
You are a skilled assistant in replacing entities found in abstracts with
their respective IDs.
```

Figure 4.10. System prompt for replacing entities with IDs.

```
You are a skilled assistant in finding id pairs with a relationship, in a
given abstract.
```

Figure 4.11. System prompt for finding relations.

4.3.2.2. User Prompts. We used two user prompts, one for each of the two calls in an iteration of our code:

```
Given the following abstract:\n{abstract} and entity,id pairs: {entities}\n
can you replace the entities with their IDs?
```

Figure 4.12. User prompt for replacing entities with IDs.

```
Given the following abstract:\n{abstract}, can you find which id pairs
have a relationship? Only print id pairs in format (id1, id2)?
```

Figure 4.13. User prompt for finding relations.

5. RESULTS

5.1. Normalization Results

5.1.1. Accuracy of Contrastive Learning

We used three different accuracy calculations to measure the success of Contrastive Learning. One pair of our data has the structure of (mention, id), (dictionary similarity mention, dictionary similarity id). Our first calculation is the accuracy of Contrastive Learning itself, calculating accuracy based on the similarity scoring of test data pairs.

The second accuracy calculation is the accuracy of Contrastive Learning on the normalization task, which we call Contrastive Similarity because we use Contrastive Learning to find the most similar entity. To do that, we construct the candidate set, consisting of all the mentions in our training data. For all the mentions in the test data, we find the most similar candidate from the candidate set using Contrastive Similarity.

Lastly, to increase BioNEN's overall accuracy, we have taken the results of exact matches. Where the exact match is missing, we used Contrastive Similarity to normalize the mention.

Unlike previous methods Contrastive Similarity performance is not static, it slightly fluctuates with each run. We ran each different configuration ten times, and averaged the percentages to account for small fluctuations in performance.

Table 5.1. Performance of Contrastive Learning with Different Loss Functions and BioBERT Versions on BC5CDR Dataset.

BioBERT Version	Loss Function	Contrastive Model	Contrastive Similarity	Dictionary + Contrastive Similarity
BioBERT v1.1	InfoNCE	90.78%	79.70%	84.16%
	Euclidean	78.41%	82.24%	85.77%
BioBERT v1.2	InfoNCE	90.83%	80.04%	84.07%
	Euclidean	78.80%	81.72%	85.92%

When we use InfoNCE, the accuracy of the Contrastive Model itself increases, but Euclidean performs better for the normalization tasks. We think this is because the Euclidean loss function better represents the similarity of entity embeddings.

5.1.2. Comparison of Contrastive Learning with Previous Methods

The tool we produced still supports getting results with pure Contrastive Similarity. However, it proved to be inferior in terms of accuracy to applying Contrastive Similarity only on data where exact matches fail, which we call Dictionary + Contrastive Similarity.

We compared performance using BioBERT v1.1, which is the best performer for previous methods and performed well for Contrastive Similarity with the Euclidean loss function. Similar to Contrastive Learning accuracy calculation, we ran each configuration ten times, and averaged the percentage performances.

Table 5.2. Performance of BioNEN + Contrastive Similarity on BC5CDR and NCBI Disease Datasets.

Dataset	Loss Function	Contrastive Model Accuracy	Exact Match	Clustering	Context Similarity	Dictionary Similarity	Dictionary + Contrastive Similarity
BC5CDR	InfoNCE	90.86%	67.99%	72.68%	73.04%	74.56%	84.16%
	Euclidean	78.35%	67.99%	72.68%	73.04%	74.56%	85.77%
NCBI Disease	InfoNCE	83.41%	63.74%	68.13%	69.23%	73.08%	81.26%
	Euclidean	82.70%	63.74%	68.13%	69.23%	73.08%	81.87%

We see significant performance improvement for both datasets, with both loss functions, while the accuracy of the model itself ranges from 78.35% to 90.86%.

5.2. Recognition Results

When used in abstracts, LLM produces a set of recognized entities. To compare with other methods, recall and precision calculations are done on sets produced from recognized entities.

Because Contrastive Similarity Recognition is extremely slow, we used a subset of BC5CDR data. We used a threshold for Contrastive Similarity of 0.96, BioBERT v1.1 as a BERT model, and GPT-4o-mini as an LLM model.

Table 5.3. Performance of LLM methods, compared to Dictionary and Contrastive Similarity Recognition.

Metric	Dictionary Recognition	Contrastive Similarity Recognition	LLM Filtering on CSR	LLM on entity candidates	LLM on abstracts
Recall by entity	54.55%	67.05%	26.14%	30.68%	20.45%
Recall by id	75.00%	77.27%	35.23%	50.00%	36.36%
Precision by entity	10.93%	3.26%	25.27%	55.10%	85.71%
Precision by id	26.65%	11.93%	35.16%	71.43%	95.24%

We see that normalization-based methods have high recall but low precision, while LLM-based methods have better precision but lower recall. Contrary to what we expected, the combination of the two, LLM Filtering on Comparative Similarity Recognition, was low in both recall and precision.

5.2.1. Error Analysis for Recognition

LLM results are as expected. Since we used general-purpose models such as ChatGPT4o, recall is lower because not all disease names are being recognized. However, precision is extremely high, especially when we use the LLM directly on abstracts.

When we combined LLM with Contrastive Similarity Recognition and filtered the results from Contrastive Similarity, we observed an increase in precision at the expense of recall. However, the increase in precision does not justify the significant drop in recall, and hence, this method is not evaluated as a strong performer.

