DATA AUGMENTATION FOR NAME ENTITY RECOGNITION

KYAW ZIN TUN

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Master of Engineering

2022

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

26/08/2022	
Date	Kyaw Zin Tun

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

26/08/2022

Date

Assoc Prof. Chng Eng Siong

Authorship Attribution Statement

This thesis contains material from 1 paper accepted at conference in which I am listed as a second author.

Chapter 4 is submitted as Vasantharajan Charangan, Kyaw Zin Tun, Thi Nga Ho, Sparsh Jain, Tong Rong, Chng Eng Siong. MedBERT: A Pre-Trained Language Model for Biomedical Named Entity Recognition. 2022 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC) - APSIPA ASC 2022 (Full Paper Track). Submission ID 1570839765.

The contributions of the authors are as follows:

- A/Prof Chng provided the initial project direction.
- Charagan and I proposed key research idea and co-designed the model architecture.
- Charangan and Nga conducted the experiments as well as prepared the manuscript drafts.
- Dr. Tong Ron and Sparsh Jain contributed to the analysis of the results as well as reviewed the manuscript drafts.

28/08/2022

Date

Kyaw Zin Tun

Acknowledgement

Assoc Prof. Chng Eng Siong has been an ideal teacher, mentor, and thesis supervisor, offering advice and encouragement with a perfect blend of insight and patience. I'm truly proud of, and grateful for, my time working and studying under him.

I am also indebted to several colleagues from SpeechLab@NTU for their support. Especially I would like to thank my colleagues and friends: Ho Thi Nga, Lim Zhi Hao, Vu Thi Ly and Nabilah Binte Mohammed Ismail for covering my works while I was occupied with studies. Special thanks to Priyam Basu, Vasantharajan Charangan, Clarita Chua and Nakhate Anand Theertha for the research collaboration and brainstorming ideas.

My appreciation also goes out to my family and my fiancee for their love and motivation all through my studies. The journey wouldn't have been possible without your continuous and unconditional support.

Abstract

The objective of this thesis is to develop text augmentation approaches for Name Entity Recognition tasks under low-resource domain settings. The field of Name Entity Recognition has advanced rapidly due to the contributions of Deep Learning. Deep Learning techniques have become the mainstream approach for the majority of Natural Language Processing tasks. Neural Network models are able to learn data more efficiently and produce state-of-the-art results compared to traditional approaches. However, one constraint of Deep Learning approach is the need for a large volume of annotated data. The Name Entity Recognition (NER) often faces low-resource issues i.e there are insufficient annotated examples with entities. When NER systems based on Deep Learning techniques are trained with relatively small dataset, such models are unable to learn good representation for the name entities, and hence these models' performance degrades drastically.

Text augmentation is the approach to generate additional artificial text derived from existing data. The idea is to increase the size of training data and ultimately improve the model performance. In this thesis, we aim to explore the domain-independent text augmentation approaches for NER tasks via text generation using two approaches: Finite State Transducer and Abstractive Text Summarization techniques. The objective is to evaluate the effectiveness of Finite State Transducer and Abstractive Text Summarization techniques data augmentation for NER by comparing performance on the original datasets against augmented datasets. Finite state transducer is a template based approach while abstractive text summarization will use Google's Pegasus deep NN model. The proposed approach consists of following steps. Firstly, with the use of OpenGrm Thrax Grammar compiler and OpenFST library, sentences in the original dataset were handcrafted into regular expressions and the name entities values are replaced by a set of variables. Each variable then stores the possible values of the particular entity found in the dataset. By performing word replacements for each entity variable, more sentences can be generated using the template to create a never-seen combination of entity values. Secondly, we use the pre-trained Google's Pegasus summarization tool to transform the original sentence into several semantically similar sentences. Finally, the text generated by these two methods are combined to form the augmented text.

To verify the effectiveness of these two approaches, we will train three state-of-the-art BERT-based NER models, namely, BERT, DistilBERT, RoBERTa systems. Our results on the Groningen Meaning Bank corpus showed that text augmentation fine-tuning improved F1 score of BERT and DistilBERT NER model by 0.3%and 0.7% respectively over the baseline system , while RoBERTa based model performance reduced by 0.2%. Our conclusion is that the performance of our proposed text augmentation is model dependent as it showed improvement on smaller pre-trained models such as BERT and DistilBERT but not on the relatively large RoBERTa model.

Contents

List of Figures

List of Tables

Acronyms

BERT Bidirectional Encoder Representations from Transformers

BI-LSTM Bidirectional Long Short-Term Memory

CapsNets Capsule Network

CNN Convolution Neural Networks

CoNLL Conference on Computational Natural Language Learning

CRF Conditional Random Fields

DistilBERT A distilled version of BERT

DNN Deep Neural Network

FST Finite State Transducer

GMB Groningen Meaning Bank

GPT Generic pre-trained transformer

GRU Gated Recurrent Unit

HMM Hidden Markov Models

LSTM Long Short-Term Memory

LUKE Language Understanding with Knowledge-based Embeddings

MELM Masked Entity Language Modeling

MEM Maximum Entropy Model

NER Named Entity Recognition

NLP Natural Language Processing

PMI-IR Pointwise Mutual Information and Information Retrieval

POS Part of Speech

RNN Recurrent Neural Networks

RoBERTa A Robustly Optimized BERT Pretraining Approach

SVM Support Vector Machines

T-NER Transformer-based Named Entity Recognition

Chapter 1

Introduction

Named Entity Recognition (NER) is the process of locating the proper nouns or important terms from the given text and assigning the unique label (entity) to them [1]. An entity could be a topic or category such as Location, Person's Name or Organization. NER can be considered a multi-class classification problem and NER predictions are currently trained with large amounts of data. For instance, in [2], NER evaluation system was trained with over 200 thousand labeled tokens.

Typically, neural networks and deep learning algorithms require a multitude of labeled data for training [3]. However, such a high volume of data in the preferred format may not be readily available in the real-word applications. It is evident in biomedical or science domains where data needs to be properly curated and annotated by the experts which is time consuming and costly [4]. This data scarcity problem, known as low-resource problem, is a common issue and many approaches have been proposed to solve. One example to resolve the issue is using a transfer learning approach in which the representation is learned on the self-supervised source tasks and then it is adapted to the target task's representation [5]. Other researchers also adopted the different techniques, using data augmentation. In which the training set is expanded by modifying training instances without changing the labels [6].

Data Augmentation approaches on sentence level and pairs have recently gained popularity in Natural Language Processing (NLP) applications such as classification of text [7], machine translation [8] and natural language inference [9]. A common technique in creating the augmented instances involves modifying a word or sequence of words in the original sentence, such as a

direct word replacement [10]; or random deletion [11]; or word position swap [12]; or generate completely new sentences via variational autoencoders [13]; or back-translation models [14].

In contrast to the above-mentioned sentence-boundary oriented NLP tasks, NER predictions are based on the token level. Given a sentence, NER system will predict certain words as the entities and assign them with the predefined labels. Hence, it is challenging to apply any text transformative technique on the sentence as tokens associated to the entities might be transformed completely. In this thesis, we will try to fill this research gap by exploring masked text generation methods.

1.1 Motivation

The objective of this thesis is to find an approach to augment the new or existing datasets with different permutations while maintaining the structure and semantics for NER training. This thesis will address how challenging and data-hungry it is to train a NER model for broadcast news domains and for low-resource domains such as medical domains.

Conventional approaches when it comes to NER data requisition follow the process of procuring off-the-shelf electronic datasets or generating entirely new datasets from scratch or a combination of both. The former approach is comparatively efficient and less labor intensive than the latter but it faces the potential issue of domain mismatch. While the latter approach would create the most representative data for the target domain, it would be extremely costly to generate a large volume of data for NER training.

Hence, this thesis aims to realize an approach by leveraging the Finite State Transducer (FST) with context free grammar and Deep Learning techniques to augment and conjure new data points by generating a sentence template, identifying important tokens or states and ultimately reconstructing multiple permutations of the existing data points.

1.2 Contributions

In this thesis, two text augmentation approaches are proposed to overcome the low-resource data problem found in NER tasks.

Text Augmentation with FST: In this study, FST is used to create grammars and as a keyword substitution tool. We divide the NER datasets into training and testing to create the templates. The segments pertaining to the relevant NER tags must be labeled from the training data. By substituting the words within the NER tags with variables that can be updated by the FST text generator with a fresh list of possibilities for the tag, these segments become templates for the FST.

Text Augmentation with Pegasus Abstractive Text Summarizaton: In this approach, we applied Google's Pegasus pre-trained model to create semantically similar sentences from given sentences. Released in 2020, Google's Pegasus model was built for abstractive text summarization. Although it was not designed for text generation, forcing Pegasus to process inference from a short sentence causes it to paraphrase into a new sentence that is semantically comparable, and so we "trick" Pegasus into becoming a one-to-one sentence text generator with semantically identical input and output sentences.

The paper consolidating above-mentioned approaches was submitted to the conference, 14th Asian Conference on Intelligent Information and Database Systems, 2022 (Submission 346, Main Track).

1.3 Report Organization

This thesis is divided into five chapters and an overview of each chapter is as follows:

- Chapter 2 provides a thorough review of related works in NER field which span from rule-based approach to current deep learning approach. The chapter also explores various popular annotation schemes employed for NER training and testing. The chapter concludes with an overview of the evaluation matrix used for entity recognition systems.
- Chapter 3 gives an overview of the preliminary evaluation results of different state of the art NER models on some popular and prominent NER datasets. However, the datasets are subsequently discarded from further investigation as they do not represent low-resource domain. The chapter concludes that BERT NER implementation will be used for augmentation experiments.
- Chapter 4 gives an overview of the proposed Text Augmentation method. Then it provides details of the experiments where 2 new NER datasets with low frequency terms are

introduced along with BERT NER systems, BERT, DistilBERT and RoBERTa. The chapter concludes with analysis and discussion of the experiment, verifying whether the proposed augmented method affects the NER model performance.

- Chapter 5 concludes the thesis and summarizes the future work.

Chapter 2

Literature Review

NER is one of the prominent fields of Natural Language Applications. It has been widely used on multiple real world applications such as highlighting important terms on news, Extracting structure from unstructured text and Power Recommendation across multiple domains ranging from financial or medical fields. Range of techniques have been developed to drive NER results across the years with deep learning techniques achieving the state-of-the-art result. Hence, to provide a comprehensive literature review of research done on NER, Section 2.1 briefly described classical approaches to Text Classification. The following Section 2.2 reviewed various stages of NER techniques and its evolution. Then, Section 2.3 explored the 2 contemporary NER Tagging Schemes: Position-based and Constituent-based Tagging scheme. Section 2.4 expands on the metric used to evaluate the performance of the NER system. Final Section 2.5 reviewed data augmentation techniques applied on text classification and NER tasks.

2.1 Deep Learning based Text Classification

Text classification is a classical problem in NLP with the goal of classifying text into predefined categories.The application of text classification via automation are Part of Speech (POS) tagging, sentiment analysis, news categorization and topic classification. Approaches to text classification can be categorized into rule-based , machine learning based and deep learning based [15]. In each approach, the inputs to the classifiers are tokens, sentences, documents or combinations of all. Token classification or commonly known as NER is one of the sub-tasks of information extraction and text classification in which the candidate tokens are extracted and given predefined class

labels. As the goal of this thesis is to explore the data augmentation techniques with respect to improving NER performance, in this section, we will focus on the literature review of legacy and deep learning based approaches for NER. Readers are recommended to the Minaee et al. [15] article for more details on deep learning based text classification models.

2.2 Overview of Named Entity Recognition approaches

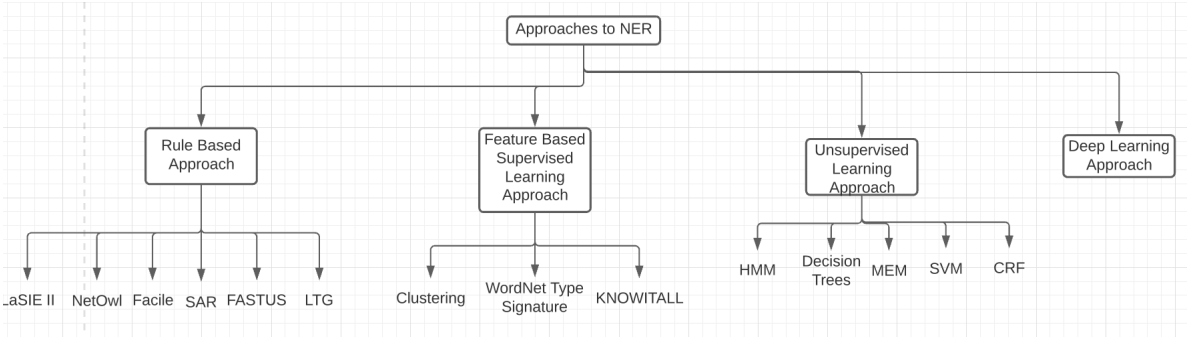

FIGURE 2.1: Approaches to NER

The traditional approaches in solving NER can be divided into three categories: the rule based approach, unsupervised learning based approach and the feature based supervised learning approach. [16]. However, over the recent times, Deep Learning NER systems have risen to prominence and achieved state of the art results. In comparison to the features dependent methods, deep learning is advantageous as the neural networks are designed to discover hidden features automatically. Firstly, we briefly describe the traditional approaches to solving NER and later understand what deep learning is, and subsequently survey Deep Learning based methods with respect to NER.

2.2.1 Rule-Based Approach

A Rule-based NER method depends on hand-crafted rules. Such crafted rules can be devised from domain-specific documents and syntactic-lexical designs. The rule-based systems directly encode the linguistic knowledge of the data set directly into a collection of simple rules [17]. The rule-based NER program generally consists of two parts; cleaning and extraction. In the cleaning phase, we discard unnecessary information from every record to handle the extraction of named entities. In the extraction phase, we recognize the data from the "refined" memo. We identify the unnecessary data for both phases and recognize the patterns of occurrences of named

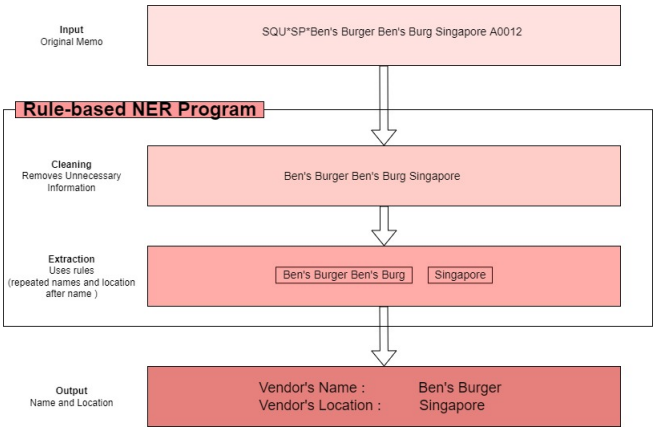

FIGURE 2.2: Rule-Based NER example.

entities for extraction. Rule-based models perform efficiently when the dictionary is exhaustive [18]. Such systems frequently experience high precision as a result of domain-specific rules and incomplete dictionaries of which the results cannot be conveyed to different domains [17].

Early works done in this approach for NER

Some of the popular rule-based NER systems include LaSIE-II in 1998 [19], FASTUS in 1998 [20], SAR in 1998 [18], Facile in 1998 [21], LTG in 1999 [22] and NetOwl in 2005 [23] systems, each of which follows different system architecture and rule patterns for identifying named entity tags.

Some of the later works done in NER using this approach are: Ralph Grishman in 1995 [24] developed a rule-based NER method that employs specialized dictionaries, incorporating the names of all nations, principal cities, organizations, traditional maiden titles, etc. Kim in 2000 [25] recommended using the Brill rule deducing method for speech data. The technique stated above applied Brill's POS tagger to generate rules automatically. In the biomedical field, ProMiner, a dictionary-dependent model [26] was proposed to recognize protein naming and potential genes in biomedical documents. Quimbaya et al. in 2016 [27] suggested a dictionary-based procedure for NER in computerized health reports.

Advantages of using Rule-Based Approach to solving NER

While statistical machine learning is popularly applied as a black-box technology concerning traceability and clarity of decisions, rule-based methods support a primarily declarative approach leading to clear and expressive models. Additional advantages of the rule languages include readability, maintainability, and the likelihood of directly conveying domain knowledge into rules.

The potential of including the understanding of a field expert into the extraction process instead of adopting the most promising training data, hyper-parameters, or weights, thus indirectly incorporating the domain knowledge, is a notable advantage contrasted to other approaches to information extraction.

However, rule-based and machine learning strategies need not be practiced exclusively but can complement one another very well. Supervised learning always needs substantial training data; one of the primary steps is to sample and generate training data sets. In areas where pre-labeled samples are rare or non-existent, this step is the most time-consuming task. To obtain a significant quantity of high-quality training examples, rule languages can be used. This methodology is a form of bootstrapping modern machine learning technology [28].

Limitations of using Rule-Based approach to solving NER

Some of the disadvantages of rule-based systems are that The Rule-Based system necessitates a deep understanding of the field and a lot of manual work. Creating rules for a complicated system is very challenging and time-consuming. The system will generate the result as per the rules, so the learning potential of the method by itself is significantly less. If an application we desire to build is too complicated, creating the rule-based system can take time and investigation. Complex pattern identification is a challenging job for this approach [29].

2.2.2 Unsupervised Learning Based Approach

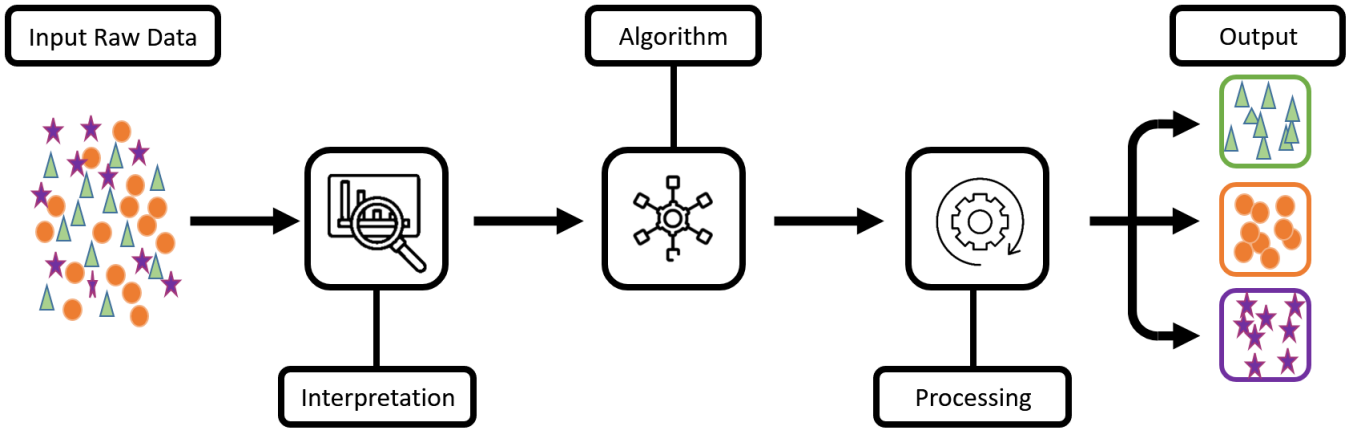

FIGURE 2.3: Unsupervised Learning Architecture.

