



The comparison between the tools for Named Entity Recognition

Zhang Wenjie

This thesis is submitted to Auckland University of Technology
As the partial fulfillment of the requirements for the degree of
Master of Computer and Information Sciences

2020

School of Engineering, Computer and Mathematical Science

Table of the contents

Table of the contents.....	II
List of figures.....	I
List of tables.....	II
Abstract.....	III
Attestation of Authorship.....	V
Acknowledgement.....	VI
Chapter 1: Introduction of the NLP	1
1.1 What is NLP and its difficulties we may face?	1
1.2 How can NLP help us and what is the use of this technique?	3
1.3 NLP in the past and current time.....	4
1.4 Named Entity Recognition	7
1.5 A brief history of NER	10
Chapter 2: Introduction of the Deep Learning.....	12
2.1 Deep Learning	12
2.2 Pre-trained models	19
Chapter 3: literature review over the techniques which were used for the experiment.....	22
3.1 Convolutional Neural Networks.....	22
3.2 Bi-directional Long Short-Term Memory (BiLSTM)	30
3.3 Conditional Random Field (CRF)	34
3.4 BERT.....	37
Chapter 4: Experiment and analysis	39

4.1 Working environment.....	39
4.2 A brief introduction about the mutually used dataset and pre-processing of the data for the BERT	39
4.3 The results of the experiment:.....	41
4.4Limitation of the BERT experiment.....	43
Conclusion and future research	44
Reference	45

List of figures

Figure1.....	8
Figure2.....	9
Figure3.....	11
Figure4.....	12
Figure5.....	12
Figure6.....	13
Figure7.....	16
Figure8.....	19
Figure9.....	20
Figure10.....	20
Figure11.....	21
Figure12.....	23
Figure13.....	26
Figure14.....	32
Figure15.....	32
Figure16.....	38
Figure17.....	40
Figure18.....	41
Figure19.....	41
Figure20.....	43

List of tables

Table1.....	41
Table2.....	41

Abstract

NLP (Natural language processing) is currently being widely used in our modern daily life, such as spam email detecting, prediction of potential criminals by analysing the provided information, sentiment analysing and many so on. In the old-time, the popular way to solve NLP tasks can be done by hands or by computers with some strictly given disciplines, these ways are tiresome and often too slow if we have to deal with a huge amount of data or information. Until now, some of the companies and researchers are still using these ways to do their jobs. Besides that, thanks to the development of the computer hardware, we have some relatively new, prevalent and highly accurate ways to help us to finish these tasks, from Machine Learning (such as SVM---supported vector machines) to DNN (Deep Neural Networks) and pretrained models (like ELMO, OpenAI GPT), therefore in modern data companies and online shopping enterprises, the previous ways are gradually decreasing in use to do these tasks because they are obsolete and not suitable to be used in the data era anymore.

But now, we have the brand-new approach to cope with the NLP tasks, which is called BERT, it is the abbreviation of **B**idirectional **E**ncoder **R**epresentations from **T**ransformer. The BERT was introduced and inaugurated by the Google development team in late 2018, it is a pretrained language model for multiple language processing tasks, and also for multiple languages by using the unsupervised learning way. This model is powerful for doing the realistic language processing tasks and it has already broken 11 records in the NLP area in 2018. And, it is simple to use, it just needs fine-tuning and only one additional layer for output without other specific architectures for some specific tasks (such as labelling, classification or question answering). More information of this new technique will be introduced in a detailed way.

The method used for completing this thesis is based on literature review, and a comparison between BERT and one prevailing and elaborate DNN technique which is called BILSTM-CRF-CNN (**Bi-directional Long Short-Term Memory-Conditional Random Field-Convolutional Neural Networks**) in dealing with Name Entity Recognition (NER) and the results of the comparison will be shown. Also, my opinions about the BERT, the suggestions, and potential future issues of future researches are given.

Keywords---- BERT, NLP, BILSTM-CRF-CNN, NER

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the citations and acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Signature:  Date: 10/01/2020

Acknowledgement

Many thanks to my supervisor Dr Parma. He helped me a lot during the writing of the thesis, without his encouragement this thesis cannot be finished on time, because I changed my research object in the middle of the semester.

And once again, thanks for Dr Parma's help.

Signature:  Date: 10/01/2020

Chapter 1: Introduction of the NLP

Natural language processing (NLP) is actually a very challenging task for computers to deal with. Because there are many languages and each language has a different grammar, the way of writing (e.g.: characters for Chinese, Japanese and letters for German, English and many others), also, there are many kinds of tasks or problems which we may face in our daily lives (e.g.: Extraction of information and article analysis). By using the right approach/es, useful and critical information can be extracted from articles, speeches without spending too much time on reading or analyzing the whole materials or we can even “talk” with our electronic devices (chatbots). In erstwhile times, researchers and scientists used many complicated or even intricate methods to break the whole language into different forms, such as, speeches, words, syntax, or phrases to analyse the language and do the tasks of NLP. Today, by the continuous and rapid development of deep learning, neural networks have already become a main part of the same processing.

1.1 What is NLP, and its difficulties we may face?

In [6], it gives the simple definition of NLP is that: it is the capability which is given by the humans to the computers, machines or systems to make sure they can truly and correctly understand human languages and do some tasks which are given by the humans in the same way that people do.

In [7], it provides the definition from the author of NLTK (Natural Language Toolkit, it is a very popular python library for NLP tasks): “*We will take Natural Language Processing - or NLP for short - in a wide sense to cover any kind of computer manipulation of natural language. At one extreme, it could be as simple as counting word frequencies to compare different writing styles. At the other extreme, NLP involves “understanding” complete human utterances, at least to the*

extent of being able to give useful responses to them". In [8], it gives the definition in a detail is that: NLP is the analysis of data which is formed by language (syntax, phrases, or words). Most usually the data is in the text form such as speeches which are written on paper, articles, documents or many other publications, and we use some specific methods to process and generate some results that we aim for, but, of course, the conversation between humans and machines needs the use of NLP. The aim of NLP in a general way is to set up some structures to the unstructured natural languages, in order to represent the texts, by using the applications of the theories of modern linguistics. The structure can be either in the syntactic form in order to grasp the relationships of grammar among components of the text or other materials, or in the semantic form to capture the meaning which is represented by the sentences of text.

Also, in [9], it defined NLP as a combination of many computational techniques to analyse and represent human language automatically which is based on theory. So, the NLP, generally speaking, is a technique which consists of many other techniques, such as, Deep learning and modern linguistics which can greatly help people to analyse any materials that are comprised of languages.

The evolution of NLP is great, decades ago, scientists and researchers could only use punching machines, cards and batch processing to finish the NLP task, therefore it could cost up to 7 minutes to analyse just one sentence, and now, we have highly advanced methods and algorithms which were developed by Google team or many others, by the help of these developments, we can process millions of webpages in less than one second [9]. NLP provides the capabilities for computers to make the performance over a wide range of natural language corresponded tasks at every level, the range includes parsing and POS (part of speech) tagging, sequence labeling, dialogue systems, and many others.

The NLP is actually a very complicated technique, for humans, it is relatively easy to comprehend the language which is either spoken or written form by other people,

but for the machines and computers it can be another case [6]. For example, if the sentence says: “Food is very important for mankind that we cannot live without it.”, It is conspicuous that the “that” in the sentence means the “food” at the beginning of the whole sentence, in the English language the “that” is called relative word and the whole sentence is called relative clause. For computers and machines or any other smart devices, to understand the language/s which were either in spoken or written form are difficult, it can be a big problem. In the normal communication between people, many things are always omitted or said indirectly, these omitted or implied words or significance are substituted by the form of signals, special expression, or just silence [6]. Also, humans are capable of comprehending the intention and the tones of the language/s but devices or any other machines cannot do this. The second critical problem is that sentences may contain ambiguity. This problem may occur at the level of the word, level of the syntax, or even at the level of the meaning. Also, in [6] it provides an example of the word: “won’t”, there is always an ambiguity about contraction. The NLP systems may consider the contraction of words as one word or maybe two words and with different senses which are not the exact senses, these rules seem easy to humans, but in reality, most of the people are also very poor at fully and totally understanding and giving a clear and correct description of the rules that govern languages. Also, the languages are not like pictures, they are continuously developing and evolving which makes them even harder for computers to understand, even sometimes for humans.

1.2 How can NLP help us and what is the use of this technique?

The NLP has already attracted much attention for coping with automatic analysis of human language. By years of development, this technique and those applications which are derivatives of it are currently been using in many fields, for example [1]: Machine translation, extraction of critical information, summarization of literature, chatbot, medical analysis, detection of spam emails.

This technique is somehow a combination of computer science and linguistics which aims to “teach” the computers to comprehend the speeches and writings (sentences, articles, or just a single word). It can greatly help the people and make it easier for them to communicate with computers, robots, or much other equipment [1], especially for elders, because many of them do not know how to use smartphones and other smart devices or they have difficulties in typing words, due to the problems of eyes, but, with NLP and NLP based applications, like Siri and Google assistant, they can use their phones by speaking to them. Also, the most popular and welcomed application which uses the NLP technique is the chatbot. It can provide the relatively correct and appropriate or sometimes interesting answers or replies to the questions which are asked or sent by the real human users, we can attain this result by using the correct NLP models and training process.