An interesting and counter-intuitive finding for us was that the precision for Dictionary Recognition was low. This low precision also translated into low precision for Contrastive Similarity, prompting a deeper analysis of the errors originating from Dictionary Recognition and Contrastive Similarity Recognition.

5.2.1.1. Dictionary Recognition & Contrastive Similarity Recognition Errors. Dictionary Recognition achieves relatively high recall, particularly when we use the IDs of the mentions found. This is significant because, in the end, we aim to produce ID pairs for found interactions.

Precision is seemingly underperforming for both Dictionary Recognition and Contrastive Similarity Recognition. Contrastive Similarity Recognition, as expected, increases recall at the expense of precision, therefore pushing precision to even lower grounds. Beyond false positives, multiple factors falsely contribute to the performance issues:

- Novel findings
- Linguistic differences for the same entity

5.2.1.2. Novel Findings. It is surprising to observe low precision for Dictionary Recognition. Our candidate set is derived from n-grams in the test set, making it impossible for Dictionary Recognition to identify entities that do not exist in the abstract. The dictionary used is curated in BioNEN [1] from disease databases and includes only diseases. Therefore, the absolute precision should approach 100%, with rare exceptions where a recognized disease is used as a different entity or word in a Biomedical abstract.

This is unlikely, given that related work is written within the Biomedical domain.

When we inspect results for Dictionary Recognition, we encounter novel findings. For example, the first entry [58] in our test set:

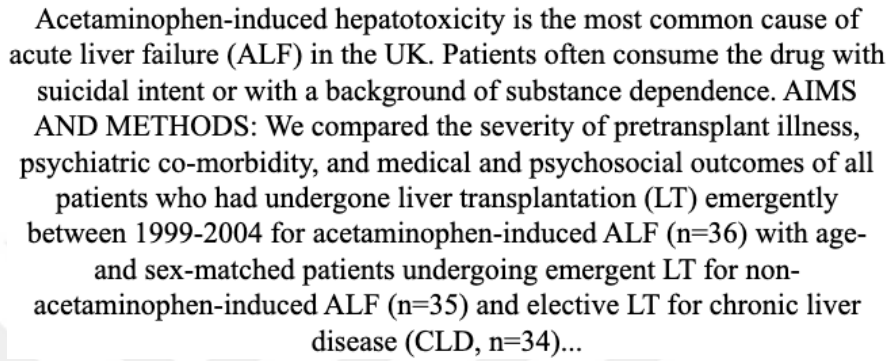


Figure 5.1. An abstract from BC5CDR dataset.

The disease recognition result includes: "hepatotoxicity": "D056486". However, this mention is not present in the dataset. From these findings, we deduce that normalization accuracy was high even though the same methods were used, as there were no possibilities of novel findings during the normalization task. However, during recognition, novel findings are very likely, which highlights the importance of human verification for recognition tasks.

5.2.1.3. Linguistic Differences. Another drawback of reapplying a successful normalization tool for recognition is that dictionaries are typically created to handle linguistic differences. Specialized dictionaries improve the accuracy of normalization tasks by recognizing different forms of disease, including prefixes, suffixes, and abbreviations. However, they reduce the accuracy of recognition tasks.

For example, extended versions of disease names are often recognized due to dictionary matching, such as "acute liver failure (ALF)." In contrast, the test set may refer to the condition only as "ALF." Similarly, non-exact matches are frequent in both our dictionary and Contrastive Similarity, such as "ventricular tachycardia among", which is recognized as "ventricular tachycardia" and normalized with the ID

"D017180". While this may be considered correct in context, it is marked as a false positive in exact matching because the dataset only lists the condition as "ventricular tachycardia". Recall by ID mitigates the limitations of exact recall comparison.

A potential solution to improve calculated statistics is to recreate the dictionary specifically for recognition tasks. This would involve eliminating variations of diseases unlikely to appear in a test set of already annotated abstracts. However, this approach risks over-tuning the results, and it is not a reliable way to address these true positives to be incorrectly labeled as false positives.

The proper solution is to employ extensive human evaluation of the results. While it would be the most accurate method, we lack necessary resources and it is beyond the scope of this thesis.

5.3. Relation Extraction Results

We have used the test dataset from BioRED with 100 abstracts, and run our Python code on both ChatGPT4o and ChatGPT4o-mini, and calculated Precision, Recall, and Accuracy. We see that ChatGPT4o's average performance was lower than its smaller version.

Table 5.4. Performance of LLM RE.

Model	Average Precision	Average Recall	Average Accuracy
ChatGPT4o-mini	38.48 %	49.74 %	26.85 %
ChatGPT4o	31.89 %	43.62 %	22.59 %

5.3.1. Error Analysis for Relation Extraction

Overall, we attribute our errors to the:

- Novel findings
- False Positives
- False Negatives
- Test set articles with few annotations

Many entities are recognized for some abstracts in the test set, but only a few relations exist in BioRED, where LLMs find more relations within the text. This could mean that we found novel relations on BioRED that previous methods could not find. On top of that, some abstracts simply have one or two annotated relations, in which case missing them skews our results lower because we don't take a weighted average; we simply apply our solution to all papers, and average Precision, Recall, and Accuracy throughout all papers in the test set.