Unsupervised learning refers to applying Artificial Intelligence algorithms to recognize patterns in the knowledge base comprising data points that are neither classified nor labeled. Supervised learning comes with the challenge of needing the huge volume of annotated data on the features set that is the most representative of the data. Such annotated data might be readily available for many domains and languages. Unsupervised procedures for NER have been developed to deal with the scarcity of annotated corpus across areas and languages [30].

Early works done in Unsupervised learning Approaches for NER

[31] in 1991 and [32] in 1992 are early attempts in this field. The authors introduced the idea to extend ontology with domain-specific data automatically. In [31] in 1991 and [32], authors invented NER systems based on heuristic rules and lexical sources like WordNet [33]. Rau in 1991 [31] implemented a program that extracts organization titles automatically from financial news. The paper implemented a system by merging heuristics, exclusion lists, and comprehensive corpus review. Numerous researchers have developed Unsupervised NER methods. Some of the earliest methods utilized minimal amounts of training data. One method in 1999 [34] for classification and name entity extraction, required labeled seeds and seven characteristics such as orthography, the context of the entity, words enclosed within named entities. Alfonesca and Manandhar, in 2002 [35], said that the principal gain of their procedure is its capability to be employed in any language. The authors introduced an approach based on a "word-sense disambiguation" system. The process employed in their procedure is Aggire's approach [36]. They adopted Aggire's approach to generate topic headers as they are not available in WordNet. They investigate the problem of identifying an input term with a relevant named entity class. Named entity classes are obtained from WordNet. The method allots a class signature to every WordNet synset by listing commonly co-occurring terms in a vast corpus. Later, given an input term in a document, the term's context is compared to class signatures and categorized under the most alike class.

One literature in 2006 [37] came up with an unsupervised model for construction of gazetteers and ambiguity resolution of named entities based on the approaches by the earlier researches in 2005 [38] and in 1999 [34] that merged the target gazetteers with publicly available gazetteers documents to attain the high F1 score of 59%, 61% and 88% on MUC-7 [39] Organization, Person and Location entities respectively. Nadeau et al. [37] came up with an unsupervised NER method of two modules. Module A applied generation of a comprehensive list of common

entities where Module B deployed a heuristic entity disambiguation approach to cluster the entities for a given context.

In 2013, Zhang and Elhadad [40] suggested an unsupervised method for recognizing named entities in biomedical corpus. Rather than supervision, the technique heavily relies on domain-specific terms, word distribution in the corpus as well as word correlations. This unsupervised method is able to generalize well and produce NER effectiveness on the conventional biomedical datasets.

The prominent systems in Unsupervised NER

One of the most prominent systems for NER is KNOWITALL [38]. Much literature in information extraction has been done on tiny datasets utilizing hand-label training samples. Manual training samples have allowed statistical tools such as Hidden Markov Models (HMM) or Conditional Random Fields (CRF) to obtain data from complicated expressions. In distinction, KNOWITALL focuses on unsupervised web information extraction methods. It takes a collection of defined signatures as input but no manually annotated trained samples and integrates the extraction method from a small collection with typical patterns. Additionally, it employed a unique "generate-and-test" design to achieve high precision, relying on occurrence statistics of the web corpus.

Pointwise Mutual Information and Information Retrieval (PMI-IR) was used in 2005 [38] to evaluate the association of words for a given named entity or a class. The first iteration of PMI-IR [41], is used in an information retrieval system to measure mutual-information among words or phrases in web queries. The greater the PMI-IR, the higher chances of words co-occurring. O. Etzioni et al. [38] elevated PMI-IR approach where it could form the association between candidate entity class such as "London" with discriminator sentences or phrases like "is a city" or "is located". KNOWITALL leverages this approach to perform domain-independent extraction on the web corpora in the unsupervised manner [38].

KNOWITALL employs eight domain agnostic IE patterns to produce possible entities or candidate facts. Given the pattern "NP1 such as NPLIST2", it can be assumed that all the noun phrases values appears in NPLIST2 belongs to the candidate fact of NP1. The possibility of the entity will then be then assessed by deploying PMI computed against the large volume of text from web corpus. KNOWITALL assign the probability value to the extracted candidate fact based on

the calculated PMI score. At its core, KNOWITALL is dependent on the bootstrapping step where the domain agnostic pattern extraction is materialized from extraction rules and words phrases used as a discriminator.

2.2.3 Feature Based Supervised Learning Approach

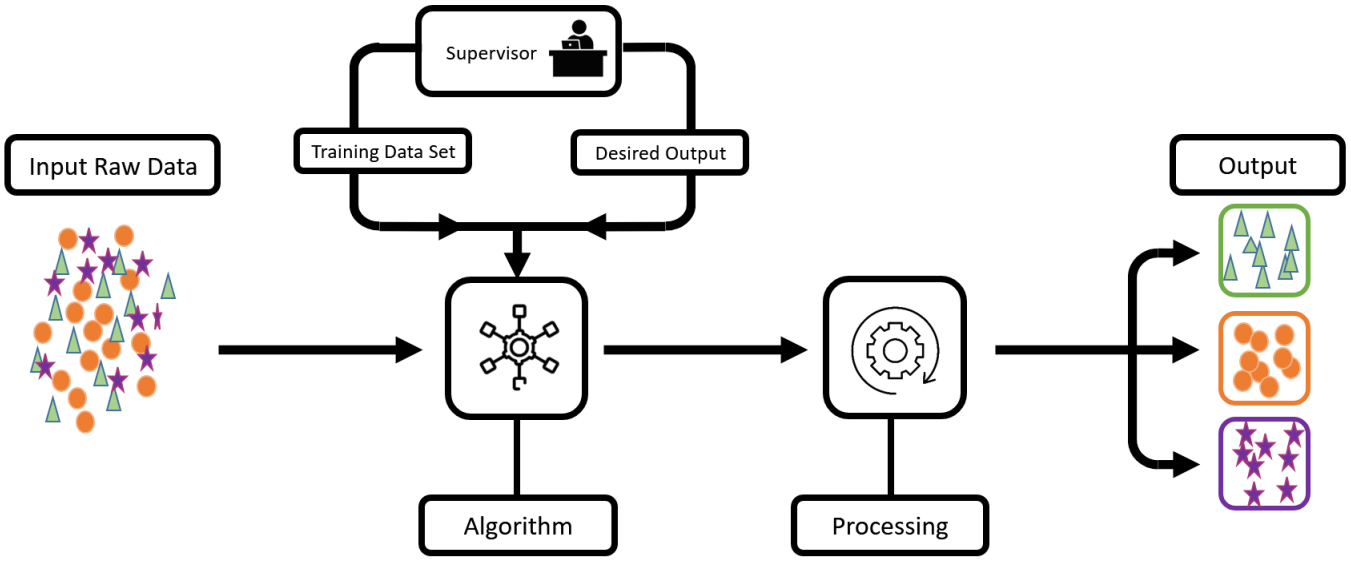

FIGURE 2.4: Supervised Learning Architecture.

Supervised learning approaches deploy algorithms which learn a pattern by studying labeled training data [42]. Name Entity recognition is considered as a multi-class classification or sequence labeling problem that could be solved with supervised learning. Features can be extracted from the given annotated data while the model will learn the class-defining pattern and predict class labels for the unseen data. Hence, in supervised learning systems such as NER, data mining and feature engineering tasks are detrimental to the model performance. Typical input to the supervised learning is called a feature which is a text representation in the vector space.

An input word can be transformed into numeric or boolean values in matrix or vector representation. This process is called feature vector representation. Word boundary characteristics list lookup features, and document and corpus characteristics have been broadly used in several supervised NER models. Various machine learning models such as Conditional Random Field, Decision Tree, Hidden Markov Model, Maximum Entropy Model and Support Vector

Machine [43] widely used feature vector representations to learn discriminative characteristics that maximize the likelihood of training data [42].

Hidden Markov Model (HMM)

HMM is the initial model employed for solving the NER problem by Bikel et al. in 1999 [44]. Bikel proposed a system, IdentiFinder, to identify NER. Based on Bikel's formulation of the problem in the Identifinder model, only an individual label can be attributed to a word in context. Consequently, the system attributes either one of the desired classes to each term or the label "NOT-A-NAME" to express "none of the desired classes." The HMM-based chunk tagger yielded an accuracy of 96.6% on the MUC-6 dataset and 94.1% on the MUC-7 dataset. Zhou and Su in 2002 [45] used an HMM based NER system which achieved F1 scores of 94.1% and 96.6% on MUC-7 and MUC-6 datasets respectively. The features used in this model include the list of words from several gazetteers and trigger words for name entities.

Maximum Entropy Model (MEM)

Unlike the HMM, the MEM is a discriminative model. The model learns the weight of discriminative features for classification given a collection of features and training data. The goal of maximum entropy models is to maximize the entropy of the data so that the training data can be generalized as much as possible. Borthwick in 1999 [46] proposed a Maximum Entropy Name Entity system which utilizes a unique collection of data sources to make its tagging decision. It uses a wide collection of gazetteers entries of individual or multi-word expressions like first name, organization name, corporate suffixes. It applies a broad diversity of features such as binary, lexical and section features, external systems output and reference resolution. Curran and Clark in 2003 [47] employed the another MEM to the NER problem where they deployed the softmax function to calculate the probability P(y—x). The paper reported the 84.89% and 68.48% accuracy scores for the English and German test datasets of the CoNLL-2003 shared task respectively. By combining multiple features, Malouf in 2002 [48] made a comparison between HMM and Maximum Entropy. Their most robust method , built with 13281 first names from dictionaries, incorporated capitalization which determines whether a word appears for the first time in a phrase or has previously appeared with a known last name. The model obtained a 73.66% and 68.08% F score on Spanish and Dutch language of CoNLL 2002 datasets.

Support Vector Machines (SVM)

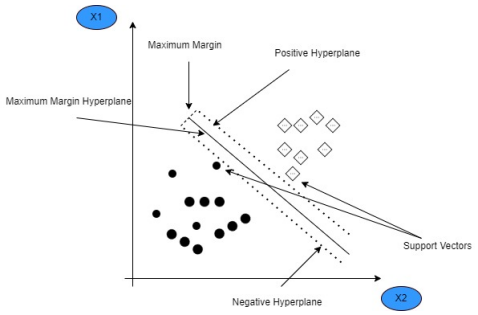

FIGURE 2.5: SVM Architecture

Cortes and Vapnik [49] introduced the Support Vector Machine (SVM), which is based on learning a linear hyperplane that divides positive samples from negative by a large margin. Because of the large margin, When the distance between the hyperplane to either point of the division is at highest value, the hyperplane is considered to be at a vast margin and ideal. [50] investigated NER problem via SVM as multi-class classification where 8 SVM classifiers are model. Each hyperplane will determine whether a word is either B-Beginning, I-Inside or O-Outside for location, organization, person or others tags. The SVM model was able to achieve 60.97 and 59.52 accuracy scores for Spanish and Dutch CoNLL 2002 datasets respectively. Li et al. in 2005 [51] further improved this SVM model by expanding window length and considering neighboring words where features of the neighboring words are weighted during the classification. Two SVM classifiers are trained to predict the start and end word of the entities. The model achieved an F score of 88.3% on the English CoNLL-2003 dataset.

Conditional Random Field (CRF)

CRF is another example of a discriminative model that studies the context and surrounding neighbors for the prediction tasks. [52]. Feature induction model was [53] proposed in 2003 and achieved 84.04% and 68.11% accuracy for CoNLL 2003 English and German datasets respectively. In the field of medicine and drug NER,[54] applied CRF using lexicon features and word embedding trained on corpus such as MedLine, DrugBank and etc. to achieve the start of the art results. On the same NER task, [55] proposed a dictionary-based CRF using concepts and ontologies from chemical entities.

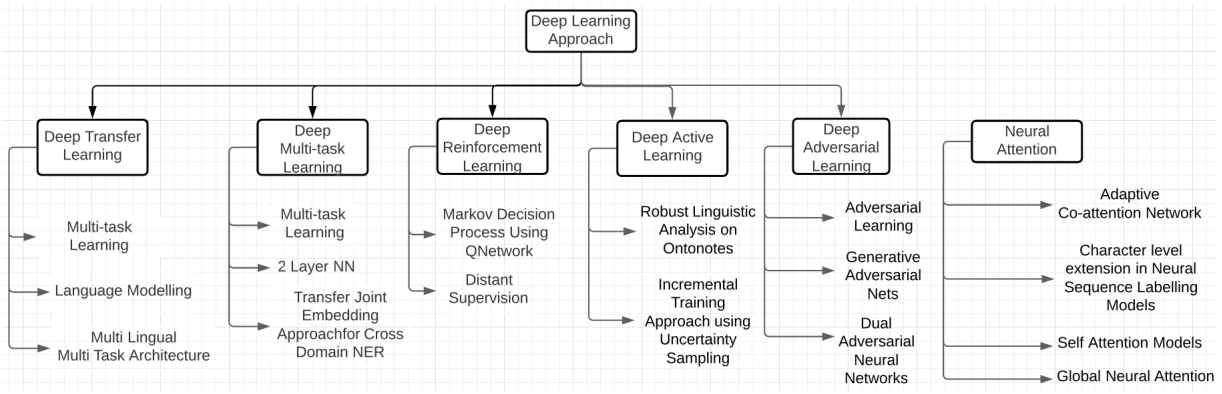

FIGURE 2.6: Deep Learning Approaches to NER

2.2.4 Deep Learning Based Approaches

As described in Figure 2.6, Deep learning paradigm encompasses approaches such as Transfer Learning , Multi-task Learning, Active Learning , Adversarial Learning and Neural attention. The following sections will review and compare each learning model with respect to NER.

2.2.4.1 Deep Learning based NER

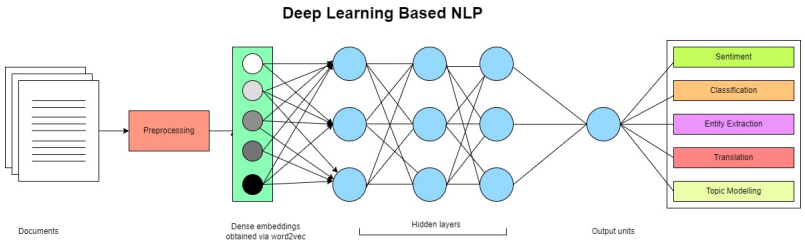

FIGURE 2.7: Deep Learning Architecture.

Recently deep learning NER practices have grown predominant and obtained state-of-the-art outcomes. Unlike supervised methods, deep learning is advantageous in automatically recognizing hidden characteristics. In the following section, we first briefly present deep learning, including why deep learning is used for NER followed by deep learning based NER strategies.

Deep learning, a refined framework of machine learning, utilizes multi-layer artificial neural networks to study patterns. This neural network processes inputs by cascading numerous layers of nonlinear transformations to estimate a final result. The first few initial layers study simple traits, while the deeper layers determine complicated characteristics emerging from the lower layers. A simple feed forward neural network consists of 3-layers template namely, an input layer,

hidden layers, and a result layer. The nodes on the hidden layers use the previous layer's output and pass outcomes to the subsequent layer.

Advantages of Deep Learning based NER

There are three main advantages of employing deep learning procedures to NER. First, NER profits from the nonlinear transformation, producing nonlinear mappings from data to outcome. Unlike the linear models such log-linear HMM, deep learning based systems can study complex characteristics from training data through nonlinear activation functions. Next, deep learning saves notable work on outlining NER features. The conventional feature-based strategies need a large quantity of engineering talent and field expertise. Deep learning based systems automatically determine valuable representations and underlying features from raw input data. Thirdly, gradient descent can train Deep Neural Network (DNN) NER systems in an end-to-end model. This attribute allows us to produce likely complicated NER practices [43].

Various architectural systems for Deep Learning NER

Deep learning models for NER are mainly categorized into Convolution Neural Networks (CNN) [56], Recurrent Neural Networks (RNN) [57] and Transformer architectures. Even though RNN is competent in handling long sequences of inputs, they seldom fail due to fading gradient problems [58], [59]. As a result, more complex variations of RNN such as Long Short-Term Memory (LSTM) [60], Bidirectional Long Short-Term Memory (BI-LSTM) [61], and Gated Recurrent Unit (GRU) [62] were introduced to tackle vanishing or exploding gradient problems.

The CNN-CRF system [63] introduced the use of DNN for NER, where it is used to extract "word-level information from input word embeddings". Then, as a tag decoder, a CRF layer is put on top of it. Using the embeddings combined with external data, [63], the system achieved an 89.59 F1 score in the NER challenge by creating a fixed-sized contextual window.

Variations of RNN/LSTM models [64] were proposed to solve NER problem. The LSTM network, BI-LSTM network, LSTM-CRF, and BI-LSTM- CRF are examples of these models. Word embeddings and other word characteristics were fed into the BI-LSTM to construct the word-level representation in BI-LSTM-CRF. This representation was then transmitted to the output CRF layer, which was used to determine output tags. In comparison to the CNN-CRF model[63], the

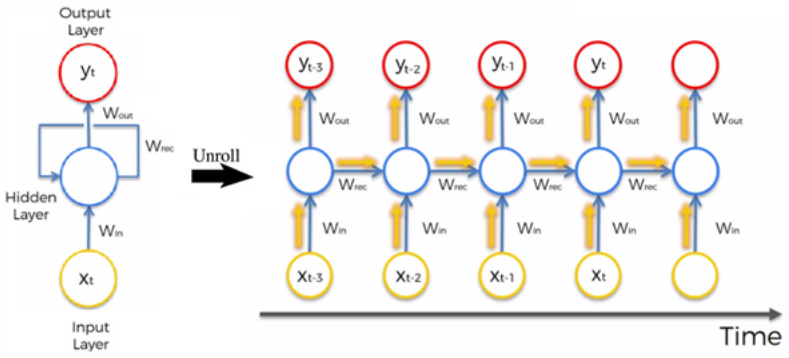

FIGURE 2.8: Recurrent Neural Network Architecture.

BI-LSTM-CRF system that is not highly dependent on word embeddings, achieved an F1 score of 90.1% using both Senna embeddings [63] and gazetteer features.

