MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES

DEVELOPMENT OF DATA AUGMENTATION METHODS TO IMPROVE PERFORMANCE OF SUPERVISED MACHINE LEARNING MODELS IN NATURAL LANGUAGE PROCESSING

ABDUL MAJEED ISSIFU

MASTER THESIS

Computer Engineering Department

ADVISOR

Assoc. Prof. Dr. Murat Can GANİZ

ISTANBUL, 2022

MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES

DEVELOPMENT OF DATA AUGMENTATION METHODS TO IMPROVE PERFORMANCE OF SUPERVISED MACHINE LEARNING MODELS IN NATURAL LANGUAGE PROCESSING

ABDUL MAJEED ISSIFU

523618922

MASTER THESIS

Computer Engineering Department

ADVISOR

Assoc. Prof. Dr. Murat Can GANİZ

ISTANBUL, 2022

ACKNOWLEDGMENTS

All thanks and praises are due to Allah for giving me life and the ability to work on this thesis and my academic research. I give a special thanks to my advisor Assoc. Prof. Dr. Murat Can Ganiz for his guidance and support, his cordial and amicably help in the development of this research work, publications, and thesis completion.

I want to thank all my friends and colleagues at the @BigDataLab Laboratory in Marmara University who supported me morally and socially during the workflow.

Besides, I would like to thank my family and my friends for their continuous support and understanding. I want to deliver a special thanks to my dad and sister Alima for their inspiration and encouragement to further my studies and make the family proud.

July 2022 **Abdul Majeed ISSIFU**

TABLE OF CONTENTS

ÖZET

DOĞAL DİL İŞLEMEDE DENETİMLİ MAKİNE ÖĞRENİMİ MODELLERİNİN PERFORMANSINI ARTTIRMAK İÇİN VERİ ZENGİNLEŞTİRME YÖNTEMLERİNİN GELİŞTİRİLMESİ

Başlangıçta, “Basit veri zenginleştirme” (Easy Data Augmentation) metin sınıflandırma görevleri için geliştirilmiştir. Temel olarak bu yaklaşım dört yöntemi kapsar: Eşanlamlı Değiştirme, Rastgele Ekleme, Rastgele Silme ve Rastgele Değiştirme. Bunlar derin sinir ağı modellerinde doğruluğu artırmak için hizalanır. Bu çalışma, bu yöntemleri medikal alanda Adlandırılmış Varlık Tanıma görevleri için genişletmeyi amaçlamaktadır. Adlandırılmış varlıkların (cümlelerdeki bir kelime veya kelime gruplarından veya ailelerden oluşan) doğası, veri zenginleştirme alanında bazı zorluklar getirse de, adlandırılmış varlık tanıma başarımını iyileştirilmesine öne sürmektedir.

Bu yöntemleri biyomedikal kıyaslama veri kümelerinin boyutunu artırmak ve biyomedikal adlı varlık tanıma modellerinin performansını geliştirmek için kullanıyoruz. BERT gibi dönüştürücü modelleri üzerinde yapılan çalışmaları değerlendirmek için deneyler yaptık. Aktarım yoluyla öğrenme ile, veri kümeleri üzerinde bir biyomedikal dil modeli olan BioBERT'e ince ayar yaptık. BioBERT ve BERT modelleri ile tüm veri setlerinde genel bir iyileştirme ve BC5CDR-hastalık veri setinde sırasıyla %5.95 ve %8.49 F1 puanı artışı sağladık.

ABSTRACT

DEVELOPMENT OF DATA AUGMENTATION METHODS TO IMPROVE PERFORMANCE OF SUPERVISED MACHINE LEARNING MODELS IN NATURAL LANGUAGE PROCESSING

Originally, Easy Data Augmentation holds its development to tasks of text classification. Basically, it encapsulates four methods: Synonym Replacement, Random Insertion, Random Deletion, and Random Swap aligned to improving accuracy on several deep neural network models. This study aimed at deploying these methods to new domains by augmenting Named Entity Recognition datasets from the medical domain. Although the nature of the named entities (consisting of a word or word groups or families in sentences) posed some challenges to the augmentation task, a case is advanced that an improvement of the named entity recognition performance is achievable.

We use these methods to increase the size of biomedical benchmark datasets and improved the performance of biomedical named entity recognition models. We carried out experiments to evaluate the work on transformer model like BERT. With transfer learning, we fine-tuned BioBERT, a biomedical language model on the datasets. We achieved a general improvement on all datasets and a 5.95% and 8,49% increment of F1-score on BC5CDR-disease dataset with BioBERT and BERT models respectively.

LIST OF SYMBOLS

Q: Computational cost

N: Number of times of augmentation for each method

n: Number of folds of augmentation per each sentence

$\boldsymbol{\lambda}$**:** Coefficient for controlling the degree of interpolation or extrapolation

$\vec{\boldsymbol{Q}}$: Query vector for k-nearest neighbor

$\vec{W}$: Query vector for k-nearest neighbor

$\boldsymbol{\omega}$: word vector selected for search

<bos>: Beginning of a sentence

$\boldsymbol{w_j'}$**:** The novel word embedding vector

$\boldsymbol{w_k}$**:** Centroid vector from

LIST OF ABBREVIATIONS

NLP: Natural Language Processing

POS: Part-of-speech

NER: Named-Entity Recognition

CBOW: Continuous bag-of-words model

ULMFiT: Universal Language Model Fine-Tuning

TF-IDF: Term Frequency - Inverse Document Frequency

PHI: Protected Health Information

EDA: Easy Data Augmentation

cTAKES: Clinical Text Analysis and Knowledge Extraction Systems

OOV: Out of vocabulary

MaSCiP: Multidisciplinary Association of Spinal Cord Injury Professionals

i2b2: Informatics for Integrating Biology and the Bedside

UMLS: Unified Medical Language System

NCBI: National Center for Biotechnology Information

BC5CDR: BioCreative-V Chemical-Disease Relation

BIO: Beginning Inside Outside

MaSciP: Multidisciplinary Association for Spinal Cord Injury Professionals

GloVe: Global Vectors

CV: Computer Vision

GPT-2: Generative Pre-trained Transformer 2

SOTA: State-of-the-Art

NN: Neural Network

CNN: Convolutional Neural Network

GRU: Gated Recurrent Units

LSTM: Long Short-Term Memory

biLSTM: Bidirectional Long Short-Term Memory

DL: Deep Learning

Seq2seq: Sequence to Sequence

BERT: Bidirectional Encoder Representations from Transformers

BioBERT: Biomedical Bidirectional Encoder Representations from Transformers

SciBERT: Scientific Bidirectional Encoder Representations from Transformers

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Machine learning and deep learning models perform adequately on various NLP tasks and have so far gotten high accuracy. Sentiment analysis (XLNet [1]), Text classification (ULMFiT [2]), Named entity recognition (LUKE [3]), Question answering (Gated-Attention Reader [4]), Summarizing (RNES [5]) and many more have shown important advancement in the field of NLP. Many of these models are supervised and thus largely depend on labeled data, which is scarce and costly to obtain. The use of semi-supervised learning, transfer learning, and data augmentation are the current remedy to this problem. In semi-supervised learning, researchers use a small amount of labeled data in combination with a large number of unlabeled input instances during the training of machine learning and deep learning models. This way, semi-supervised learning uses the knowledge learned from the small amount of labeled data to label the available vast unlabeled data. Data augmentation is, of course, a relatively simpler solution to solving the problem aforementioned. It also has the advantage of allowing researchers to use any kind of supervised algorithm to increase data instances without finding new samples from scratch.

Data augmentation is especially important currently in general artificial intelligence research. In deep learning, it generally reduces overfitting of models. Another very crucial reason for studying and developing data augmentation methods is to artificially increase supervise learning training data. This also helps in generalization of the models.

Data augmentation is heavily studied in the computer vision domain and resulted in fairly advanced methods, such as Auto-Augment [6], Learning Augmentation Policies from Data, Random Erasing Data Augmentation [7], and Albumentations [8]. Data augmentation studies in NLP are relatively new and we have much fewer methods in comparison. Google researchers [9] propose augmentation techniques for text classification where back translation and TF-IDF were used. Many, if not all, the text augmentation approaches in NLP are intended for text classification and there are only a few attempts for NER such as [10]. NER is different from sentence-level or document-level NLP tasks as it requires token-level sequence labels. It is not straightforward to apply data augmentation methods developed for example sentence-level NLP tasks such as text classification. On the other hand, NER is a crucial NLP task that gets attention, especially in the medical domain.

The exponential rise in data generation in these recent days has subsequentially affected the research industry. The field of medicine and clinical industries is one of the domains that also produce data in unstructured text format. Medical text data comes in the form of discharge summaries, radiology reports, clinical notes, etc. These enormous amounts of data are of very importance to the field of deep learning and machine learning.

One of the important tasks of Natural Language Processing (NLP) is Named Entity Recognition (NER). NER is a form of information extraction where named entities are extracted from raw text documents. Named entities here represent predefined objects, words, or word groups that are labeled such as a person, location, date, organization, etc. This task is sometimes called, Entity Identification, Entity Chunking, or Entity Extraction, and its main aim is to locate and designate the above-mentioned entities. Use cases of NER can be seen in big companies where many requests are received daily. These requests include sales, installations, maintenance, complaints, troubleshooting, etc. NER help in extracting information and understanding the specific request of the customer. The use of NER can also be applied to resumes, where applicants' resumes are filtered out for finding apt candidates for a specific job.

Medical data contains valuable information in the form of narratives or hospital/clinical discharge summaries. There is a large amount of research on how to use this vital information of patience in the hospital to develop artificially intelligent systems and to improve the health care of individuals. Since the data is both crucial and constrained for research, getting them is not easy and labeling them for research needs human expertise. Nevertheless, access to medical data does not come on a silver plate since it contains information that could be used to identify specific individuals. So, the data is highly protected as Protected Health Information (PHI). In this study, we use four basic methods of data augmentation originally proposed by Wie et al. [11] to create more realistic and diverse augmented medical data for named entity recognition.

Biomedical text data used as benchmark datasets are what we used in this work. They include diseases, chemicals, genetics, proteins, and species entities in general biomedical text data. We developed data augmentation methods for named entity recognition on biomedical data. Four methods commonly used for augmentation of text for classification are what we used and adapt them all to NER. We developed biomedical NER models for each of the benchmark datasets. Testing of the augmentation is done with these developed models. We record and

compare the performance of each model on the original dataset and its corresponding augmented version. We obtained promising results that show clearly that text augmentation can improve the performance of deep learning models and especially biomedical named entity recognition models.

The organization of this work is as follows. Section 2 summarizes the background and related work. EDA and NER, the model for data augmentation are provided in Approach section 3. Section 4 includes detailed information about the experimental setup, the datasets used in the work, and the results and discussion can be found in section 5. Finally, this thesis concludes in Section 6 with future work and references in section 7 and section 8 respectively.

2. RELATED WORK

In this section, we shall talk about the various related work and methodologies that have so far been proposed in this field of study. We looked at the concepts in a variety of sections. This covers data augmentation and its application on models in the medical field, Named Entity Recognition (NER), and NER in the medical domain.

2.1. General Text Augmentation

Data augmentation (DA) in Natural Language Processing (NLP) has no clear-cut method to directly increase the size and diversity of labeled data, nevertheless, researchers are tirelessly working on techniques to improve DA methods in NLP. The techniques proposed so far have shown impressive results and have significantly improved the performance of many diverse NLP downstream tasks. Back-translation is one of the promising methods, proposed by google AI where they take data samples x in a language A translate it to another language B, and then translate it back to A to obtain augmented data sample x' [9]. In the same research, the authors calculate TF-IDF scores of tokens in a training set and then replaced uninformative words with more informative ones, to generate new instances for topic classification tasks. Zhang et al. [11], Wei, and Zou [12] in their work used methods where tokens are replaced by their synonyms from WordNet and/or a predefined language model [13]. The general objective of data augmentation especially in the text is to significantly increase the diversity of the available text data for training models, without actually collecting new labeled text data.

Many of the methods of text augmentation are for classification tasks in NLP. To make data augmentation for NER tasks, Dai et al. [36] in their paper "An Analysis of Simple Data Augmentation for Named Entity Recognition" proposed:

- Label-wise token replacement: This is a method wherein in each sentence, tokens that share the same labels are randomly replaced with tokens from the original data set.
- Mention replacement: Replace a mentioned entity and its label tokens from the original data set that shares the same entity label.
- Shuffle within segments: Here a sequence of tokens is divided into segments and then shuffle the tokens in each segment without changing the labels. Segmentations are done at every start of a mentioned entity in the sequence. An example of a sequence and the process

of forming augmented data is using the shuffle with segment method is shown in Figure 2.1**Error! Reference source not found.**

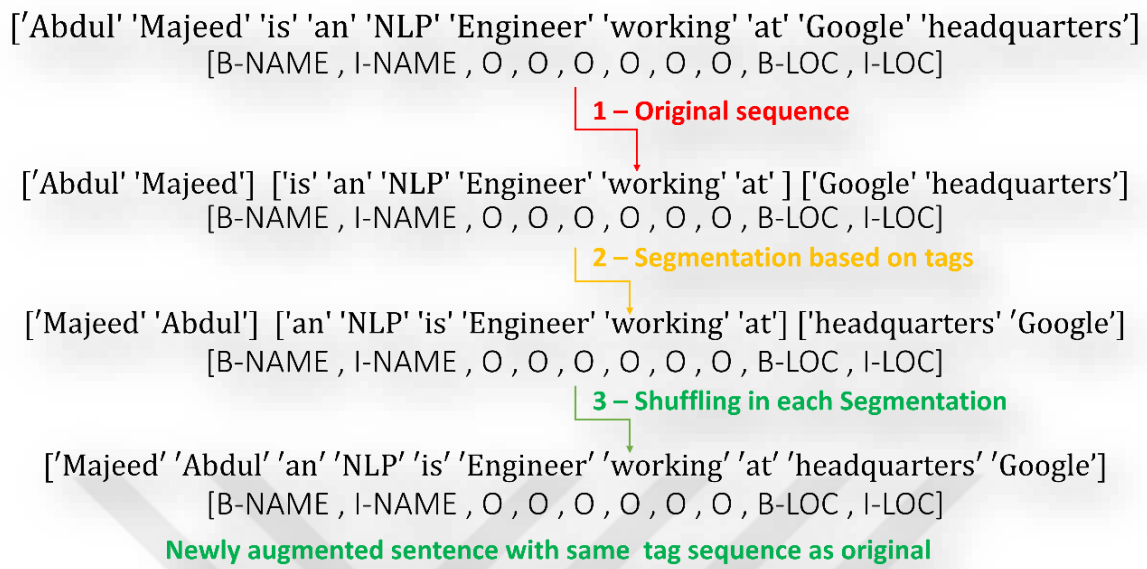

Figure 2.1: Augmentation of a sentence using shuffle within segments methods. From 1 – 3 shows the process of augmentation.