We inspected performance on an example paper [59] from the BioRED dataset with 18 relations:

Congenital long QT syndrome (LQTS) with in utero onset of the rhythm disturbances is associated with a poor prognosis. In this study we investigated a newborn patient with fetal bradycardia, 2:1 atrioventricular block and ventricular tachycardia soon after birth. METHODS: Mutational analysis and DNA sequencing were conducted in a newborn. The 2:1 atrioventricular block improved to 1:1 conduction only after intravenous lidocaine infusion or a high dose of mexiletine, which also controlled the ventricular tachycardia. RESULTS: A novel, spontaneous LQTS-3 mutation was identified in the transmembrane segment 6 of domain IV of the Na(v)1.5 cardiac sodium channel, with a G-->A substitution at codon 1763, which changed a valine (GTG) to a methionine (ATG). The proband was heterozygous but the mutation was absent in the parents and the sister. Expression of this mutant channel in tsA201 mammalian cells by site-directed mutagenesis revealed a persistent tetrodotoxin-sensitive but lidocaine-resistant current that was associated with a positive shift of the steady-state inactivation curve, steeper activation curve and faster recovery from inactivation...

Figure 5.2. An abstract from BioRED dataset.

We see for this abstract, Precision was 53.33%, Recall was 44.44% and Accuracy was 32.00, and performance is closer to our average performance of LLMRE.

5.3.1.1. Novel Findings. LLMRE found a relation between LQTS and arrhythmias, which are both heart diseases. When we had a closer look, an interesting finding was that, in BioRED, there is a positive correlation found between LQTS and a sequence variant V1763M and another positive correlation between arrhythmias and V1763M. By transition, we can say that if LQTS is positively correlated with V1763M, and if V1763M is positively correlated with arrhythmias, LQTS is also positively correlated with arrhythmias. Indeed, there is a positive correlation to be found between LQTS and arrhythmias. As expected, LLMs are powerful in deriving complicated relationships in the given text, and the transitive correlation where the original test dataset was missing was found with LLM Relation Extraction.

Another novel finding is the relation between the patient and LQTS. This relation is sentences apart, and missing in the test set, however we find it with our LLM recognition. Moreover LLMs present relation of the patient with bradycardia, atrioventicular block, ventricular tachycardia, and all are the associated sickness'es with the patient in the study.

5.3.1.2. False Negatives. While the LLM found the transitive novel relation, it was missing the base relations between V1763M, LQTS, and arrhythmia. In the paper, we see the V1764M is referred to as Na(v)1.5/V1763M; when we do the replacement step, we therefore get "Na(v)1.5/p—SUB—V—1763—M", and Na(v)1.5 is replaced with 6331, and we have "63311/p—SUB—V—1763—M" which is not a real id anymore, and id itself it the subtext of the text produced. In the second step of recognition, LLM discovers that both diseases are related to the same entity. Therefore, they are related, but due to linguistic differences in how the id was presented, they can not discover the base relations in the form they are found in the test set.

5.3.1.3. False Positives. LLM recognition finds a relationship between lidocaine and tetrodotoxin. In the abstract, it says, "A mutation had revealed a tetrodotoxin-sensitive, but lidocaine-resistant current." in this particular sentence, both have a relation with the mutation; however, we are not informed on the transitivity of the association; this does not prove lidocaine, and tetrodotoxin is always negatively correlated or not. A more accurate way of annotation would be to produce the relations between the two and the mutation itself; however, ironically, those are missing in the test dataset and would have been novel findings, which would be misidentified as incorrect by the test dataset.

5.3.1.4. Precision, Recall, Accuracy Histograms. We must highlight that our performance calculations only indicate the level of agreement with BioRED annotations due to novel findings. In BioRED, human annotators verified and separated the novel findings from false positives. We can not afford such an evaluation within the scope of this thesis, but level of disagreement is visible in the histograms below.

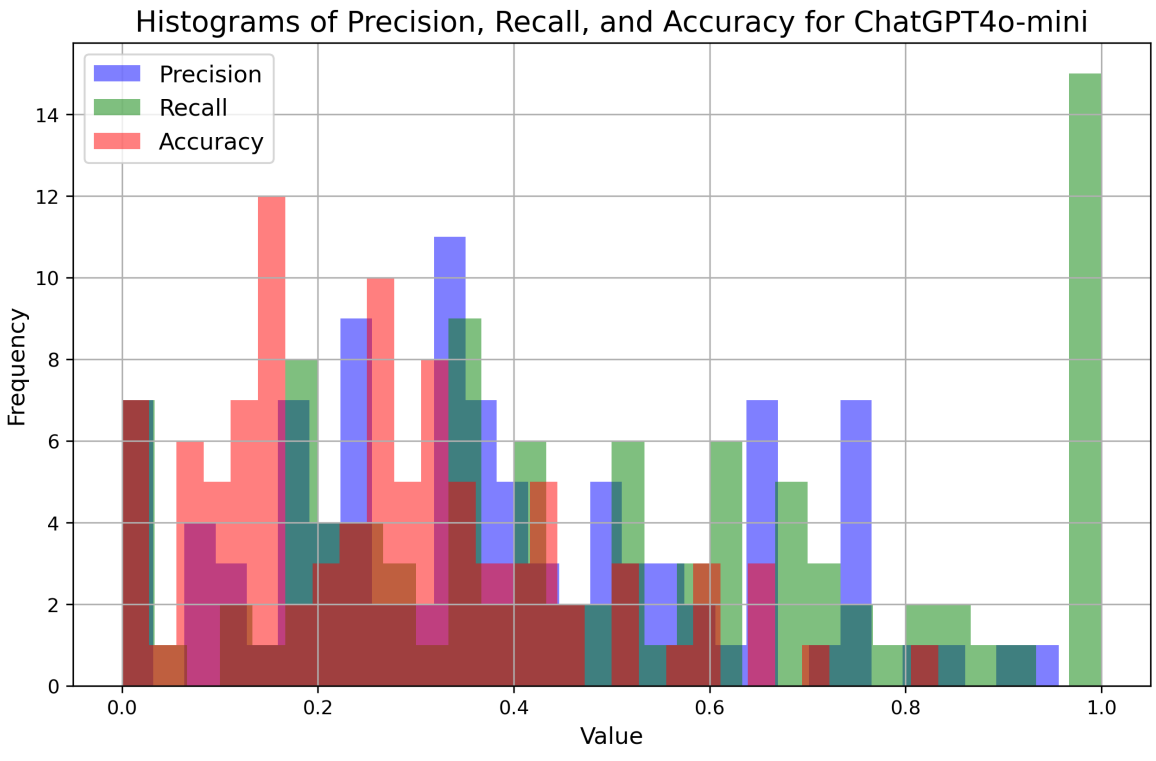


Figure 5.3. Precision, recall and accuracy of ChatGPT4o-mini on individual abstracts.