1.3 NLP in the past and current time

In [2], the UK computer scientist of University of Cambridge gave 4 phases of the history of the development of the NLP: phase 1: from the late 1940s to 1960s, phases 2: late 1960s to 1970s, phases 3: from late 1970s to 1980s and phase 4: 1990s

Phase 1:

At this stage, the NLP tasks mainly focused on machine translation. In 1954, the IBM cooperation demonstrated a very basic machine translation from Russian to English. Also, the journal which is called Mechanical Translation was published in the same year. In 1952, the first conference about NLP was held and in 1956 the second one was held. In 1967, the researcher who is called Plath indicated that the time consumption for the analysis of long sentences was averagely around seven minutes, even people used the best machine and most sophisticated algorithm at that time. The researches were conducted internationally, including Japan, USA, UK, Soviet Union. The main languages for researches were Russian and English [2].

At this stage, the NLP was still relatively primitive, many researchers engaged in it and tried to discover more of it. Their contribution was great because the foundation of the NLP was established by them. This phase ended in 1966[2] because the ALPC (Automatic Language Processing Advisory Committee) made a conclusion that Machine Translation was meaningless and could not attain any further achievement.

Phase 2:

At this phase, the AI (Artificial Intelligence) played a role in it. In 1961, some works about data construction and many other kinds used AI as a part of the system, also these works integrated the BASEBALL QA system (Question and Answer). At that time, both the language input and the tasks were relatively too simple, if we compare them with the contemporary Machine Translation [2]. The system was developed by Minsky in 1968 [2] and SIR (Semantic Information Retrieval program) which was developed by Bertram Raphael [2],[4] in 1964, these two systems, showed the capability in interpretation and response towards the language input. In 1970 [3], Bill Woods created LUNAR a type of finite-machine which is called augmented transition networks and it was modified from TN (Transition Networks), also in 1973, Winograd developed SHRDLU. These programs or systems are the successors of previous ones with many advanced capabilities in coping with processing natural language related tasks.

At this era, scientists thought that the contemporary linguistics could not contribute to any more ideas and the Artificial Intelligence could somehow help these thoughts were enlightened by Minsky.

Phase 3:

In the early 80s [2], it was still apparently very difficult to build a predictable and extensible NLP system to deal with supposed application. It is even harder to cope with the applications which are heavily restricted or in terms of processing tasks

and domain of discourse. There were many obstacles that needed to get break through.

Also, in the early 1980s [1], the theory of formal grammar for computation purposes became a wildly popular and fast growing area as the scope of researches combined with logics for the representation of meaning which made the knowledge be able to fit the users' intentions and other desires, and with conversation features, for example, emphasis or finding the theme of the conversation. But it was still somehow difficult to understand expressions of linguistics and presented the indication of theme and sentiment, and dialogue on a great scale by using this paradigm.

At the end of the 1980s [2], the combination of the grammar and logic as an approach finally became sentence processors for general-purpose and they were powerful at the contemporary stage. Such as, Core Language Engine (CLE) which was developed by SRI (Stanford Research Institute) and theory of Discourse Representation made itself an approach to deal with more complicated discourses with much more content under the grammatic-logical framework.

At this stage, the AI gradually became more important in NLP and the foundation of the NLP had been established. Although, there were many other difficulties and obstruction were needed to be overcome. Some of the techniques and notions, we are still using in coping with the tasks of NLP in the 2010s, e.g.: AI and Parsers.

Phase 4:

The Lexicalism (it is a standpoint of theory in modern generative linguistic which is used in the processing that forms many complex words and these words are accounted by a set of lexical rules [5] which is, of course, different with the semantic rules) appeared in the 1980s [2] and it has become very popular and influential in NLP area. Also, in the late stage of the 1990s, the processing method which is called statistical language processing became the most major approach. It has not only

involved analysis of data; indeed, it has involved the direct application of statistical methods towards NLP [2].

The other feature of this field was that the progress was significant with practical tasks. For example, researchers used finite state parsing and surface patterns, by these techniques the useful output can be generated. Also, the extraction of information and automatic summarization were the main research fields.

From the 2000s to recent age:

Many projects and works from around the year 2000 have involved the applications of ML (Machine Learning) techniques, e.g.: Bayesian models, maximum entropy [3]. The ML techniques also are combined with annotated corpora for the purpose of system training and annotated texts which are subject to the different criteria of morphology, syntactic structures, and semantic structures. This technique has already been used as important tool in dealing with some specific tasks, for example, parsing, distinguishing the meaning between different words, QA system, extraction of information, and chatbot.

1.4 Named Entity Recognition

Names are a kind of proper nouns, in many western countries' languages, their beginning letters are usually written in the capital form and have other syntactic characteristics that make them distinct from other kinds of nouns, such as, names that can be used without a determiner [10]. At the side of semantic, we can consider the names as a single entity, like countries' names or companies' names. Names in sentences usually act as an important role among different kinds of languages, at the aspects of legal texts or documents [10]: we use names to identify the different parties, lawyers, courts, judge, statues, jurisdictions, and many others which are covered during the legal proceedings. Therefore, we need to make the computers or other machines have the capability to figure out which sequences of words are

names and match these names to what they refer to can be very essential for many jobs of NLP.

We, as humans, have the capability to find out names easily and figure out and comprehend what they refer to, this is a part of our inherent abilities. The whole process of locating named entities in any kind of written materials and making a classification of them is called named entity recognition [10].

The decision by the independent MP **Andrew Wilkie** to withdraw his support for the minority **Labor** government sounded dramatic but it should not further threaten its stability. When, after the **2010** election, **Wilkie**, **Rob Oakeshott**, **Tony Windsor** and the **Greens** agreed to support **Labor**, they gave just two guarantees: confidence and supply.

Person
Date
Location
Organi- zation

Fig 1 The example of the NER [11]

In [11], the paper defines that the Named Entity Recognition (NER) is a subtask of extraction of information and it is very important. We use this to find the names in the articles or any other written materials and of course make a classification of these names which are found by us. It is important, because, for example, QA systems, these answers are usually named entities, many relations of extraction of information are related to named entities and sentiment about the specific companies or customers' experience about the products can be highlighted if we use NER to check the keywords which reflect the sentiment.

In [12], it gives a more detailed definition of NER: it is one of the most important parts of NLP tasks. NER is designed to find and figure out expressions of some special significance which are written in texts by using NL (natural language).

These expressions vary a lot, from some specific names of persons, companies, or any other kinds of organizations to dates or anything else, and these expressions usually contain the information of texts that we think it is important. We can apply this technique to different kinds of important tasks. First of all, it can be used as a quick tool to search the full texts and filter out the information which we think it might be useful or critical, and also, we can apply it over the tasks as a tool for pre-processing of other NLP tasks, these tasks can make good use of those labeled named entities which are retrieved from NER processing and use them well in the next procedures, by doing that, the performance can be enhanced. There are many tasks can be integrated with NER [12], such as, MT (Machine Translation), QA systems or applications, summarization of written materials, LM (Language Modeling), and SA (Sentiment Analysis).

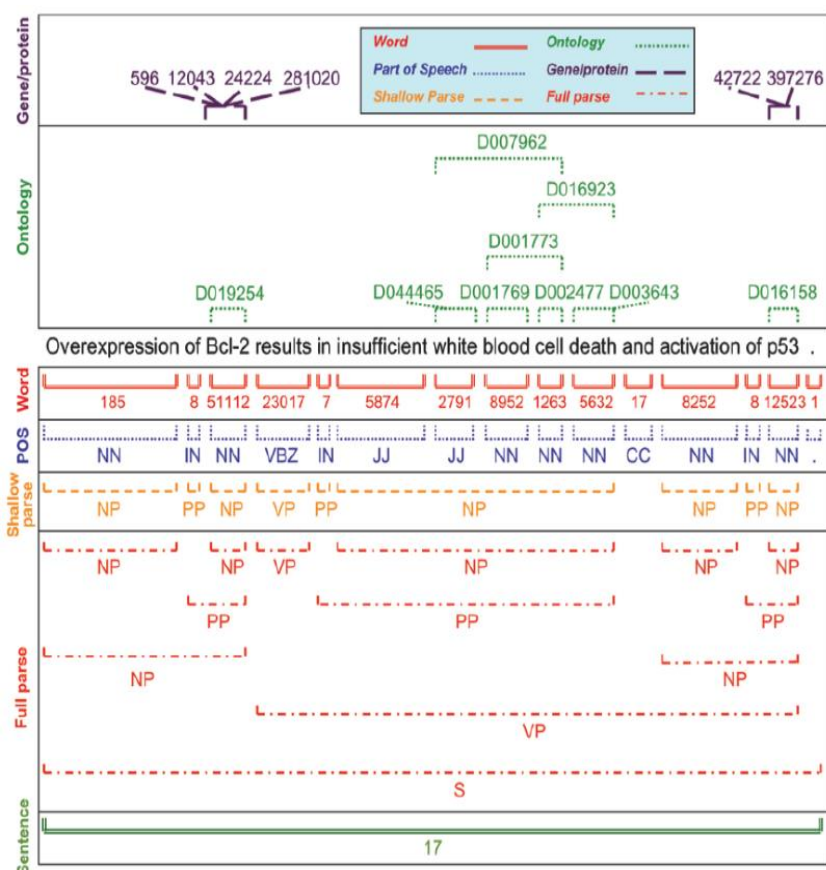


Figure 2: A practical example of using NER in the study of biological science [5].