Their methods improved the performance of both the transformer model and RNN model after experimenting with two domain-specific data sets MaSciP [14] and i2b2-2010 [15].

Tian Kang et al. [16] in their work, presents an extension of "Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks" (EDA) [12] methods, by featuring Unified Medical Language System (UMLS) [17], and adapting the methods for named entity recognition tasks to improve performance of models in both classification and named entity recognition in the biomedical domain. UMLS, a knowledge-based system, is not easily accessible, and setting it up is both time-consuming and highly costly. This makes it not suitable for low-resource settings. Nevertheless, UMLS-EDA [16] enables substantial improvement for NER tasks and improves the performance of state-of-the-art text classification models. In our approach, we extend the methods used in EDA, by modifying them to a low-cost and easy-to-use setting, for medical text augmentation.

Data augmentation methods for text data can be grouped into those applied in feature space of embeddings and that of the data itself. While feature space approaches tackle the subject matter in the embeddings of the available text, data space methods of data augmentation deal

with the changing of the text itself in its original format. Figure 3 shows a summary of the taxonomy and categorization of various methods of data augmentation.

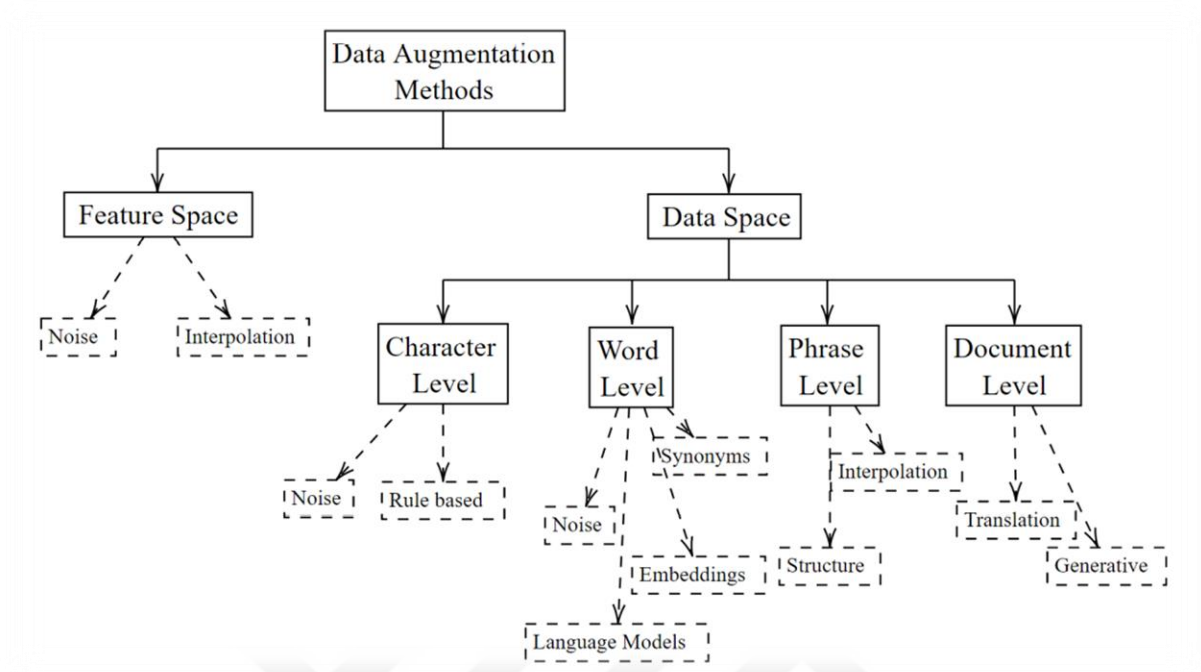

Figure 2.2: Summary of the Taxonomy and categories of various data augmentation methods (adapted from [18]**).**

2.1.1. Rule-based Techniques for Data Augmentation

The use of manually laid down rules and algorithms to increase the input data of any deep learning or machine learning model is classified as a rule-based technique for data augmentation. These techniques are one of the first proposed methods for DA in all downstream NLP tasks. They are easy to implement and improve the performance of the underlining models but at the same time costly to implement [19].

In Game-theory, Monte Carlo Tree Search (MCTS) can be used to predict moves and then counterattack an opponent's strategy to reach a Nash equilibrium or winnable state. Taking into consideration a one-player game setting, decisions are paramount to the selections of tokens from one state to another. Quteineh et al. [20] use this concept of MCTS to make data augmentation for small text datasets. Tokens are represented in a tree form. Representation is based on the pre-calculated probability of their appearances in the general corpus. A tree search is conducted. More informative search result from the tree is selected and concatenated to form new data. Figure 2.3 shows an example of MCTS traversing down the tree. As it moves down

the tree, it creates paths of which, tokens of the same path form a new sentence. This can be seen as the path in red in Figure 2.3. Though this method is good for text augmentation in sentiment analysis, it will be difficult to apply it to NER. This is because, at every state of the game, entities in the sentence will be given more priority and are likely to be selected than that of other tokens.

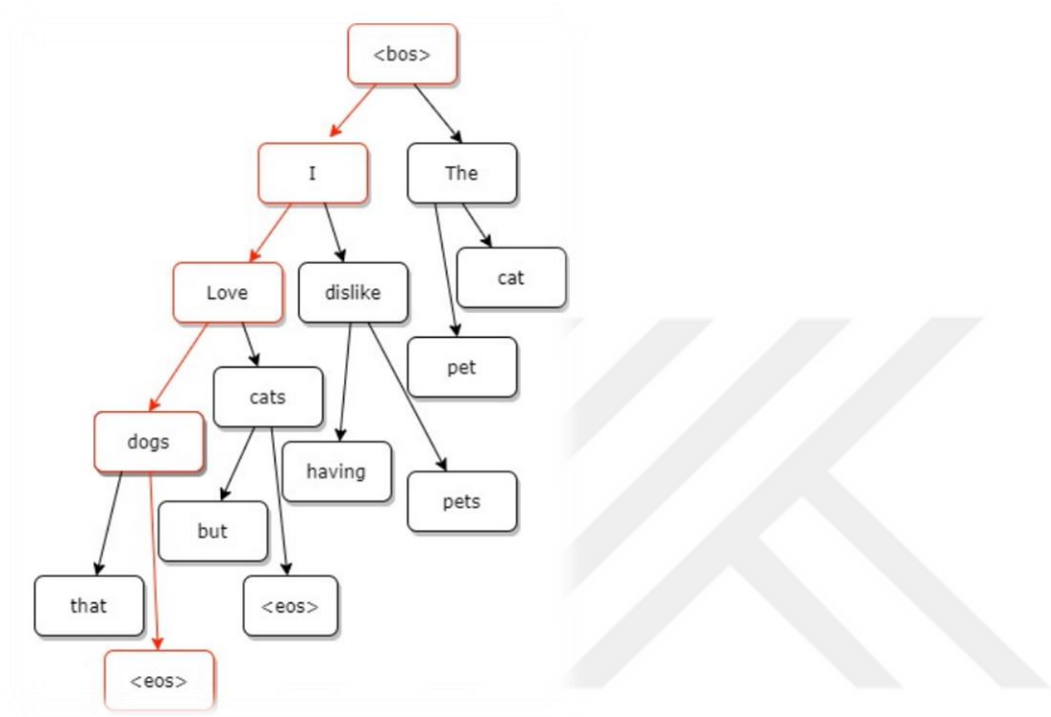

Figure 2.3: Monte Carlo Tree Search traversing down a tree from the root node <bos> to the end node <eos> (Adopted from [20]**).**

2.1.2. Noise Injection as a method of data augmentation

Randomly injecting noise in a text can be used as data augmentation. This comes in a close continuous change in the text. Insertion of characters, deletion, and modification of punctuations are just a few ways of injecting noise into the text.

Injection of noise into text is considered data augmentation in neural networks since it is possible that the scalability of the model is increased by producing new instances. A powerful regularization method in neural networks, dropout is used to prevent complex co-adaptations

of the model on the noise injected training data [21]. Using misspelling datasets and typos in place of the correct words is also considered data augmentation.

Many NLP models including neural machine translation extract stem and morphological information based on character and other sub-words units in the data provided. They then used this information to generalize and to unseen words and conjugations [22]. NMT models cannot comprehend simple noisy data due to this.

In the work of Belinkov and Bisk [22], they use naturally occurring mistakes in text data and synthetically generated noise to train NMT which increased the robustness of the model. These noises include typos, misspellings, grammatical errors, manually deleting starting and ending characters of words [23], keyboard typos, etc.

2.1.3. Deep Learning Methods for Data Augmentation

Deep learning generally outperforms traditional machine learning models and methods in solving current AI problems. Interestingly deep learning techniques can apply in the generation of synthetic and high-quality data instances artificially, to help in their training. Using deep learning methods for data augmentation could be done in a variety of ways including the manipulation of the embedding layers, the use of the transfer learning of language models, the use of text generative models, and many more. Many data augmentation methods especially those that rely on the simple transformation of data instances are only applicable in their respective fields e.g., computer vision (CV), and cannot be applied in other fields e.g., natural language processing (NLP) and vice versa. A more interesting thing about deep learning methods for data augmentation is that they can be used in both CV and NLP.

2.1.3.1. Embeddings in Deep Learning for Data Augmentation

The use of word embedding substitution as a data augmentation method surprisingly gave substantial results in name entity recognition (NER). Habitually used of word embeddings include Word2Vec [24], GloVe [25], fastText [26] and word embedding systems such as SENNA [27], [28]. SENNA is a standalone, self-contained system used to output word embeddings for Part-Of-Speech Tagging (POS) system, word relatedness modeling task, word analogy task, and word similarity task [27], [28]. In this approach of data augmentation, new

data instances are generated based on the vector similarities. The vectors could be of selected words, sentences, or even a document, in the embedding layer of the neural network. In [29], Yang et al. use lexical and semantic embeddings of words and sentences to generate novel data. They trained a multi-class classifier of annoying behaviors using Twitter as a corpus. The nearest neighbor of a word vector is calculated using cosine similarity. This novel vector is used to replace the original word. The word selection, as well as replacement, are both done in a non-deterministic manner. That is for all tokens in a vocabulary W, they search for the k-nearest-neighbor (knn) word ω for the query terms using cosine similarity between the query $\vec{Q}$ and target word vectors $\overrightarrow{W}$ using equation (1).

$$Knn(of\ all\ tokens) = arg\max_{\omega \epsilon W} cosine\ \left(\vec{Q}, \overrightarrow{W}\right) \quad (1)$$

Pre-trained models also play many roles when it comes to the use of embeddings for data augmentation as we mentioned above. A pre-trained model in the nutshell is a model that is previously trained using large amount of text data. These pre-trained models are used to solve similar problems that they were developed to solve.This also known as transfer learning,where the weights of a pretrained model is used in the training of a different model. This is acquired by adjusting few settings to suit our solution that is fine-tuning to our task. In the field of text classification, new embeddings could be generated from the input sequences, token by token, and keep the original label of the sequence intact. This way, new training inputs are generated at the embedding layers of the model, and the training instances are increased which eventually increases the performance of the classifier model.