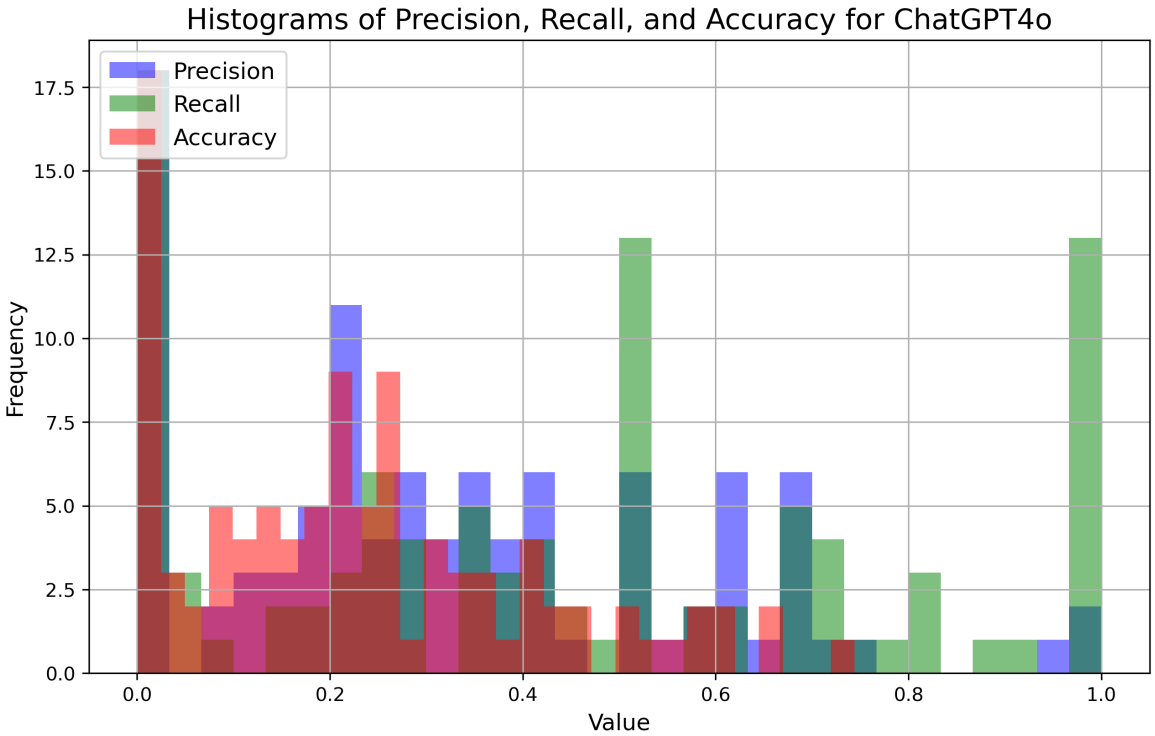


Figure 5.4. Precision, recall and accuracy of ChatGPT4o on individual abstracts

Recalls are higher, and Accuracy is lower, indicating disagreement between datasets. This could either be due to false positives or novel findings, and we saw examples of both in our inspection.

6. TOOLS

6.1. A Simple Annotator with Built-in NEN, NER, RE

We have implemented a simple annotator that processes research abstracts to produce an annotated text version. This tool recognizes entities, applies normalization, and lists the interactions between these entities in a normalized fashion.

6.1.1. Steps for Setup and Execution

6.1.1.1. Setting Up OpenAI API Authentication. OpenAI API is a paid API; therefore, running LLMRE has an actual monetary cost. To run it, please follow the instructions in the OpenAI documentation and set your access token in your environment variable. For Mac OS, add:

```
export OPENAI_API_KEY=""
```

Figure 6.1. Command for setting OPEN_AI_API_KEY.

to bash_profile.

6.1.1.2. Mapping BioNEN Models for Entity Normalization. An instance of BioNEN needs to be trained for each entity type.

- The input for training should be organized in the data/ folder, with a mapping defined in a file named mapping.txt.
- The format of mapping.txt should follow these guidelines:
  - N (number of entity types): The number of entity types to be trained.
  - For each entity, provide the following, each on a separate line:
    Entity_Class
    path_to_specialized_normalization_dictionary

path_to_pubtator_file_with_abstracts_containing_Entity_Class

Example mapping.txt file:

```
2
Disease
path_to_disease_normalization_dictionary
path_to_pubtator_file_with_abstracts_containing_disease
Species
path_to_species_normalization_dictionary
path_to_pubtator_file_with_abstracts_containing_species
```

6.1.1.3. Training the Models. Once the mappings are set up, run the following command to train the models:

```
python train.py
```

Upon completion, the trained models and related artifacts will be stored in the models/ folder.

6.1.1.4. Preparing Input. Paste all abstracts for which the annotations are desired to be generated into input.txt, and separate them with a blank line.