Lample [65] proposed a similar LSTM model to [66] where a BI-LSTM was used for character feature extraction. A forward and backward LSTM is provided with the character embeddings for each character in a word in direct and reversed order. The character-level data is combined with skip-n-gram generated pre-trained word embeddings [67]. During the training step, word embeddings were fine-tuned, resulting in a F Score of 90.94%. With the use of GLOVE, Global Vector for Word embeddings, [68] presented a unique architecture comprising BI-LSTM, CNN, and CRF that could extract features at word and character level. Without using any character-type traits, the model obtained 91.21% F1 score [66].

One literature [69] came up with a hierarchical NER model that used a combination of deep GRUs or CRF layers to predict the output tag. The GRUs leveraged on [63] word embeddings to learn the hierarchical patterns at character level and semantic information at word level. The model achieved a 91.20% F1 score after fine-tuning word embeddings throughout the training phase.

Complex convolutional or recurrent networks with encoders and decoders are commonly used in sequence to sequence labeling tasks. Vaswani introduced a groundbreaking Transformer architecture [70], which completely redefined recurrence and convolutions. Transformers utilize attention and fully connected layers to generate encoder and decoder networks. Transformer model has shown to be competent at various tasks [71], [72], [73] by attaining better results despite taking shorter training time.

Generic pre-trained transformer (GPT) [74] was developed on the Transformer architecture to solve language comprehension problems. GPT employs a 2 stages training approach. To select the initial parameters, GPT first employs a language modeling task using Transformers on raw data. On the 2nd stage, utilizing the supervised goal, GPT adapts these parameters to a target task, resulting in minimum changes to the pre-trained model. Contrary to GPT, Bidirectional Encoder Representations from Transformers (BERT) model [75] proposes conditioning both left and right context at all levels to pre-trained deep bidirectional Transformers.

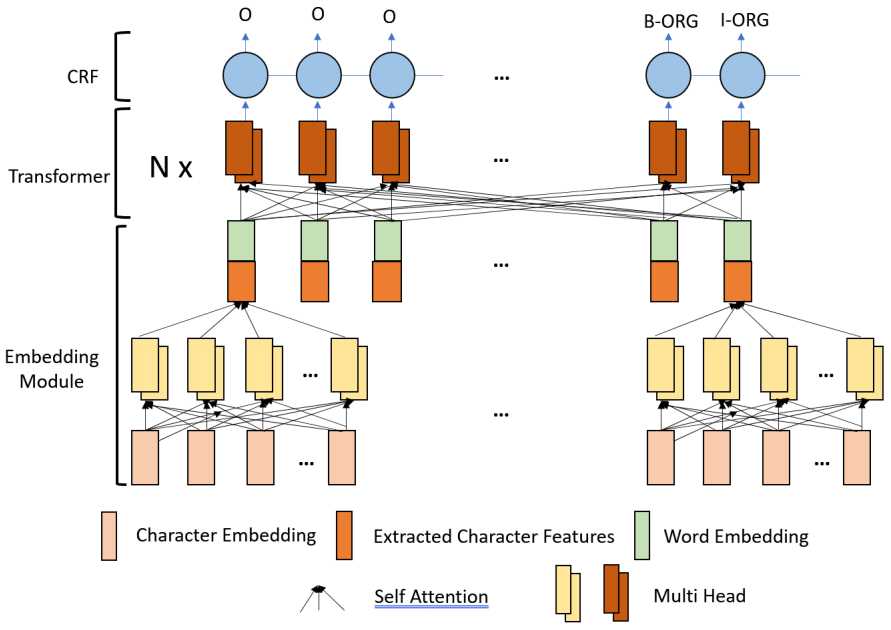

FIGURE 2.9: Transformers Architecture.

The usage of these language model embeddings trained with the Transformer model has been the new standard approach for NER tasks. This is partly due to traditional embeddings such as Word2Vec or Glove having limitations with contextualization. Multiple experiments [76], [77], [78], [79], [80], [81] have proven that the combination of traditional and language model embeddings yield better results. Moreover, with the addition of an extra layer at the end of the network, these pre-trained embedding can be further fine-tuned for various NER tasks. This is supported by [82] where the study advocated the use of fine-tuning BERT models to solve machine reading comprehension tasks.

Comparison between different Architectures

Primarily, no agreement has been reached about whether external knowledge should be combined into deep learning based NER systems. Some studies [76], [77], [78], [83] indicated that the

use of expert knowledge could improve NER performance. Nevertheless, limitations could be self-evident in obtaining external knowledge (labor-intensive and computationally costly); and integrating external knowledge negatively influences the learning and generalization of deep learning models. Secondly, Transformer models produce better results over LSTM model when it is pre-trained on the large corpora but it is evident that the model performance degrades drastically in the absence of training data [84], [85]. In terms of computation speed, Transformer can encode and decode much faster than RNN models when input length and dimensionality are proportional [71]. Thirdly, another critical limitation of RNN and LSTM architecture relies on the sequential nature of processing where the current activation node requires the output of the previous node. This further limits the capability of parallelization and impacts the network speed.

To summarize, the choice of NN architecture is dependent on the data and task. Transformer excels in learning parameters for long sequences of text due to the use of attention mechanisms and the training data is in abundance. On the other hand, LSTM can be effective with a few hidden layers for NER prediction when dealing with data scarcity even though it has issues with scalability [43].

NER with Deep Multi-task Learning

[86] introduced Deep Multi-task learning method designed for learning a set of similar tasks together. By studying the relationship among various tasks, multi-task learning programs are supposed to produce more favorable outcomes than those that study each task independently. A study [87] developed a grouping strategy system to simultaneously predict POS, name entity, and semantic role tags. The algorithm of the multi task system intuitively learns possible feature representations useful for all joint tasks during the training. Another study [69] introduced a multi-task learning system for learning language dependent patterns, by training part of speech, chunk and name entity tasks jointly. Rei [88] observed the sequence labeling model delivers consistent performance enhancement by introducing an unsupervised language modeling goal in the training method. Another multi-task system for multi-lingual tasks was proposed in 2018 [89] which could transfer knowledge learned on join tasks to the main model on a low resource environment. Apart from viewing NER collectively with sequence labeling related tasks, a multi-task learning method could be employed to jointly extract of classes and relationships [90], [91], or divide and learn NER prediction into 2 sub tasks namely entity segmentation and category prediction [92], [86]. In the medical setting, NER is regarded as a task in a multi-task

environment [93], [94] due to the diversity in various datasets. This is due to the several datasets in the domain sharing identical information on character and word-level. Hence, it is more efficient to employ multi-tasks learning that can leverage on data and produce better generalized data representation.

NER with Deep Transfer Learning

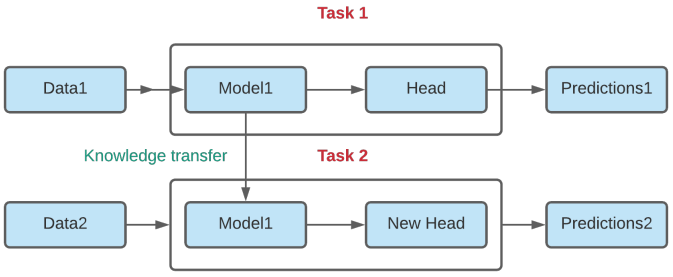

FIGURE 2.10: Transfer Learning Architecture.

Transfer learning or domain adaptation, intends to produce a machine learning job on a target area using knowledge acquired from a source domain [95]. On NER, the conventional method is by bootstrapping algorithms [96], [97], [98]. Contemporary approaches [99], [100], [101], [102], [103], [104] are introduced for cross domain and low resource NER utilizing DNN. A Transfer Joint Embedding [100] is one approach that uses class embedding methods to transform output labels and high dimensional input features to lower dimensional latent space for cross-domain multi-class NER tasks. Another study [105] examines the correlation among the entities assuming closely related entity classes share lexical and contextual information with the use of 2 layers neural network. The strategy can be applied to the NER task when the target domain shares similar entities to the source domain.

It is common that many neural network models often share different sections of parameters or characteristics among source and target tasks in transfer learning. [106] is the first study conducted to learn the transferability of layers and presented 3 parameters used for sharing architecture: cross-application, cross-domain and cross-lingual situations. For instance, when the shared label sets can be established between the two tasks, a shared CRF layer could be deployed. Empirical results reveal notable developments on multiple low-resource datasets. [107] advances the prior approach [106] by introducing sentence feature representation. The experiment on WNUT-2017 shared task NER proves the improvement of F1 score (40.78%) which is the second highest score with the use of joint training. Domain adaptive multi-tasks was introduced by

Zhao et al. [108] in which the fully connected layer is tuned to various datasets and CRF characteristics are calculated independently. Zhao's technique has the benefit of filtering out situations with varied distributions and misaligned annotation criteria during the data selection process.

In contrast to parameter-sharing methods, transfer learning with A fine-tuning model [100] was proposed for the NER task. A model trained with the data the source task can be fine-tuned to the target job. Recent work by Lin and Lu [102] substantiates the use of fine-tuning methods in NER with the application of 3 adaptation layers namely: word level adaptation, sentence level adaptation and output adaptation layers. Studies [109], [110], [111] confirmed the use of transfer learning, especially in biomedical NER, reduce the volume of data required for annotation.

NER with Deep Active Learning

The core idea underlying active learning is that if a machine learning system can select the data from which it learns, its model may produce better results with much less data from training [112]. Despite the performance boost, deep learning techniques are often data hungry and data preparation is an expensive task. Hence, an integrated active learning approach to deep learning should result in less data annotation work. As the name suggests, a typical active learning requires multiple rounds and is the continual retraining on the baseline deep learning model with new data. As a result, to combat the cost of complete retraining from scratch, Shen et al. [113] explored the potential batch training for NER task. Prior to updating the network weights and querying for labels in the next round, new label data is to be mixed with existing data for a few epochs. After collecting the desired annotations, the neural network parameters are adjusted based on augmented dataset training. Active learning with an uncertainty sample method [114] can be adopted for the selection of annotated sentences. The study revealed that the performance of an active learning model trained on 24% and 30% training data (English and Chinese) is 99% comparable to the state of the art deep learning model trained with a full dataset. Furthermore, the active learning model only requires as little as 12% to 16% of training data to beat shallow models trained on the entire training set [115].

NER with Deep Reinforcement Learning

Reinforcement learning is a subset of machine learning where the model is rewarded or penalized based on the decision it makes. Correct decisions will be rewarded so the model learns to repeat the same action on the subsequent task while the decisions led to error will be penalized with

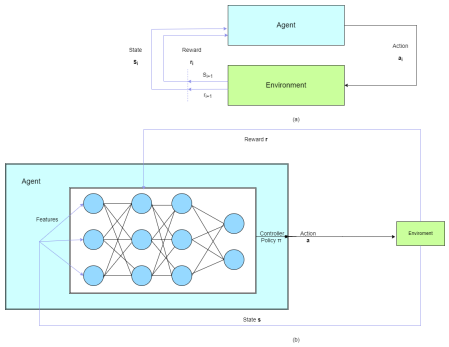

FIGURE 2.11: Reinforcement Learning Architecture.

the goal of maximizing the rewards [116], [117]. RL consists of two components: agents and environment where agent will provide inputs to the system and environment is the probabilistic finite state machine that observes and rewards the positively or negatively based on the inputs. With the objective of maximizing the cumulative rewards, an agent's final goal is to learn a decent state-update function and policy. The data generated via distant supervision was utilized in a recent study to conduct new type NER in new domains [118]. The instance selector uses reinforcement learning and receives input from the named entity tagger, with the goal of selecting positive statements to reduce the impact of noisy annotated data.

NER with Deep Adversarial Learning

Adversarial learning [119] was an unsupervised learning model that is designed to learn on adversarial instances. Two networks, generative and discriminative networks , are competing with each other with the objective of obtaining the robust model. Generative model creates fakes instances from latent space of the given inputs while discriminative classifies whether the candidate image is real [120]. Classification error is a core measure used to evaluate the entire network performance. NER tasks can leverage adversarial learning in 2 ways. In the first approach, [121], [122], [123] examined the method where they interchangeably use data from either source or target domain as adversarial inputs. For instance, recent works on cross-domain NER [122], [123] deployed the use of external domain data as adversaries samples to promote domain-invariant features. For the second approach, adversarial samples can be generated by mixing source data with a perturbation. For instance, with the application of dual adversarial transfer network [124], low-resource NER problems could be dealt with.

NER with Neural Attention

Neural attention model was loosely inspired from the behavior of human interaction on the visual mechanism [125]. For instance, when viewing the object, humans tend to enhance the resolution on a particular section while reducing the surrounding area's resolution. Adopting the similar concept, attention mechanism allows the network to concentrate on selected regions of the inputs. This is beneficial to NER tasks as the neural attention model could capture the most meaningful words related to the entities from the inputs. [126] investigated the use of single sequence weight for self-attention mechanism for NER task while another study [127] introduced an attention-based neural NER system to exploit document-level information, especially the document-level data extracted from documents is identified by the bidirectional language model with neural attention. An adaptive co-attention system [128] for twitter NER , introduced in 2018, was a multi-modal system employing the co-attention process. The model combines the use of attention mechanism on both visual and textual data to "capture the semantic interaction among various modalities".

NER with Capsule Network

While CNN implementation of adding successive layers of convolutional layers and pooling operation achieve competitive results, it continue to lose spatial information the further the layers are stacked. To address this problem, in 2011, Hinton et al. introduced the concept of Capsule Network (CapsNets) for a better hierarchical model [129] [130]. Capsules Network utilizes modules or capsules to represent the pose of the objects which include object's translation and rotation. The core idea to preserve of the hierarchical "pose" relations between objects parts. Capsule network recently gained attention in NLP field of text classification where each capsule is used as a vector to represent a sentence or a document. [131] [132] proposed a dynamic routing based Capsule network for text classification which achieved comparable results with LSTM/Bi-LSTM and CNN classifiers. For the NER task, [133] presented a mix of Bidirectional GRU and capsule network where capsule network is used as an entity classifier. Instead of mapping the GRU output features to the traditional linear layer, this approach utilized capsules to predict. Each capsule represents an entity vector which can preserve more information compared to scalar representation. The proposed model showed improvement of 5% F score compared to traditional LSTM baseline model on MSRA dataset.

2.3 Annotations used for Named Entity Recognition

Various annotation designs have been applied to different datasets over the years.However, selecting the best annotation method is a difficult task [134]. We will discuss some of the annotation schemes that are commonly used in the literature. These schemes are the following:

Position-based Tagging Scheme

IO: The IO scheme of annotation is the most straightforward scheme applied to NER. Tokens in the corpus are attached to either I (inside tag) and O (outside tag) in this annotation. The Inside tag is given for the words belonging to entity, whereas the O tag is for ordinary terms. There is a flaw with this scheme, as it struggle to tag connected entities of the identical type. The annotation was used by Mozharova and Loukachevitch [135] to annotate Russian Text to analyze the impact of IO and IOB annotation schemes for NER.

IOB: The IOB design is also commonly referred to as the BIO scheme, and it has been affirmed by the Conference on Computational Natural Language Learning (CoNLL) [136]. The IOB format is standard for tagging terms in a chunking task in Computational Linguistics such as the NER task. It designates the beginning (B) tag to the words that starts the Entity , inside (I) tag to consecutive entity words after B tag and or outside (O) tag for non-entity words.

IOE: The IOE scheme of annotation operates almost similarly to IOB, though it uses (E) tag instead of (B) tag to indicate the end entity word.

IOBES: IOBES is an alternative to the IOB system that increases the amount of knowledge about named entity boundaries. Apart from tagging the beginning words (B), the inside (I), the end (E), and the outside (O) of a named entity, (S) is used to identify single term entities.

BI: The BI tagging scheme annotates the entities in a similar manner to IOB. Non-entity words are additionally marked with B-O or I-O to denote the start and inside.

IE: The IE system works is similar to the IOE annotation with the difference that it annotates the end of non-entity terms with the tag E-O and the rest as I-O.

BIES: This annotation system is similar to the IOBES annotation. Additionally, it also annotates the non-entity words employing the same method. It follows BI and IE scheme of tagging non-entity words and an addition of S-O tag to identify single term of non-entity.

Constituents-based Tagging Scheme

TOMN: The TOMN scheme was introduced by Zhong and Cambria [137] [138] to eliminate the inconsistent tagging behaviour when it comes to tagging time expression. The scheme contains 4 tags namely Time token, Outside, Modifier, Numeric. For instance, using the classical tagging approach, the token "October" in "1st October 2021" and "October 2021" would be assigned as "I-Date" and "B-Date" respectively. This inconsistencies will reduce the prediction accuracy. With TOMN scheme, the token will be given "T" tag and hence it will retain the predictive power.

UGTO: Inspired from the work of TOMN scheme, the UGTO tagging scheme focuses on name entities and uncommon word tagging [139]. Uncommon tag (U) is encoded for uncommon words such as 'Boston' , Generic tag (G) is assigned to words that are part of the name entities such as 'of' 'the' while (T) encodes the trigger words such as 'University' in the phrase 'Boston University'. (O) tag is given for words outside of the name entities. Similar to TOMN, UGTO aims to overcomes the inconsistency of positional NER tagging scheme.

Though various annotation paradigms have been mentioned in the literature, determining the ideal annotation scheme for a particular system is a complicated problem, and it needs to be explored based on the language and text structure and several works have been done in this area [140].

2.4 Evaluation of an NER System

NER models are generally assessed by examining their results against ground truth results which are manually annotated. The Examination can result in either an exact match or a flexible match. NER typically comprises two sub-tasks: boundary detection and entity class determination.

In the exact-match evaluation [136], a correctly identified occurrence needs a model to precisely recognize the entity's boundary and class. For comparison purposes, the total numbers of False positives (FP), False negatives (FN), and True positives (TP) are typically used to generate the metrics such as Precision, Recall, and F-score.

FP refers to an entity identified by the model but it is not present in the ground truth. FN refers to the entity not identified by the model yet it is present in the ground truth. TP refers to the entity identified by the NLP model and present in the ground truth.

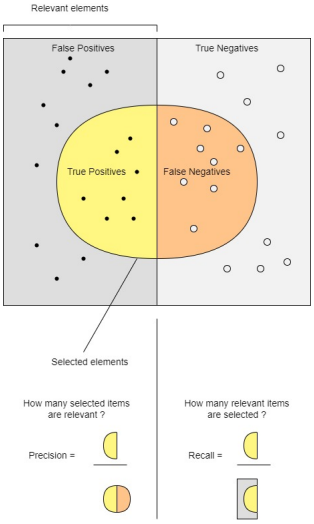

FIGURE 2.12: Evaluation of NER.