2.1.3.2. Interpolation and Extrapolation method for data augmentation

Generating synthetic data examples is what matters most in data augmentation. For a given sentence, obtaining a new embedding vector of the original vector in the embedding space eventually forms a new data input. This is done by first obtaining the nearest neighbors and the centroid of the given word in the embedding space. The novel word embedding vector w_j' is calculated from the centroid w_k using equations 2 and 3 for interpolation and extrapolation, respectively. The green points (l1, l2, and l3) in Figure 2.4 and Figure 2.5**Error! Reference source not found.**, represent the selected nearest neighbors. These nearest neighbors are used to calculate te centroid marked as the red indicator (l6). The blue point (l4) in **Error! Reference s ource not found.** and **Error! Reference source not found.** indicates the original embedding

vector, and the yellow mark (l5) is the resulting novel vector. The black points in the figures are the other word embeddings.

$$w_j' = (w_k - w_j)\lambda + w_j \quad (2)$$

$$w_j' = (w_j - w_k)\lambda + w_j \quad (3)$$

The original word w_j is subtracted from the centroid in interpolation and for extrapolation the centroid w_k is subtracted from the original word. The parameter λ which ranges from $[0 - 1]$ and $[0 - \infty]$ controls the degree of interpolation and extrapolation, respectively. Figure 2.4 and Figure 2.5 shows more details of this method as used in the work of Papadaki [30] and the survey of Giridhara et al. [31]

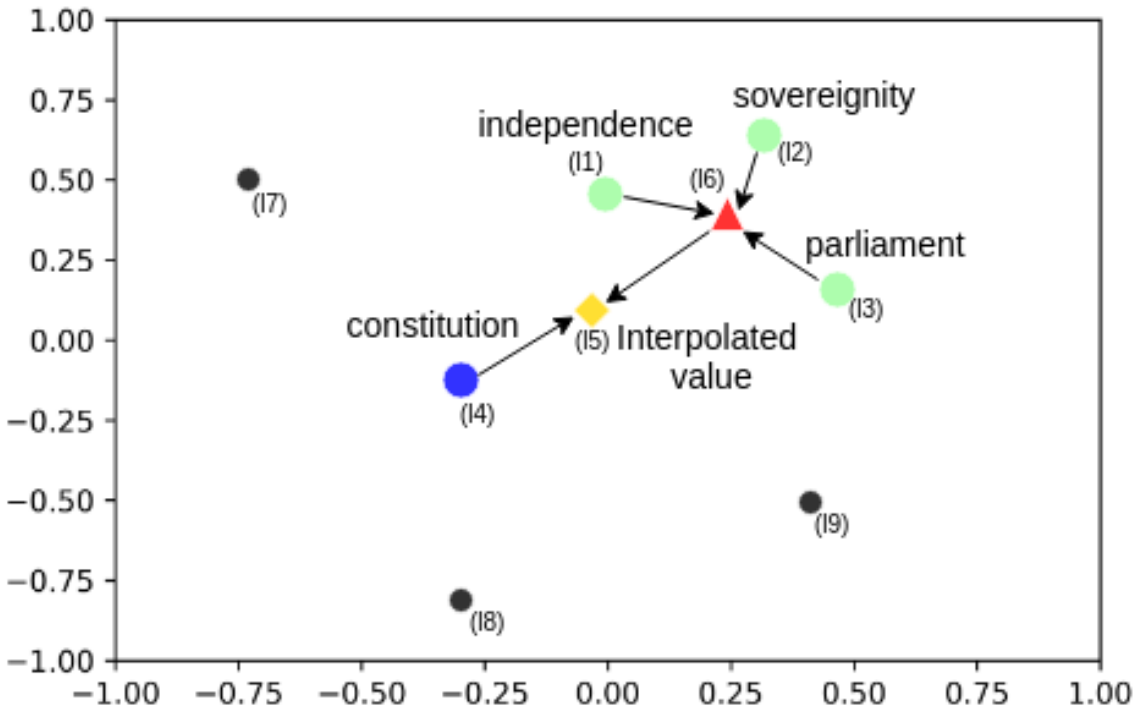

Figure 2.4: Creating a new embedding vector using Interpolation of centroid and word embedding (Taken from [31]**).**

The technique of removing a non-linear hidden layer in a deep learning neural network to decrease complexity and projection is quite an interesting method for data augmentation. This is manifested in the work of Caroline et al. [32] where they used Word Embedding Substitution

(WES) for low-resource Arabic language [32]. They gain a performance increment on Arabic English language pair dataset with an increment of 1.51% in F1-score.

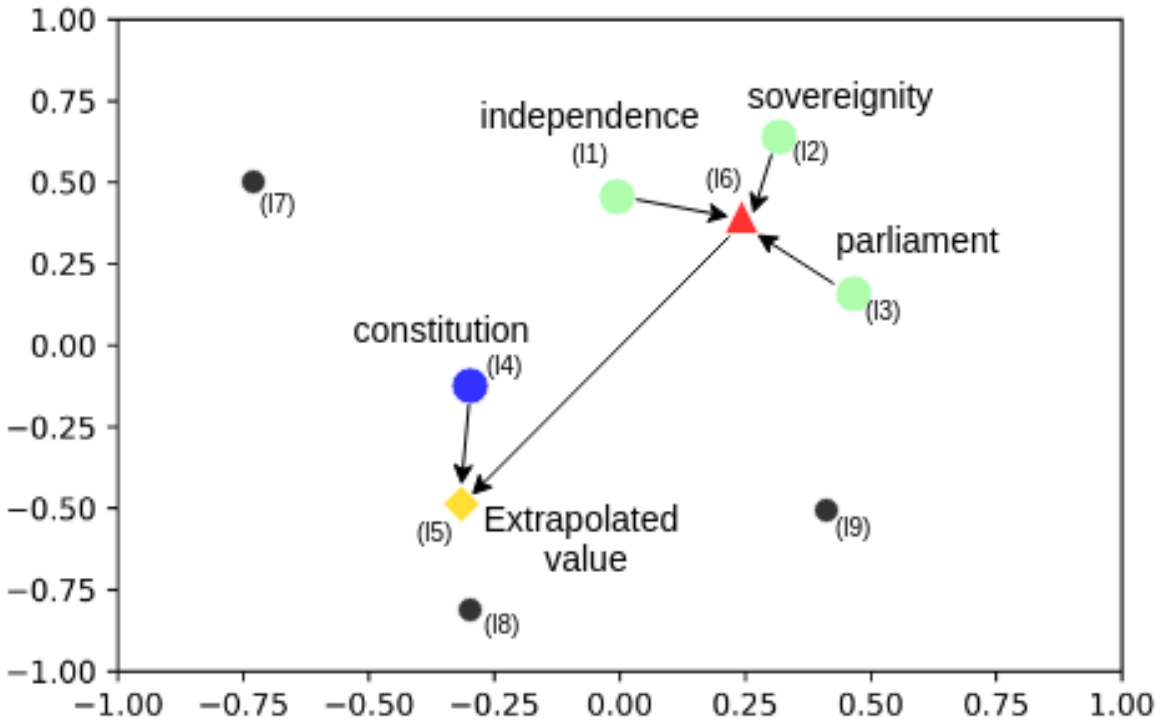

Figure 2.5: Creating a new embedding vector using Extrapolation of centroid and word embedding (Taken from [31]**).**

Manipulation of the embedding vectors helps in generating new instances for training, which in turn helps in generalization, sparsification, and data augmentation of the available data and model. The work of Zhang et al.[33], Zhang et al. [34] and Jindal et al. [35] show that manipulation of the embeddings of a model creates new instances for training. Assuming we have two randomly selected data points in the embedding space, (x_i, y_i) and (x_j, y_j), where x_i and x_j are the input data samples and y_i and y_j are the embeddings of the labels. With linear interpolation of input data, Zhang et al. [33][34] MixUp generates new virtual training samples by creating a mix function using the equations (4) and (5).

$$x = mix\left(x_i, x_j\right) = \lambda x_i + (1-\lambda)x_j \quad (4)$$

$$y = mix\left(y_i, y_j\right) = \lambda y_i + (1-\lambda)y_j \quad (5)$$

This formulation of mixed algorithms generates synthetic data in models trained for image processing but has less performance in text data due to the inability of computing discrete

tokens. A modification of the MixUp formula by Jindal et al.[35] adapts this idea to text augmentation by training a model on interpolations of hidden states with reference to embeddings of training instances. The modified formulation is shown in equation (6) and (7).

$$mix\left(y_i, y_j\right) = \frac{\mathrm{t}(x_i - \mu_i) + (1-t)\left(x_j - \mu_i\right)}{\sqrt{t^2 + (1-t)^2}} \quad (6)$$

$$where\ t = \frac{1}{1 + \frac{\sigma_i}{\sigma_j} . \frac{1-\lambda}{\lambda}} \quad (7)$$

2.1.3.3. Language models for text augmentation

For many years now language models and generative models achieved state-of-the-art (SOTA) on a majority of downstream NLP tasks. Usage of language models for example BERT [36] and text generative models like GPT2 [37] have become almost a norm. This is due to the representation of words done with context taken into consideration. Given two inputs *River [bank]* and *[bank] deposit*, Word2Vec [24] model, for example, will map the word "bank" to the same representation since it has no context consideration. Other models like RNN [38] will tokens before the masked word when representing them. BERT [36] will naturally consider the context in which the token appears. Both words before and after the masked token are taken into consideration. That means BERT will be able to predict *"bank"* with a representation of the financial institution not a *"bank"* at the river when given the *sentence "I made a [bank] deposit".*

With that said, masking some percentage of tokens in the training set, predicting those masked tokens with language models like BERT will always give new tokens and for that matter new augmented data as can be seen in Figure 2.6.

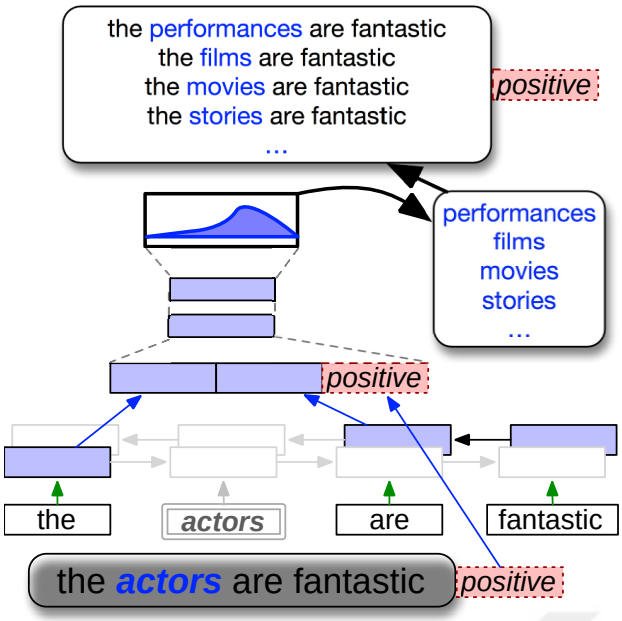

Figure 2.6: Data augmentation with language models (Taken from [39]**).**

2.1.4. Back-Translation for Data Augmentation

One of the tasks of NLP is neural machine translation. This implies the translation of a language into another language. Machine translation is built with encoder-decoder architecture where a word sequence is sent as inputs to the encoder. The encoder formulates a real-value vector mapping of these sequences and then sends it to the decoder. The parameters of both encoder and decoder are learned together to increase the target sentence's likelihood, given the corresponding source sentences. To train a machine translator, a parallel corpus is needed. This is a corpus that contains both the original sentences and their corresponding translation in a different language.

In machine translation, since the decoder is a neural language model conditioned on the encoder, outputs different sequences which could be of different lengths than the input sentence, translating these sequences again back to the original language will generate a new but remarkably similar sequence in the original language.

Back-translation [9], [40], [41] used for data augmentation means translating input data x in language A to a different language B to obtain y and then translating y back to A obtaining new input data x'. In the work of Xie et al. [9], they presented unsupervised data augmentation, and one of their methods is back translation. Two models are trained separately on unsupervised data and augmented unsupervised data. The consistency loss of these two models is calculated.

TF-IDF word replacement and back translation are the two methods used for text augmentation in this work which generated quality data used for training classification models. An example of back translation is manifested in Figure 2.7, where sample text is back-translated from other languages, forming new sentences.

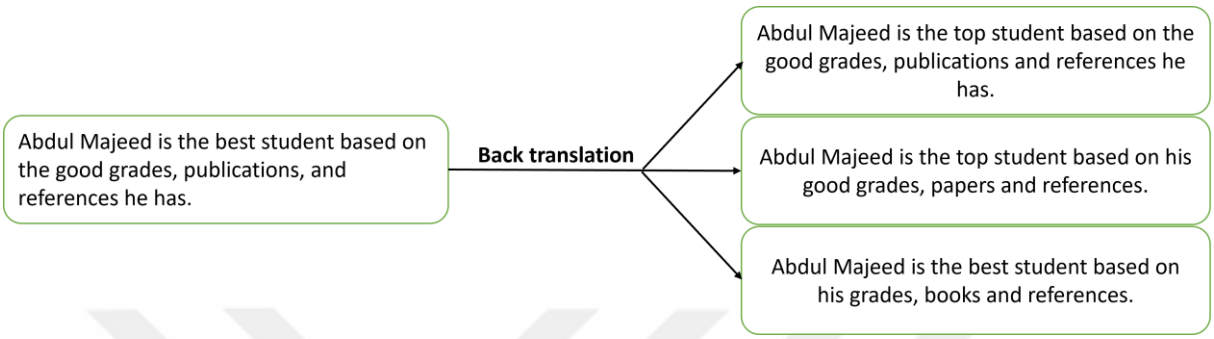

Figure 2.7: Sample data augmentation using back translation.

2.2. Named Entity Recognition (NER)

As we mentioned above, NER identifies and groups key information in text documents into predefined types/entities. Even though NER is a vital tool for information extraction and retrieval [42], it has also been used for other NLP downstream tasks like question answering [2] and text summarization [5]. Much of the research is now moving towards solving domain-specific NER tasks. BioBERT [43] is a NER model for biomedical text data (ranked number one for NER on NCBI-disease [44]), and SciBERT [45] is a NER model for scientific text data (ranked number one for Relation Extraction on SciERC [46]). Though there are other domain-specific NER tasks, most of the advancements are seen in the medical sector of research. Figure 2.8 shows how a NER algorithm can highlight and extract entities from a given text document.

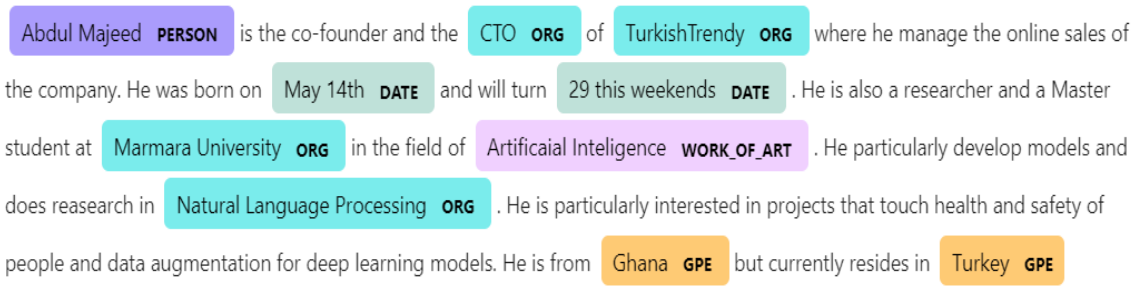

Figure 2.8: NER algorithm tagging out entities from a given text.