6.1.1.5. Annotating Abstracts. To annotate new abstracts, run the following command without any input parameters:

```
python annotate.py
```

The tool will:

- Invoke the annotator via the command-line interface.
- Produce entity annotations for each abstract and display the results in the Unix terminal.

6.1.2. Output

For each abstract, the tool will output:

- Annotated entities with their normalized forms.
- The interactions between the recognized entities within the paper.

6.2. Research Toolset: BIONEN, BIONER and LLMRE

BioNEN was a previously developed [1] Python-based tool for Entity Normalization. It was a tool that achieved high accuracy in Entity Normalization and provided a simple-to-use command line interface. We have extended BioNEN and added support for Contrastive Similarity, pushing the tool's accuracy even higher while keeping the interface simple and accessible to its possible users.

We have introduced BioNER, a sister script of BioNEN that trains BioNEN inside but uses it to solve Entity Recognition. BioNER automatically normalizes recognized entities. While BioNEN aims to achieve high accuracy, BioNER aims to achieve high recall and stay an easy-to-use tool that also gives the option of using LLMs for higher precision at the cost of lower recall.

We also introduced LLMRE. LLMRE is a script for discovering relations in an abstract for which the recognition and normalization tasks are already completed.

6.2.1. Dependencies

6.2.1.1. Dependencies of BioNEN, and BioNER. The dependency requirements for both scripts are the same. BioNER first trains BioNEN and uses new implementations for Dictionary and Contrastive Similarity lookup that use the same set of libraries as BioNEN. The tool still requires previous libraries such as NLTK, scikit-learn, numpy, and transformers, but now it also requires the torch library, as it trains and uses a neural network inside.

The full set of libraries and their versions at the time of writing are:

- nltk: 3.8.1
- scikit-learn: 1.3.0
- transformers: 4.32.1
- pandas: 2.0.3
- numpy: 1.24.3
- torch: 2.2.1

Moreover, results are produced with Python 3.11.5.

6.2.1.2. Dependencies of LLMRE. LLMRE requires the following libraries:

- openai: 1.38.0
- tqdm: 4.65.0

Though we have provided the versions above, there is a high chance it will work with the latest versions of OpenAI and TQDm.

6.2.2. Changes to Command Line Interface for BioNEN/BioNER

The ease of use for BioNEN came from its simple CLI interface, which only required a terminal to work on. The basic structure of the CLI as described in [1] is:

```
$ python bionen.py --[model_name] "MODEL_NAME" --[dict_file]
"DICTIONARY FILE" --[dfs_data] "TEST_FILE" --[epsilon]
"DBSCAN_EPSILON"
```

Figure 6.2. BioNEN command structure.

The basic structure is the same for BioNER except that we now need to also give a training file. The file that tests BioNEN is what we train BioNEN with Contrastive Similarity on, so it is passed as a training file, and the test file is another test file for calculating BioNER's recall and accuracy on the recognition task.

The basic structure of BioNER:

```
$ python bioner.py --[model_name] "MODEL_NAME" --[dict_file]
"DICTIONARY FILE" --[dfs_data_training] "TRAINING_FILE"  --
[dfs_data_test] "TEST_FILE" --[epsilon] "DBSCAN_EPSILON"
```

Figure 6.3. BioNER command structure.

Both BioNEN and BioNER now support –pure_contrastive_similarity as an optional parameter. If this parameter is not passed, by default, Contrastive Similarity is used to normalize only the mentions, not normalized by the exact matches. If the parameter is passed, Contrastive Similarity is used to normalize all mentions, which still has very high accuracy but is lower than using mentions normalized by exact matches with Dictionary.

Similarly, both tools now require –loss_function parameter, which can either be "Euclidean" or "InfoNCE."

An example command to run the BioNEN with pure contrastive similarity:

```
$ python bionen.py --model_name "dmis-lab/biobert-v1.1" --dict_file
"ncbi_mesh_do_bc5cdr_umls.pk" --
dfs_data="data/TestSet.DNorm.PubTator.txt" --epsilon=0.05 --
function='Jaccard' --pure_contrastivc_similarity --
loss_function='Euclidean'
```

Figure 6.4. BioNEN example command with pure contrastive similarity.

Example of a BioNER run with mandatory additional training file:

```
$ python bionen.py --model_name "dmis-lab/biobert-v1.1" --dict_file
"ncbi_mesh_do_bc5cdr_umls.pk" --
dfs_data="data/TestSet.DNorm.PubTator.txt" --epsilon=0.05 --
function='Jaccard' --pure_contrastivc_similarity --
loss_function='Euclidean'
```

Figure 6.5. BioNER example command with additional training file.

Since we already explained the differences in the interface, and [1] has the details on the functionality of all arguments, we will not repeat that information here. However, BioNER has a few extra arguments we still need to mention.

6.2.2.1. BioNER Only Arguments. –relevance_filter is an optional parameter to filter created entity candidates with the BART model.

–open_ai_model_name is a required parameter for LLM recognitions.

6.2.3. Command Line Interface for LLMRE

Basic structure:

```
$ python llmre.py --[model_name] "MODEL_NAME" --[test_file]
"TEST FILE"
```

Figure 6.6. LLMRE command structure.

The model name should be the name of an available OpenAI model. The test file should be in PubTator format.

An example command to run the LLMRE:

```
python llmre.py --test_file=Test.PubTator --model_name=gpt-4o-mini
```

Figure 6.7. LLMRE example command.

7. CONCLUSION AND FUTURE WORK

It is a fact that the information we have available to us in any domain, including Biomedical data, is already beyond our capabilities to process without the help of AI/ML methods. LLMs are the first kind of AI models that process data at a large scale, using billions of words to train billions of parameters.