Two primary measures for examining the performance of any information extraction method are precision and recall measures. The precision metric is the proportion of the system's outcomes that are correctly identified. The Recall metric is the proportion of the total entities precisely identified by the NER model.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

The classic F measure or F score is the metric that couples both the above metrics of precision and recall.

$$F - score = 2 * \frac{(Precision + Recall)}{(Precision * Recall)}$$

Additionally, the macro-averaged F score and micro-averaged F score analyze the model's prediction capability among multiple entity classes. The macro-averaged F score individually measures the F score on various entity types, to calculate the mean of the F scores while the Micro-averaged F score totals the false positives, false negatives, and true positives of all entity types to generate the statistics. The Micro-average results can be massively influenced by the quality of identifying entities in a vast number of classes in the text.

An annotation is considered to be a relaxed match when the entity is labeled correctly even if the word boundary does not match exactly provided that the word boundary intersects with ground truth borders; a true boundary is considered irrespective of the entity's class assignment. Various complex evaluation methods have been used in various works such as ACE2005 [141] that are not natural and complicate error analysis. Therefore, in recent studies, more sophisticated evaluation techniques have not been popularly adopted [43].

2.5 Overview of Data Augmentation approaches

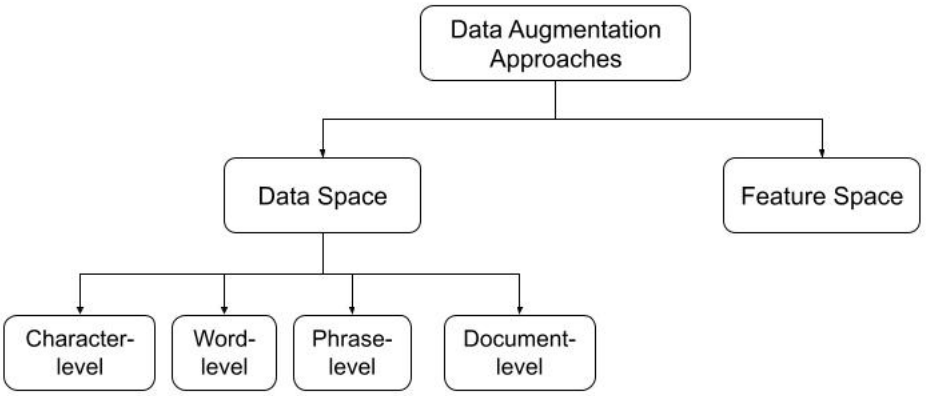

FIGURE 2.13: Approaches to Data Augmentation

Data augmentation methods refer to approaches of generating extra synthetic data from the existing dataset. The goal of data augmentation is to the tackle over-fitting problem which is apparent in low-resource languages as the model tend to over-generalize on relatively small training data. The fundamental technique includes modifying the structure of the sentence, sequences of words or even tokens of existing data with the aim of regularizing data.[142] [143]. Data augmentation is widely utilized in computer vision, for the tasks of cropping, flipping, and color jittering as well as in NLP tasks such as language modeling, back translation, and text classification. As the input space NLP is rather discrete, it is more challenging to build successful augmented instances that capture the necessary invariances. Hence, in this section, we will explore recent data augmentation approaches used for NLP and NER tasks.

As shown in Figure 2.13, data augmentation methods can be categorised into two categories: Data Space augmentation and feature space augmentation. In data space augmentation, the transformation of data usually happens in raw textual format whereas feature space augmentation

deals with transformation of feature representation of input data. In the following subsections, we will explore how each category of augmentation is applied in both text classification and NER tasks.

2.5.1 Data space Augmentation

2.5.1.1 Character-level augmentation

Character level augmentation revolves around the idea of adding synthetic noises to the input data in order to improve model's robustness when dealing with adversarial samples. [144] proposed a method to either randomly add characters in a word (e.g. "noise" changed to "noisie") or randomly rearranging the character positions (e.g. "noise" changed to "nsioe"). Similarly, this study [145] uses random deletion, swapping, and inserting of text characters to fine-tune text generators. They also disregard the initial and final character of a word for this purpose. They intrinsically assess variety, fluency, semantic context preservation, and sentiment consistency to determine appropriateness for text generators. In every way, the applied approach outperforms the baseline.

2.5.1.2 Word-level augmentation

Word-level augmentation is one of the popular techniques in data augmentation whereby the transformation is done by substituting the particular word with its synonym. Such process is commonly referred to as Synonym replacement and multiple articles [146] [147] [148] have applied it on the field of data augmentation using lexical dictionary such as WordNet [149] or thesaurus.

Embedding replacement is similar to synonym replacement, but this technique looks for words that are closer in the textual context and does not change the essential content of the text [150]. The words are transformed into the vector representation in the latent space where other vectors of the similar words are closer to each other. From the given sentence, the hypothetical words that are closed in the vector space are selected to create new instances of text. In comparison to the synonym substitution approach, this data augmentation methodology has the advantage of being more thorough and taking into account the context of texts. This implies that substitutes are no longer constrained by a database, such as WordNet, and therefore grammatically more accurate phrases may be constructed [151].

2.5.1.3 Phrase-level augmentation

Phrase or sentence-level augmentation deals with the structure of the input data to generate augmented text. Structures such as "grammatical formalities, dependence and component grammars or POS tags" are used to generate augmentation and as a results, these techniques are restricted by language and tasks.

The authors [152], for example, are interested in augmenting low resource datasets for the POS task. Sentences are condensed by focusing on topics and objects using the "cropping" approach. Flexible phrases are moved using the "rotation" approach. According to the authors, this strategy is dependent on certain grammatical phrase patterns in different languages and most likely adds noises in English. Both techniques are appropriate for a wide range of low-resource languages. Vania et al. [152] also evaluated them for the augmentation of training data for low-resource dependency parsers.

2.5.1.4 Document-level augmentation

Document level augmentation utilizes the round-trip translation approach typically found in the translation model.

Round-trip translation 3 is a method for creating paraphrases using translation models. The source text in sentence or document format is translated into another language via forward translation and then back into the source language [153]. The reasoning behind this is that text translations are frequently variable due to the complexity of natural language [154], and as a result, more semantically similar sentences with various structures are reconstructed.

Because of its intrinsic label preservation and effective paraphrase capabilities, the technique seems promising. The content of the text is kept through translation, and only stylistic details based on the author's characteristics are modified [155].

2.5.2 Feature space Augmentation

Unlike data space augmentation, Feature space augmentation was introduced to perform the data transformation in the features vectors [156]. The approaches includes noise addition, interpolation and extrapolation similar to data space approaches. Kumar et al. [157], for example, proposed

four methods for feature space augmentation to solve them intent classification. The first approach adopts addition of random multiplicative and additive noise to feature representations as defined in [158]. However, in their approach, produced representations are not translated back into the data space. Linear Delta is another approach that computes the difference between two instances and adds it to a third. Data interpolating is used in the third method. The authors' fourth technique adapts the Delta-Encoder of Schwartz et al. [159] for text data. An autoencoder model is trained to learn the deltas between instance pairs of the identical class, which is then used to produce examples of an new unknown class. In a typical testing scenario, the strategies only modestly increase classification results, whereas all methods show favourable results in a few-shot setting.

2.5.3 Data Augmentation for NER

NER tasks in low-resource domains share the same issue of overfitting when training data is scarce. In order to avoid the issue and improve the generalization capability of a model, most recent papers employ a combination of data augmentation techniques discussed in Section 2.5.1. Hence, in this section, we will review recent data augmentation approaches widely deployed in enriching NER systems.

Dai and Adel [160] proposed 4 methods of NER augmentation namely label-wise token replacement, Synonym replacement, Mention replacement and Shuffle within segments. This approach combines both character and word level augmentation discussed in Section 2.5.1. and Section 2.5.1.2 respectively. The core idea behind each technique is to replace tokens with synonymous words from WordNet [149] and the token selection is dictated by binomial distribution or entity words. To simulate the low-resource issues, multiple LSTM and Transformer-based models ware trained using 50, 150, 500 and full instances of training data including augmentations. The evaluation results tested on MaSciP [161] and i2b2 [162] datasets show the F1 score improvement over the baseline system whenever the augmentation is used. Another observation is that applying all augmentation methods together outperformed any single data augmentation for small data partitions (test with 50,150 and 150 training instances). This technique does not suffer from label misalignment issue as each entity token is randomly replace with another entity token however, the entity diversity is not improved.

Ding et al. [163] introduced generative language model-based augmentation approach using **rnnlm** for low-resource tagging tasks. This sentence-level augmentation approach linearized

labeled sentences before training the language model to learn the context and distribution entity words. First, for a given labeled sentence, the entity tag are inserted before the start of each entity token with BOS and EOS tokens added to define sentence boundary.The entire process is known as linearization. Next, the linearized sentences are then fed into 1-layer RNNLM ,introduced by [164], to maximize next word prediction. For the generation step, only beginning of sentence token is inserted and the sentence is constructed in an auto-regressive manner. The proposed method achieve state of the art F1 score of 83.74(en), 62.44(de), 80.74(es), 72.71(nl), 77.39(vi) and 69.86(th) for multiple languages of CoNLL-2003 dataset. This is due to the diverse nature of language model which could learn context from training samples and generate unseen combination of text with entity.

Cross-domain augmentation approach [165] was explored to leverage data from high resource domains and apply learned linguistic patterns such as structure, style and noise on the low resource domains. In this feature-based augmentation approach, a linearized sentence pair [163] from source and target domain ,is used as an input to the autoencoder model. The model performs "word-by-word" denoising reconstruction followed by detransforming reconstruction. Denoising reconstruction stage forces the model to capture textual patterns for each sentence while detransforming reconstruction transforms the sentence from one domain to another while preserving the semantics. The generated sentences are then used to fine-tune BERT model for NER evaluation. By setting OntoNote5.0 [166] are the source/high-resource domain and Temporal Twitter Dataset [167] as target/low-resource domain, the proposed method is able to achieved F1 score of 65.25% using only 5% of target dataset with potential of hitting 77.59% as more samples are added to autoencoder model training. However, the method does not outperform the model trained with target domain data, the resulted model provide better lower bound baseline for semi-supervised learning.

A recent study by Zhou et al. [168] presented a Masked Entity Language Modeling (MELM) for low resource NER. This novel framework is built upon pre-trained Masked Language Model [169] to solve a common label misalignment issue found in low-resource text augmentation by explicitly conditioning the entity labels. The proposed approach pre-processed data by injecting entity labels (e.g.<B-ORG>, <I-ORG>)and masking entity tokens (e.g.<MASK>) into the training sentences. While [163] inserted an entity label token before the entity token, this method added extra label token at the end of the entity token for each training sentence (e.g.<B-ORG>European <B-ORG>) before masking the tokens. These linearized sentence are consumed to fine-tune MELM to generate augmented data with diverse entities via masked

entity prediction. The proposed method is compared against MELM without linearization, [160] and [163] baseline methods on both monolingual and cross-lingual NER task of CoNLL2003. Under the low-resource dataset setting, the MELM with linearization approach outperforms all other approaches on both tasks. It achieved significant performance gains on the dataset with lowest sample size (100 sentences) with F1 score of 70.44 on average. At higher sample size, the MELM performance is comparable to other baselines.

Chapter 3

Comparison of existing NER models on 2 public datasets

In this chapter, we first explored two widely used datasets CoNLL-2003 and BC5CDR for NER tasks [170]. Both CoNLL-2003 and BC5CDR are two of the most popular datasets on paperswithcode.com which appeared in 319 and 89 open-access papers respectively in recent five years[171]. CoNLL-2003 is a generic English language dataset while BC5CDR is built for the medical domain. Subsequently, we evaluated each dataset with top state-of-the-art pre-trained NER models mentioned in paperswithcode.com. CoNLL-2003 was benchmarked on LUKE, T-NER and Flair Cross-Weigh pre-trained models whereas biomedical BC5CDR was benchmarked on Spark NLP and BioBERT. The goal was to verify if reported results could be replicated using the pre-trained models and to explore the ease of retraining/fine-tuning with own datasets. The experimental results showed LUKE performed the best on CoNLL-2003 dataset while SparkNLP slightly edges the BioBERT pre-trained model for BC5CDR dataset. The chapter concluded that nowadays pre-trained models can be easily downloaded to reach state-of-the-art results, and there is not enough support for further fine-tuning and pre-training. Hence, due to ease and reliability of fine-tuning, Google's BERT implementation is selected for the proposed text augmentation experiments.

3.1 Datasets

The following are some of the prominent datasets that are publicly available for the English language NER task.

3.1.1 CoNLL-2003

CoNLL-2003 [136] dataset was developed for language independent NER shared tasks, commonly known as CoNLL-2003 Shared Task. The dataset consists of 4 entities: person name, organization name, location and miscellaneous, spanned across 8 files covering English and German language. The English dataset contains news articles from Reuters Corpus while the German data set is made up of Frankfurter Rundschau articles. There is a training set, a development set, a test set, and an extensive file with non-annotated data for each language. Every word in the dataset has been placed on a separate line, and there is a blank line following each sentence.

The first object on every sentence is a word, following that is its POS tag; the third item is a syntactic chunk tag, and the last column denotes its named entity tag. The named entity labels have the form I-TYPE, indicating that the word is inside an expression of type TYPE. When two words of the identical class sequentially occur, the first word will be identified with a tag B-TYPE to record that it starts a different phrase. The tag O refers to the words that do not belong to the entity phrase. "train.txt", "valid.txt", and "test.txt" in the dataset have sentences accompanying their tags. For this experiment, only the named entity tags and the words along with their named entities are needed and extracted into an array. This tagging scheme is the IOB scheme.

TABLE 3.1: CoNLL-2003 Dataset Description

English Dataset	Articles	Sentences	Tokena	LOC	MISC	ORG	PER
Training Set	946	14987	203621	7140	3438	6321	6600
Development Set	216	3466	51362	1837	922	1341	1842
Test Set	231	3684	46435	1668	702	1661	1617

3.1.2 BC5CDR

Built by Li et al. at 2015 [172], the BC5CDR Disease (BC5-Disease) dataset contains three different collections of articles with chemicals and their relationships annotated, in the English language. There are 1500 PubMed articles in the BC5CDR corpus, with 4409 annotated chemicals,

5818 illnesses, and 3116 chemical-disease relationships. BC5CDR is a popular biomedical NER dataset.

3.2 Evaluauation of NER Systems

3.2.1 CoNLL-2003 Dataset

3.2.1.1 Language Understanding with Knowledge-based Embeddings (LUKE)

LUKE [173] is a pre-trained contextualized design of words and entities based on the transformer. It was presented in the research LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. It attains state-of-the-art outcomes on major NLP benchmarks, such as CoNLL-2003 (NER) and Open Entity (entity typing).

LUKE model proposes unique pre-trained contextualized descriptions of words and entities based on the bidirectional transformer[174]. Words and entities in a text are treated as separate tokens by the model to produce contextualized descriptions. The model is trained using the BERT pre-training tasks of Masked Language Model (MLM) with the entity-annotated Wikipedia corpus. The model will then predict both masked words and entities. Another contribution of LUKE is the introduction an entity-aware self-attention mechanism to recognize words or entity types during the attention scores estimation.

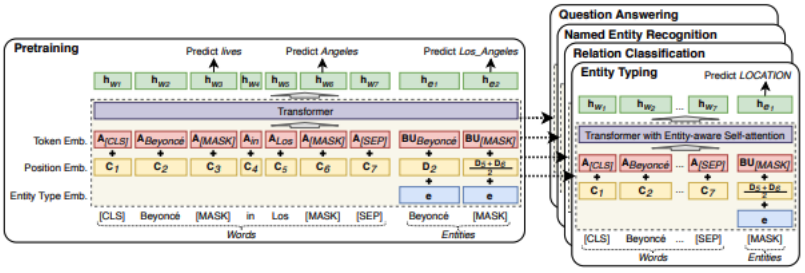

FIGURE 3.1: LUKE Architecture [173]

LUKE system as depicted in Figure 3.1 is modeled upon a multi-layer bidirectional transformer [174]. It converts words and entities from sentences as input tokens and calculates contextualized representation for every token. Formally, given a series of n terms w1, w2, ..., wn and k entities e1, e2, ..., ek, the model calculates M-dimensional word representations and entity representations. The entities can be Wikipedia entities or special entities.

Entity Aware Self Attention Mechanism

The core concept of transformer [174] is the self-attention mechanism which associates tokens with other dependent on the attention points among each tokens pair. For a stream of input vectors (x1, x2, x3, .., xN), the associated output vectors (y1, y2, y3, ..., yN) is calculated from the weighted aggregate of the transformed input vectors. Unique token (word or entity) identifier is assigned to each input and output vector.

Since LUKE manages two kinds of tokens (word-level and entity-level token), the model assumes that it is helpful to utilize the knowledge of target token types when calculating the attention scores. The model then enhances the mechanism by providing an entity-aware query technique that employs a unique query matrix for each conceivable pair of token types.

The computational expenses of the original tool and the stated tool are the same except for the added cost of calculating gradients and updating the parameters of the added query matrices at the training stage.

TABLE 3.2: Evaluation of LUKE NER model on CoNLL-2003 Dataset

LUKE CoNLL-2003	Precision	Recall	F1 Score	Support
LOC	0.9558	0.9478	0.9518	1666
MISC	0.8553	0.8688	0.862	701
ORG	0.9287	0.9496	0.9391	1647
PERSON	0.9683	0.9719	0.9701	1602
micro-average	0.9386	0.9453	0.942	5616
macro-average	0.927	0.9345	0.9307	5616
weighted-average	0.9389	0.9453	0.9421	5616

As shown in the Table 3.2, the downloaded LUKE NER pre-trained model is able to replicate state-of-the-art results (F1 0.93) reported on [175] with F1 score of 0.94 on CoNLL-2003 dataset.

3.2.1.2 Transformer-based Named Entity Recognition (T-NER)

T-NER [176] is another variation of transformer-based NER designed for fine-tuning NER Language Model. T-NER allows the research and analysis of the cross-domain and cross-lingual generalization capacity of Language Models fine-tuned on NER, and allows convenient implementation due to the support with Python library.

This paper shows the library's potential by integrating 9 public NER datasets (CoNLL, OntoNotes, WNUT, BC5CDR, etc) into a consolidated custom dataset, and estimating the cross-domain and cross-lingual results across the datasets. The outcomes from the initial analyses reveal that

in-domain outcomes are usually competitive across datasets. Nevertheless, it is challenging to train a model that generalizes well even with a large pre-trained language model, which can learn domain-specific peculiarities if fine-tuned on a consolidated dataset. An essential aim of this model was to build a self-contained, comprehensive arrangement to train, evaluate, and efficiently utilize NER models, not only for research objectives but also practical use cases in the industry.

TABLE 3.3: Evaluation Results of T-NER on CoNLL-2003 Dataset

T-NER	Precision	Recall	F1 Score	Support
LOC	0.91	0.83	0.87	1660
MISC	0.75	0.7	0.72	702
ORG	0.83	0.89	0.86	1661
PERSON	0.95	0.95	0.95	1610
micro-average	0.88	0.87	0.87	5633
macro-average	0.86	0.84	0.85	5633
weighted-average	0.88	0.87	0.87	5633

As seen from Table 3.3., F1 score of 0.87 showed that the downloaded T-NER pre-trained model was able to attain similar F1 score, 0.90, of T-NER reported in [176].