The techniques and approaches used for NER so far could be classified into two major categories.

- The traditional approach to solving NER tasks and
- Deep Learning (DL) methods of solving NER tasks

2.2.1. Traditional approaches in NER

The traditional approaches are the earlier ways of solving NER tasks. They include:

- Rule Based: In this approach, NER models rely totally on scrawl rules e.g., domain-specific gazetteers [47], syntactic-lexical patterns [48], and dictionary-based [9], [49], [50].
- Unsupervised and Semi-supervised Learning Approaches: As the name suggests, fully unlabeled (unsupervised) or partially labeled (semi-supervised) data are clustered into groups, and extraction of named entities is done based on contextual similarities. [48], [51]–[53]. Ando and Zhang [54]. in their work showed the use of semi-supervised learning with linear models. They trained the model on annotated data, and then added non-annotated inputs, later on, resulting in an F1 score of 89.31%

The traditional approaches have high precision vs. low recall in their model test and evaluation results. They are also both time-consuming to build and domain knowledge is needed.

2.2.2. Deep learning approaches in NER

Deep learning (DL) is a subfield of general machine learning that uses algorithms inspired by the structure and functions of the human brain, popularly known as artificial neural networks. DL-based approaches for NER are currently the most preferred and have so far remained the State-of-the-Art (SOTA) for most of the benchmark datasets. This could be because unlike rule-based approaches, DL has a large number of processing layers in the architectural design, which may learn and discover features automatically.

The neural network (NN) architecture in DL computes a weighted sum of the input data from a layer before it, then passes the results to a non-linear activation function. The non-linear mapping of inputs to outputs in DL makes it suitable for NER to be able to learn complex features from the given data. General training of the DL model for NER can be done in the following 3 steps as seen in **Error! Reference source not found.**.

1. Distributed representation of inputs (Word Embedding): This is the stage where all text is converted to vectors as a representation of the text for deep learning neural network architectures. Words of the same meaning are given the same representation. Words with similar meanings turn to be closer to each other in their vector space.
2. Context Encoder: This is the actual neural network model. Here the model is created. It takes in the word representation from step one and based on the context of the words, learns features. Instead of looking at words one by one, here every token is taken into consideration based on their context of appearance in the sentence. Language models, sequential models, and transformer models are examples.
3. Tag decoder: This is the final stage where the labels/tags are grouped with entities. CRF is a good example, which maps the input sentences with their predicted entity tags.

A general taxonomy of DL-based NER is manifested in **Error! Reference source not found.** m any DL architectures are proposed for NER tasks. These architectures include LSTM, CNN, LSTM-CRF, GRU, and some combination of other architectures. Fine-tuning the pre-trained language model for the NER task is a preferred approach currently followed in DL-based NER approaches.

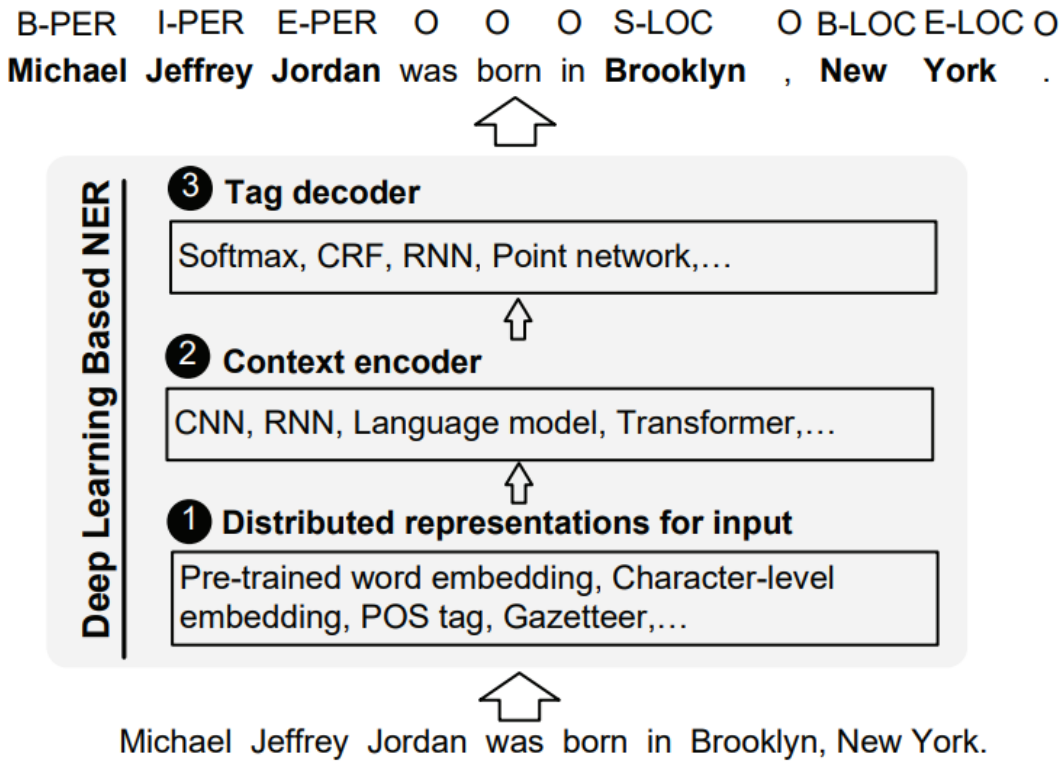

Figure 2.9: Overview of DL-based NER from the input to the output (Taken from [23]**).**

The bidirectional LSTM (biLSTM) + CRF architecture was popular obtaining best results before thee introduction of transformers models in solving NER tasks. The biLSTM model consists of two LSTMs, each of which processes a given sequence in a different and opposite direction. The ability to work with sequential input and the capacity to recognize long-term dependencies because of a memory cell is the fundamental features of LSTM.

The hidden states vectors $H = (h_1, h_2, \ldots, h_n)$ are the output of the LSTM, which receives a series of vectors $X = (x_1, x_2, \ldots, x_n)$ as input. Where x_t, h_t, and c_t are input, hidden state, and cell state, at time *t* respectively. A general representation of the biLSTM architecture is shown in Figure 2.10.

A conditional random field (CRF) model makes a prediction as a graphical model to account for the influence of nearby data. A linear chain CRF can make predictions utilizing this improved context after receiving the biLSTM's output. The combination of these two architectures gives biLSTM+CRF [55].

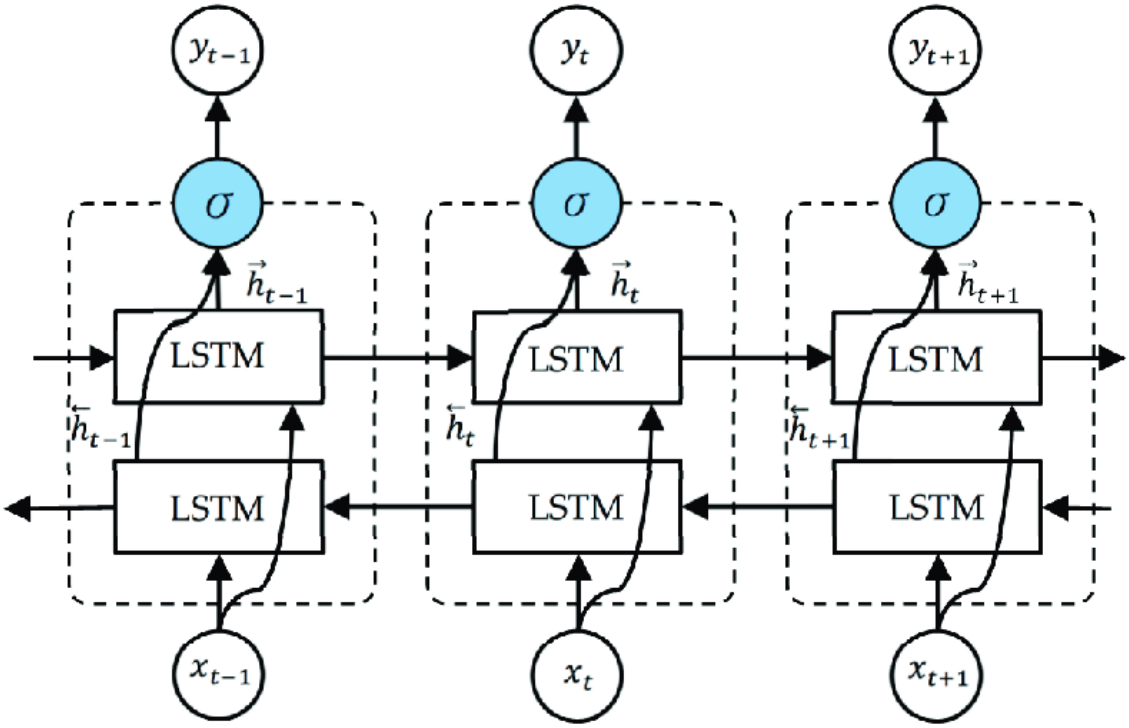

Figure 2.10: A General Architecture for biLSTM algorithm (Adapted from [56]**).**

2.2.3. Attention Mechanism

In the traditional sequence to sequence (Seq2Seq) models, where encoder-decoder architecture is used, all the intermediate states of the encoder are discarded using only its final states (vector) to initialize the decoder. This encoder-decoder architecture performs well in short sentences and poorly when the length of the sentence increases. This is because intermediate states are discarded.

The attention mechanisms are the building blocks to solving the task of neural machine translation (NMT). The encoder-decoder architecture takes a sequence of words as inputs. The encoder encodes the sequence into a fixed-sized single vector, which forms the representations of the sequences technically known as embeddings.

To decode a vector in the encoder layer, the model pay *attention* to the previous state and its intermediate state. Hence the name attention. The decoder here has a selective way of looking back and paying attention to the more relevant words in embeddings. This is mostly done by Backpropagation.

Many language models use the transformer architecture. The Transformer architecture is an improvement of the encoder-decoder with attention mechanisms. Transformer model has

encoder-decoder. Its encoding and decoding components has 6 or more encoders and decoders stack on top each other respectively as seen in Figure 2.11.

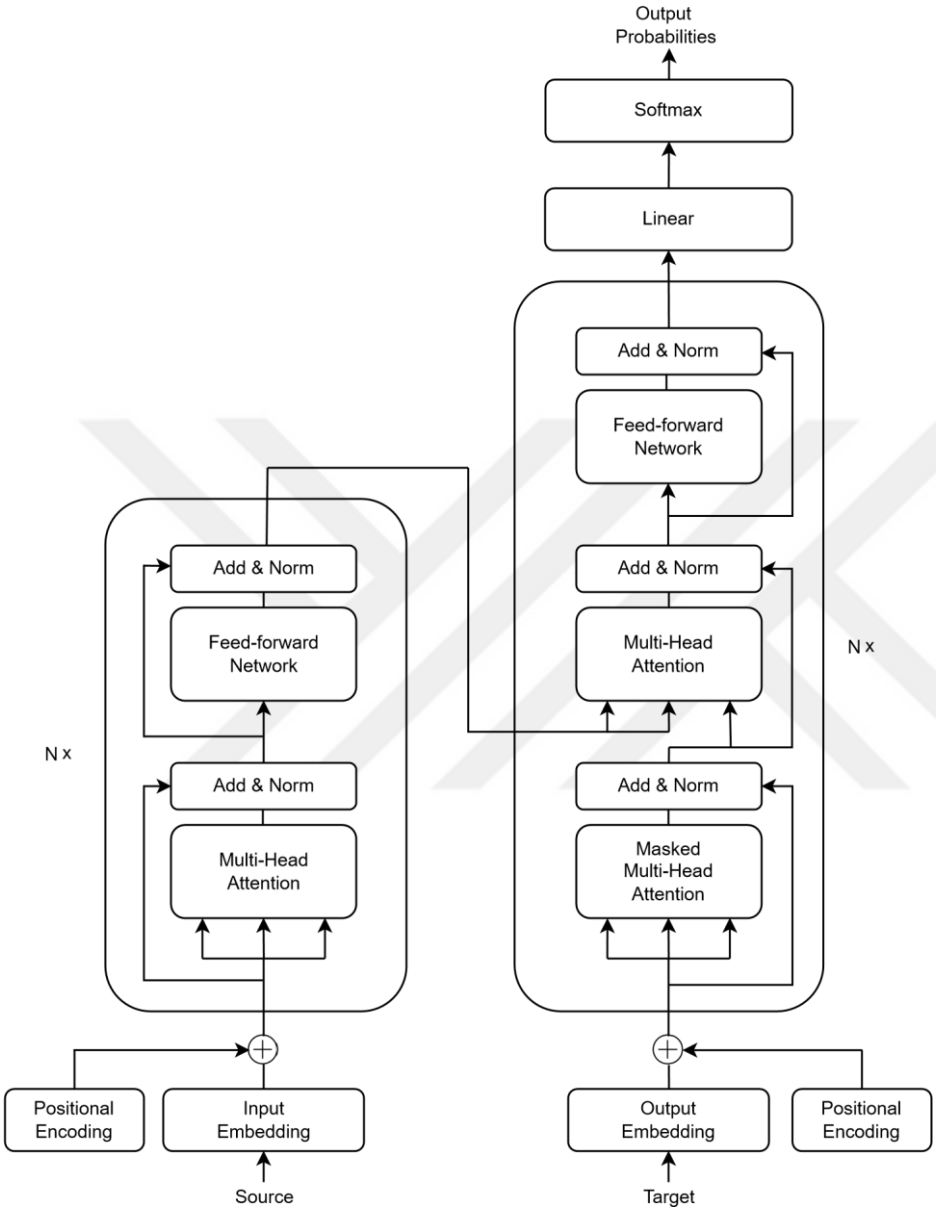

Figure 2.11: Architecture of a Transformer (Taken from).