With such vast amounts of data, to develop a cure, a scientist might need to research chemicals, diseases, genes, species, and many more entities and relations between those entities. We need to be able to annotate and retrieve entities, extract relations, and map these entities to their scientific representations. For these purposes, we studied NER, NEN, and RE methods in this thesis.

We have improved the performance of a previously developed normalization tool. We used our normalization tool to recognize entities and produced a new tool. We added LLM capabilities to the newly produced recognition tool. Finally, we produced an LLM-based Relation Extraction tool. Our research toolset, which consists of these three tools, can be separately used to normalize, recognize entities, and extract relations. However, these tools are in research format. They produce statistics on the found data and are not focused on processing abstracts to produce annotations in a human-readable form.

Following this up, we produced an easy-to-use annotator, which uses selected best performers from research done on normalization and recognition and combines them with LLM extraction of interactions. The resulting application takes a paper and prints an annotated version with recognized and normalized entities, as well as entity pairs that are found to have a relation in between.

If we had the means, we could produce a huge dataset covering the most popular entity types and including normalizations for the entities for respective types. We would

also supply our annotator with a large training set of annotated papers containing these entity types. Supplying this data would increase our capabilities in annotating papers on different areas of Bioscientific research.

With limited data, we can specialize the annotator to a specific domain and assess performance; doing so mainly relies on having the necessary data for scoped-down entities. For example, to inspect host-pathogen interactions, we need a dictionary of hosts, a dictionary of pathogens, annotated papers with host and pathogens, and host-pathogen interactions to be used in training and test datasets.

LLMs are generalists that solve a wide variety of tasks, but they can be fine-tuned using specialized data. For example, we can fine-tune an LLM for Named Entity Recognition and Relation Extraction tasks, improving performance on both.

We can produce a search tool that annotates the given abstracts with our interaction finder and lists abstracts that contain a given relation.

Relations between different entity types in a paper produce a graph. We can implement a user interface for our interaction finder to visualize found interactions and a graph search that works similarly to interaction search. We can implement the necessary UI for users to construct a graph by simply dragging and dropping entities to form relations and searching for them in abstracts of scientific papers.

REFERENCES

1. Berke, K. and A. Özgür, "Symptom Normalization Using Unsupervised Learning and Text Similarity.", *Proceedings of the BioCreative VIII Challenge and Workshop: Curation and Evaluation in the era of Generative Models, American Medical Informatics Association 2023 Annual Symposium, New Orleans, United States*, 2024.

2. Gantz, J. and D. Reinsel, "The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East", *International Data Corporation iView: International Data Corporation Analyze the Future, Massachusetts, United States*, Vol. 2007, No. 2012, pp. 1–16, 2012.

3. "The Global Publishing Report 2021", United Nations Educational, Scientific and Cultural Organization, `https://uis.unesco.org/en/topic/publishing`, 2021, accessed: 2024-01-20.

4. Heath, A. P., E. Green, Y. A. Lussier and E. Ratner, "Big Data: The Rise of the Biomedical Data-Driven World", *Nature Biotechnology*, Vol. 38, pp. 996–999, London, United Kingdom, 2020.

5. OpenAI, "GPT-4 Technical Report", *arXiv Preprint*, Vol. 2303, arXiv: 2303.08774, 2023.

6. Chowdhery, A., S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "PaLM: Scaling Language Modeling with Pathways", *arXiv Preprint*, Vol. 2204, arXiv: 2204.02311, 2022.

7. Touvron, H., T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar and Others, "LLaMA: Open and Effi-

cient Foundation Language Models", *arXiv Preprint*, Vol. 2302, arXiv: 2302.13971, 2023.

8. Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language Models Are Few-Shot Learners", *arXiv Preprint*, Vol. 2005, arXiv:2005.14165,2020.

9. Luo, R., S. Sun, Y. Xia, T. Qin, S. Zhang, S. Piao, W. Chen and T.-Y. Liu, "BioGPT: Generative Pre-Trained Transformer for Biomedical Text Generation and Mining", *Briefings in Bioinformatics*, Vol. 23, No. 6, Oxford, United Kingdom, 2022.

10. Sayers, E. W., E. E. Bolton, J. R. Brister, K. Canese, J. Chan, D. C. Comeau, R. Connor, K. Funk, C. Kelly, S.-H. Kim *et al.*, "Database Resources of the National Center for Biotechnology Information", *Nucleic Acids Research*, Vol. 50, No. D1, pp. D20–D26, 2022.

11. Jaccard, P., "Étude Comparative de la Distribution Florale dans une Portion des Alpes et des Jura", *Bulletin de la Société Vaudoise des Sciences Naturelles*, Vol. 37, pp. 547–579, 1901.

12. Levenshtein, V. I., "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals", *Soviet Physics Doklady*, Vol. 10, No. 8, pp. 707–710, 1966.

13. Winkler, W. E., "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage", *Proceedings of the Section on Survey Research Methods, American Statistical Association*, pp. 354–359, Virginia, United States, 1990.

14. Jones, K. S., "A Statistical Interpretation of Term Specificity and Its Application in Retrieval", *Journal of Documentation*, Vol. 28, No. 1, pp. 11–21, 1972.

15. Mikolov, T., K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word

Representations in Vector Space", *Proceedings of the International Conference on Learning Representations*, Scottsdale, Arizona, United States, 2013.

16. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser and I. Polosukhin, "Attention Is All You Need", *Advances in Neural Information Processing Systems*, Vol. 30, pp. 5998–6008, 2017.