In [5], a real-life example of a direct application of NER in dealing with some information was provided to us. In Fig.5, we can see annotations of terms of ontology, gene, and protein. There are many of such terms in the upper figure are related to named entities, because of different objects that are generally being distinct and referred to by names. This is for processes or events which usually are required to identify the relations of a higher order. The typical keywords or phrases of named entities in the biological science area which are usually designed to be extracted are genes, chemicals, virus, bacteria, different kinds of diseases or methods of experiment [5]. There are many handy and ready approaches that can be used to perform the whole process of NER.

The most basic and simplest way is to set up a dictionary of the names we care about for a special category of entities which can be linked between them, and we just need to make a performance of string matching within the dictionary. Other methods which involve SML (supervised machine learning) use string matching within dictionary as one important part of the whole process, but it also involves the consideration of the contextual words, syntactical features, and other textual clues to generate the highly precise identification of possible instances of terms from the category [5].

In this paper two different ways that can be used to do NER tasks will be discussed and the result of the comparison between them will also be provided and analysed.

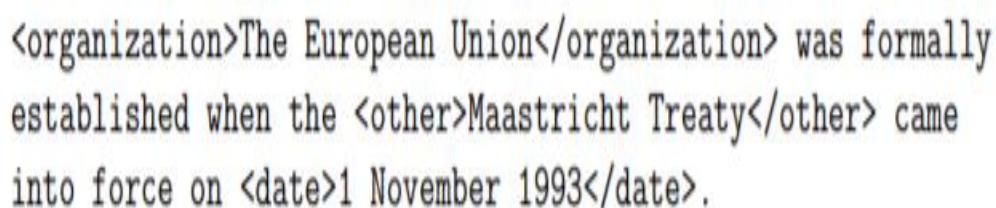
1.5 A brief history of NER

The objective of research of automatic identification of named entities in texts provides a vast pool which contains many different kinds of tactics, approaches, and some potential suggestions. One of the first research papers in this area was published by Lisa F. Rau in 1991 [13] during the 7th IEEE Conference on Artificial Intelligence Applications. In her paper, she designed and described the function of

the system is to “*extract and recognize [company] names*”. This system uses the combination of heuristics and handcrafted rules (rule-based) to do the NER tasks.

Tasks which are related to NER were introduced at MUC-6 (the sixth in a series of Message Understanding Conference) in November of 1995 for the first time [12]. Since 1995, the techniques of doing the NER tasks have shifted from rule-based methods to more advanced statistical methods with many more varieties of different modern features. And now, we have Deep Learning and some other powerful tools, like the newest tool which is called BERT, can help us to boost the accuracy of the performance.

In 2012 accuracy of performance of NER was around 90% for English [12]. The accuracy of performance varies greatly among other languages based on the properties of the targeted languages. Thus, it is relatively very important to explore some new measures to fill up this gap in the accuracy of performance among different kinds of languages (Latin letters based, character-based or some other types).



<organization>The European Union</organization> was formally established when the <other>Maastricht Treaty</other> came into force on <date>1 November 1993</date>.

Fig3. An example that presents a typical output after the process of NER [12].

Chapter 2: Introduction of the Deep Learning

2.1 Deep Learning

Deep learning (DL) is a complicated application of artificially intimated neural networks (Commonly, we just call it neural networks) to training and making the computers learn and achieve the results that we want to get for some specific tasks by using networks, usually these networks contains multiple layers [14]. By the development of the Deep Learning, it is getting deeper continuously and much more power of learning of the neural networks are being exploited, before that, the neural networks were considered could only be practical with one or two layers and could only process a really small data.

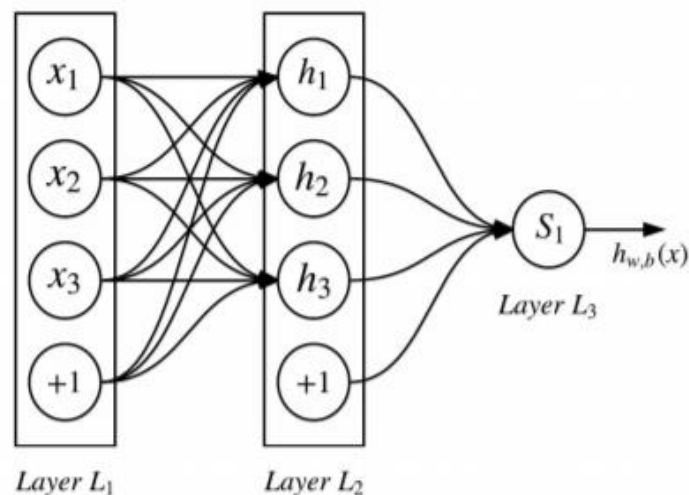


Fig4. The simple structure of the Feed Forward Neural Networks [14]

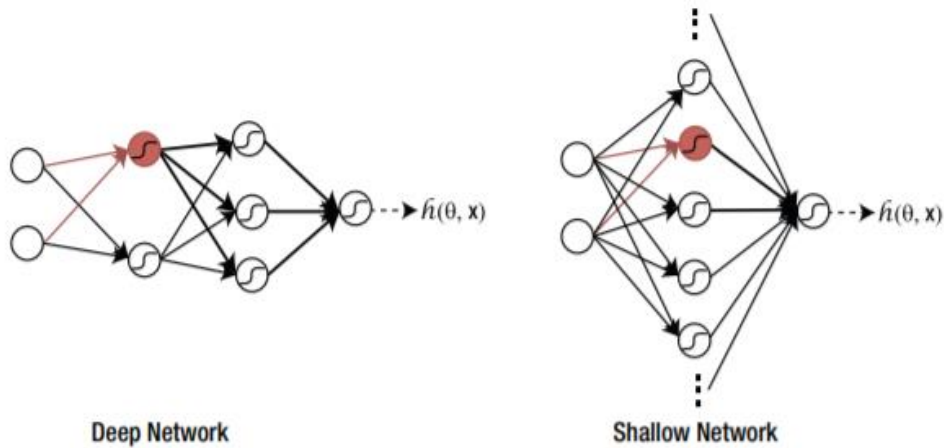


Fig5. The comparison of Deep Neural Network and the Shallow one [6]

In the technical side, the neural networks were the product by the inspiration of the structure of the brain's nerves and neurons. In detail, the organized units for the information processing procedures (called perceptron) in layers comprise the whole neural networks, and these perceptrons can work together. The networks can be trained and learn to perform the tasks follow the requirements which are instructed

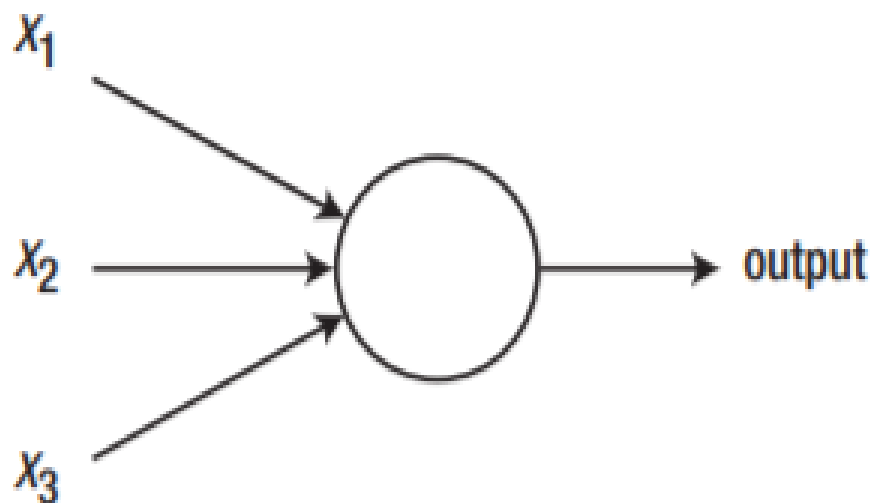


Fig6 [6]. The simple example of the perceptron

by the users (e.g. facial detection, the predication of the future stock prices, or classification) by the adjustment of the weights between the connection of these units. And, this process shows the resemblance of the learning process of a normal brain [14].

Deep learning has proved itself as a powerful tool which we compare it with the traditional ML (Machine Learning) methods in 2006 [15][16]. In 2012's ILSVRC (The ImageNet Large Scale Visual Recognition Challenge competition), the accuracy of performance by using CNN (convolutional neural networks) surpassed the second place by 10% [17]. Since then, researches that focus on this area became more and more, and the development of it is steady and continuous.

Deep learning (we also can call it the feature learning and representation learning) is actually a set of algorithms of ML which we want to train it and make sure it can get multiple-layered models from inputs of data in the forms of neural networks. The deep neural networks, technically, are the combination of several levels of mathematical operations and these operations are non-linear. Before 2006 and the [16] was published by Dr. Hinton and his co-researchers, people thought the deep learning was a hard task and may cost a lot, but the results may not be as good as using the traditional ML methods. And now, Deep Learning is a mature and powerful tool which is currently being used in many applications and many areas. The bond between this technique and our daily lives has formed.

In [18], Dr. Y Bengio suggested an idea of deep learning which is described as a greedy and layer wise pre-training process and it is unsupervised, it is designed to learn a hierarchy of features one level at a time. By using unsupervised learning, the whole process can be trained, learn, and generate models by itself that we can use the input of data without being labeled.