3.2.1.3 Flair Model + Cross-Weigh

Eugene Siow [177] used stacked embeddings (containing word and flair contextual embeddings) to train a flair model for NER. CoNLL-2003 train and dev set was used in addition to the CoNLL++ of CrossWeigh article [178] where test set was further cleaned. CrossWeight approach by Wang et al. [179] achieved the state-of-the-art results on this dataset with 94.3% F1 score. For the model evaluation, a model without pooled-embeddings, CrossWeigh, and training to a max 50 epochs was used to achieve 93.5% micro F1 score that is only 0.7% lower than the state-of-the-art result.

TABLE 3.4: Evaluation Results on CoNLL++ Dataset

	Precision	Recall	F1 Score
LOC	0.955	0.954	0.9547
MISC	0.8397	0.8479	0.8438
ORG	0.9119	0.9172	0.9145
PER	0.9826	0.9759	0.9792
Micro-Average			0.9354
Macro-Average			0.9231

Table 3.4 confirms that F1 score (0.93) of the downloaded pre-trained model is identical to the reported F1 score, 0.93 [175] .

3.2.2 BC5CDR Dataset

3.2.2.1 Spark NLP

The BiLSTM-CNN-Char structure is the deep neural network used in original Spark NLP for the task of NER, which significantly improved the similar approach introduced by [180]. The idea behind the hybrid B-LSTM and CNN approach is that the network is able to automatically learn word and character level features so the feature engineering steps could be eliminated. First, fixed-length vectors that represent characters features are fed to the CNN network. Word levels vectors are formed from the concatenation of characters vectors. Next, these word level features are fed into a forward and backward LSTM network, respectively. A final linear layer and a log-softmax layer map the output of each network at each time step into log probabilities for each tag class. The final result is calculated by adding these two vectors together [180]. In total, to extract word level features, 50-dimensions pre-trained word embeddings are used while 25-dimension character embeddings are used for character features extraction and representation. Additionally, external lexicon information is used following the recommendation of Ratinov and Roth [181].

In Spark NLP, the authors [182] have adjusted the structure as follows: Habibi et al., in 2017 [183], analyzed the performance of the LSTMCRF strategy on 33 datasets including 5 distinct entity types against SOTA NER models and CRF implementation. The results showed LSTM-CRF implementation with Wiki-PubMed-PMC word embedding gained 5% F1 score when compared against the baseline models on average. With the same NN architecture, a skip-gram based biomedical embedding (20 dimension and 2.2 millions vocabularies) was trained on PubMed articles to create a representation that can learn context [184]. The embedding was made available to use with the Spark NLP toolkit.

TABLE 3.5: Test Set Evaluation Results on BC5CDR Dataset

Spark NER	Precision	Recall	F1 Score	Support
B-Disease	0.84	0.86	0.85	960
I-Disease	0.82	0.88	0.85	1087
O	0.99	0.99	0.99	22450
Micro-Average	0.88	0.91	0.9	24497
Weighted Average	0.98	0.98	0.98	24497

As described in Table 3.5, the downloaded Spark NER model obtains a F1 score of 0.9. However, this is over-emphasized by the significantly higher number of O tags. Hence, a better F1 score

can be calculated by averaging B-Disease and I-Disease. Final F1 score of 0.85 indicates that the result is comparable to the reported F1 score, 0.89 [185].

3.2.2.2 BioBERT

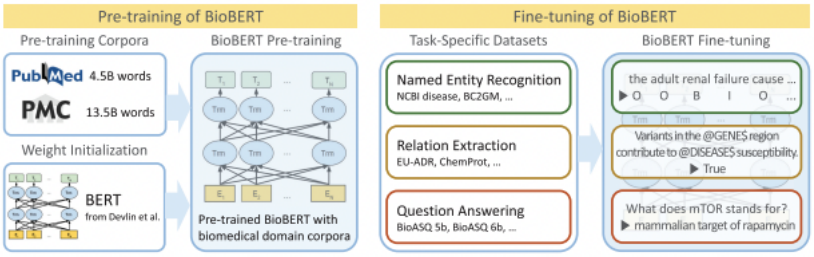

FIGURE 3.2: BioBERT Model [186].

The authors of the BioBERT model [186] explored the adaptation approach using the pre-trained language model BERT for biomedical corpora. They proposed BioBERT, a Biomedical BERT in which the BERT model is pre-trained on various biomedical corpora. The BioBERT model showed significant improvement over other state-of-the-art models and BERT when it is pre-trained on biomedical data for the biomedical text mining tasks such as 1) NER with 0.62% F1 score improvement, 2) connection Extraction with 2.80% F1 score gain, and 3) question answering with 12.24% MRR improvement. These outcomes indicated that the performance of NLP tasks can be improved when BERT model is pre-trained on related data such as biomedical data in this instance. The model by Eugene Siow [177] follows the model by Nooralahzadeh et al. [187]. This model is trained on top of the BioBERT model as a base and uses Simple Transformers library and has an F1 score of 0.893, which is slightly worse (difference of 0.19) than the current state-of-the-art model in this dataset.

TABLE 3.6: Test Set Evaluation Results on BC5CDR Dataset

Spark NER	Precision	Recall	F1 Score	Support
B-Disease	0.88	0.91	0.90	960
I-Disease	0.87	0.89	0.89	1087
O	0.99	0.99	0.99	22450
Micro-Average	0.88	0.91	0.9	24497
Weighted Average	0.877	0.90	0.89	24497

The model's F1 scores on the BC5CDR dataset is 0.89 which is comparable to the reported F1 score, 0.92, of BioBERT V1.0 model [186].

3.3 Summary

This chapter introduced 2 commonly used datasets namely CoNLL-2003 and BC5CDR for NER tasks. CoNLL-2003 was benchmarked with 3 NER pre-trained models, LUKE T-NER and Flair whereas BC5CDR dataset was evaluated with Spark NLP and BioBERT pre-trained models. For CoNLL-2003 dataset, the benchmarks indicated that all 3 models were comparable, with LUKE obtaining the higher F1 score (0.94), followed by Flair Cross Weight model (F1 0.93) and T-NER (F1 0.90). As for BC5CDR dataset, the experiments showed Spark NLP, a fine-tuned BERT-base model, edged the performance of BioBERT with F1 score of 0.98. The learning point is that the experiments were able to be carried out due to the accessibility of the datasets and pre-trained models on paperswithcode.com where the site documented the codebase for all available models under each dataset. However, the models could not be easily pre-trained or fine-tuned for further experiments due to the lack of support and documentation. The BioBERT paper demonstrated that the baseline BERT performance could be improved by either pre-training or fine-tuning and Google's BERT is well-documented for NLP tasks. Hence, we concluded that the proposed augmentation experiments will be carried out by fine-tuning several BERT models.

Chapter 4

Text Augmentation with Finite State Transducers and Abstractive Text Summarization

In this chapter, we propose two text augmentation approaches as shown in Figure 4.5 : 1) text generation using keyword replacement by FST, and 2) paraphrasing approach using the Pegasus abstractive summarization model. In Section 4.1, we describe the proposed text augmentation technique using FST. In Section 4.2, we present the text augmentation framework with Google's Pegasus model. In Section 4.3, we describe the experimental benchmarks of our augmentation pipeline on SOTA transformer based NER architectures namely - BERT, A Robustly Optimized BERT Pretraining Approach (RoBERTa) [188] and A distilled version of BERT (DistilBERT)[189]. OpenFST and the Thrax compiler were used for implementing the FST grammars, while PyTorch and Simple Transformer APIs from HuggingFace were used for implementing Pegasus and NER models. Lastly, Section 4.4 summarizes this chapter.

4.1 Text Augmentation with Finite State Transducers

Finite automata have been used in many fields of computational linguistics [190]. Their use can be demonstrated through both linguistic and computational parameters.

[0]The model used in this chapter has been accepted to APSIPA ASC 2022 (Submission 1570839765, Full Paper Track).

From a linguistic point of view, the application of automata is convenient because it allows one to describe the most relevant local phenomena encountered in empirical language researche. They also result in linguists' presenting lexical rules or compact presentations of idioms and cliches. There are graphical tools to visualize and modify automata, which aid in correcting and completing grammars[191]. Other phenomena like parsing context-free grammars can also be analyzed using finite state machines, as the underlying mechanics of most methods used in parsing are related to automata.

The use of finite state machines in computation is primarily motivated by time and space efficiency considerations. Time efficiency is usually achieved by deterministic automata where the input size linearly determines the computation time for the output machine. Space efficiency is achieved using the classical minimization algorithm of deterministic automata. Compiler construction is one of the applications that show the efficiency of deterministic finite automata in practice[192].

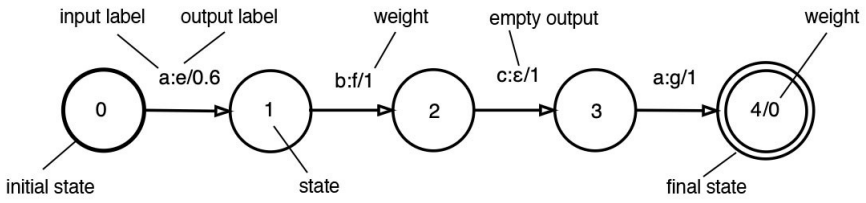

FIGURE 4.1: Weight Finite State Transducer (WFST).
Source: [193]

FST is a finite state automaton that both produces outputs and reads inputs, which is suitable for parsing[194]. FSTs consist of a finite number of states connected by transitions labeled with input/output pairs. A FST starts from a beginning state, jumps to different states depending on inputs, and produces outputs according to its transition table. Sequential string-to-string converters are commonly used in various fields of natural language processing. Both determination and minimization algorithms have been defined for the p-subsequential converter class, which includes sequential string-to-string converters.

Our Approach with FST. We generate grammars using FST and use FST as a keyword replacement tool. To generate the templates, we partition the NER datasets into training and testing. From the training data, the segments belonging to the required NER tags must be labeled. These segments become the templates for the FST by replacing the words within the

Original Template - *I came from VAR-LOC on VAR-DAY*

Keywords belonging to each Variable

VAR-LOC	VAR-DAY
India	Monday
Singapore	Tuesday

Sentences generated

I came from India on Monday
I came from Singapore on Monday
I came from India on Tuesday
I came from Singapore on Tuesday

FIGURE 4.2: FST generation example

NER tags with variables which can be modified by the FST text generator with a new list of candidates for the tag.

For example, the sentence *"I live in the United States of America"* would be tagged as *"I live in the **[LOC]** United **[/LOC]** **[LOC]** States **[/LOC]** **[LOC]** of **[/LOC]** **[LOC]** America **[/LOC]**"*. To generate the template, we first combine the multi-word entity into *"I live in the **[LOC]** United States of America **[/LOC]**"*. Hence, the template for this sentence is *"I live in **VAR-LOC**"*. We conduct this word-to-variable conversion for all the training sentences to obtain the corresponding FST templates. To generate augmented data, we apply the Thrax compiler [195] and provide a list of new target words to replace, e.g, VAR-LOC = INDIA, SINGAPORE, AUSTRALIA, COUNTRY-NAME. to replace each variable in each template. As shown in Figure 4.2, from one template with VAR-LOC and VAR-DAY list with two entity terms ("India", "Singapore" and "Monday", "Tuesday" respectively), we could generate four extra data points.

While FST is a highly optimized and undemanding approach computationally, the generated augmentations vary little from the original template. At its core, it has been utilized as a keyword replacement technique. Hence, it performs well when evaluated against the ROGUE metric, but such data can often lead to model overfitting due to highly homogeneous training data. Thus, the model might fail to perform well on unseen test data. For this reason, we look

into Pegasus as an additional augmentation technique that enables us to get variation in the sentence structure of the generated sentences.

4.2 Text Augmentation with Abstractive Text Summarization, Pegasus

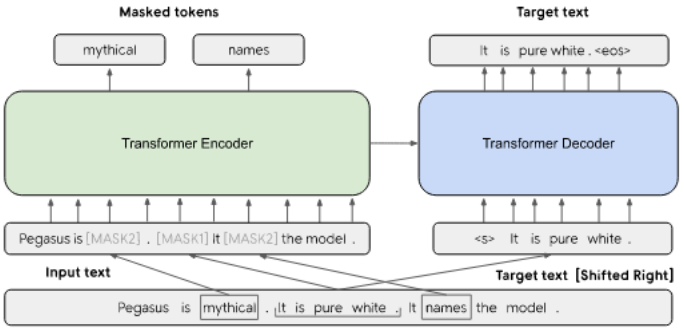

FIGURE 4.3: Pegasus Architecture.
Source: [196]

The Google Pegasus model was released in 2020 and it can be applied for the task of abstractive text summarization [197]. Although it was not built for the task of text generation, by forcing Pegasus to abstract from a short sentence, the model paraphrases the text to a semantically similar sentence. Hence, we 'trick' Pegasus to become a one-to-one sentence text generator where the input and output sentence are semantically similar.

Abstract text summarization is another challenging task in natural language processing, as from the given long text, the model needs to understand the context, compress the information and produce a succinct text [198]. The main paradigm for training machine learning models is sequence-to-sequence learning (seq2seq), where the input sequences are mapped to the output sequences as the neural network learns the pattern [199]. Recent transformer-encoder-decoder models have gained favour because they model long-sequence dependencies more efficiently when summarizing[200] [201].

The Pegasus model is a Transformer encoder-decoder model trained with the objective to improve abstract summarization task[197]. Transformer models combined with self-supervised pre-training have shown powerful language learning abilities. When fine-tuned, this framework

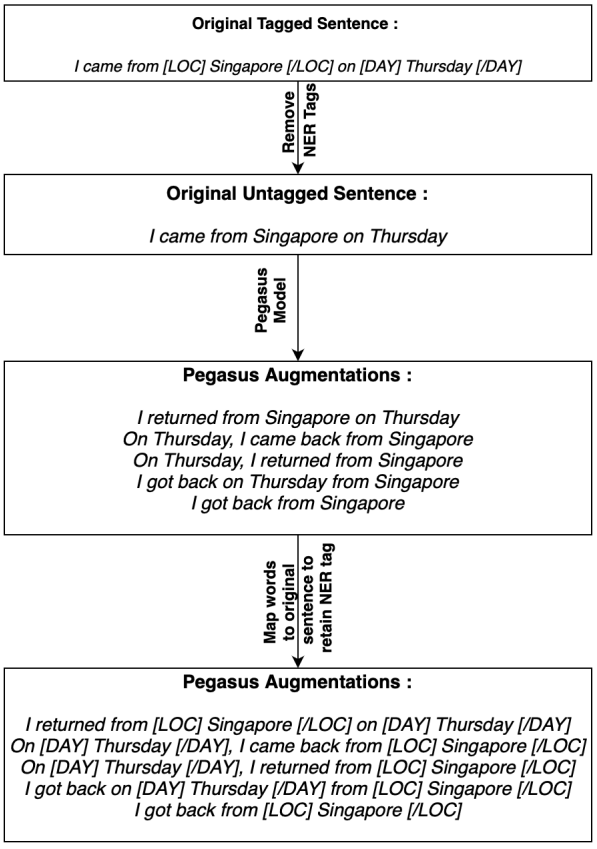

FIGURE 4.4: Pegasus text generation example

achieves optimal performances across different language tasks[202] [203]. In the Pegasus's pre-training process, the task is to recover missing sentences in a document after deliberate sentence removals. The pre-trained example input is a document containing missing sentences, and the output consists of concatenated missing sentences. The Rogue metric was used to detect the most important sentence with respect to the rest of the paragraph followed by subsequent removal and prediction in the training phase. Even with just 1,000 fine-tuning samples, it was able to perform better than baseline encoder-decoder networks which immensely increases the usefulness of text summarization models as it significantly lowers the scale and cost of supervised data collection.

Our Approach with Pegasus. The Pegasus model is used to create alternatives from given sentences. The following lists the steps taken to produce alternate sentences using Pegasus.

1. First, we take the original sentences from the training data without NER tags. Given the original sentence *"I came from **[LOC]** Singapore **[/LOC]** on **[DAY]** Thursday **[/DAY]**"*, these NER tags are removed such that the processed sentence becomes *"I came from*

Singapore on Thursday". This sentence is then passed through the Pegasus model to generate possible alternatives.

2. We can set the number of alternatives for Pegasus to produce. In our case, we take only the top 5 alternatives. In this example, Pegasus returned:

   (a) "I returned from Singapore on Thursday"

   (b) "On Thursday, I came back from Singapore"

   (c) "On Thursday, I returned from Singapore"

   (d) "I got back on Thursday from Singapore"

   (e) "I got back from Singapore"

3. To create text augmentation for NER, we have to process these paraphrased sentences to recover the relevant NER tags. To perform this, we simply locate the terms/words in the original text which are NER, and find them in corresponding alternatives. For instance, we are trying to find "Singapore" and "Thursday" in the alternatives. Note that it is possible that some NER terms may be removed, i.e. in the example below item (4.e), "Thursday" was removed in the generated sentence.

4. The following lists the recovered alternative sentences with NER tags:

   (a) "I returned from [**LOC**] Singapore [**/LOC**] on [**DAY**] Thursday [**/DAY**]"

   (b) "On [**DAY**] Thursday [**/DAY**], I came back from [**LOC**] Singapore [**/LOC**]"

   (c) "On [**DAY**] Thursday [**/DAY**], I returned from [**LOC**] Singapore [**/LOC**]"

   (d) "I got back on [**DAY**] Thursday [**/DAY**] from [**LOC**] Singapore [**/LOC**]"

   (e) "I got back from [LOC] Singapore [/LOC]"

As depicted in Figure 4.5, our proposed Text Augmentation pipeline consolidates both Text augmentation approaches: Finite State Transducers and Abstractive Text Summarization. The original data with NER tags are fed to 1)FST pipeline to generate new sentences with the same grammatical structures but with similar vocab of the same entity type (i.e. I travelled from "Singapore" to I travelled from "India") and 2)Pegasus pipeline to generate sentences with the same entity terms but with different grammatical structures (i.e. I returned from "Singapore" on "Thursday" to On "Thursday", I came back from "Singapore"). The final data is the combined outputs which will be used as training data to carry out NER experiment to verify whether the augmented text improves NER performance over original data.