17. Devlin, J., 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*, pp. 4171–4186, Minneapolis, USA, 2019.

18. Ester, M., H.-P. Kriegel, J. Sander and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, Portland, Oregon, United States, 1996.

19. Dogan, R. I., R. Leaman and Z. Lu, "NCBI Disease Corpus: A Resource for Disease Name Recognition and Concept Normalization", *Journal of Biomedical Informatics*, Vol. 47, pp. 1–10, 2014.

20. Li, J., Y. Sun, R. J. Johnson, D. Sciaky, C.-H. Wei, R. Leaman, A. P. Davis, C. J. Mattingly, T. C. Wiegers and Z. Lu, "BioCreative V CDR Task Corpus: A Resource for Chemical Disease Relation Extraction", *Database*, Vol. 2016, 2016.

21. Miranda-Escalada, A., D. Farré-Masip and M. Krallinger, "Overview of the Semantic Representations of Biomedical Documents Track: Semantic Textual Similarity and Symptom NER at BioASQ 8b", *Proceedings of the Conference and Labs of the Evaluation Forum*, Thessaloniki, Greece, 2020.

22. Lample, G., 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: Human Language Technologies, pp. 260–270, San Diego, California, United States, 2016.

23. Akbik, A., T. Bergmann, D. Blythe, K. Rasul, S. Schweter and R. Vollgraf, "FLAIR: An Easy-to-Use Framework for State-of-the-Art NLP", *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pp. 54–59, Minneapolis, United States, 2019.

24. Lee, J., 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*, Vol. 36, No. 4, pp. 1234–1240, 2020.

25. Neumann, M., D. King, I. Beltagy and W. Ammar, "ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing", *Proceedings of the 18th BioNLP Workshop and Shared Task*, pp. 319–327, Florence, Italy, 2019.

26. Krallinger, M., O. Rabal, S. A. Akhondi *et al.*, "Chemical Named Entity Recognition with Long Short-Term Memory Models", *BioCreative VI Challenge Evaluation Workshop Proceedings*, pp. 357–365, Bethesda, Maryland, USA, 2017.

27. Li, Z., J. Gao, Y. Peng, D. Wei and Y. Xiang, "CNN-BiLSTM-CRF Models for Biomedical Named Entity Recognition", *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 207–217, Melbourne, Australia, 2018.

28. El-Tantawy, W. H., "Biochemical Effects of Solidago Virgaurea Extract on Experimental Cardiotoxicity", *Journal of Physiology and Biochemistry*, Vol. 70, No. 1, pp. 33–42, 2014.

29. Rios, A. D., B. Heinzerling and M. Strube, "Dictionary-Based Biomedical Entity

Normalization Using Neural Word Embeddings", *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 1038–1046, New Orleans, Louisiana, USA, 2018.

30. Li, X., H. Sun and R. Nayak, "Learning to Rank for Biomedical Entity Normalization", *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1705–1714, Melbourne, Australia, 2018.

31. Logeswaran, L., M.-W. Chang, K. Lee, K. Toutanova, J. Devlin and H. Lee, "Zero-Shot Entity Linking by Reading Entity Descriptions", *arXiv Preprint*, Vol. 2005, arXiv:1906.07348, 2019.

32. Vijayaraghavan, P., R. Li and X. He, "Self-Attention Networks for Entity Normalization", *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 872–877, Florence, Italy, 2019.

33. Luo, L., P.-T. Lai, C.-H. Wei, C. N. Arighi and Z. Lu, "BioRED: A Rich Biomedical Relation Extraction Dataset", *Briefings in Bioinformatics*, Vol. 23, No. 5, p. bbac282, 2022.

34. Islamaj, R., C.-H. Wei, D. Cissel, N. Miliaras, O. Printseva, O. Rodionov, K. Sekiya, J. Ward and Z. Lu, "NLM-Gene: A Richly Annotated Gold Standard Dataset for Gene Entities That Addresses Ambiguity and Multi-Species Gene Recognition", *Journal of Biomedical Informatics*, Vol. 118, p. 103779, 2021.

35. Bunescu, R., R. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani and Y. W. Wong, "Comparative Experiments on Learning Information Extractors for Proteins and Their Interactions", *Artificial Intelligence in Medicine*, Vol. 33, No. 2, pp. 139–155, 2005.

36. Wei, C.-H., H.-Y. Kao, Z. Lu *et al.*, "GNormPlus: An Integrative Approach for Tagging Genes, Gene Families, and Protein Domains", *BioMed Research Interna-*

tional, Vol. 2015, 2015.

37. Collier, N., T. Ohta, Y. Tsuruoka, Y. Tateisi and J.-D. Kim, "Introduction to the Bio-Entity Recognition Task at JNLPBA", *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications (NLPBA/BioNLP)*, pp. 73–78, Geneva, Switzerland, 2004.

38. Yeh, A., A. Morgan, M. Colosimo and L. Hirschman, "BioCreative Task 1A: Gene Mention Finding Evaluation", *BMC Bioinformatics*, Vol. 6, pp. 1–10, 2005.

39. Smith, L., L. K. Tanabe, R. J. n. Ando, C.-J. Kuo, I.-F. Chung, C.-N. Hsu, Y.-S. Lin, R. Klinger, C. M. Friedrich, K. Ganchev *et al.*, "Overview of BioCreative II Gene Mention Recognition", *Genome Biology*, Vol. 9, pp. 1–19, 2008.

40. Wei, C.-H., H.-Y. Kao and Z. Lu, "SR4GN: A Species Recognition Software Tool for Gene Normalization", *PLOS ONE*, Vol. 7, No. 6, p. e38460, 2012.