There are many reasons to make the whole neural networks to be deep, one of the important reasons is that, from the mathematical aspect, a nonlinear function can be represented by deep structures with fewer parameters and more efficiency. The investigations of the computation of circuits complexity provided the most formal arguments about the power of “deep structure” [15]. The investigations indicate that if there is a function which has a relatively compact representation and if it has a deep architecture, then, a very large architecture may be needed for the representation, if the architecture is not deep enough. In simple terms, multiple results of computation of complexity of circuits gave firm indications that functions that can be represented compactly, if the architecture is deeper enough, otherwise, the representation can be very large, if the representation is generated by an architecture which is not deep enough. The reason why the deep structure can make a compact representation is that every parameter of the architecture might be randomly selected or learned. These results indicate that the depth of architecture is very essential for the efficiency of the statistical aspect. Also, a representation based on hierarchy might be allowed by deep representations. And, multiple levels of latent variables provide the sharing of statistical strength in a combinatorial way.

By the inspiration of the depth of the architecture of the brain, researchers had researched decades to find a way to train the deep and multiple layered neural networks, but they failed until 2006. In 2006, researchers showed positive results of the experiment of the neural networks with one or two hidden layers, but the results of the deeper ones were still not good enough and relatively poorer than the shallow ones. The breakthrough is also considered happened in 2006, Dr. Hinton introduced Deep Belief Networks (DBNs) [19], it used a greedy learning algorithm to trains one layer at a time, also, an unsupervised learning algorithm were applied on for each layer plus the using of Restricted Boltzmann Machine (RBM). After that, some related algorithms which autoencoders were applied on were suggested, and those algorithms exactly follow the same principle which is using unsupervised learning to guide the training of intermediate levels of the whole representation, and

For NLP, one of the most commonly and successfully used deep neural networks is called Recurrence Neural Network (RNN). Stanford researching team of NLP successfully used RNN to cope with sentiment analysis for compositionality of semantics [20], it boosted the accuracy of the performance of the classification of whether a single sentence is positive or negative from 80% to 85.4%. Also, it is used for parsing, by the help of it, the F1 score of PCFG (Probabilistic context-free grammar) of the Stanford Parser was boosted by 3.8% and get 90.4%. The language models which involves neural network were also improved by applying recurrence towards the hidden layers [15], by making it overtake the smoothed n-gram models not only in respect of static perplexity, by decreasing it from 140 to 102. Also, in respect of the error rate of the words in speech recognition, it improved the accuracy by decreasing error rate from 17.2% and 16.9% (KN5 baseline and discriminative language model) to 14.4% over the benchmark tasks of the Wall Street Journal. This technique can also be applied to statistical machine translation, and as a result, it improved the BLEU (Bilingual Evaluation Understudy) score by nearly 2 points. Similarly, recursive auto-encoders which are used in the generalisation of recurrent neural networks were used in paraphrase identification in a full sentence, it attained success by almost increasing the F1 score doubly in comparison with the erstwhile state of the art method for paraphrase identification. DL can also be applied to the performance of the disambiguation of word sense [21], by using DL, the accuracy of the performance was increased from 67.8% to 70.2%.

There is a lot of hype and many absurdly hefty claims about deep learning techniques and its applications, but, in our real situation as the data mentioned above, this technique has attained marvelous successes over many complicated and challenging tasks and the improvement of the performance accuracy of the same tasks by using DL is very conspicuous in NLP area.

In [7], it indicates several benefits that DL has in coping with the problems of NLP:

The first benefit of using DL in NLP is that it has the capability to make the replacement of previous linear models with some new DL models which are better in learning from the data and exploiting nonlinear relationships, which is called the Drop-in replacement of models. In Yoav Goldberg's beginners' book about neural networks for NLP researchers, he indicated that the DL technique has already attained very promising results.

The second benefit is that DL techniques can boost the development of other new models. One convincing example is the use of the applications of RNN. They have the ability to learn precisely and can generate output from extremely long sequences of words. This approach is powerful which allows the researchers and scientists to make a big breakthrough of the traditional way of static modeling, and as the consequence of choosing the use of RNN, people achieve the best results with higher accuracy in return. In Goldberg's book and in the chapter about using the DL for NLP, he gave the comments that totally new NLP modeling chances can be provided and allowed by the advanced and complicated neural networks, for example, RNN.

Around 2014, some successes appeared in the field which are shifting from linear models with sparse inputs to nonlinear neural network models with dense inputs [22]. Other successes are more advanced, the traditional notions were changed, and many opportunities for new modeling were provided. Particularly, a bunch of manners that involve RNN enhances the reliance on the Markov property, which are wildly used in sequence models, by this, they can process the random long sequences and generate effective extractors of the feature after the processing. This improvement makes a great leap in the performance accuracy and reliability among NPL modeling, machine translations, and other NLP related applications.

The rest of the benefits are:

Feature Learning: That is, that deep learning methods can learn the features from natural language required by the model, rather than requiring that the features be

specified and extracted by an expert. And, End-to-End model with deep learning technique can be used on NLP, and it provides a more general model and achieve much better results.

Beside of the RNN, in the field of NLP, the CNN and LSTM (Long Short-Term Memory, it is very similar to the RNN) are other tools which are used by researchers usually. In this paper, the CNN and LSTM were used as part of the experiment and in the next chapter, these two neural networks will be explained in a detailed way.

2.2 Pre-trained models

In 2010 [23], Dr. Benjio and his co-researchers indicated that the pre-training can greatly enhance the accuracy of performance of the Deep Neural Networks, although the experiment they carried out mainly focused on the image processing tasks, one of the most important benefits is with the pre-training process, the results of generalization can be much better for the following process.

Generally speaking, there are two main types of pre-trained models for NLP tasks: Multi-purposed (e.g.: BERT and OpenAI), Word Embeddings (e.g.: ELMo). Here are some examples of pre-trained models:

Word embeddings are the most important and basic type of pre-trained models for the NLP related tasks by using DL to deal with. It turns each word to vectors which can be processed in the form of matrix or tensors.

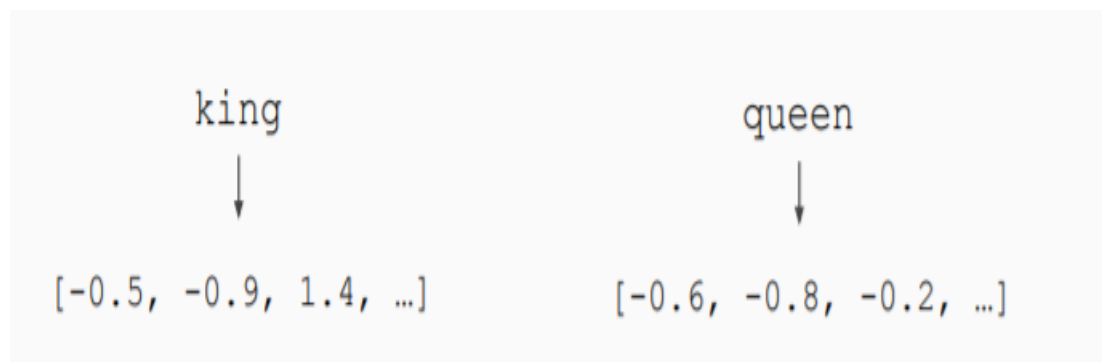


Fig8. The example of the words which are processed by word embedding [24]

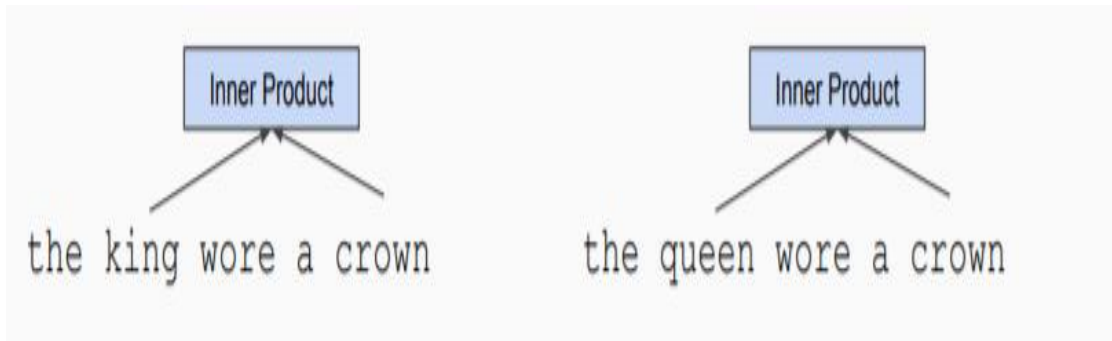


Fig9. We usually get results of the Word embeddings by using the corpus of text and with the co-occurrence statistics [24]

But the word embedding models sometimes have the problem that is they are usually applied in the way which is called text free. Therefore, if there is a word which appears in two different sentences and has a different meaning, but the vectors of the word are the same, for example, bank account and riverbank. And, the solution for it is to train the corpus with contextual representations. Here are two examples with this feature:

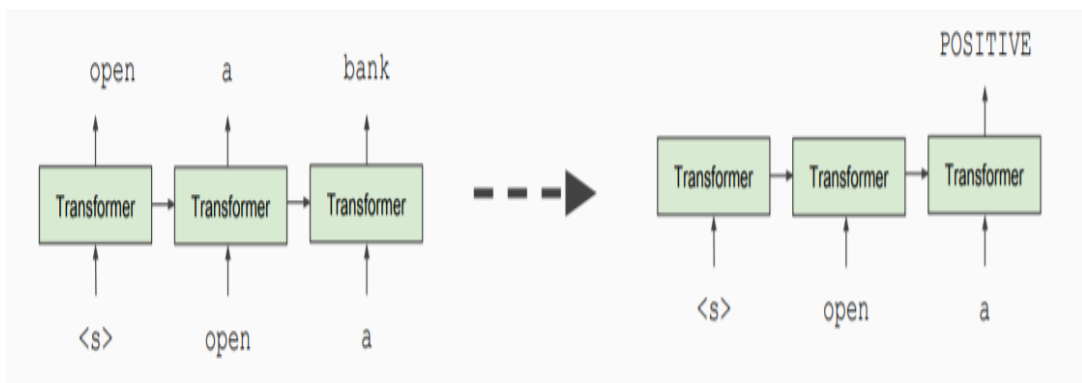


Fig10. GPT, which is developed in 2018 by OpenAI. In GPT, developers train the Deep language models with Transformer with 12 layers rather than the LSTM, then do the fine-tuning process based on the specific classification tasks [24].