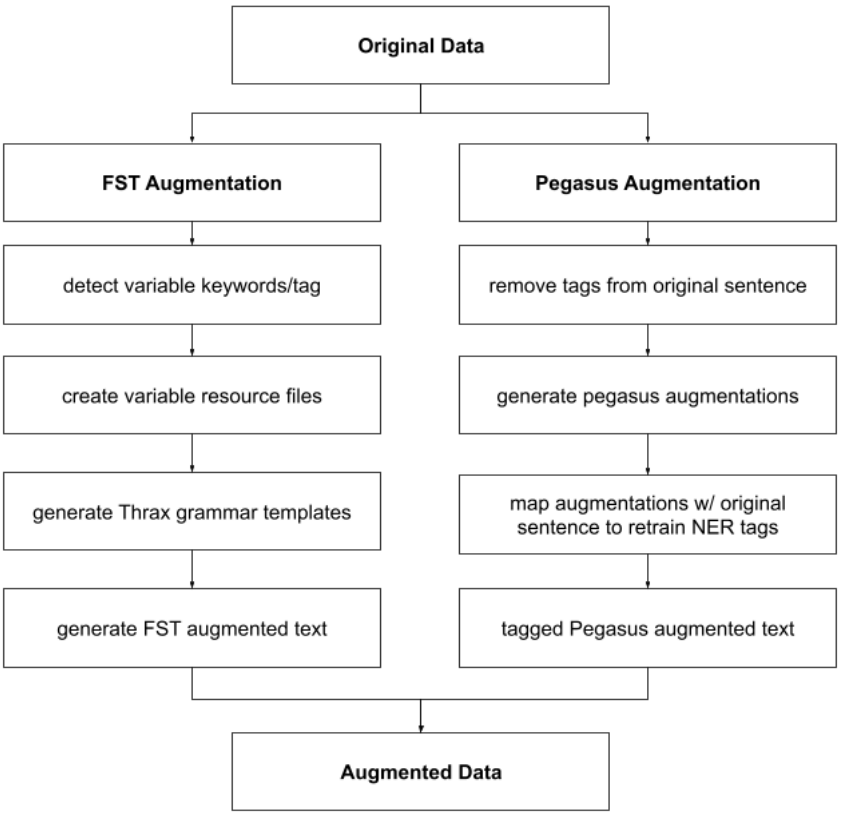

FIGURE 4.5: Proposed Text Augmentation Pipeline

4.3 Experiments

In this section, we discuss the Groningen Meaning Bank (GMB) dataset, NER models and the configurations used to evaluate the proposed text augmentation framework, followed by the analysis on the experimental results.

4.3.1 Datasets

The **GMB** can be downloaded from https://gmb.let.rug.nl/data.php. Currently, there are 5 versions of GMB corpus released [204]. Each release has a different number of datapoints, for example, in ver 2.1.0 (year 2013), the number of datapoints is close to 1 million tokens, with 47 thousand sentences. We will use this version in our experiment.

The GMB has 8 NER tags, as shown in Table 4.1: Geographical location (geo), Organization (org), Person (per), Time (tim), Geopolitical location (gpe), Artifact (art), Natural phenomenon (nat), Event (eve). The text data comes in both raw and tokenized format which consists of POS

	Terms Frequency		
Entity	**Original Train set**	**Augmented Train set**	**Validation set**
Geographical (geo)	36098	46600	8960
Organization (org)	29375	38103	7552
Person (per)	27590	32972	6651
Time (tim)	21448	26801	5413
Geopolitical entity (gpe)	12786	17366	3283
Artifact (art)	574	1363	125
Event (eve)	471	815	90
Natural Phenomenon (nat)	192	358	60
Total	128534	164378	32134

TABLE 4.1: Groningen dataset Tags Data

tags for POS tagging tasks, named entities tags following IOB scheme, and word representation formats. The corpus contains a significant number of semantically annotated English texts with extensive semantic representations and follows a modified Sekines Extended Named Entities. With its rich annotation, this dataset has been widely used for NER and POS tasks.

The tagging scheme for the NER uses the BIO scheme. Specifically, each entity label is prefixed with either letter B or I. For example, B-org and I-org denotes the beginning and ending of an organization entity. The O tags are for words which are not of interest as NER. There are about 1 million data points in the dataset. Sentence number, word, and POS are the features available. The tag column (target column) classifies the corresponding word in a sentence into available tags. There are 47,959 unique sentences and 35,179 unique words in the dataset.

4.3.2 NER Models Architecture and Configuration

NER Models Architecture

3 BERT variant models are pre-trained on the NER task to find the best baseline model for the respective dataset. The 3 chosen models are BERT[75], DistilBERT [189], and RoBERTa [188].The main differences between these models are the sizes of the model and the data the model was previously trained on.

BERT is based on Transformers and its acronym stands for Bidirectional Encoder Representations from Transformers. It is a deep learning model with an attention mechanism and has been widely used for many NLP tasks. Basically, it is applied to convert words to word embeddings.

TABLE 4.2: BERT Model Variations

Model Architecture	**BERT** (bert-base-uncased)	**RoBERTa** (roberta-base)	**DistilBERT** (distil-base-uncased)
Model Details	12 layer 768 hidden 12-heads 110M parameters	12 layer 768 hidden 12-heads 125M parameters	6 layer 768 hidden 12-heads 66M parameters
Model Datasets	Pretrained on English Wikipedia and Bookscorpus	Pretrained on 160GB BERT 16GB BERT data 144GB additional data	Pretrained on English Wikipedia and Bookscorpus

For our NER task, these embeddings are fed into a 12-layers Transformer as shown in Figure 4.6 and the final output is generated via softmax into the 8 target NER classes for the GMB dataset.

The DistilBERT model is a light-weight model of BERT. Distillation is a technique used to train a small model from a large one. Specifically, the DistilBERT is trained to reproduce the output distribution of BERT but using a smaller model, with only 6 layers with 66 million parameters as opposed to BERT's original 12 layers with 110 million parameters.

RoBERTa, on the other hand, is a larger version of BERT. Its acronym stands for "Robustly Optimized version of BERT". During training, Roberta is trained with a larger batch size and much larger training dataset (10 times the size of BERT). However, its parameter size is almost the same (only 10% larger than BERT).

NER Models Configuration

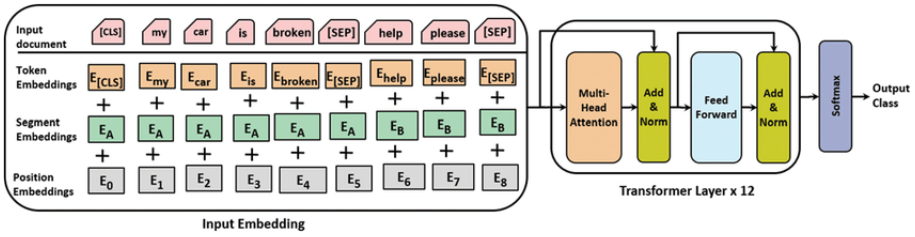

FIGURE 4.6: BERT Base classifier Architecture [205]

Word level embeddings, positional embeddings and segment level embeddings are aggregated into the deep bidirectional layers of the BERT model. The output of the model is the BERT representation, which is the hidden states of every token of the input. Then, the output is fed into a linear layer classifier to determine the entity of the input token. Fine-tuning is done by feeding newly generated sentences on the already pre-trained weights of the BERT model.

The entire architecture is shown in Figure 4.6 where there is an Input Embedding layer, 12 Transformer layers and a Linear Classification layer.

Our initial BERT, DistilBERT and RoBERTa models were retrieved from HuggingFace. To compare the effectiveness of our data augmentation, these models are fine-tuned using our training data (with and without text augmentation). The model was trained and evaluated using the Trainer API, which contained pre-defined training and validation batch sizes, learning rates, and epoch counts. In consideration of the BERT authors' advice, a batch size of 8 was trained across three epochs with a learning rate of $2x10^{-5}$.

4.3.3 Experimental Results and Discussion

The goals of the experiments are:

1. To verify if our fine-tune BERT model is comparable to the existing published BERT NER implementation on the GMB dataset.
2. To evaluate if text augmentation via FST and Pegasus are useful to improve BERT NER training.

The public GMB dataset is fine-tuned to the 3 different BERT Variant models. The dataset has also been augmented using Pegasus and added to the original training set to be fed into the NER models. The performance metric used to evaluate the NER model is the macro-average F-score. Macro-average F-score is calculated by individually calculating the precision and recall of every entity type, and averaging the scores.

TABLE 4.3: Baseline Benchmarks

Bert NER Model	**Accuracy**	**F1**
Nvidia BERT	96.8	74.2
bert-based-uncased	96.7	73.4

4.3.4 Published BERT model's result on GMB

The published result of NVIDIA BERT model [206] is shown in row 1 of Table 4.3. They achieved an accuracy of 96.8% with an F1-score of 74.2. To verify if our Hugging Face BERT model is acceptable, we fine-tune this model with the GMB training data and early stopping criterion.

The best model found was with 3 epochs of fine-tuning. The resultant fine-tuned model achieved 96.7% Accuracy and 73.4 F1-score in row 2 of Table 4.3. The results verify that the Hugging Face BERT model fined tuned with original training data is comparable to the NVIDIA BERT model and will be utilized for our augmentation experiments.

4.3.5 Fine-Tuning with GMB Original+Augmentation Data

To evaluate the effectiveness of size of augmentation data, we split the GMB corpus into 80/20 split to partition the corpus. Augmentation with FST Thrax compiler and Pegasus was only conducted on the training dataset where test data was left untouched. We experimented with different sizes of augmentation data, ranging from 2,000, 4,000, 8,000, and 10,000 sentences. The experimental results showed that Accuracy saturates from 8,000 sentences. We report the results using 10,000 sentence augmentation in row 3 in Table 4.4.

4.3.6 Discussion of results

Among the BERT variant models, as depicted in row 2 of Table 4.4, RoBERTa performs better than the rest of the BERT variant models on Accuracy and F1 score when fine-tuned with original training data. The 2% improvement of F1 score from 0.742 to 0.761 can be observed when using RoBERTa on the GMB dataset. This range corresponds to the results of the RoBERTa article, which showed that it outperformed BERT and other transformer models by about 2% to 20% on similar NLP datasets [188].

For the experiment where BERT models were fine-tuned with original and augmentation data, as shown in Table 4.4, all BERT variants except RoBERTa model achieved an overall higher Accuracy and F1-score over baseline HuggingFace models fine-tuned on original training data. DistilBERT out of all the BERT variants had the highest improvement with F1 score of 74.1 outperforming baseline HuggingFace model by 0.7%. The BERT model showed second best improvement for the GMB augmented experiment, with a macro-average F1-score of 73.7. These results align with the expectation as DistlBERT is originally pre-trained with the fewest amount of data and the extra augmentation data is able to enhance the model's generalization. RoBERTa ,on the other hand, saw a 0.2% F1-score performance drop with the augmentation fine-tuning when compared to baseline HuggingFace RoBERTa model.

Model	Description	BERT Models	Accuracy	F1 score
Nvidia	Reported NVIDIA BERT model on GMB	NVIDIA BERT	96.8	74.2
Baseline Hugging Face	HuggingFace model Fine-tuned on Training data	BERT DistilBERT RoBERTa	96.7 96.7 97.2	73.4 73.4 76.1
Our Augmentation Approach	HuggingFace model Fine-tuning on Training + Augmented data	BERT DistilBERT RoBERTa	96.7 96.7 97.1	73.7 (+0.3%) 74.1 (+0.7%) 75.9 (-0.2%)

TABLE 4.4: BERT NER results on Augmented vs Baseline datasets

Data augmentation improved DistilBERT and BERT models performance albeit marginally but there was no improvement for the RoBERTa pre-trained model. One possible reason is that the sample size of augmented data is relatively small for RoBERTa model to capture better representation. According to Longpre et. al [207], their results have also shown that augmentation only improved the BERT model while RoBERTa did not obtain any benefits. Their analysis for this finding was due to the scale of pre-training size, and the augmentation will only benefit the model when it provides linguistic patterns that have not been seen during pre-training. Another explanation is that contextual embeddings learnt linguistic patterns from pre-training data boost performance on NLP tasks with unseen data [208]. As a result, we assumed that the data augmentation will only improve NER tasks when it introduces new linguistic patterns. This also further explains how BERT and DistilBERT showed the highest improvement with augmentation as the dataset they have been pre-trained on is significantly smaller than RoBERTa's dataset shown in Table 4.2. Fine-tuning of large scale models such as RoBERTa would be more appropriate for domains with complex linguistic patterns and uncommon lexicons such as biomedical settings as explored by [209] which achieved best results for chinese medical NER.

4.4 Summary

In this section, a novel text augmentation pipeline using two augmentation approaches was proposed to improve NER tasks on low-resource domains. This pipeline leverages thrax-FST and Pegasus text summarizing toolkits to perform word replacements and sentence level paraphrasing to generate more data-points. The resulting data-points will then be used as additional training data to enrich the NER model.

In order to verify whether this text augmentation pipeline improves the NER results, namely F1-score and Accuracy, a GMB dataset was used and 3 experiments were conducted. Firstly, we benchmarked our BERT-based NER model implementation against NVIDIA NER implementation on GMB dataset to study the model discrepancy. The benchmarks established that our selected NER model is comparable on Accuracy and F1 score to NVIDIA fine-tuned NER implementation, attaining 73.4 F1 score. In the second experiment, the original GMB dataset was fine-tuned and tested on 3 BERT variant models where the RoBERTa model performed better on Accuracy and F1-score than BERT and DistilBERT. In the final experiment, the combination of original and augmented datasets were fine-tuned and tested on the same 3 BERT variant models. The experiment showed that models fine-tuned with augmented dataset achieved an F1-score on BERT and DistillBERT by 73.7 and 74.1 F1-score respectively outperforming the baseline result by 0.3% and 0.7% while the augmentation degraded on RoBERTa with 0.2% lower F1 score.

To summarise, this proposed text augmentation pipeline improved NER performance when fine-tuned on the relatively smaller NER model such as BERT and DistilBERT but performed poorly on a large BERT model,RoBERTa, that is pre-trained with ten times more data. Although data augmentation can be useful, it is also model dependent.

Chapter 5

Conclusion and Future Work

5.1 Contribution

NER models reached a breakthrough in the performance due to the introduction of neural networks and deep learning models. However, training such models requires vast amounts of data and it is challenging for low-resource social or medical domains with frequent emergence of new terms. Data augmentation is one of the techniques to combat such issues and hence, in this thesis, we aimed to improve NER performance on low-resource domains via Text Augmentation. To overcome the data scarcity problem, we proposed a novel text augmentation pipeline that leverages on Finite State Transducer and Pegasus Abstractive Text Summarization.

This thesis initially reviews and verifies the performance of the state-of-the-art models for 2 popular datasets for English news stories and biomedical domain, CoNLL-2003 and BC5CDR respectively. Out of the 5 state-of-the-art models (LUKE, T-NER, Flair Model + Cross-Weight, Spark NLP, BioBERT), LUKE transformer-based model obtains the highest F1-score of 0.9421 on English CoNLL dataset while BiLSTM-CNN based Spark NLP achieves the highest Precision, Recall and F1-Score on BC5CDR dataset. However, the datasets were not representative of low-resource datasets and hence, were not considered in Text Augmentation experiments. The benchmarking results showed that BERT showed improved performance when pre-trained or fine-tuned on the target dataset. Hence, we selected BERT implementation for text augmentation experiments.

FST text augmentation. In this work, we proposed text augmentation via the entity word replacement. Particularly, the method extracted distinct terms associated with each entity type

from the original NER dataset. Given a sentence with NER tags, random sentence permutations were generated by replacing word(s) of Entity A with other words that belonged in Entity A.

Pegasus text augmentation. By leveraging Pegasus Text Summarization toolkit, NER sentences without the tags were pushed to perform sentence level paraphrasing to generate more data-points. The resulting data-points will then be used as additional training data to enrich the NER model.

Augmented NER result. In order to verify our hypothesis, we compared the NER performance on combined original and augmented dataset against original only dataset. On the GMB dataset, augmented dataset achieved overall higher F1 score on BERT and DistillBERT by 73.4 and 74.1 respectively while its performance degraded on RoBERTa with 0.2 lower F1-score when compared to the baseline system fine-tuned only on the original training set. Our conclusion is that the performance of our proposed text augmentation is model dependent as it showed improvement on smaller pre-trained models such as BERT and DistilBERT but not on the relatively large RoBERTa model.

5.2 Future Work

For the future work, we planned two ways to improve the quality of augmentation. Firstly, the proposed augmentation with Pegasus is not robust enough to re-tag the name entities on the generated text when the candidate terms from the original text are drastically changed. This is commonly known as token-label misalignment issue where the method substitutes entities with an augmentation that mismatch the source label. One instance is where the entity terms are changed to another synonymous word(s) ("new zealanders"[GPE] to "kiwis"). Phrase reconstruction including entity terms is another instance that causes the re-tagging issue. For example, "the citizen of London[LOC]" could change to "Londoners"[GPE] in which the re-tagging module need to be smart enough to change entity labels from LOC to GPE. In order to solve this issue, we plan to incorporate POS tagger and Dependency parser to first identity POS tags of the entity terms, reapply them on same POS tags of the generated text if original terms is not found. Second approach is to include relevant data from cross-domains. The proposed text augmentation pipeline relies heavily on the variety of entity terms in the corpus. The more the distinct entity values, the better the quality of the augmented data. For instance, given the data points in the corpus with N distinct sentences with 1 Entity type (e.g. "Country") containing M possibles

lexicons, by applying the proposed FST augmented approach, we could generate N * M sentences. Increasing N would require manual effort to reconstruct new semantically sentence. Logically, it is more efficient to increase M by sourcing other possible lexicon values of Entity type (e.g. "Country") from out-domain NER corpora. However, the extra consideration must be applied when sourcing cross-domain entities that can be homonymous. For instance, the candidate terms of "Product" entity in Finance domain would differ from that of Medical domain.

Bibliography

[1] S. Coates-Stephens, "The analysis and acquisition of proper names for the understanding of free text," *Computers and the Humanities*, vol. 26, no. 5/6, pp. 441–456, 1992, ISSN: 00104817. [Online]. Available: `http://www.jstor.org/stable/30204635` (visited on 05/19/2022).

[2] X. Wang, Y. Jiang, N. Bach, *et al.*, "Automated Concatenation of Embeddings for Structured Prediction," in *the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (**ACL-IJCNLP 2021**)*, Association for Computational Linguistics, Aug. 2021.

[3] C. P. Samuel R. Bowman Gabor Angeli and C. D. Manning, "A large annotated corpus for learning natural language inference," *EMNLP*, 632–642, 2015.

[4] B. H. Xiang Dai Sarvnaz Karimi and C. Paris, "Cost-effective selection of pretraining data: A case study of pretraining bert on social media," *arXiv*, 2020.

[5] S. Ruder, "Neural transfer learning for natural language processing," Ph.D. dissertation, National University of Ireland, Galway, 2019.

[6] J. Wang and L. Perez, "The effectiveness of data augmentation in image classification using deep learning," *CoRR*, 2017.