2.2.4. Transformer and Generative Language models for data augmentation

BERT [36], unlike other models that do not take into consideration the context of words, is a contextualized word representation language model. It is based on a masked language model and pre-trained using a bidirectional transformer. Bidirectional Encoder Representations from

Transformers for Biomedical Text Mining (BioBERT) is a great medical model that is based on BERT, pre-trained on large biomedical corpora. Mostly general language models perform less on medical data due to the fact that there are terms, disease names and domain-specific used words that are only understood by experts in the medical domain. To make language models like BERT perform well on biomedical data, Lee et al. [43] pre-trained their model on PubMed (4.5 billion words), PMC Full-text articles (13.5 billion words), PMC Full-text articles (2.5 billion words), and BooksCorpus (2.5 billion number of words) and named it as BioBERT as can be seen in Figure 2.12.

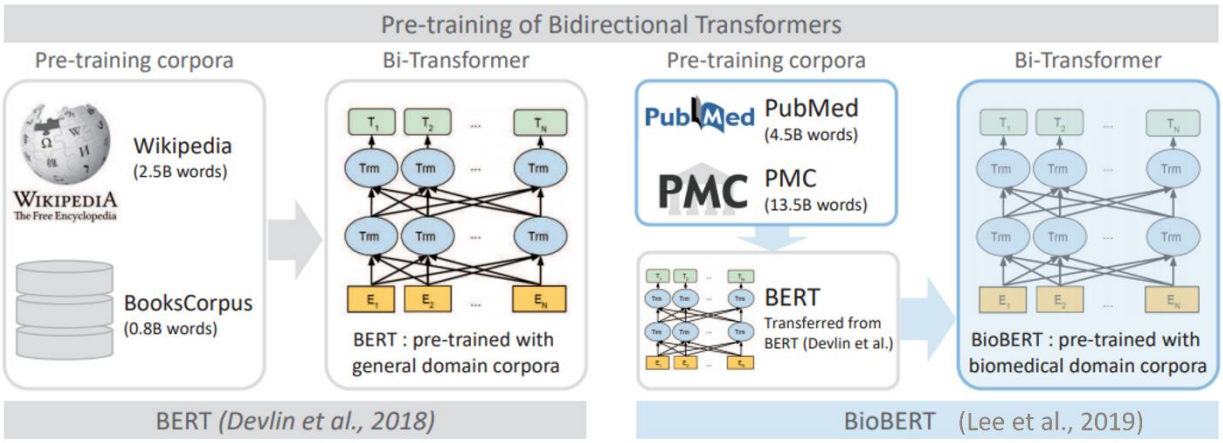

Figure 2.12: BioBERT a contextualized language representation model, based on BERT (Adapted from[43]**).**

In pre-training, they first initialize with BERT and for tokenization, they go for WordPiece tokenization [57]. WordPiece algorithm helps prevent the "out of vocabulary" and infinite vocabulary problem. Instead of the frequency of words, WordPiece uses the combination of pairs of sub words regarding the probability of the language model at hand. The WordPiece algorithm is shown in Figure 2.13.

Input: strings S, vocabulary size k

procedure Wordpiece(S, k)

V ← unique characters in S

while |V| < k **do** (Merge tokens)

a_1, a_2 ← pair that causes largest increase in likelihood

$a_{new} \leftarrow a_1 \oplus a_2$ (new token)

V ← V ∪ {a_{new}}

Replace a_1, a_2 with a_{new}

end while

return V

end procedure

Figure 2.13: Algorithm for WordPiece.

With minimal architectural modification, BioBERT is fine-tuned on downstream NLP tasks including, Question-Answering, Embeddings and NER tasks. In our work, we also used this pre-trained model on biomedical benchmark datasets.

2.3. Information Extraction in medical and clinical data

The importance of information extraction is manifested in the research currently going on, especially in the medical domain. Gathering structured data from medical records, analysis of it, and information extraction enables the automation of tasks, as in the smart content classification of diseases, integrated research on future infections, management, and delivery. Data-driven activities like mining patterns and trends in patient medical history are just but a few to mention. Many Natural Language Processing systems, approaches, models, and research techniques have been developed to extract vital information from medical records. These include the use of ontology-based resources, concept mapping, grammar structure matching [58], semantic parsing [59] approaches, and rule-based [60] and machine learning systems [49]. The rule-based approaches used to extract information in medical text data are less robust since for every new corpus, the rules must be revamped to preserve the best performance of the

model; this requirement increases the maintenance cost accordingly [50]. A smoking status detection system for patients was developed by Savova et al. [50] and later embedded into the clinical Text Analysis and Knowledge Extraction System (cTAKES).

Due to the lucrative nature and vitality of NLP in medicine, we see many medical institutions establishing research centers for deep learning and data science to improve the work of NLP in the biomedical field. The i2b2, tranSMART Foundation, etc. are such institutions that organizes "The Shared Tasks for Challenges in NLP for Clinical Data" regularly.

3. APPROACH

In this study, we followed a straightforward approach in developing data augmentation methods by modifying some of the available methods and adapting them to name entity recognition tasks. We also developed models fine-tuned from pretrained biomedical language model BioBERT. This model is used in our experiment to justify the correctness of our augmented data. We train the model with the benchmark biomedical datasets each, records the results and then save the model. Secondly, we perform text augmentation on the training datasets. These augmented training data is used in training the previous model architectures.

Most NER models use the BIO / IOB tag format (B: Beginning of entity mentioned, I: Inside and part of the entity mentioned except the first, O: Other words with no entity mentioned) which makes data augmentation in NER more tedious. We adapted EDA methods to NER by transforming word sequence and the corresponding BIO tag sequences properly so that the consistency of both sequences is maintained before and after augmentation. Each of the following augmentation methods is applied to each given sentence in the training data N times. Stop words are not included.

- **Random Deletion**: Randomly remove each word in the sentence with a probability p if and only if it is not the beginning of the entity (**B**) tag. If the word is inside an entity (**I**), delete it and its corresponding I tag, else delete an O tag. The pseudocode for this algorithm is shown in Figure 3.1

Algorithm 1 random deletion

Input: (W, L, p)
Output: (W', L')
  if *number of words* $== 1$ **then** Do not delete only one word sentence
    return W, L
  end if
  $W' \leftarrow$ *empty list*
  $L' \leftarrow$ *empty list*
  for each *index*, *word* in W **do**
    $r \leftarrow$random.uniform(0, 1)
   if *word's label* **start with** $B - tag$ **then** don't delete the words
      append *word* **to** W'
      append *label* **to** L'
      continue
   end if
   if r **greater than** p **then**
      append *word* **to** W'
      append *label* **to** L'
   end if
end for
if *all words are deleted* **then**
   return *random word* **and** *label*
end if
return W', L'

Figure 3.1: Pseudocode for Random Swap technique of text augmentation.

- **Random Insertion**: select a random word, find its random synonym, and insert that synonym in the sentence. If the insertion position is at the beginning of an entity (**B**), skip it, else if it is at the position inside an entity **I** replace it, with a corresponding *I* tag. Otherwise, replace the synonym with a corresponding ***O*** tag. The pseudocode of the algorithm is shown in Figure 3.2

Algorithm 2 random insertion

Input: (W, L, n)
Output: (W', L')
  $W' \leftarrow$ copy of W
  $L' \leftarrow$ copy of L
  for each i in range n **do**
    $synonyms \leftarrow empty\ list$
    $counter \leftarrow 0$
    while $len(synonyms) < 1$ **do**
      $random\_word \leftarrow W'[random]$
      $synonyms \leftarrow get\_synonmys(random\_word)$
      $counter \leftarrow counter + 1$
      if $counter$ **greater than** 10 **then**
        return W'
      end if
    end while
    $random\_synonym \leftarrow synonym[0]$
    $random\_idx \leftarrow random\ int$
    insert $random\_synonym$ **to** W'
    if $random\_idx$ **equals** $len(L') - 1$ **then**
      insert $'O'$ **to** L'
    else if $L'[random_idx + 1]$ **equals** $'O'$ **then**
      insert $'O'$ **to** L'
    else
      insert $L'[random\_idx - 1]$ **to** L'
    end if
  end for
  return W', L'

Figure 3.2: Pseudocode for Random Insertion technique of text augmentation.

- **Random Swap**: Randomly select two tokens and swap their positions. the indices of the two selected words must always be different. Tag sequence and word count are maintained as it is. The pseudocode of the algorithm is shown in Figure 3.3

Algorithm 3 random swap

Input: (W, L, n)
Output: (W', L')
  $W' \leftarrow$ copy of W
  $L' \leftarrow$ copy of L
  for each i in range n **do**
    Select $random\_index1$ and $random\_index2$
    while $random\_index1$!= $random\_index2$ **do**
      swap $element\ at\ random\_index1$ **and** $element\ at\ random\_index2$
    end while
  end for
  return W', L'

Figure 3.3: Pseudocode for Random Swap technique of text augmentation.

- **Synonym Replacement**: randomly select n number of words and replace them with their corresponding synonym from WordNet. Tag sequence and word count are maintained as it is. The number of words to be augmented, n is a product of the hyper parameter α and the total length of the sentence at hand l as seen in equation (8). The pseudocode of the algorithm is shown in Figure 3.4

$$n = \alpha * l \tag{8}$$

Algorithm 4 Synonym replacement

Input: (W, L, n)
Output: (W', L')
  $W' \leftarrow$ copy of W
  $L' \leftarrow$ copy of L
  $num\_replaced \leftarrow 0$
  for each w in W **do**
    $synonyms \leftarrow SYNONYM(w)$: get the synonyms of w from WordNet
    if $synonyms \geq 1$ **then**
      $synonym \leftarrow$ $SYNONYM$: randomly select one from synonyms list
      if $len(synonym) \geq 1$ **or** $synonym \in w$ **or** $w \in synonym$ **then**
        continue
      end if
      Append synonym to new word list
      $W' \leftarrow$ synonym
      $num\_replaced \leftarrow num\_replaced + 1$
      if $num\_replaced \geq n$ **then**
        break Only replace up to n words
      end if
    end if
    Return none if new words and old words are same
    return W', L'

Figure 3.4: Pseudocode for Random Swap technique of text augmentation.

To illustrate more of the methods used, **Error! Reference source not found.** shows an example o f how a given sentence and its tag sequences are affected. SR, RI, RS and RD in the figure represents the methods of augmentation, where SR represents Synonym Replacement, RI: Random Insertion, RS: Random Swapping, and RD: Random Deletion.

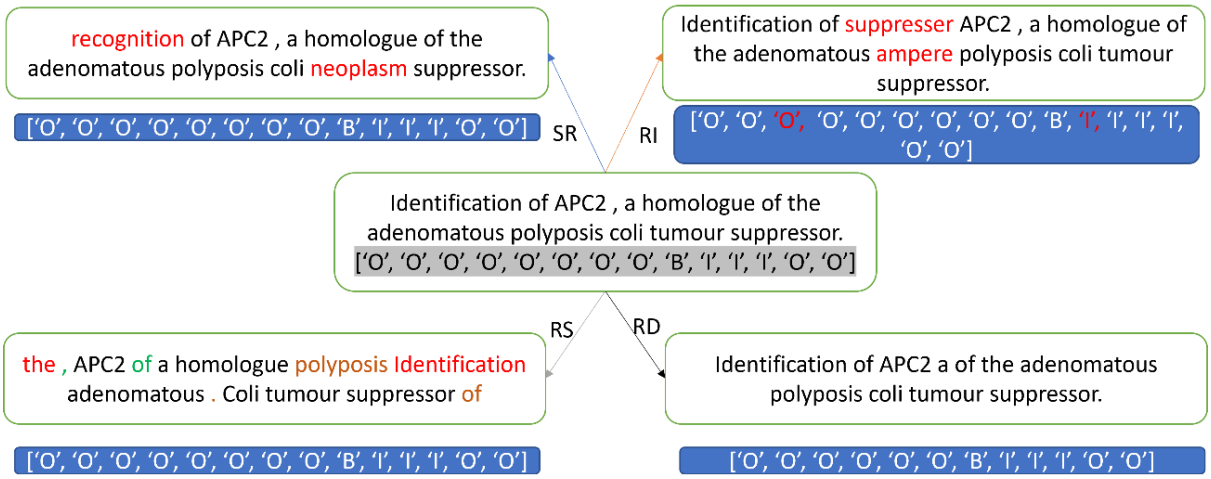

Figure 3.5: Overview of the effects of augmentation methods on sample data. The words marked in red indicate the changes from data augmentation.

We try various combinations of the number of augmented data and the number of epochs of training. Our experiments show that small datasets when augmented yields better results than relatively large datasets.