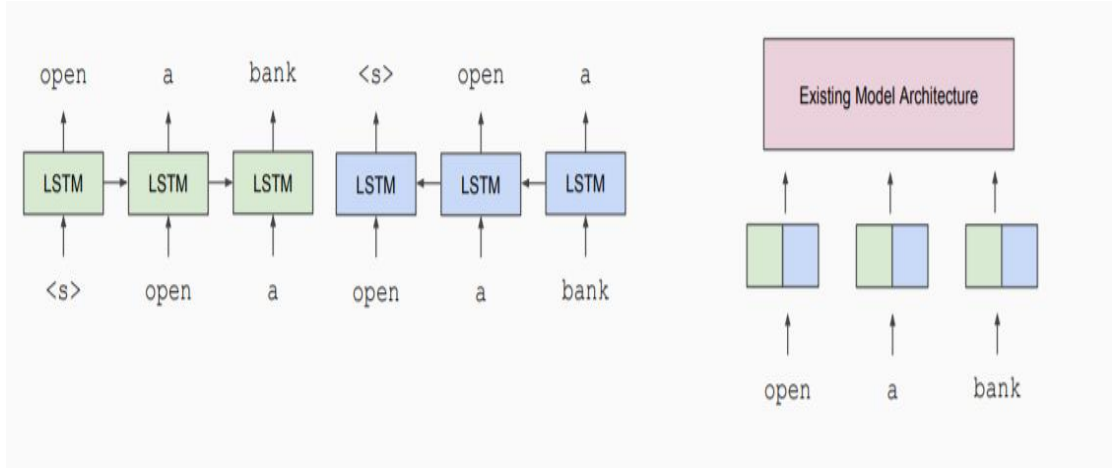


Fig11 [24]. ELMo, which is developed in 2017 by AI2 and the University of Washington. Researchers used the right to left LSTM and left to right LSTM to train the model separately and use the model as a pre-trained embedding in real tasks.

In this paper, another model was used in the experiment for the comparison which is called BERT, and the details about it will be introduced and provided in the next chapter.

Chapter 3: literature review over the techniques which were used for the experiment

In this chapter, the techniques which were used for the experiment of comparison will be explained in a detailed way. The techniques which were used are BiLSTM, CNN, Conditional Random Field (CRF) and BERT (Bidirectional Encoder Representation from Transformers).

3.1 Convolutional Neural Networks

The CNN is not a miraculous and difficult tool like what the people commonly think as, because the notion of it. After many years, it becomes very easy to be understood and the usages of CNN are very popular in many areas, especially in image processing are not very new. This technique was applied on the tasks in the visual area since the late 1980s in the last century [25]. At the primitive stage, the applications of CNN were only a few of them were used. It finally became very welcomed among the researchers and became prevailing since the mid-2000s. By the help of the development in the performance power and capability of computation which enhanced algorithms, and the advent of the era of huge information flow pushes us to pick up this technique. Indeed, the continuous improvement among the more advanced techniques of algorithms for enhancing the ability and the power of the computation of computers or machines do make the big help for the CNN touch a flourishing state and push it to grow significantly and continuously from 2012 [25].

In recent years, the applications of CNN have been used over almost all kinds of tasks which contains the relation of the computer vision. It has provided many manners for the tasks of image processing which surpass the traditional and

somehow obsolete ways of processing. The technical architecture of CNN is not very difficult to be comprehended, it is relatively easy to be understood. Generally, the neural networks involve many parts and those parts like the bricks, we just need to lay them on the top of each other.

For tasks of image processing, the architecture is : At the beginning, it is the stage for the convolutional layer, the next layer is called Max-Pooling layer, this layer is applied for the performance of the image chroma subsample (for instance, suppose the size is 2 times 2 for all patches, the patch only makes the selection of the maximum value, by this process, as the result, a four-times reduction in the image size can be given), and last “brick” is called Fully Connected Layer [26].

Usually, CNN is considered as a framework of supervised learning, and many open-sourced and extremely large training datasets (e.g.: ImageNet) are available for this network which can be used for image processing [26].

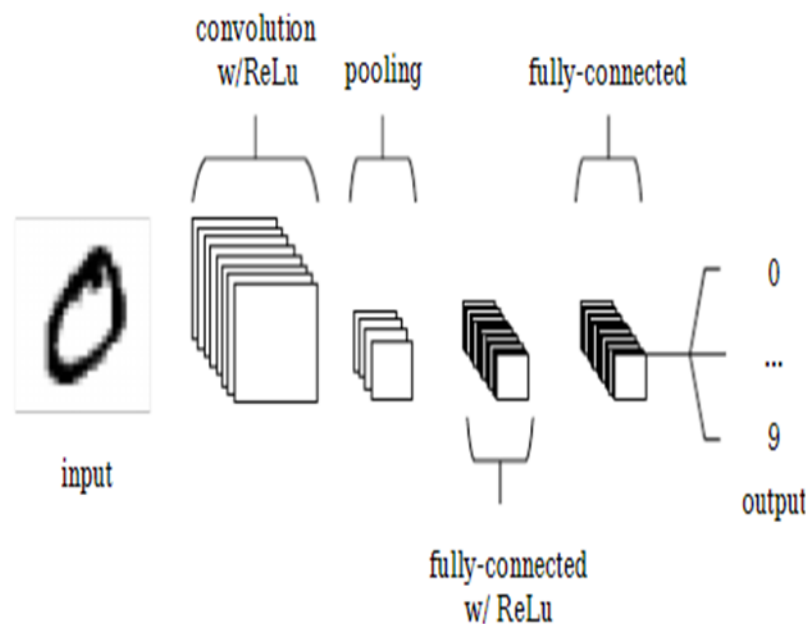


Fig12: The simple example that shows the structure of the handwriting recognition tasks [27]

The layer of CNN that is used in image processing:

1.1 Convolutional layer

It is the beginning layer of the process of convolution (before that there is an input layer), and it is used to determine the output of neurons, and the neurons which are locally linked to the regions of the input image by filtering those image pixels between the region and the weights which match the input image. The process which sets all the negative value of the matrix to 0 (The rectified linear unit) helps the whole processing by providing an activation function (e.g.: sigmoid and stepper function) to the output of activation which is generated by the input layer [27].

1.2 Pooling layer

This layer follows the above layer, it simply uses the pre-processed input from the erstwhile layer (the input may be processed by several times), and 2 times 2 (or n times n) filter will be applied on it. For an n times n square filter, it in this stage is used to figure out the maximum values of every n image pixels. After this processing, a large number of parameters within the activation can be reduced.

1.3 Fully connected layers

The role of these final layers which does the same jobs in the other Artificial Neural Networks. It can produce classes scores which comes from the process of activations. For classification tasks, the scores are used. Also, the ReLu can continuously be used among these layers in order to achieve better performance [27].

2. Training

The training process is an essential part of CNN, as the word “training” of it, this process lets the computer learn from the input data and instructions that we desire and deliberately want it to. Commonly, many algorithms were applied to learn by CNN and other standard Artificial Neural Networks, and they are also be used to make gradual adjustments over the free parameters (the biases and weights) in order to obtain and optimize the output that we desire. The most usual and widely

applied method of learning for the training process is called Backpropagation. This method does the computation over the gradient of the targeted function in order to find out the way to adjust the free parameters of the networks which can set the errors that incur adverse effects over the whole accuracy of the performance to the minimum [25]. A usually happened problem which is called overfitting, we may face it during the training of CNN, and in other Deep Neural Networks. The overfitting means that the performance of a hold-out test (validation) set is relatively poor and blow the expectation after the neural network has been trained with any sized training set. This may cause an unfavorable impact towards the capability of the after-trained model on the generalization of new data and it is a core issue for Deep Neural Networks that can be simply dealt with by using the method which is called regularization [25]. During the training process, there are two main indexes in this process, they are epochs and losses (These two indexes are used for indicating the occurrences of when the failure happens and how big is the failure during the training process). Another issue to mention is that the “losses” during the process of the validation may be a little bit higher than the “losses” during the training process.

The use of CNN for sentence modeling can be chased back to the research and thesis which was done and written by Dr. Collobert and Weston [28][29]. This research used the method which is called multi-task learning to produce the output of multi-predictions for tasks of NLP, such as POS (part-of-speech tag) tags, sentences, and phrases chunking, NER, semantic roles, words which are similar with respect to the sentiment and language model. A look-up table was applied over the transformation of each word into a vector of dimensions which is defined by the user. Therefore, the sequence of input $\{s_1, s_2, s_3, \dots s_n\}$ of n words can be transformed into a sequence of vectors $\{w_{s1}, w_{s2}, w_{s3}, \dots w_{sn}\}$, it is the result we desire which by applying the look-up table over each of its words. This can be considered as an early word embedding method that the weights were learned during the process of the network training. In [28], it also indicates that Dr.