[7] J. Wei and K. Zou, "Eda: Easy data augmentation techniques for boosting performance on text classification tasks," *EMNLP-IJCNLP*, 6382–6388, 2019.

[8] Z. D. Xinyi Wang Hieu Pham and G. Neubig, "Switchout: An efficient data augmentation algorithm for neural machine translation," *EMNLP*, 856–861, 2018.

[9] D. D. E. P. Junghyun Min R. Thomas McCoy and T. Linzen, "Syntactic data augmentation increases robustness to inference heuristics.," *ACL*, 2339–2352, 2020.

[10] J. Z. Xiang Zhang and Y. LeCun, "Character-level convolutional networks for text classification," *NIPS*, 649–657, 2015.

[11] J. Wei and K. Zou, "Eda: Easy data augmentation techniques for boosting performance on text classification tasks," *EMNLP-IJCNLP*, 6382–6388, 2019.

[12] G. G. Sahin and M. Steedman, "Data augmentation via dependency tree morphing for low-resource languages," *EMNLP*, 5004–5009, 2018.

[13] Y. S. Kang Min Yoo and S. goo Lee, "Data augmentation for spoken language understanding via joint variational generation," *AAAI*, 2019.

[14] M.-T. L. R. Z. K. C. M. N. Adams Wei Yu David Dohan and Q. V. Le, "Qanet: Combining local convolution with global self-attention for reading comprehension," *ICLR*, 2018.

[15] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning–based text classification: A comprehensive review," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–40, 2021.

[16] N. D and S. S, "A survey of named entity recognition and classification," *Lingvist. Investig*, vol. 30, no. 1, 3–26, 2007.

[17] Kim, Ji-Hwan, and P. C. Woodland, "A rule-based named entity recognition system for speech input.," *Sixth International Conference on Spoken Language Processing.*, 2000.

[18] A. C, H. L, H. T, and R.-S. M, "Sra: Description of the ie2 system used for muc-7," *MUC-7*, 1998.

[19] H. K, G. R, A. S, *et al.*, *University of Sheffield: Description of the lasie-ii system as used for muc-7,*, 1998.

[20] A. E. D, H. R. J, B. J, *et al.*, "Sri international fastus system: Muc-6 test results and analysis," *MUC-6*, 237–248, 1995.

[21] B. J. W, R. F, and M. D, "Facile: Description of the ner system used for muc-7," *MUC-7*, 1998.

[22] M. A, M. M, and G. C, "Named entity recognition without gazetteers," *EACL*, 1–8, 1999.

[23] K. G and I. K, "Description of the netowl extractor system as used for muc-7," *MUC-7*, 21–28, 2005.

[24] R. Grishman, "The nyu system for muc-6 or wheres the syntax?," 1995. DOI: 10.21236/ada460232.

[25] Kim, Ji-Hwan, and P. C. Woodland, "A rule-based named entity recognition system for speech input.," in *Sixth International Conference on Spoken Language Processing.*, 2000.

[26] H. D, F. K, M. H-T, Z. R, and F. J, "Prominer: Rule-based protein and gene entity recognition," *BMC Bioinform*, 14th ser., vol. 6, no. 1, 2005.

[27] Q. P. A, M. S. A, R. G. A, *et al.*, "Named entity recognition over electronic health records through a combined dictionary-based approach," *Procedia Comput. Sci*, vol. 100, 55–61, 2016.

[28] Waltl, Bernhard, Georg, Bonczek, and F. Matthes, "Rule-based information extraction: Advantages, limitations, and perspectives," *Jusletter IT*, vol. 2, 2018.

[29] M. Marrero, J. Urbano, S. Sánchez-Cuadrado, J. Morato, and J. M. Gómez-Berbís, "Named entity recognition: Fallacies, challenges and opportunities," *Computer Standards Interfaces*, vol. 35, no. 5, 482–489, 2013. DOI: 10.1016/j.csi.2012.09.004.

[30] S. S and N. C, "Definition, dictionaries and tagger for extended named entity hierarchy.," *Proceedings of the language resources and evaluation conference (LREC)*, 1997–1980, 2004.

[31] L. Rau, "Extracting company names from text," *[1991] Proceedings. The Seventh IEEE Conference on Artificial Intelligence Application*, DOI: 10.1109/caia.1991.120841.

[32] S. Coates-Stephens, "The analysis and acquisition of proper names for the understanding of free text," *Computers and the Humanities*, vol. 26, no. 5-6, 441–456, 1992. DOI: 10.1007/bf00136985.

[33] C. Fellbaum, "Wordnet," *Theory and Applications of Ontology: Computer Applications*, 231–243, 2010. DOI: 10.1007/978-90-481-8847-5_10.

[34] M. Collins and Y. Singer, "Unsupervised models for named entity classification," *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora. 1999.*, 1999.

[35] E. Alfonseca and S. Manandhar, "An unsupervised method for general named entity recognition and automated concept discovery," *Proceedings of the 1st international conference on general WordNet, Mysore, India.*, 2002.

[36] A. E, A. O, H. E, and M. D, "Enriching very large ontologies using the www.," *Proceedings of the Ontology Learning Workshop, ECAI*, 2000.

[37] D. Nadeau, P. D. Turney, and S. Matwin, "Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity," *Advances in Artificial Intelligence Lecture Notes in Computer Science*, 266–277, 2006. DOI: 10.1007/11766247_23.

[38] O. Etzioni, M. Cafarella, D. Downey, *et al.*, "Unsupervised named-entity extraction from the web: An experimental study," *Artificial Intelligence*, vol. 165, no. 1, 91–134, 2005. DOI: 10.1016/j.artint.2005.03.001.

[39] N. Chinchor, "Muc-3 evaluation metrics," *Proceedings of the 3rd conference on Message understanding - MUC3 91*, 1991. DOI: 10.3115/1071958.1071961.

[40] S. Zhang and N. Elhadad, "Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts," *Journal of Biomedical Informatics*, vol. 46, no. 6, 1088–1098, 2013. DOI: 10.1016/j.jbi.2013.08.004.

[41] Y. Shinyama and S. Sekine, "Named entity discovery using comparable news articles," *Proceedings of the 20th international conference on Computational Linguistics - COLING 04*, 2004. DOI: 10.3115/1220355.1220477.

[42] V. Yadav and S. B. Bethard, "A survey on recent advances in named entity recognition from deep learning models.," 2019. [Online]. Available: arXivpreprintarXiv:1910.11470.

[43] 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*, 1–1, 2020. DOI: 10.1109/tkde.2020.2981314.

[44] D. M. Bikel, R. Schwartz, and R. M. Weischedel, "An algorithm that learns what's in a name.," *Machine Learning*, vol. 34, no. 1/3, 211–231, 1999. DOI: 10.1023/a:1007558221122.

[45] G. Zhou and J. Su, "Named entity recognition using an hmm-based chunk tagger," *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL 02*, 2001. DOI: 10.3115/1073083.1073163.

[46] Borthwick and A. Eliot, "A maximum entropy approach to named entity recognition.," 1999.

[47] J. R. Curran and S. Clark, "Language independent ner using a maximum entropy tagger," *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003* -, 2003. DOI: 10.3115/1119176.1119200.

[48] R. Malouf, "A comparison of algorithms for maximum entropy parameter estimation," *proceeding of the 6th conference on Natural language learning - COLING-02*, 2002. DOI: 10.3115/1118853.1118871.

[49] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, 273–297, 1995. DOI: 10.1007/bf00994018.

[50] P. Mcnamee and J. Mayfield, "Comparing cross-language query expansion techniques by degrading translation resources," *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR 02*, 2002. DOI: 10.1145/564376.564406.

[51] Y. Li, K. Bontcheva, and H. Cunningham, "Svm based learning system for information extraction," *Lecture Notes in Computer Science Deterministic and Statistical Methods in Machine Learning*, 319–339, 2005. DOI: 10.1007/11559887_19.

[52] A. M. Lafferty John and F. C. Pereira., "Conditional random fields: Probabilistic models for segmenting and labeling sequence data.," 2001.

[53] A. Mccallum and W. Li, "Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons," *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003* -, 2003. DOI: 10.3115/1119176.1119206.

[54] S. Liu, B. Tang, Q. Chen, and X. Wang, "Effects of semantic features on machine learning-based drug name recognition systems: Word embeddings vs. manually constructed dictionaries," *Information*, vol. 6, no. 4, 848–865, 2015. DOI: 10.3390/info6040848.

[55] M. W. Tim Rockta schel Torsten Huber and U. Leser., "The impact of domain-specific features on the performance of identifying and classifying mentions of drugs.," 356–363, 2013.

[56] L. Y, B. B, D. S. J, *et al.*, "Backpropagation applied to handwritten zip code recognition.," *Neural Comput*, vol. 1, no. 4, 541–551, 1989.

[57] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, 179–211, 1990. DOI: 10.1207/s15516709cog1402_1.

[58] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, 157–166, 1994. DOI: 10.1109/72.279181.

[59] P. R, M. T, and B. Y, "On the difficulty of training recurrent neural networks.," *International Conference on Machine Learning*, 1310–1318, 2013.

[60] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[61] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, 602–610, 2005. DOI: 10.1016/j.neunet.2005.06.042.

[62] K. Cho, B. V. Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014. DOI: 10.3115/v1/w14-4012.

[63] C. R, W. J, B. L, K. M, K. K, and K. P, "Naturallanguage processing (almost) from scratch.," *Proceedings NAACL-HLT*, 2016.

[64] H. Z, X. W, and Y. K, "Bidirectional lstm-crf models for sequence tagging.," 2015. DOI: arXiv:1508.01991.

[65] L. G, B. M, K. K, S. S, and D. C, "Neural architectures for named entity recognition.," *Proceedings NAACL-HLT*, 2016.

[66] J. P. Chiu and E. Nichols, "Named entity recognition with bidirectional lstm-cnns," *Transactions of the Association for Computational Linguistics*, vol. 4, 357–370, 2016. DOI: 10.1162/tacl_a_00104.

[67] L. W, T. Y, A. S, *et al.*, "Not all contexts are created equal: Better word representations with variable attention.," *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing.*, 1367–1372, 2015.

[68] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016. DOI: 10.18653/v1/p16-1101.

[69] Y. Z, S. R, and C. W, "Multi-task cross-lingual sequence tagging from scratch.," 2016. [Online]. Available: http://arxiv.org/abs/1603.06270.

[70] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. arXiv: 1706.03762. [Online]. Available: http://arxiv.org/abs/1706.03762.

[71] V. A, S. N, P. N, *et al.*, "Attention is all you need," *NIPS*, 5998–6008, 2017.

[72] L. J. P, S. M, P. E, *et al.*, "Generating wikipedia by summarizing long sequences," 2018. [Online]. Available: arXivpreprintarXiv:1801.10198.

[73] K. N and K. D, "Constituency parsing with a self- attentive encoder," *ACL*, 2675–2685, 2018.

[74] T. B. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," 2020. arXiv: 2005.14165 [cs.CL].

[75] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. arXiv: 1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805.

[76] T. Liu, J.-G. Yao, and C.-Y. Lin, "Towards improving neural named entity recognition with gazetteers," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. DOI: 10.18653/v1/p19-1524.

[77] Z. Jie and W. Lu, "Dependency-guided lstm-crf for named entity recognition," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019. DOI: 10.18653/v1/d19-1399.

[78] C. Xia, C. Zhang, T. Yang, *et al.*, "Multi-grained named entity recognition," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. DOI: 10.18653/v1/p19-1138.

[79] Y. Luo, F. Xiao, and H. Zhao, "Hierarchical contextualized representation for named entity recognition," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 8441–8448, 2020. DOI: 10.1609/aaai.v34i05.6363.

[80] Y. Liu, F. Meng, J. Zhang, J. Xu, Y. Chen, and J. Zhou, "Gcdt: A global context enhanced deep transition architecture for sequence labeling," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. DOI: 10.18653/v1/p19-1233.

[81] Y. Jiang, C. Hu, T. Xiao, C. Zhang, and J. Zhu, "Improved differentiable architecture search for language modeling and named entity recognition," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019. DOI: 10.18653/v1/d19-1367.

[82] X. Li, J. Feng, Y. Meng, Q. Han, F. Wu, and J. Li, "A unified mrc framework for named entity recognition," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. DOI: 10.18653/v1/2020.acl-main.519.

[83] J. Zhuo, Y. Cao, J. Zhu, B. Zhang, and Z. Nie, "Segment-level sequence modeling using gated recursive semi-markov conditional random fields," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016. DOI: 10.18653/v1/p16-1134.

[84] Q. Guo, X. Qiu, P. Liu, Y. Shao, X. Xue, and Z. Zhang, "Star-transformer," *Proceedings of the 2019 Conference of the North*, 2019. DOI: 10.18653/v1/n19-1133.

[85] Y. H, D. B, L. X, and Q. X, "Tener: Adapting trans- former encoder for name entity recognition," 2019. [Online]. Available: arXiv:1911.04474,.

[86] R. Caruana, "Multitask learning," *Learning to Learn*, 95–133, 1998. DOI: 10.1007/978-1-4615-5529-2_5.

[87] C. R, W. J, B. L, K. M, K. K, and K. P, "Atural language processing (almost) from scratch," *J. Mach. Learn. Res*, vol. 12, 2493–2537, 2011.

[88] M. Rei, "Semi-supervised multitask learning for sequence labeling," *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017. DOI: 10.18653/v1/p17-1194.

[89] Y. Lin, S. Yang, V. Stoyanov, and H. Ji, "A multi-lingual multi-task architecture for low-resource sequence labeling," *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018. DOI: 10.18653/v1/p18-1074.

[90] S. Zheng, F. Wang, H. Bao, Y. Hao, P. Zhou, and B. Xu, "Joint extraction of entities and relations based on a novel tagging scheme," *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017. DOI: 10.18653/v1/p17-1113.

[91] P. Zhou, S. Zheng, J. Xu, Z. Qi, H. Bao, and B. Xu, "Joint extraction of multiple relations and entities by using a hybrid neural network," *Lecture Notes in Computer Science Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, 135–146, 2017. DOI: 10.1007/978-3-319-69005-6_12.

[92] G. Aguilar, S. Maharjan, A. P. L. Monroy, and T. Solorio, "A multi-task approach for named entity recognition in social media data," *Proceedings of the 3rd Workshop on Noisy User-generated Text*, 2017. DOI: 10.18653/v1/w17-4419.

[93] N. Peng and M. Dredze, "Multi-task domain adaptation for sequence tagging," *Proceedings of the 2nd Workshop on Representation Learning for NLP*, 2017. DOI: 10.18653/v1/w17-2612.

[94] G. Crichton, S. Pyysalo, B. Chiu, and A. Korhonen, "A neural network multi-task learning approach to biomedical named entity recognition," *BMC Bioinformatics*, vol. 18, no. 1, 2017. DOI: 10.1186/s12859-017-1776-8.

[95] P. J. S and Y. Q, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng*, vol. 22, no. 10, 1345–1359, 2010.

[96] J. J and Z. C, "Instance weighting for domain adaptation in nlp," *ACL*, 2007th ser., 264–271,

[97] D. Wu, W. S. Lee, N. Ye, and H. L. Chieu, "Domain adaptive bootstrapping for named entity recognition," *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing Volume 3 - EMNLP 09*, 2009. DOI: 10.3115/1699648.1699699.

[98] A. Chaudhary, J. Xie, Z. Sheikh, G. Neubig, and J. Carbonell, "A little annotation does a lot of good: A study in bootstrapping low-resource named entity recognizers," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019. DOI: 10.18653/v1/d19-1520.

[99] C. Jia, X. Liang, and Y. Zhang, "Cross-domain ner using cross-domain language modeling," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. DOI: 10.18653/v1/p19-1236.

[100] S. J. Pan, Z. Toh, and J. Su, "Transfer joint embedding for cross-domain named entity recognition," *ACM Transactions on Information Systems*, vol. 31, no. 2, 1–27, 2013. DOI: 10.1145/2457465.2457467.

[101] L. Y. J, D. F, and S. P, "Transfer learning for named-entity recognition with neural networks," 2017. [Online]. Available: arXiv:1705.06273.

[102] B. Y. Lin and W. Lu, "Neural adaptation layers for cross-domain named entity recognition," *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018. DOI: 10.18653/v1/d18-1226.

[103] Y. Cao, Z. Hu, T.-S. Chua, Z. Liu, and H. Ji, "Low-resource name tagging learned with weakly labeled data," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019. DOI: 10.18653/v1/d19-1025.

[104] X. Huang, L. Dong, E. Boschee, and N. Peng, "Learning a unified named entity tagger from multiple partially annotated corpora for efficient adaptation," *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 2019. DOI: 10.18653/v1/k19-1048.

[105] L. Qu, G. Ferraro, L. Zhou, W. Hou, and T. Baldwin, "Named entity recognition for novel types by transfer learning," *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016. DOI: 10.18653/v1/d16-1087.

[106] Y. Z, S. R, and C. W. W, "Transfer learning for sequence tagging with hierarchical recurrent networks," *ICLR*, 2017.

[107] P. V. Däniken and M. Cieliebak, "Transfer learning and sentence level features for named entity recognition on tweets," *Proceedings of the 3rd Workshop on Noisy User-generated Text*, 2017. DOI: 10.18653/v1/w17-4422.

[108] H. Zhao, Y. Yang, Q. Zhang, and L. Si, "Improve neural entity recognition via multi-task data selection and constrained decoding," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018. DOI: 10.18653/v1/n18-2056.

[109] X. Wang, Y. Zhang, X. Ren, *et al.*, "Cross-type biomedical named entity recognition with deep multi-task learning," *Bioinformatics*, vol. 35, no. 10, 1745–1752, 2018. DOI: 10.1093/bioinformatics/bty869.

[110] J. M. Giorgi and G. D. Bader, "Transfer learning for biomedical named entity recognition with neural networks," *Bioinformatics*, vol. 34, no. 23, 4087–4094, 2018. DOI: 10.1093/bioinformatics/bty449.

[111] Z. Wang, Y. Qu, L. Chen, *et al.*, "Label-aware double transfer learning for cross-specialty medical named entity recognition," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018. DOI: 10.18653/v1/n18-1001.

[112] S. B, "Active learning," *Synth. Lect. Artif. Intell. Mach. Learn*, vol. 6, no. 1, 1–114, 2012.

[113] Y. Shen, H. Yun, Z. Lipton, Y. Kronrod, and A. Anandkumar, "Deep active learning for named entity recognition," *Proceedings of the 2nd Workshop on Representation Learning for NLP*, 2017. DOI: 10.18653/v1/w17-2630.

[114] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," *Sigir '94*, 3–12, 1994. DOI: 10.1007/978-1-4471-2099-5_1.

[115] U. O, P. S, M. A, *et al.*, "Towards robust linguistic analysis using ontonotes," *CoNLL 2013*, 143–152, 2013.