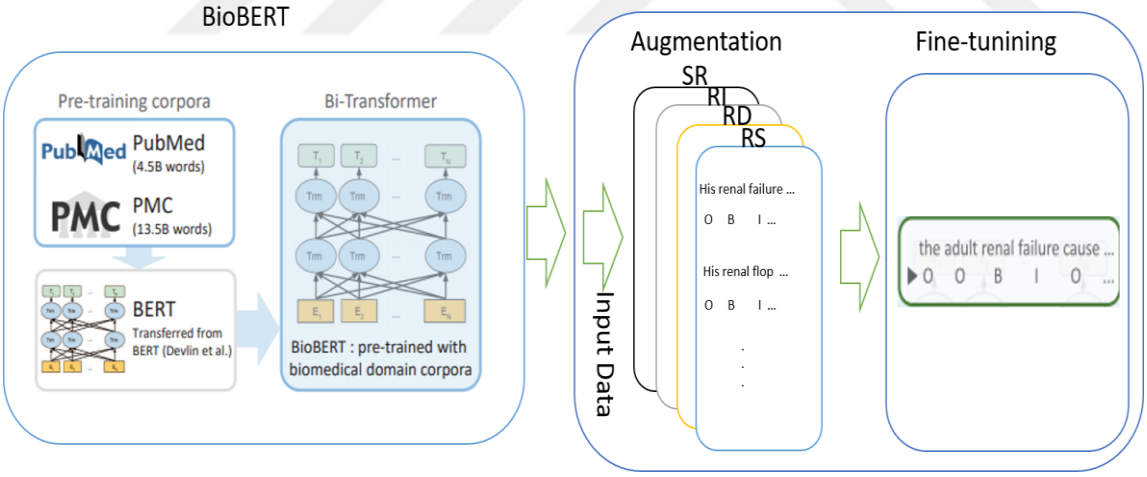

Figure 3.1: General approach used: pre-trained model with data augmentation and fine-tuning for NER (Adapted from [43]**).**

4. EXPERIMENTS

4.1 Datasets

To evaluate our work, we used the standard benchmark datasets usually used to evaluate NER tasks in the medical domain. These includes:

i. NCBI-disease corpus [44]
ii. BC5CDR (BioCreative V CDR corpus) [61]
iii. BioCreative II Gene Mention Recognition (BC2GM) [62]
iv. JNLPBA, Linnaeus [63]
v. BC4CHEMD [64]
vi. Species-800 [65] datasets.

i. NCBI-disease

NCBI-disease is a disease corpus in which the annotation is disease mention and/or concepts. This was introduced for disease name recognition and normalization. NCBI disease corpus contains 793 fully annotated PubMed citations constituting more than six thousand (6K) sentences. More than half of the sentences available contain disease names and about 2136 unique disease mentions, mapping to 790 unique database identifiers. Annotations of diseases mentioned are in four major categories:

1. Specific Disease: clear cell renal cell carcinoma
2. Disease Class: cystic kidney diseases
3. Composite mentions: prostatic, pancreas, skin, and lung cancer
4. Modifier: hereditary breast cancer families.

The most common disease mentions and disease concepts in the NCBI disease corpus, with the corresponding number of abstracts they appear in [44] are presented in **Error! Reference s ource not found.**. An example of text and the annotation is shown in **Error! Reference source not found.**, a screenshot of the annotation tool presented by Islamaj et al. [44].

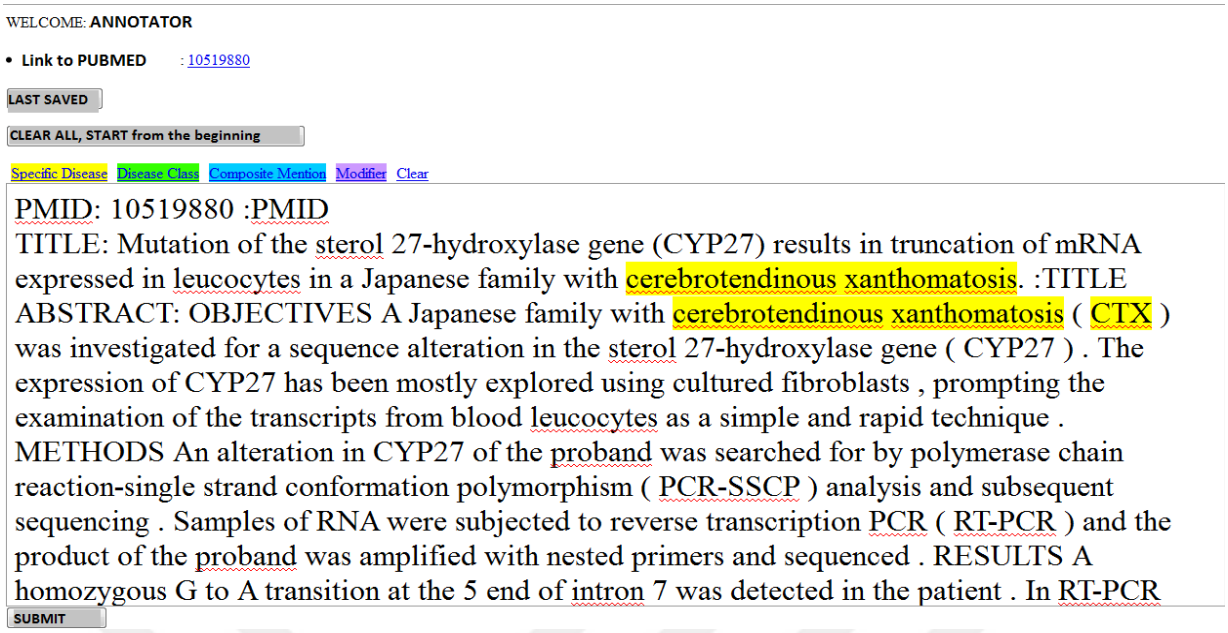

Figure 4.1: Sample text and annotation process of NCBI-disease corpus (Taken from [44]**).**

Table 4.1: The NCBI disease corpus' common disease mentions, their concepts, and the number of abstracts in which they appear (Taken from [44]**).**

Disease mentions	Number of Abstracts	Disease concepts	Number of Abstracts
Cancer	44	**D030342** – Genetic Diseases, Inborn	113
Tumor	43	**D009369** – Neoplasms	112
Breast cancer	41	**D061325** – Hereditary Breast and Ovarian Cancer Syndrome	52
Diabetes mellitus	39	**D001943** – Breast Neoplasms	46
Myotonic dystrophy	35	**D009223** – Myotonic Dystrophy	42
G6PD deficiency	33	**D005955** – Glucosephosphate Dehydrogenase Deficiency	36
Duchenne Muscular Dystrophy	33	**D020388** – Muscular Dystrophy, Duchenne	33
Ataxia-telangiectasia	30	**D011125** – Adenomatous Polyposis Coli	33
APC	29	**D001260** – Ataxia Telangiectasia	31
Duchene muscular dystrophy	27	**D010051** – Ovarian Neoplasms	27

ii. Species-800

Species-800 is a species entities corpus comprising eight hundred PubMed abstracts manually annotated. One hundred abstracts each are collected from Bacteriology, Botany, Entomology, Medicine, Mycology, Protistology, Virology, and Zoology to form the corpus. SPECIES and ORGANISMS being standalone command-line software applications can identify and map the taxonomic mentions in plain text. These are the two taggers used in tagging the s800 corpus. In their work, Evangelos et al. [65] provide a supplementary table for the journal selections for the s800 categories as shown in Table 4.2 For each category, they selected between one and four journals, from which they randomly picked one hundred Medline abstracts in total from the years 2011 and 2012.

Table 4.2: Journal selection for the S800 categories that formed the S800 dataset (Taken from [65]**).**

Category	Journal	Abstracts
Bacteriology	Journal of Bacteriology	100
Botany	New Phytologist	50
	Plant Cell & Environment	50
Entomology	Insect Molecular Biology	59
	Journal Insect Science	24
	Environmental Entomology	17
Medicine	The Lancet	41
	The Lancet Infectious Diseases	15
	The Lancet Neurology	22
	The Lancet Oncology	22
Mycology	Fungal Genetics and Biology	100
Protistology	European Journal of Protistology	30
	International Journal of Systematic and Evolutionary Microbiology	30
	Protist	40
Virology	Journal of Virology	63
	Journal of Virological Methods	37
Zoology	Journal of Animal Ecology	100

iii. BC2GM

BC2GM is a gene/protein corpus in which the annotation is Gene. This dataset was originally provided for gene mention recognition. It has more than 20,703+ named entities and about 20,000 sentences in the total number of documents.

iv. BC5CDR

BC5CDR corpus consists of 1500 PubMed articles with 4409 annotated chemicals, 5818 diseases, and 3116 chemical-disease interactions. Medical Subject Headings (MeSH) concept identifiers are used as controlled vocabulary to annotate chemical and disease entities in all the selected articles of PubMed. The annotation is done manually with the assistance of tools like PubTor, DNorm, and tmChem. An example of PubTor format annotation (PMID 354896) is shown in **Error! Reference source not found.**.

```
354896|t|Lidocaine-induced cardiac asystole.
354896|a|Intravenous administration of a single 50-mg bolus of lidocaine in a 67-year-old
man resulted in profound depression of the activity of the sinoatrial and atrioventricular
nodal pacemakers. The patient had no apparent associated conditions which might have
predisposed him to the development of bradyarrhythmias; and, thus, this probably
represented a true idiosyncrasy to lidocaine.
354896  0   9   Lidocaine   Chemical    D008012
354896  18  34  cardiac asystole    Disease D006323
354896  90  99  lidocaine   Chemical    D008012
354896  142 152 depression  Disease D003866
354896  331 347 bradyarrhythmias    Disease D001919
354896  409 418 lidocaine   Chemical    D008012
354896  CID D008012 D006323
```

Figure 4.2: PubTator format annotation (Adapted from [61]**).**

v. JNLPBA

JNLPBA consists of both RNA and DNA, Gene/Protein, Cell line, and Cell Type. Gene/Protein is annotated as Protein and the rest are as their named-entity recognition. The corpus is created with a controlled search of 2,000 abstracts on MEDLINE.

vi. Linnaeus

Linnaeus is software for general-purpose dictionary matching, whose output forms one of the species corpora with species annotation. The original project was created for entity mention recognition. This dataset consists of documents from MEDLINE, PMC, BMC, OTMI, etc.

vii. BC4CHEM

BC4CHEM is a chemical corpus in which the annotation is Chemical. Originally provided for chemical mention recognition task. It consists of 10,000 PubMed abstracts with 84,355 annotations, though we used part of the original dataset which contains 79,842 chemical annotations.

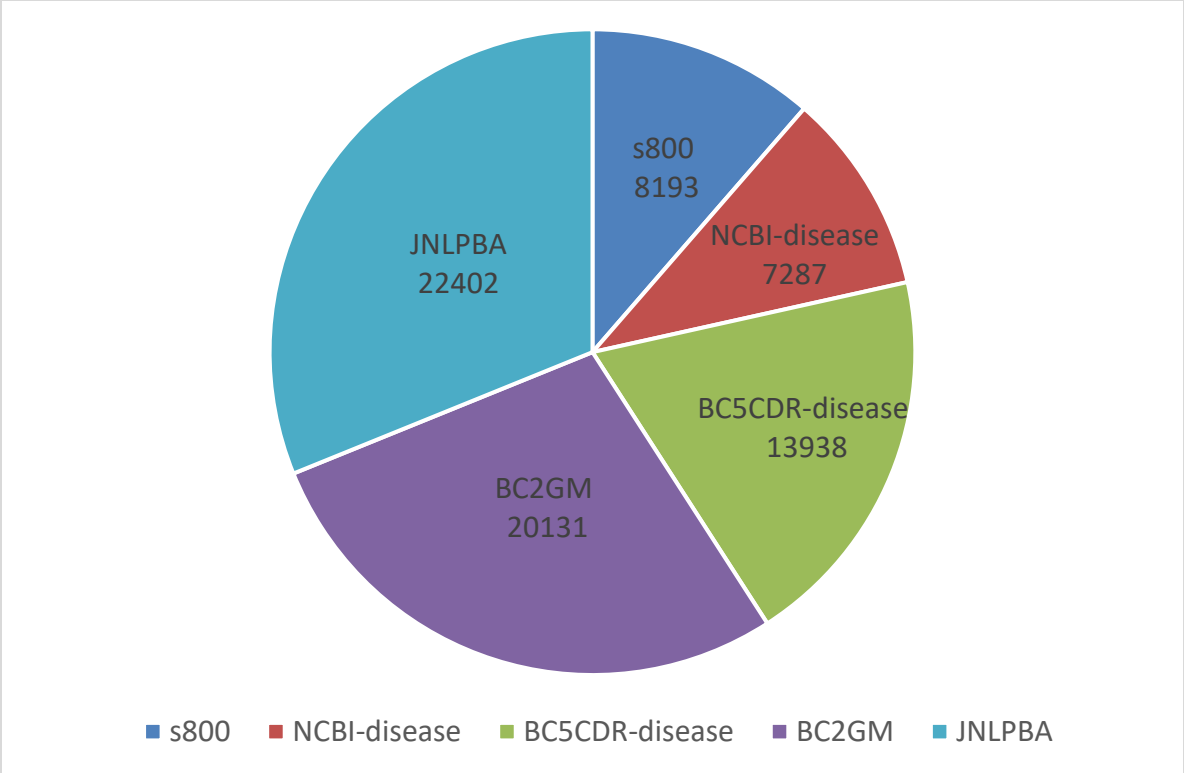

Figure 4.3: Instance distributions in some of the benchmark datasets.

Fully annotated biomedical data sets at the mention and concept level are used. All data sets are used in their pre-processed version from Lee et al. [43]. We fine-tune BioBERT on these datasets with and without their augmented data. **Error! Reference source not found.** shows t he statistics of the biomedical benchmark datasets used in our work. The number of sentences in each of the datasets that we used is also shown in **Error! Reference source not found.**. We u se these datasets for both data augmentation and fine-tuning BioBERT in our work.

Table 4.3: Overview of the biomedical benchmark datasets used in our work. "# Annotations" shows the number of Named Entities.