Collobert made the research a leap in order to provide a much more general framework which is based on CNN in order to solve a hefty amount of NLP tasks in the modern era. By the success of these researches, the CNN become much more popular among the researchers and scientists which focus on NLP. It is conspicuous that CNN had already proven their power for computer image processing tasks, therefore, it was easy for people to believe in that it can also enhance the performance in the NLP area. The CNN is capable of extracting n-gram features with salience from the input sentence to generate an informative latent semantic analysis of the sentence for the later downstream tasks. This application was used by many NLP pioneered researchers and scientists, which caused a huge boom of networks which are based on CNN in the following researches.

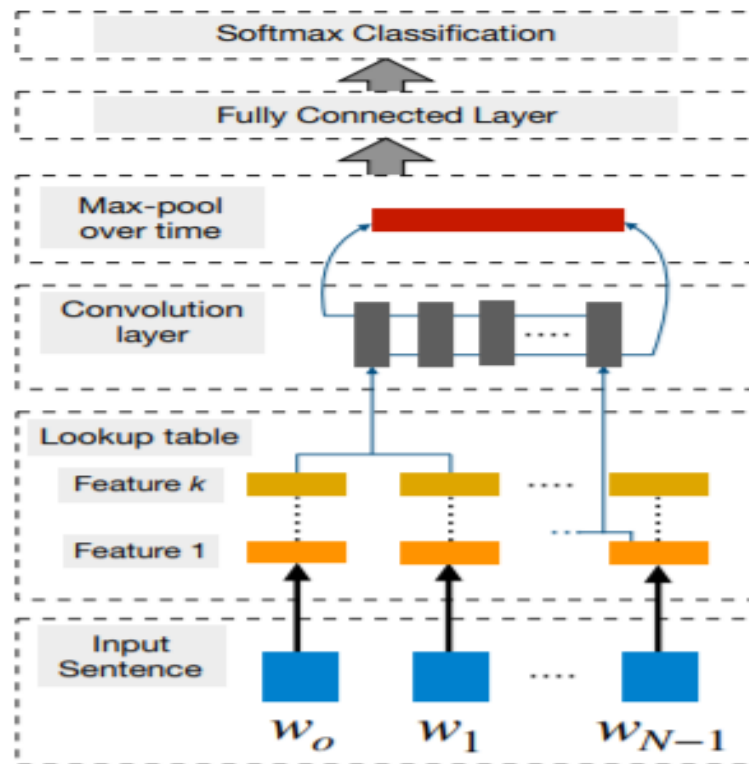


Fig 13: The performance of word wise class prediction which is done by the CNN

[28]

The following structure is a very simple one which is used in NLP tasks (for the modeling of the sentence) [28]:

For every sentence, we suppose $\mathbf{W}_i \in \mathbf{R}_d$ as the representation of the word embedding for the i^{th} word in each sentence, the d is the dimension of vector of the word embedding. Let us suppose that the sentence has n words in total, then, the whole sentence can now be represented in the form of the embedding matrix: $\mathbf{W} \in \mathbf{R}^{n \times d}$. The $\mathbf{W}_{i:i+j}$ is referred to the concatenation of multiple vectors: $\mathbf{W}_i, \mathbf{W}_{i+1}, \mathbf{W}_{i+2}, \mathbf{W}_{i+3}, \dots, \mathbf{W}_j$. The process of Convolution works on this embedding layer of input. There is a filter $\mathbf{k} \in \mathbf{R}^{hd}$ included, and it is applied to a window of h words in order to generate a new feature. Therefore, if there is a feature which is called \mathbf{x}_i is produced by the application of window of words $\mathbf{W}_{i:i+(h-1)}$, then \mathbf{x}_i in mathematical form can be presented as $\mathbf{x}_i = f(\mathbf{W}_{i:i+(h-1)} \cdot \mathbf{k}^T + \mathbf{b})$. The $\mathbf{b} \in \mathbf{R}$ here is the term of bias and the f of the function is called activation function and it is non-linear, such as, the ReLU (Rectified Linear Unit), sigmoid, and the hyperbolic tangent. Also, the filter can be used over all possible windows by using the totally same weights to generate the map of feature, therefore $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n-h+1}]$.

In the CNN, the amount of the convolutional filters, or they are called kernels, normally hundreds of them with different widths and they slide over the whole matrix of word embedding. Every filter does the job of extraction of a particular pattern of the n -gram. Normally, a max-pooling strategy follows the convolutional layer, $\hat{\mathbf{x}} = \max\{\mathbf{x}\}$, it can apply the subsampling over the input by doing a max operation on every filter. There are two major reasons for this process:

The First reason is that the output with fixed length can be provided by the max-pooling, and this kind of output is usually required for classification. Therefore, no matter what the size of the filters is, the max pooling process can always generate a fixed dimension of output from the input. The second reason is that the pooling

can decrease the dimension of the output, usually by the SVD (singular value decomposition), and has no impact towards other stuff, thus, the most salient n-gram features of the targeted sentence can be preserved. The process is done in an approach which is called translation invariant, so, every filter which is used during the pooling process is capable of extracting a specific feature (e.g.: negations or approvals in sentiment analysis) from anywhere in the targeted sentence and combine it to the final representation of the sentence.

For CNN it is essential to use the word embeddings, it can be fetched randomly or from the pre-training process over an extremely large unlabeled data of text corpus. The pre-trained one is usually be found helpful to the practical performance, especially under some unfavourable conditions, such as, when the labeled corpus data has a limited amount [30]. With the help of the word embedding, the max-pooling layer follows the convolutional layer and it is often in a layer-above-layer form in order to generate deep CNN. As a result of using CNN, it can provide the improvement and enhance the analysis of the sentence to figure out a mathematical representation in a vector form and consist of much semantic information.

The above architecture provides the capability for CNN to do the modeling of the complete the sentences and generate the representations of sentence as the result. But, in addition to sentence modeling, many other NLP tasks, such as, NER tagging, POS tagging, and SRL (Semantic Role Labeling), have the requirement of predictions which are based on words. To make the CNN suitable for these tasks, a method which is called Window approach is used, it supposes that the tag of a word is mainly dependent upon the words beside it. For every targeted word, therefore, a window with a fixed size surrounding itself is put on and only the ranging of the sub-sentence within the window will be used for the following procedures. After that, the standalone CNN is used and applied towards the sub-sentence which is mentioned above and the predictions as the results are allocated to the targeted word which located right in the center of the window. Based on this

method, in [31], researchers used a multi-layer deep CNN to detect and put tag over every word in targeted sentence whether the word is a possible aspect or not. As the result showed in the paper, the classifier gave pretty good accuracy of performance in the tasks of detection of aspect. The primary goal of classification at word-level is generally to allocate a series of labels to the whole targeted sentence. In such tasks, some modern prediction techniques can be deployed, such as conditional random field (CRF) are usually applied in order to get better capture of dependencies between neighboring class labels and finally generate a sequence of labels with Lexical Cohesion of the words with a maximum score to the targeted sentence [32]. On the other hand, to get a greater contextual range, the normal window approach usually works with a time-delay neural network (TDNN) as a combination [29]. In that case, the convolutions are performed among all windows throughout the sequence. Generally, we can put a definition over a kernel with a certain width in order to constrain the convolutions. Therefore, for the traditional window approach, it only cares about the words that are in the window around the word, which is going to be labeled, on the contrary, TDNN cares all windows of words in the targeted sentence. The structure of the layers of TDNN is similar to CNN, thus, the local features can be extracted in lower layers and higher layers for global features [29].

In addition, in [14] it gives the summary that Convolutional layers of this type of neural network act as the extractor of the features, and the extraction of local features can be done because the receptive fields (it is the representations of features in the form of a matrix) of the hidden layers are restricted to be local. It means that CNN has a particular local correlation in spatial level by putting a pattern of connectivity between neurons of neighboring layers. Therefore, by this characteristic, it is greatly useful for tasks which are about classification in the NLP area, thus, researchers can use it to find solid local clues which are related to class membership, although, these clues may appear in different positions in the input.

3.2 Bi-directional Long Short-Term Memory (BILSTM)

Indeed, the BILSTM is the derivative of the LSTM (long short-term memory), so, for LSTM, it is a very special variant of RNN [14], and it has the capability to learn the long-term dependencies. As one of the characteristics, the form of RNNs contains a chain of repeating modules. For standard and ordinary RNNs, the structure of the repeating module is normally simple, whereas, the repeating module of LSTM is much more elaborate and intricate. As the substitute for having one neural layer there are four layers that do the interaction in a way which is special. In addition to these layers, it also has two states which are called the hidden state and cell state.

In [33], it indicates that the LSTM can deal with the bottlenecks of the stability which is the problem often occurs in classic RNNs (especially the vanishing gradient), make it is very practical as an application in real life. This variant can easily use the temporal dependence well, which is learned from the data. Besides, LSTM also can use its internal memory effectively, such that the predictions are based on the recent context which follows the sequence of input and not the new input that has just been thrown to the network. For example, each observation can be presented to the networks at a time from the sequence of the input and the relevant observations which have already been seen by it can be learnt, and it can finally figure out how to use these observations to make the prediction. The simple and basic structure of the LSTM is very different from other structures of deep learning, such as, CNN. In detail, each cell of LSTM has three gates which are called 1. input gate, 2. output gate and 3. forget gate, it is very obvious, those “gates” are non-usual to be seen in other neural networks. The LSTM use these gates to do the regulations over the dataflow of training information by several selections: 1. adding information at the state of the input gate, 2. removal at the state of the forget gate, or 3. permitting it to move through to the next cell at the state of the output gate. By those gates and the selection and control of the flow of information are

the main reasons why LSTM does not have the problem of vanishing gradient which is very typical and common to be encountered in the other kinds of RNNs. By this favourable reason, LSTM is a useful and powerful tool for the modeling of non-stationary datasets.