41. Wei, C.-H., R. Leaman and Z. Lu, "SimConcept: A Hybrid Approach for Simplifying Composite Named Entities in Biomedical Text", *IEEE Journal of Biomedical and Health Informatics*, Vol. 19, No. 4, pp. 1385–1391, 2015.

42. Leaman, R., C.-H. Wei and Z. Lu, "tmChem: A High-Performance Approach for Chemical Named Entity Recognition and Normalization", *Journal of Cheminformatics*, Vol. 7, No. 1, pp. 1–10, 2015.

43. Wei, C.-H., B. R. Harris, H.-Y. Kao and Z. Lu, "tmVar: A Text Mining Approach for Extracting Sequence Variants in Biomedical Literature", *Bioinformatics*, Vol. 29, No. 11, pp. 1433–1439, 2013.

44. Lafferty, J., A. McCallum and F. C. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289, Williamstown, Massachusetts, United States, 2001.

45. Peng, W., C. Yan and T. Jiang, "Supervised Contrastive Learning for Medical Relation Extraction", *arXiv Preprint*, Vol. 2009, arXiv:2009.08473, 2020.

46. Huang, K., Y. Yin, Y. Long, W. Du, X. Yuan, J. Cheng, F. Chen and L. Yao, "CLIP-Event: Contrastive Learning from Information Propagation for Clinical Event Detection", *arXiv Preprint*, Vol. 2010, arXiv:2201.05078, 2020.

47. Rao, R., N. Bhattacharya, N. Thomas, Y. Duan, X. Chen, J. Canny, P. Abbeel and Y. Song, "Self-Supervised Learning of Protein Structure", *bioRxiv*, preprint number: 2021.02.12.430858, 2021.

48. Banville, H., O. Chehab, A. Hyvarinen and A. Gramfort, "Uncovering the Structure of Clinical EEG Signals with Self-Supervised Learning", *Journal of Neural Engineering*, Vol. 18, No. 4, p. 046020, 2021.

49. Hadsell, R., S. Chopra and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping", *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Vol. 2, pp. 1735–1742, IEEE, 2006.

50. Can den Oord, A., Y. Li and O. Vinyals, "Representation Learning with Contrastive Predictive Coding", *arXiv Preprint*, Vol. 1807, arXiv:1807.03748, 2018.

51. Schroff, F., D. Kalenichenko and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", *Proceedings of the Institute of Electrical and Electronics Engineers Conference on Computer Vision and Pattern Recognition*, pp. 815–823, Boston, Massachusetts, United States, 2015.

52. Chen, T., S. Kornblith, M. Norouzi and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations", *Proceedings of the International Conference on Machine Learning*, pp. 1597–1607, PMLR, Virtual Conference, 2020.

53. Khosla, P., P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu and D. Krishnan, "Supervised Contrastive Learning", *arXiv Preprint*, Vol.

2004, arXiv:2004.11362, 2020.

54. Islamaj, R., D. Kwon, S. Kim and Z. Lu, "TeamTat: A Collaborative Text Annotation Tool", *Nucleic Acids Research*, Vol. 48, No. W1, pp. W5–W11, 2020.

55. Yang, W., K. Vijay-Shanker and Z. Lu, "PubTator: A Web-Based Text Mining Tool for Assisting Biocuration", *Nucleic Acids Research*, Vol. 41, No. W1, pp. W518–W522, 2013.

56. Lewis, M., Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov and L. Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension", *arXiv Preprint*, Vol. 1910, arXiv:1910.13461, 2019.

57. Jonkers, R. E., R. P. Koopmans, E. Portier and C. J. van Boxtel, "Debrisoquine Phenotype and the Pharmacokinetics and Beta-2 Receptor Pharmacodynamics of Metoprolol and Its Enantiomers", *PubMed*, Vol. 256, No. 3, pp. 959—-966, 1991.

58. Karvellas, C. J., N. Safinia, G. Auzinger, N. Heaton, P. Muiesan, J. O'Grady, J. Wendon and W. Bernal, "Medical and Psychiatric Outcomes for Patients Transplanted for Acetaminophen-Induced Acute Liver Failure: A Case-Control Study", *Liver International*, Vol. 30, pp. 826—-833, 2010.

59. Chang, C.-C., S. Acharfi, M.-H. Wu, F.-T. Chiang, J.-K. Wang, T.-C. Sung and M. Chahine, "A Novel SCN5A Mutation Manifests as a Malignant Form of Long QT Syndrome With Perinatal Onset of Tachycardia/Bradycardia", *Cardiovascular Research*, Vol. 64, No. 2, pp. 268–278, 2004.

APPENDIX A: PERFORMING THRESHOLD OPTIMIZATION ON BIONER

While one can simply run the tool for all temperatures, there is a more cost-effective way that requires slight alteration of the tool. However, this alteration is only done for research purposes. As the tool is already quite complex with classifiers and neural networks inside, we did not want to further complicate it by adding this batch-processing functionality.

To cost-effectively try all thresholds, contrastive_similarity_recognition method needs to be updated. For each candidate, this method finds the most similar mentions and IDs and their similarities, and we check if this similarity is above the required threshold.

```
for mention, top_id, similarity in zip(top_mentions, top_ids,
   top_similarities):
    if top_id and similarity > self.similarity_threshold:
        if candidate not in found_scores or found_scores[
           candidate] < similarity:
            found_entities[candidate] = top_id
            found_scores[candidate] = similarity
            break
```

In this function, a threshold check can be done for all candidate thresholds and found entities, and their scores can be added to their respective dictionaries, giving all results in one run of the BioNER.