Corpus	**Named-Entities**	**Number of Annotations/ Named entities**	**Number of Documents**
NCBI Disease	Disease	6,881	793 PubMed abstracts
Species-800	Species	3,708	800 PubMed abstracts
BC2GM	Gene/Protein	20,703	20,000 sentences
BC5CDR-disease	Disease	12,694	1,500 articles
BC5CDR-chem	Drug/Chem.	15,411	
JNLPBA	Gene/Protein, DNA, Cell-type, Cell-line, RNA	35,460	2,000 Medline abstracts
linnaeus	Species	4,077	MEDLINE, PMC
BC4CHEM	Chemical	79,842	10,000 abstracts

4.2 Experimental Setup

Our work presents simple data augmentation methods for named-entity recognition tasks in medical data mining. We adapt the four straightforward, but robust methods of text augmentation methods used to generate diverse and high-quality data to train and enhance the performance of medical domain models for name entity recognition tasks. UMLS-EDA adapted EDA to suit NER tasks by adding UMLS, with the notion that its satisfactory performance in text classification can also be realized in NER. We provide a low-cost and straightforward way of utilizing a small amount of domain-specific data to enhance models' performance with augmentation and transfer learning.

Transfer learning so far has proven to be one of the best ways to improve the performance of deep learning and neural network models at a minimal amount of cost. The idea of using pre-trained models eradicates the cost involved in training a language model from scratch and saves time and energy in looking for data to feed huge neural network models. With transfer learning one needs to only modify the last layers of of a pretrained model to suit specific task without training the whole models again. This prompted the idea of fine-tuning the pre-trained model Bidirectional Encoder Representations (BERT) [36] in the medical domain. Bidirectional Encoder Representations from Transformers for Biomedical Text Mining (BioBERT) being the first domain-specific BERT-based model is an example of transfer learning in the medical domain. This model improves and gain the state-of-the-art model with 0.62% F1 score improvement in biomedical named-entity recognition [43]. With this notion mind, and the fact that data augmentation improves the performance of NLP models, we combined the two in our approach to boost the performance of biomedical models.

Augmentation operations in computer vision inspired the methods used now in NLP. EDA [12] proposed universal data augmentation methods for NLP. They used four methods to perform augmentation on a randomly selected token in a sentence. Their approach was for text classification, therefore needs amendment to suit named-entity recognition tasks for token-level prediction.

In general, the standard evaluation metrics used in supervised machine learning models are Precision, Recall, F1-score, and Accuracy. In our work, we opt to use only Precision, Recall and F1-score evaluation metrics to evaluate the performance of our models at a token level.

Precision shows the percentage of true labels among all the labels. That is how precise or accurate the model is. This metric shows how many of the predicted entities are correctly labeled. As can be seen in equation 8, it is the ratio of true positives to all other identified positives. A named entity is correct only if it is an exact match of the corresponding entity in the test set.

$$Precision = \frac{\#True_{Positive}}{\#True_{Positive} + \#False_{Positive}} \quad (8)$$

Recall measures the percentage of true labels in the dataset being recalled. As seen in equation 9, it is the ratio of true positives to the sum of true positives and false negatives.

$$Recall = \frac{\#True_{Positive}}{\#True_{Positive} + \#False_{Negatives}} \quad (9)$$

F1-Score is the harmonic mean of precision and recall. For the evaluation metrics in our work, we use entity-level precision, recall, and F1 score. As seen in equation 10, it is the ratio of the product of precision and recalls to their summation all multiplied by two.

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (10)$$

Table 4.4: Hyperparameters for BioBERT fine-tuning.

Hyper-parameter	Value
mini-batch size	32
epochs	30 & 100
max. sequence length	192
weight decay rate	0.0 - 0.01
optimizer	Adam

The language models that we use in our experiments are

1. BioBERT version 1.1 (BioBERT-Base v1.1 (+ PubMed 1M)). This version of BioBERT is pre-trained on BERT-base-Cased with a custom vocabulary size of 30k.
2. BERT (bert-based-uncased). A general language model pre-trained on Wikipedia and Book Corpus. This version of BERT has 12 encoders stuck on each other and 12 bidirectional self-attentions. We used the "uncased" version of BERT since it is case insensitive.

In the fine-tuning, we use only 1 NVIDIA GeForce RTX 2080 Ti. The hyper-parameters we use are shown in **Error! Reference source not found.**.

5. RESULTS AND DISCUSSION

In this section we present the general output from all our experiments. We shall interpret the findings and discuss the results in detail.

We show how data augmentation can be useful in the uplifting and enhancement of medical data and models. We present the results in groups according to the entities mentioned in the datasets. These are Diseases datasets, Chemicals datasets, Species datasets and Protein/gene datasets

The training of the models is done in 30 number of epochs as well as 100 number of epochs. Results for both BERT and BioBERT models trained for 30 epochs are presented in Table 5.1, Table 5.2, and Table 5.3. As for training for 100 number of epochs, it done for only BioBERT models and does not include BERT.

Table 5.1: Results of BioBERT and BERT on disease datasets. Number of epochs = 30.

			BioBERT				**BERT**			
Dataset	**Number of Entities**	**Metric**	**Original Data**	**n = 2**	**n = 10**	**n = 16**	**Original Data**	**n = 2**	**n = 10**	**n = 16**
NCBI disease	6,881	P	**86.36**	86.35	86.35	86.35	**83.96**	83.65	83.65	83.65
		R	**89.06**	88.96	88.96	88.95	86.67	**86.88**	**86.88**	**86.88**
		F	**87.69**	87.63	87.63	87.63	**85.29**	85.23	85.23	85.23
BC5CDR-disease	12,694	P	84.49	**93.05**	**93.05**	84.56	82.81	**92.20**	**92.20**	82.38
		R	87.57	**93.52**	**93.52**	87.77	83.72	**92.22**	**92.22**	84.02
		F	86.00	**93.26**	**93.26**	86.13	83.26	**92.21**	**92.21**	83.19

The result in the various tables shows best score within a model as bold font, the overall best score of the two models is bold and underlined. For example, in Table 5.1, the F1-score of

BioBERT model trained on 2-fold and 10-folds of BC5CDR-disease augmented data outperforms the rest. Therefore, its written in bold and underlined. Likewise, an F1-score of 92.0% of same datasets is the best score in BERT model, so recorded in bold but not underlined. All the results are recorded in percentage of 2 decimal point.

Table 5.2: Results of BioBERT and BERT on chemical datasets. Number of epochs = 30.

			BioBERT				**BERT**			
Dataset	**Number of Entities**	**Metric**	**Original Data**	**n = 2**	**n = 10**	**n = 16**	**Original Data**	**n = 2**	**n = 10**	**n = 16**
BC5CDR-chem	15,411	P	92.79	**92.83**	**92.83**	**92.83**	92.20	92.20	92.20	92.20
		R	**94.21**	93.35	93.35	93.35	92.23	92.23	92.23	92.23
		F	**93.49**	93.09	93.09	93.09	92.21	92.21	92.21	92.21
BC4CHEMD	79,842	P	91.68	91.68	91.68	91.68	91.07	91.07	91.07	91.07
		R	90.53	90.53	90.53	90.53	88.33	88.33	88.33	88.33
		F	91.10	91.10	91.10	91.10	89.68	89.68	89.68	89.68

Table 5.3: Results of BioBERT and BERT on Species datasets. Number of epochs = 30.

			BioBERT				**BERT**			
Dataset	**Number of Entities**	**Metric**	**Original Data**	**n = 2**	**n = 10**	**n = 16**	**Original Data**	**n = 2**	**n = 10**	**n = 16**
Species-800	3,708	P	70.29	70.29	70.29	70.29	69.21	69.21	69.21	69.21
		R	75.88	75.88	75.88	75.88	74.45	74.45	74.45	74.45
		F	**72.98**	72.98	72.98	72.98	72.73	72.73	72.73	72.73
linnaeus	4,077	P	89.53	89.53	89.53	89.53	87.47	87.47	87.47	87.47
		R	83.53	83.53	83.53	83.53	85.28	85.28	85.28	85.28
		F	86.43	86.43	86.43	86.43	86.36	86.36	86.36	86.36

Table 5.4: Results of BioBERT and BERT on protein/gene datasets. Number of epochs = 30.

			BioBERT				BERT			
Dataset	Number of Entities	Metric	Original Data	n = 2	n = 10	n = 16	Original Data	n = 2	n = 10	n = 16
BC2GM	20,703	P	82.91	82.91	82.91	**82.95**	80.65	80.65	80.65	80.65
		R	83.56	83.56	83.56	**83.83**	82.06	82.06	82.06	82.06
		F	83.23	83.23	83.23	**83.38**	81.35	81.35	81.35	81.35
JNLPBA	35,460	P	**71.19**	**71.19**	**71.19**	**71.19**	70.27	70.27	70.27	70.27
		R	82.54	82.54	82.54	82.54	82.40	82.40	82.40	82.40
		F	76.45	76.45	76.45	76.45	75.85	75.85	75.85	75.85

Table 5.5: Results of BioBERT on disease datasets. Number of epochs = 100.

Dataset	Number of EPOCH = 100					
	Number of Entities	Metric	Original Data	n = 2	n = 10	n = 16
NCBI disease	6,881	Precision (P)	85.21	86.48	86.48	86. 48
		Recall (R)	88.23	88.64	88.65	88.65
		F1-Score (F1)	86.69	**87.55**	**87.55**	**87.55**
BC5CDR-disease	12,694	Precision (P)	85.26	**92.74**	**92.74**	84.89
		Recall (R)	86.44	**93.76**	**93.76**	85.83
		F1-Score (F1)	85.85	**93.25**	**93.25**	85.35

Table 5.6: Results of BioBERT model on chemical datasets. Number of epochs = 100.

Dataset	Number of EPOCH = 100					
	Number of Entities	**Metric**	**Original Data**	**n = 2**	**n = 10**	**n = 16**
BC5CDR-chem	15,411	Precision (P)	84.86	**93.05**	**93.05**	84.57
		Recall (R)	87.04	**93.12**	**93.12**	86.75
		F1-Score (F1)	85.94	**93.09**	**93.09**	85.65
BC4CHEMD	79,842	Precision (P)	91.72	91.72	91.72	91.72
		Recall (R)	90.17	90.17	90.17	90.17
		F1-Score (F1)	90.94	90.94	90.94	90.94

Table 5.7: Results of BioBERT model on species datasets. Number of epochs = 100.

Dataset	Number of EPOCH = 100					
	Number of Entities	**Metric**	**Original Data**	**n = 2**	**n = 10**	**n = 16**
Species-800	3,708	Precision (P)	70.76	70.76	70.76	70.76
		Recall (R)	76.66	76.66	76.66	76.66
		F1-Score (F1)	73.59	73.59	73.59	73.59
linnaeus	4,077	Precision (P)	89.64	89.64	89.64	89.64
		Recall (R)	81.51	81.51	81.51	81.51
		F1-Score (F1)	85.38	85.38	85.38	85.38

Table 5.8: Results of BioBERT with Augmentation EPOCH = 100.

Dataset	Number of EPOCH = 100					
	Number of Entities	Metric	Original Data	n = 2	n = 10	n = 16
BC2GM	20,703	Precision (P)	83.61	83.61	83.61	83.61
		Recall (R)	83.29	83.29	83.29	83.29
		F1-Score (F1)	83.45	83.45	83.45	83.45
JNLPBA	35,460	Precision (P)	71.26	71.26	71.26	71.26
		Recall (R)	83.44	83.44	83.44	83.44
		F1-Score (F1)	76.87	76.87	76.87	76.87

We observe that the number of epochs affects the performance of the BioBERT model trained on augmented data. This is clear when the number of epochs is 30, BioBERT and BERT models trained on the NCBI disease dataset have their best performances in the original training data as seen in Table 5.1. The same dataset trained with 100 epochs produces an F1 score of 86.69% on the original data and an increased F1 score of 87.55% as seen in Table 5.5. When the number of augmentations is n = 2, that is 2 folds of the original data, the F1 score yields results as good as the original model. Increasing the number of n improves the performance of the model as can be seen in Table 5.5, Table 5.6, and Table 5.7. It yields an F1 score of 87.55% better than the original 86.69% before augmentation. Similarly, even after augmentation, the models still get the same precision, recall, and F1 score values as the original using the species-800 dataset. This tells us that with lesser amounts of data instances, augmentation increases the performance of the model vividly and with relatively large data, there is still a minor improvement.

There is observance that BioBERT models generally outperform BERT models on both augmented and original datasets. This could be because BioBERT is finetuned on biomedical datasets and since we are experimenting in the same domain, its performance is higher.

We got 5.95% and 8.49% increment of F1-score of BioBERT and BERT models respectively trained on BC5CDR-disease dataset. Increassing n = 16 reduces the performance of the models as can be seen in Table 5.1, Table 5.5, and Table 5.6.

6. CONCLUSION

In this study, we employ EDA, which is originally proposed for augmenting text classification datasets, and adapt it for augmenting NER datasets. NER, as a token-level sequence-based NLP task, is vastly different from sentence-level or document-level NLP tasks and therefore it is much more difficult to augment using traditional augmentation methods due to the risk of distorting sentences syntactically and semantically. Changing sentences for data augmentation using for instance back-translation may change the location of labeled tokens or may completely remove them. There is also an additional complexity due to the complex terminology and structure of sentences in the medical domain. Our approach when applied to medical NER datasets shows promising results. Our experiments show that the performance is sensitive to the augmentation factor n, which shows how much each labeled instance is augmented, and the epoch of the deep learning algorithms used in NER. A high number of epochs increase the performance of NER models. Increasing the n improves the performance of the model as can be seen in Table I and Table II. NCBI-disease dataset with seven thousand annotations shows clear improvement in the model when augmented with n=16. It yields an F1 score of 87.55% which is higher than the 86.69% F1 acquired without augmentation. Similarly, even after augmentation, the models still get the same precision, recall, and F1 score values as the original using the species-800 dataset. This tells us that with lesser amounts of data instances, augmentation increases the performance of the model vividly and with relatively large data, there is still a minor improvement. Overall, our results show that EDA can successfully be adapted for NER in the medical domain.