In the mathematical form of the LSTM [33]:

$$f_t = \sigma(W_f \cdot [h_t - 1, x_t] + b_f) \text{ -----1}$$

$$i_t = \sigma(W_i \cdot [h_t - 1, x_t] + b_i) \text{ -----2}$$

$$\hat{C}_t = \tanh \cdot (W_C \cdot [h_t - 1, x_t] + b_C) \text{ -----3}$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \text{ -----4 (* means Schur product)}$$

$$o_t = \sigma(W_o \cdot [h_t - 1, x_t] + b_o) \text{ -----5}$$

$$h_t = o_t * \tanh(C_t) \text{ -----6}$$

σ is the function of the logistic sigmoid as the function of activation. For the input gate, the label of it is i , the label for output gate is o and f for the forget gate. C for the cell state and the cell output is represented as h , and, finally, the cell input is labeled as x . W is used to represent the weights of each gate and \hat{C}_t is the label for cell state after being updated, and the b means the bias. These states are propagated forward through the network, but the method for updating and changing the weights of the gates are different, it is done by the mechanism which is called backpropagation over time. The forget gate plays a critical part in avoiding the over-fitting problem by not keeping whole information which was passed from the erstwhile time steps. But, for many tasks about sequence labeling, it can be enhanced by the access of both past and future contexts (Bi-Direction). But, the hidden layer of the LSTM takes past information only (just one direction), as a result, the network does not know anything about the future [34]. Therefore, to solve this problem and boost performance, then the BILSTM was proposed.

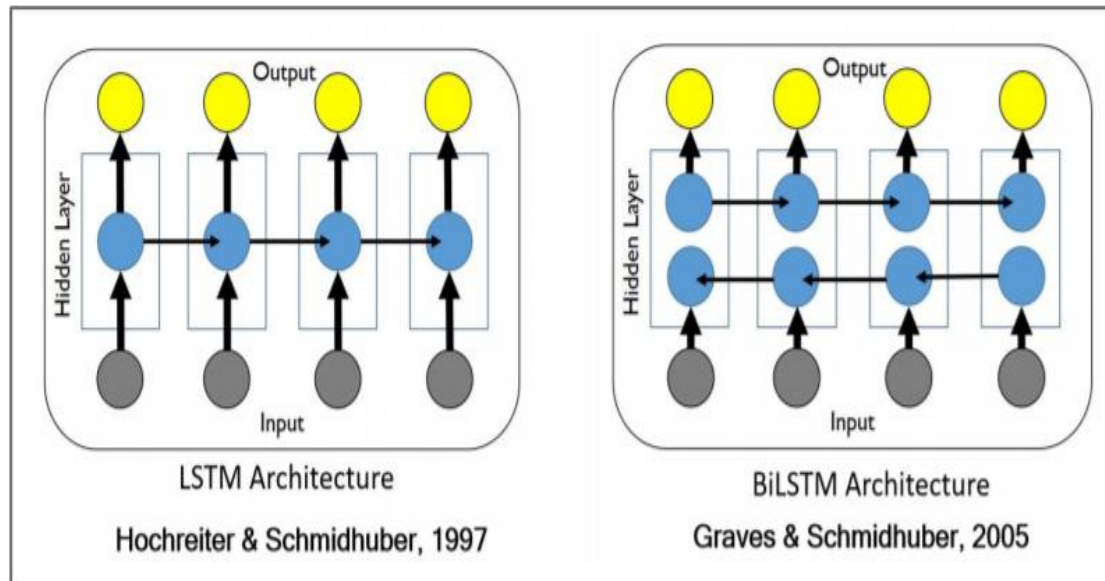


Fig14. The simple illustration of the difference between the LSTM and Bidirectional LSTM (BiLSTM). [33]

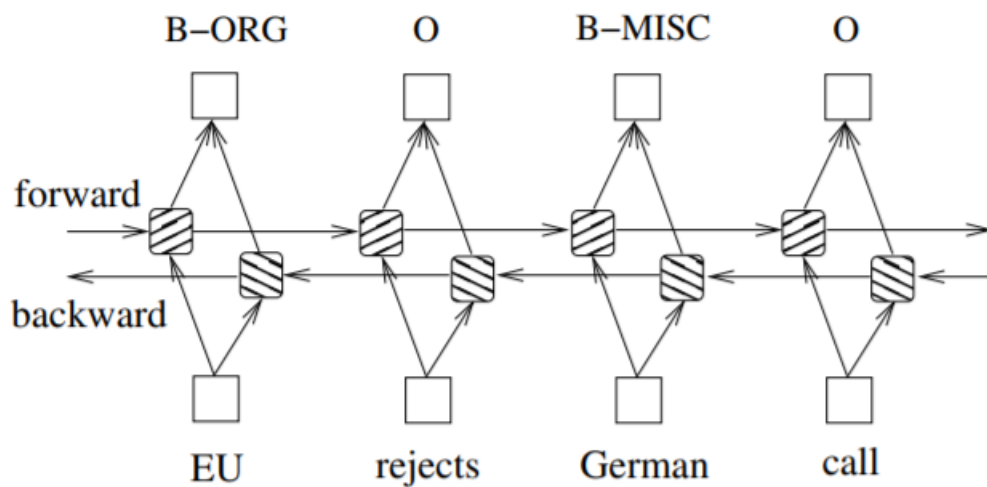


Figure 15: The BiLSTM for NER. [35]

In detail, every cell in the hidden layer of the LSTM has its input and the input is dependent on the previous cell computation of the processing of the data (therefore, each cell has a link with each other). This smart use and adjustment of memory in the targeted sequence which makes LSTM a powerful approach for modeling for

sequence. So, as an advanced form of it, the flow of information of the BiLSTM network has two ways. In this derivative, the basic idea is that each targeted sequence can be trained by two LSTM networks, one can be called forward layer and the other works in a backward direction, so it is called the backward layer, these two different hidden layers can grasp both the past and the future information respectively. Both these two layers or LSTM networks are linked to the same layer of output, and the experimental results have shown that this structure has enhanced accuracy of performance for the modeling of language and other NLP applications [33][35].

Ideally and theoretically [14], both bidirectional LSTM and LSTM are supposed to be capable of coping with long-ranged dependencies of data. But in the real situation, the long-ranged dependencies still have some problems which need to be solved. One possible problem of the classical framework with encoder and decoder may encounter is that, sometimes, the encoder might be forced to encode information that might not be fully related to the task. The problem incurs if the input is long or contains rich information and it is not possible for selective encoding. For instance, the task of summarization of the targeted source text can actually be considered as a problem of sequence-to-sequence learning, in which, the input is the source text and the output is the summarized version. Simply because it is not possible to use a fixed-size vector as an encoder for encoding all information of the targeted text because the length can be possibly very long. Similar problems also have happened in machine translation [28]. Thus, a technique which is called the Attention Mechanism can be applied. This mechanism is used in the neural networks and it is enlightened by the neural visual attention mechanism which is discovered in people. In detail, the mechanism of visual attention of humans makes us be able to focus our eyes on a definite region of an image with high resolution while looking at the surrounding image with the low resolution and we always make the adjustment of the point of focus over time. For the tasks of NLP, a similar mechanism allows the model of the language to

learn what needs to be dealt with that is based on the text input and what it has already generated by the previous procedures. By this mechanism, it does not encode the whole source text into a fixed-length vector as what standard RNN and LSTM do.

3.3 Conditional Random Field (CRF)

There are two some-how disparate approaches to harness the information of the neighboring label for the prediction of the current labels [35]. The first approach is that: for every step of time, the distribution of labels is predicated, after that, the beam search is used for the decoding process in order to find out the best sequences of the labels. i.e.: The classifier of the maximum entropy and Markov models for the maximum entropy. The second approach focuses on the level of the whole input sentence rather than just the positions of each word, therefore, the CRF gets derived from this approach.

For those tasks of sequence labeling (or other structured prediction), it is favourable to pay attention to the mutual relations between labels with their neighboring labels and jointly generate the most accurate chain of labels as the final results of the targeted input sentence. For instance, for POS tagging tasks, the adjective is very possible to be followed by a noun rather than a verb, and in NER with standard BIO2 (B=Beginning of the noun, I=Inside the noun, O= not the noun, e.g.: 61-B years-I old-O) annotation [36] **I-ORG** (Organization) does not follow **I-PER** (Person). Thus, the conditional random field (CRF) can be used to jointly model the sequence of labels, without independently decoding every label. In a mathematical way [34]:

We denote the $\mathbf{z} = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \dots, \mathbf{z}_n\}$ as a general input sequence, the \mathbf{z}_i represents the vector of the i^{th} word input. Then, we denote the $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \dots, \mathbf{y}_n\}$ as a general sequence of tags for \mathbf{z} . $\mathbf{Y}(\mathbf{z})$ represents the final set of the sequences possible tags for the \mathbf{z} . The probabilistic model of the sequence CRF which is defined in the conditional probability (it is the probability

for one event happening which has some relationship to one or many other events.)
 $p(\mathbf{y}|\mathbf{z} \text{ with } \mathbf{W}, \mathbf{b})$ over every possible sequence of label \mathbf{y} if the \mathbf{z} is given.