[116] K. P. L, L. L. M, and M. W. A, "Reinforcement learning: A survey," *J. Artif. Intell. Res*, vol. 4, 237–285, 1996.

[117] S. S. R and B. G. A, "Introduction to reinforcement learning.," *MIT press Cambridge*, vol. 135, 1998.

[118] Y. Y, C. W, L. Z, H. Z, and Z. M, "Distantly supervised ner with partial annotation learning and reinforcement learning," *COLING*, 2159–2169, 2018.

[119] L. D and M. C, "Adversariallearning," *SIGKDD*, 641–647, 2005.

[120] G. I, P. J, M. M, *et al.*, "Generative adver- sarial nets," *NIPS*, 2672–2680, 2014.

[121] L. Huang, H. Ji, and J. May, "Cross-lingual multi-level adversarial transfer to enhance low-resource name tagging," *Proceedings of the 2019 Conference of the North*, 2019. DOI: 10.18653/v1/n19-1383.

[122] J. Li, D. Ye, and S. Shang, "Adversarial transfer for named entity boundary detection with pointer networks," *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019. DOI: 10.24963/ijcai.2019/702.

[123] P. Cao, Y. Chen, K. Liu, J. Zhao, and S. Liu, "Adversarial transfer learning for chinese named entity recognition with self-attention mechanism," *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018. DOI: 10.18653/v1/d18-1017.

[124] J. T. Zhou, H. Zhang, D. Jin, *et al.*, "Dual adversarial neural transfer for low-resource named entity recognition," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. DOI: 10.18653/v1/p19-1336.

[125] B. D, 2016. [Online]. Available: http://www.wildml.com/2016/01/attention-and-memory-in-deeplearning-and-nlp.

[126] A. Zukov-Gregoric, Y. Bachrach, P. Minkovsky, S. Coope, and B. Maksak, "Neural named entity recognition using a self-attention mechanism," *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2017. DOI: 10.1109/ictai.2017.00104.

[127] G. Xu, C. Wang, and X. He, "Improving clinical named entity recognition with global neural attention," *Web and Big Data Lecture Notes in Computer Science*, 264–279, 2018. DOI: 10.1007/978-3-319-96893-3_20.

[128] Z. Q, F. J, L. X, and H. X, "Adaptive co-attention network for named entity recognition in tweets," *AAAi*, 2018.

[129] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *Artificial Neural Networks and Machine Learning – ICANN 2011*, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 44–51, ISBN: 978-3-642-21735-7.

[130] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *CoRR*, vol. abs/1710.09829, 2017. arXiv: 1710.09829. [Online]. Available: http://arxiv.org/abs/1710.09829.

[131] W. Zhao, J. Ye, M. Yang, Z. Lei, S. Zhang, and Z. Zhao, "Investigating capsule networks with dynamic routing for text classification," *CoRR*, vol. abs/1804.00538, 2018. arXiv: 1804.00538. [Online]. Available: http://arxiv.org/abs/1804.00538.

[132] M. Yang, W. Zhao, L. Chen, Q. Qu, Z. Zhao, and Y. Shen, "Investigating the transferring capability of capsule networks for text classification," *Neural Networks*, vol. 118, pp. 247–261, 2019, ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2019.06.014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S089360801930187X.

[133] J. Deng, L. Cheng, and Z. Wang, "Self-attention-based bigru and capsule network for named entity recognition," *CoRR*, vol. abs/2002.00735, 2020. arXiv: 2002.00735. [Online]. Available: https://arxiv.org/abs/2002.00735.

[134] N. Alshammari and S. Alanazi, "The impact of using different annotation schemes on named entity recognition," *Egyptian Informatics Journal*, vol. 22, no. 3, pp. 295–302, 2021, ISSN: 1110-8665. DOI: https://doi.org/10.1016/j.eij.2020.10.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110866520301596.

[135] V. Mozharova and N. Loukachevitch, "Two-stage approach in russian named entity recognition," in *2016 International FRUCT Conference on Intelligence, Social Media and Web (ISMW FRUCT)*, IEEE, 2016, pp. 1–6.

[136] E. F. T. K. Sang and F. D. Meulder, "Introduction to the conll-2003 shared task," *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003* -, 2003. DOI: 10.3115/1119176.1119195.

[137] X. Zhong and E. Cambria, "Tomn: Constituent-based tagging scheme," in *Time Expression and Named Entity Recognition*. Cham: Springer International Publishing, 2021, pp. 59–75, ISBN: 978-3-030-78961-9. DOI: 10.1007/978-3-030-78961-9_5. [Online]. Available: https://doi.org/10.1007/978-3-030-78961-9_5.

[138] X. Zhong and E. Cambria, "Time expression recognition using a constituent-based tagging scheme," Apr. 2018, pp. 983–992, ISBN: 978-1-4503-5639-8. DOI: 10.1145/3178876.3185997.

[139] X. Zhong, E. Cambria, and A. Hussain, "Extracting time expressions and named entities with constituent-based tagging schemes," *Cognitive Computation*, pp. 1–19, 2020.

[140] M. Konkol and M. Konopík, "Segment representations in named entity recognition," *Text, Speech, and Dialogue Lecture Notes in Computer Science*, 61–70, 2015. DOI: 10.1007/978-3-319-24033-6_7.

[141] D. R. G, M. A, P. A. M, R. A. L, S. S, and W. M. R, "The automatic content extraction (ace) program-tasks, data, and evaluation.," *LREC*, vol. 2, p. 1, 2004.

[142] H. Shi, K. Livescu, and K. Gimpel, "Substructure substitution: Structured data augmentation for NLP," *CoRR*, vol. abs/2101.00411, 2021. arXiv: 2101.00411. [Online]. Available: https://arxiv.org/abs/2101.00411.

[143] A. Hernández-García and P. König, "Data augmentation instead of explicit regularization," *CoRR*, vol. abs/1806.03852, 2018. arXiv: 1806.03852. [Online]. Available: http://arxiv.org/abs/1806.03852.

[144] Y. Belinkov and Y. Bisk, "Synthetic and natural noise both break neural machine translation," *CoRR*, vol. abs/1711.02173, 2017. arXiv: 1711.02173. [Online]. Available: http://arxiv.org/abs/1711.02173.

[145] S. Y. Feng, V. Gangal, D. Kang, T. Mitamura, and E. H. Hovy, "Genaug: Data augmentation for finetuning text generators," *CoRR*, vol. abs/2010.01794, 2020. arXiv: 2010.01794. [Online]. Available: https://arxiv.org/abs/2010.01794.

[146] O. Kolomiyets, S. Bethard, and M.-F. Moens, "Model-portability experiments for textual temporal analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA: Association for Computational Linguistics, Jun. 2011, pp. 271–276. [Online]. Available: https://aclanthology.org/P11-2047.

[147] X. Wang, Y. Sheng, H. Deng, and Z. Zhao, "Charcnn-svm for chinese text datasets sentiment classification with data augmentation," *International journal of innovative computing, information and control*, vol. 15, no. 1, pp. 227–246, 2019.

[148] A. Mosolova, V. Fomin, and I. Bondarenko, "Text augmentation for neural networks.," in *AIST (Supplement)*, 2018, pp. 104–109.

[149] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to WordNet: An On-line Lexical Database*," *International Journal of Lexicography*, vol. 3, no. 4, pp. 235–244, Dec. 1990, ISSN: 0950-3846. DOI: 10.1093/ijl/3.4.235. eprint: https://academic.oup.com/ijl/article-pdf/3/4/235/9820417/235.pdf. [Online]. Available: https://doi.org/10.1093/ijl/3.4.235.

[150] G. Rizos, K. Hemker, and B. Schuller, "Augment to prevent: Short-text data augmentation in deep learning for hate-speech classification," *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019.

[151] S. T. Aroyehun and A. Gelbukh, "Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling," in *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, 2018, pp. 90–97.

[152] G. G. Şahin and M. Steedman, "Data augmentation via dependency tree morphing for low-resource languages," *arXiv preprint arXiv:1903.09460*, 2019.

[153] M. Aiken, "The efficacy of round-trip translation for mt evaluation," vol. 14, Feb. 2010.

[154] C. Coulombe, "Text data augmentation made simple by leveraging NLP cloud apis," *CoRR*, vol. abs/1812.04718, 2018. arXiv: 1812.04718. [Online]. Available: http://arxiv.org/abs/1812.04718.

[155] E. Rabinovich, R. N. Patel, S. Mirkin, L. Specia, and S. Wintner, "Personalized machine translation: Preserving original author traits," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 1074–1084. [Online]. Available: https://aclanthology.org/E17-1101.

[156] T. DeVries and G. W. Taylor, "Dataset augmentation in feature space," *arXiv preprint arXiv:1702.05538*, 2017.

[157] V. Kumar, H. Glaude, C. de Lichy, and W. Campbell, "A closer look at feature space data augmentation for few-shot intent classification," *arXiv preprint arXiv:1910.04176*, 2019.

[158] G. Kurata, B. Xiang, and B. Zhou, "Labeled data generation with encoder-decoder lstm for semantic slot filling.," in *INTERSPEECH*, 2016, pp. 725–729.

[159] E. Schwartz, L. Karlinsky, J. Shtok, *et al.*, "Delta-encoder: An effective sample synthesis method for few-shot object recognition," *Advances in neural information processing systems*, vol. 31, 2018.

[160] X. Dai and H. Adel, "An analysis of simple data augmentation for named entity recognition," *CoRR*, vol. abs/2010.11683, 2020. arXiv: 2010.11683. [Online]. Available: https://arxiv.org/abs/2010.11683.

[161] S. Mysore, Z. Jensen, E. Kim, *et al.*, "The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures," in *Proceedings of the 13th Linguistic Annotation Workshop*, Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 56–64. DOI: 10.18653/v1/W19-4007. [Online]. Available: https://aclanthology.org/W19-4007.

[162] O. Uzuner, B. South, S. Shen, and S. DuVall, "2010 i2b2/va challenge on concepts, assertions, and relations in clinical text," *Journal of the American Medical Informatics Association : JAMIA*, vol. 18, pp. 552–6, Jun. 2011. DOI: 10.1136/amiajnl-2011-000203.

[163] B. Ding, L. Liu, L. Bing, *et al.*, "DAGA: data augmentation with a generation approach for low-resource tagging tasks," *CoRR*, vol. abs/2011.01549, 2020. arXiv: 2011.01549. [Online]. Available: https://arxiv.org/abs/2011.01549.

[164] M. Sundermeyer, R. Schlüter, and H. Ney, "Lstm neural networks for language modeling," in *Thirteenth annual conference of the international speech communication association*, 2012.

[165] S. Chen, G. Aguilar, L. Neves, and T. Solorio, "Data augmentation for cross-domain named entity recognition," *CoRR*, vol. abs/2109.01758, 2021. arXiv: 2109.01758. [Online]. Available: https://arxiv.org/abs/2109.01758.

[166] S. Pradhan, A. Moschitti, N. Xue, *et al.*, "Towards robust linguistic analysis using ontonotes," in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, 2013, pp. 143–152.

[167] S. Rijhwani and D. Preoţiuc-Pietro, "Temporally-informed analysis of named entity recognition," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7605–7617.

[168] R. Zhou, R. He, X. Li, *et al.*, "MELM: data augmentation with masked entity language modeling for cross-lingual NER," *CoRR*, vol. abs/2108.13655, 2021. arXiv: 2108.13655. [Online]. Available: https://arxiv.org/abs/2108.13655.

[169] X. Wu, S. Lv, L. Zang, J. Han, and S. Hu, "Conditional bert contextual augmentation," in *International conference on computational science*, Springer, 2019, pp. 84–95.

[170] *Datasets for name entity recognition*, https://paperswithcode.com/datasets?task=named-entity-recognition-ner, Accessed: 2021-12-10.

[171] *Conll-2003*, https://paperswithcode.com/dataset/conll-2003, Accessed: 2022-05-18.

[172] J. Li, Y. Sun, R. J. Johnson, *et al.*, "Biocreative v cdr task corpus: A resource for chemical disease relation extraction," *Database*, vol. 2016, 2016.

[173] 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.

[174] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[175] *Conll-2003*, https://paperswithcode.com/sota/named-entity-recognition-ner-on-conll-2003, Accessed: 2022-05-18.

[176] A. Ushio and J. Camacho-Collados, "T-ner: An all-round python library for transformer-based named entity recognition," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, 2021, pp. 53–62.

[177] E. Siow, *Practical machine learning: A collection of machine learning experiments in notebooks*, Available at: https://github.com/eugenesiow/practical-ml, 2020. [Online]. Available: https://github.com/eugenesiow/practical-ml.

[178] Z. Wang, J. Shang, L. Liu, L. Lu, J. Liu, and J. Han, "Crossweigh: Training named entity tagger from imperfect annotations," *arXiv preprint arXiv:1909.01441*, 2019.

[179] Z. Wang, J. Shang, L. Liu, L. Lu, J. Liu, and J. Han, "CrossWeigh: Training named entity tagger from imperfect annotations," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 5154–5163. DOI: 10.18653/v1/D19-1519. [Online]. Available: https://aclanthology.org/D19-1519.

[180] J. P. Chiu and E. Nichols, "Named entity recognition with bidirectional lstm-cnns," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 357–370, 2016.

[181] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, Boulder, Colorado: Association for Computational Linguistics, Jun. 2009, pp. 147–155. [Online]. Available: https://aclanthology.org/W09-1119.

[182] V. Kocaman and D. Talby, *Biomedical named entity recognition at scale*, 2020. arXiv: 2011.06315 [cs.CL].

[183] M. Habibi, L. Weber, M. Neves, D. L. Wiegandt, and U. Leser, "Deep learning with word embeddings improves biomedical named entity recognition," *Bioinformatics*, vol. 33, no. 14, pp. i37–i48, 2017.

[184] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[185] *Conll-2003*, https://paperswithcode.com/sota/named-entity-recognition-ner-on-bc5cdr, Accessed: 2022-05-18.

[186] J. Lee, W. Yoon, S. Kim, *et al.*, "Biobert: A pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.

[187] F. Nooralahzadeh, J. T. Lønning, and L. Øvrelid, "Reinforcement-based denoising of distantly supervised NER with partial annotation," in *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 225–233. DOI: 10.18653/v1/D19-6125. [Online]. Available: https://aclanthology.org/D19-6125.

[188] Y. Liu, M. Ott, N. Goyal, *et al.*, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[189] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter," *ArXiv*, vol. abs/1910.01108, 2019.

[190] T. K. Relat, "Applications of finite automata representing large vocabularies applications of finite automata representing large vocabularies," 1992.

[191] B. W. Watson, "Implementing and using finite automata toolkits," *Nat. Lang. Eng.*, vol. 2, pp. 295–302, 1996.

[192] M. Pettersson, "A term pattern-match compiler inspired by finite automata theory," in *Compiler Construction*, U. Kastens and P. Pfahler, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 258–270, ISBN: 978-3-540-47335-0.

[193] *Speech recognition — weighted finite-state transducers (wfst)*, `https://jonathan-hui.medium.com/speech-recognition-weighted-finite-state-transducers-wfst-a4ece08a89b7`, Accessed: 2022-05-18.

[194] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "Openfst: A general and efficient weighted finite-state transducer library," in *Proceedings of the 12th International Conference on Implementation and Application of Automata (CIAA 2007)*, Prague, Czech Republic, 2007.

[195] B. Roark, R. Sproat, C. Allauzen, M. Riley, J. Sorensen, and T. Tai, "The OpenGrm open-source finite-state grammar software libraries," in *Proceedings of the ACL 2012 System Demonstrations*, Jeju Island, Korea: Association for Computational Linguistics, Jul. 2012, pp. 61–66. [Online]. Available: `https://aclanthology.org/P12-3011`.

[196] *Pegasus: A state-of-the-art model for abstractive text summarization*, `https://ai.googleblog.com/2020/06/pegasus-state-of-art-model-for.html`, Accessed: 2022-05-18.

[197] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, *Pegasus: Pre-training with extracted gap-sentences for abstractive summarization*, 2020. arXiv: `1912.08777 [cs.CL]`.

[198] D. Suleiman and A. A. Awajan, "Deep learning based abstractive text summarization: Approaches, datasets, evaluation measures, and challenges," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–29, 2020.

[199] R. Nallapati, B. Xiang, and B. Zhou, "Sequence-to-sequence rnns for text summarization," *CoRR*, vol. abs/1602.06023, 2016. arXiv: `1602.06023`. [Online]. Available: `http://arxiv.org/abs/1602.06023`.

[200] L. Dong, N. Yang, W. Wang, *et al.*, "Unified language model pre-training for natural language understanding and generation," in Red Hook, NY, USA: Curran Associates Inc., 2019.

[201] S. Rothe, S. Narayan, and A. Severyn, "Leveraging pre-trained checkpoints for sequence generation tasks," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 264–280, 2020, https://doi.org/10.1162/tacl$_{a0}$0313. [Online]. Available: `https://www.mitpressjournals.org/doi/pdf/10.1162/tacl_a_00313`.

[202] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 353–355. DOI: `10.18653/v1/W18-5446`. [Online]. Available: `https://aclanthology.org/W18-5446`.

[203] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2383–2392. DOI: `10.18653/v1/D16-1264`. [Online]. Available: `https://aclanthology.org/D16-1264`.

[204] J. Bos, V. Basile, K. Evang, N. Venhuizen, and J. Bjerva, "The groningen meaning bank," in *Handbook of Linguistic Annotation*, N. Ide and J. Pustejovsky, Eds., vol. 2, Springer, 2017, pp. 463–496.

[205] S. Khatoon, M. Alshamari, A. Asif, *et al.*, "Development of social media analytics system for emergency event detection and crisis management," *Computers, Materials and Continua*, vol. 68, pp. 3079–3100, Mar. 2021. DOI: `10.32604/cmc.2021.017371`.

[206] *Nvidia named entity recognition bert*, `https://catalog.ngc.nvidia.com/orgs/nvidia/teams/tlt-jarvis/models/namedentityrecognition_english_bert`, Accessed: 2022-05-18.

[207] S. Longpre, Y. Wang, and C. DuBois, "How effective is task-agnostic data augmentation for pretrained transformers?" In *FINDINGS*, 2020.

[208] S. Arora, A. May, J. Zhang, and C. Ré, "Contextual embeddings: When are they worth it?" *CoRR*, vol. abs/2005.09117, 2020. arXiv: `2005.09117`. [Online]. Available: `https://arxiv.org/abs/2005.09117`.

[209] L. Gong, Z. Zhang, and S. Chen, "Clinical named entity recognition from chinese electronic medical records based on deep learning pretraining," *Journal of Healthcare Engineering*, vol. 2020, pp. 1–8, Nov. 2020. DOI: 10.1155/2020/8829219.