The contributions are:

We show that without using complicated and computationally expensive approaches, a simple data augmentation method can boost the performance of biomedical text mining models.

We show that with little adjustments, the methods of augmentation used in text classification can be used in biomedical named-entity recognition.

We show that data augmentation in addition to transfer learning is a suitable combination for high-performance biomedical named-entity recognition models.

We present the results in groups according to the entity mentioned in the datasets. That is, we group our results according to disease datasets, chemical datasets, speciese, genetics and protein.

7. FUTURE WORK

Some of the datasets we used could not get any improvement even after data augmentation. This could be due to the increment of noise, which leads to the poorness of datasets and eventually does not make any improvement to the model. In the future, we plan to make data augmentation of the original text by increasing both data and the cleanliness of the augmented data.

The use of synonym replacement as an augmentation method in our work solely depends on the words from WordNet. In the medical field names of diseases, organs, and some other domain-specific terminologies may not necessarily have synonyms in WordNet. Creating a mapping of medical terminologies and their respective similar words is what we plan to do. This way using nearest neighbor search in the embedding layer can replace similar medical terms in the original data to generate a more diverse and accurate dataset.

Embedding space of words displays vivid relationships and comprehensiveness of NLP in general. The use of cosine similarities, nearest neighbors etc to find similarities of words in the embedding layers are used for text augmentation. Using domain specific pretrained word embeddings hypothetically can generate more synthetic and meaningful data for text augmentation. We plan to use biomedical word embeddings like BioWordVec [66] and clinical concept pretrained word embeddings such as cui2vec [67]. This we hope can generate more reliably new data instances than embeddings of general text data.

8. REFERENCES

[1] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. v Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *arXiv preprint arXiv:1906.08237,* 2019.

[2] A. Fabbri, P. Ng, Z. Wang, R. Nallapati, and B. Xiang, "Template-Based Question Generation from Retrieved Sentences for Improved Unsupervised Question Answering," 2020. doi: 10.18653/v1/2020.acl-main.413.

[3] I. Yamada, A. Asai, H. Shindo, H. Takeda, and Y. Matsumoto, "LUKE: deep contextualized entity representations with entity-aware self-attention," *arXiv preprint arXiv:2010.01057,* 2020.

[4] B. Dhingra, H. Liu, Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Gated-attention readers for text comprehension," *arXiv preprint arXiv:1606.01549,* 2016.

[5] R. Nallapati, B. Zhou, C. dos Santos, Ç. Gulçehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," 2016. doi: 10.18653/v1/k16-1028.

[6] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. v Le, "Autoaugment: Learning augmentation policies from data," *arXiv preprint arXiv:1805.09501,* 2018.

[7] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proceedings of the AAAI Conference on Artificial Intelligence,* 2020, vol. 34, no. 07, pp. 13001–13008.

[8] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: fast and flexible image augmentations," *Information,* vol. 11, no. 2, p. 125, 2020.

[9] Q. Xie, Z. Dai, E. Hovy, M. T. Luong, and Q. v. Le, "Unsupervised data augmentation for consistency training," in *Advances in Neural Information Processing Systems*, 2020, vol. 2020-December.

[10] A. P. Quimbaya *et al.*, "Named Entity Recognition over Electronic Health Records Through a Combined Dictionary-based Approach," in *Procedia Computer Science*, 2016, vol. 100. doi: 10.1016/j.procs.2016.09.123.

[11] H. Cai, H. Chen, Y. Song, C. Zhang, X. Zhao, and D. Yin, “Data manipulation: Towards effective instance learning for neural dialogue generation via learning to augment and reweight,” *arXiv preprint arXiv:2004.02594*, 2020.

[12] J. Wei and K. Zou, “Eda: Easy data augmentation techniques for boosting performance on text classification tasks,” *arXiv preprint arXiv:1901.11196*, 2019.

[13] S. Kobayashi, “Contextual augmentation: Data augmentation by words with paradigmatic relations,” *arXiv preprint arXiv:1805.06201*, 2018.

[14] S. Mysore *et al.*, “The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures,” *arXiv preprint arXiv:1905.06939*, 2019.

[15] Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall, “2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text,” *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 552–556, 2011.

[16] T. Kang, A. Perotte, Y. Tang, C. Ta, and C. Weng, “UMLS-based data augmentation for natural language processing of clinical research literature,” *Journal of the American Medical Informatics Association*, vol. 28, no. 4, pp. 812–823, 2021.

[17] O. Bodenreider, “The unified medical language system (UMLS): integrating biomedical terminology,” *Nucleic Acids Res*, vol. 32, no. suppl_1, pp. D267–D270, 2004.

[18] M. Bayer, M.-A. Kaufhold, and C. Reuter, “A Survey on Data Augmentation for Text Classification 1.”

[19] S. Y. Feng *et al.*, “A Survey of Data Augmentation Approaches for NLP,” 2021. doi: 10.18653/v1/2021.findings-acl.84.

[20] H. Quteineh, S. Samothrakis, and R. Sutcliffe, “Textual data augmentation for efficient active learning on tiny datasets,” 2020. doi: 10.18653/v1/2020.emnlp-main.600.

[21] Y. Bengio, I. Goodfellow, and A. Courville, “Chapter 7: Regularization,” *Integration The Vlsi Journal*, 2003.

[22] Y. Belinkov and Y. Bisk, “Synthetic and natural noise both break neural machine translation,” 2018.

[23] J. Li, A. Sun, J. Han, and C. Li, “A Survey on Deep Learning for Named Entity Recognition,” *IEEE Transactions on Knowledge and Data Engineering*, 2020, doi: 10.1109/tkde.2020.2981314.

[24] K. W. CHURCH, “Word2Vec,” *Natural Language Engineering*, vol. 23, no. 1, 2017, doi: 10.1017/s1351324916000334.

[25] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global vectors for word representation,” 2014. doi: 10.3115/v1/d14-1162.

[26] T. Mikolov, “Compressing Text Classification Models,” *Iclr*, 2017.

[27] R. Collobert, “Deep learning for efficient discriminative parsing,” in *Journal of Machine Learning Research*, 2011, vol. 15.

[28] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” *Journal of Machine Learning Research*, vol. 12, 2011.

[29] W. Y. Wang and D. Yang, “That’s So Annoying!!!: A Lexical and Frame-Semantic Embedding Based Data Augmentation Approach to Automatic Categorization of Annoying Behaviors using #petpeeve Tweets *,” Association for Computational Linguistics, 2015. [Online]. Available: http://www.cs.cmu.edu/

[30] M. Papadaki, “Data Augmentation Techniques for Legal Text Analytics,” 2017.

[31] P. K. B. Giridhara, C. Mishra, R. K. M. Venkataramana, S. S. Bukhari, and A. Dengel, “A study of various text augmentation techniques for relation classification in free text,” in *ICPRAM 2019 - Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods*, 2019, pp. 360–367. doi: 10.5220/0007311003600367.

[32] C. Sabty, I. Omar, F. Wasfalla, M. Islam, and S. Abdennadher, “Data Augmentation Techniques on Arabic Data for Named Entity Recognition,” in *Procedia CIRP*, 2021, vol. 189. doi: 10.1016/j.procs.2021.05.092.

[33] H. Guo, Y. Mao, and R. Zhang, “Augmenting Data with Mixup for Sentence Classification: An Empirical Study,” May 2019, [Online]. Available: http://arxiv.org/abs/1905.08941

[34] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “MixUp: Beyond empirical risk minimization,” 2018.

[35] A. Jindal, A. Ghosh Chowdhury, A. Didolkar, D. Jin, and R. Ratn Shah, “Augmenting NLP models using Latent Feature Interpolations,” Online.

[36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[37] I. S. Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, “Language Models are Unsupervised Multitask Learners,” *OpenAI Blog*, vol. 1, no. May, 2020.

[38] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities.,” *Proc Natl Acad Sci U S A*, vol. 79, no. 8, 1982, doi: 10.1073/pnas.79.8.2554.

[39] S. Kobayashi, “Contextual augmentation: Data augmentation bywords with paradigmatic relations,” in *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2018, vol. 2. doi: 10.18653/v1/n18-2072.

[40] S. Edunov, M. Ott, M. Auli, and D. Grangier, “Understanding back-translation at scale,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, 2018, pp. 489–500. doi: 10.18653/v1/d18-1045.

[41] R. Sennrich, B. Haddow, and A. Birch, “Improving neural machine translation models with monolingual data,” in *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, 2016, vol. 1. doi: 10.18653/v1/p16-1009.

[42] P. S. Banerjee, B. Chakraborty, D. Tripathi, H. Gupta, and S. S. Kumar, “A Information Retrieval Based on Question and Answering and NER for Unstructured Information Without Using SQL,” *Wireless Personal Communications*, vol. 108, no. 3, 2019, doi: 10.1007/s11277-019-06501-z.

[43] J. Lee *et al.*, “BioBERT: a pre-trained biomedical language representation model for biomedical text mining,” *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.

[44] R. I. Doğan, R. Leaman, and Z. Lu, “NCBI disease corpus: a resource for disease name recognition and concept normalization,” *J Biomed Inform*, vol. 47, pp. 1–10, 2014.

[45] I. Beltagy, K. Lo, and A. Cohan, “SCIBERT: A pretrained language model for scientific text,” 2020. doi: 10.18653/v1/d19-1371.

[46] Y. Luan, L. He, M. Ostendorf, and H. Hajishirzi, “Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction,” 2020. doi: 10.18653/v1/d18-1360.

[47] S. Sekine and C. Nobata, “Definition, dictionaries and tagger for Extended Named Entity Hierarchy,” 2004.

[48] S. Zhang and N. Elhadad, “Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts,” *Journal of Biomedical Informatics*, vol. 46, no. 6, 2013, doi: 10.1016/j.jbi.2013.08.004.

[49] G. Zhou and J. Su, “Named entity recognition using an HMM-based chunk tagger,” in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 2002, pp. 473–480.

[50] G. K. Savova *et al.*, “Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications,” *Journal of the American Medical Informatics Association*, vol. 17, no. 5, pp. 507–513, 2010.

[51] M. Collins and Y. Singer, “Unsupervised Models for Named Entity Classification,” 1999.

[52] J.-H. Kim, I.-H. Kang, and K.-S. Choi, “Unsupervised named entity classification models and their ensembles,” 2002. doi: 10.3115/1072228.1072316.

[53] D. Nadeau, P. D. Turney, and S. Matwin, “Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2006, vol. 4013 LNAI. doi: 10.1007/11766247_23.

[54] R. K. Ando and T. Zhang, “A framework for learning predictive structures from multiple tasks and unlabeled data,” *Journal of Machine Learning Research*, vol. 6, 2005.

[55] L. Luo *et al.*, “An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition,” *Bioinformatics*, vol. 34, no. 8, 2018, doi: 10.1093/bioinformatics/btx761.

[56] Y. H. Li, L. N. Harfiya, K. Purwandari, and Y. der Lin, “Real-time cuffless continuous blood pressure estimation using deep learning model,” *Sensors (Switzerland)*, vol. 20, no. 19, 2020, doi: 10.3390/s20195606.

[57] Y. Wu *et al.*, “Google’s neural machine translation system: Bridging the gap between human and machine translation,” *arXiv preprint arXiv:1609.08144*, 2016.

[58] S. Jonnalagadda, T. Cohen, S. Wu, and G. Gonzalez, “Enhancing clinical concept extraction with distributional semantics,” *J Biomed Inform*, vol. 45, no. 1, pp. 129–140, 2012.

[59] S. Sohn, Z. Ye, H. Liu, C. G. Chute, and I. J. Kullo, “Identifying abdominal aortic aneurysm cases and controls using natural language processing of radiology reports,” *AMIA summits on translational science proceedings*, vol. 2013, p. 249, 2013.

[60] M. Torii, K. Wagholikar, and H. Liu, “Using machine learning for concept extraction on clinical documents from multiple data sources,” *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 580–587, 2011.

[61] J. Li *et al.*, “BioCreative V CDR task corpus: a resource for chemical disease relation extraction,” *Database*, vol. 2016, 2016.

[62] L. Smith *et al.*, “Overview of BioCreative II gene mention recognition,” *Genome Biology*, vol. 9, no. SUPPL. 2. 2008. doi: 10.1186/gb-2008-9-s2-s2.

[63] M. Gerner, G. Nenadic, and C. M. Bergman, “LINNAEUS: A species name identification system for biomedical literature,” *BMC Bioinformatics*, vol. 11, 2010, doi: 10.1186/1471-2105-11-85.

[64] M. Krallinger *et al.*, “Overview of the BioCreative VI chemical-protein interaction Track,” *Proceedings of BioCreative VI workshop*, vol. 450, no. 9, 2017.

[65] E. Pafilis *et al.*, “The SPECIES and ORGANISMS resources for fast and accurate identification of taxonomic names in text,” *PLoS One*, vol. 8, no. 6, p. e65390, 2013.

[66] Y. Zhang, Q. Chen, Z. Yang, H. Lin, and Z. Lu, “BioWordVec, improving biomedical word embeddings with subword information and MeSH,” *Scientific Data*, vol. 6, no. 1, 2019, doi: 10.1038/s41597-019-0055-0.

[67] A. L. Beam *et al.*, “Clinical concept embeddings learned from massive sources of multimodal medical data,” in *Pacific Symposium on Biocomputing*, 2020, vol. 25, no. 2020. doi: 10.1142/9789811215636_0027.