The complete formula is:

$$p(\mathbf{y}|\mathbf{z} \text{ with } \mathbf{W}, \mathbf{b}) = \frac{\prod_{i=1}^n \psi_i(\mathbf{y}_i - \mathbf{1}, \mathbf{y}_i, \mathbf{z})}{\sum_{\mathbf{y}' \in Y(\mathbf{z})} \prod_{i=1}^n \psi_i(\mathbf{y}'_i - \mathbf{1}, \mathbf{y}'_i, \mathbf{z})}$$

The $\psi_i(\mathbf{y}', \mathbf{y}, \mathbf{z}) = \exp(\mathbf{W}_{\mathbf{y}', \mathbf{y}}^T \mathbf{z}_i + \mathbf{b}_{\mathbf{y}', \mathbf{y}})$ are the functions of the Clique Potential, the notations $\mathbf{W}_{\mathbf{y}', \mathbf{y}}^T$ and $\mathbf{b}_{\mathbf{y}', \mathbf{y}}$ are the vectors of weight and bias with respect to the pair of labels $(\mathbf{y}', \mathbf{y})$.

For the training process of CRF, the maximum conditional likelihood estimation is applied to it. Suppose the set of training $\{(\mathbf{z}_i, \mathbf{y}_i)\}$, the likelihood is done by logarithm (So, commonly, it is called the log-likelihood) is represented in the form:
 $L(\mathbf{W}, \mathbf{b}) = \sum_i \log p(\mathbf{y}|\mathbf{z} \text{ with } \mathbf{W}, \mathbf{b})$.

The training of Maximum likelihood chooses parameters to maximize, therefore the log-likelihood $L(\mathbf{W}, \mathbf{b})$ is maximized. Then, the decoding is used to search for the highest conditional probability \mathbf{y}^* for the label sequence, and the mathematical form is $\mathbf{y}^* = \mathbf{argmax}_{\mathbf{y} \in Y(\mathbf{z})} p(\mathbf{y}|\mathbf{z} \text{ with } \mathbf{W}, \mathbf{b})$. For the model of sequence CRF (we consider the interactions between two continuous labels solely), both training and decoding processing can efficiently be dealt with by applying the Viterbi algorithm on (It is dynamic programming algorithm which is used to find the most likely sequence of hidden states by finding the optimal path through a sequence with the cost function by looking backwards through a graph of all possible paths.).

For BILSM-CRF, the CRF is connected to the end of BILSTM [37]. We suppose the \mathbf{P} to be the output of the matrix of scores which we get from the previous BILSTM process. The size of the \mathbf{P} is represented in the form: $\mathbf{n} \times \mathbf{k}$. Where the \mathbf{n} is the total number of the input word and the \mathbf{k} is the total number of the distinct label of the classification which is dependent on the standard of the tagging scheme.

And, we use $P_{i,j}$ as the representation of the score of the j^{th} label of the i^{th} word in the input sentence. Then, the score function is:

$$s(\mathbf{z}, \mathbf{y}) = \sum_{i=0}^n A_{y_i, y_{i+1}} + \sum_{i=1}^n P_{i, y_i}$$

In the upper equation the \mathbf{z} represents the input sequence and the \mathbf{y} represents the sequence of predicated tags. Since the CRF has the Markov chain property, the A in the above equation is scores of a transition matrix, thus, $A_{i,j}$ represents the transitioning score from the label i to label j . We denote the \mathbf{y}_0 and \mathbf{y}_n as the beginning and final labels of the input sentence, and we also add the predicated labels between them. Therefore, Matrix A is a square matrix and its size is $(k + 2) \times (k + 2)$.

We use SoftMax and apply it over all possible sequences of labels, and finally produce a probability of the sequence \mathbf{y} :

$$p(\mathbf{y}|\mathbf{z}) = \frac{\exp s(\mathbf{z}, \mathbf{y})}{\sum_{\hat{\mathbf{y}} \in Y_z} \exp s(\mathbf{z}, \hat{\mathbf{y}})}$$

In the training process, we figure out the maximum correct label sequence log-probability by the equation we have already mentioned before:

$$\log p(\mathbf{y}|\mathbf{z}) = s(\mathbf{z}, \mathbf{y}) - \sum_{\hat{\mathbf{y}} \in Y_z} \exp s(\mathbf{z}, \hat{\mathbf{y}})$$

where Y_z represents all possible tag sequences (it even involves those that do not follow the format of the BIO) of the targeted input sentence \mathbf{z} . By using the equation above, it is conspicuous that we can optimize our network in order to generate the correct sequence of labels as output. For the process of decoding, it can generate the final prediction of the output sequence of labels by getting the maximum score which uses the previously mentioned *argmax* formula: $\mathbf{y}^* = \text{argmax}_{\hat{\mathbf{y}} \in Y(\mathbf{z})} s(\mathbf{z}, \hat{\mathbf{y}})$.

Since, the modeling only covers the interactions bigrams between the final outputs, both the summation of the score ($s(\mathbf{z}, \mathbf{y})$) and the maximization of the posteriori of the sequence \mathbf{y}^* , both these two calculations can be computed by using Viterbi algorithm.

Also, the inputs of the CRF have a direct connection with the outputs, on the contrary, the structure of the LSTM family contains cells and other components for the recurrence. Many reports have proved that CRFs can give better accuracy of labeling [35], therefore, it is very popular and widely combined with BILSTM in such tasks.

3.4 BERT

In the late of 2018, the Google AI language team inaugurated a totally new and extremely powerful tool which can be used for multiple kinds of the tasks for multiple languages in NLP areas, including NER. This new “equipment”, has surpassed other approaches in many records, including SQuAD1.1, MultiNLI, and GLUE. Also, this model is actually a pretrained model, but it is much easier to be used than others, just need to do the process of fine-tuning for the specific tasks which are designed by the researchers.

In [38], it indicates that the processing of pre-training took a large reference of the existing approaches on the pre-training for the models of language. But, the corpus of this for BERT is way much bigger than ever before. For the BERT English version, researchers used the BooksCorpus which contains 800 million words and resources which retrieved from Wikipedia contain 2500 million words. For Wikipedia corpus, only the extraction of the text from the passages of the articles was carried out but the lists and tables were not included. With the respect to the extraction of the long and consistent sentences, it is necessary to take the corpus in the document form for the pre-training process rather than the one in the form of the shuffled sentences (e.g.: 1 Billion Word Language Model Benchmark).

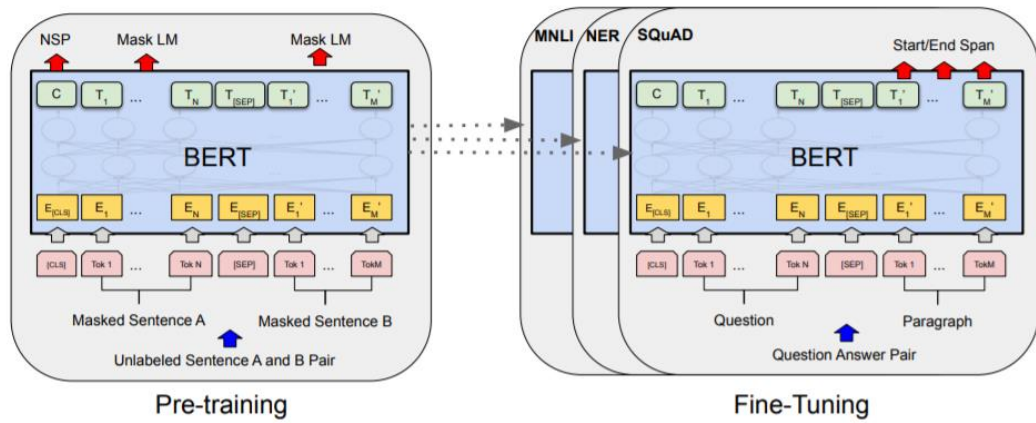


Fig16 [38]: A simple illustration which shows the steps of the process of the BERT

For the Fine-tuning part, the researchers indicate this process is somehow easy to be grasped and can be done without many difficulties because the BERT adopted the mechanism which is called self-attention mechanism from another technique which is called Transformer (First introduced in 2017 [39], it greatly enhanced performance of the NLP by only using the encoders and decoders, and no need to use the structures of the CNN and LSTM, because, the LSTM always has the potential problem which is called gradient vanishing, also, they are way too complicated to be built up) provides the ability for BERT to be suitable for many different kinds of following tasks, no matter if the following tasks use one sequence of text or in the form of pairs by just changing the inputs and outputs.

For those tasks which contain text pairs, the traditional and usual approach is to encode those pairs separately before bi-cross attention is applied. In BERT, it uses the self-attention mechanism which was mentioned above to combine these two processes into one, by using this mechanism to encode the pair of text or sentences, the bi-cross attention between the pair gets included effectively either.

The comparison between the pre-training process and the process of the fine-tuning shows that the second one is relatively easy and costs less. In researchers' paper [38], they clearly indicated that every result in it can be reproduced by using the same model, within 1 hour if the program runs on a TPU, or a couple of hours by using the GPU.

Chapter 4: Experiment and analysis

In this chapter, the experiment and results will be explained in a detailed way. This experiment is a comparison between these two technologies' performance. But we mainly focused on the BERT, because it is very new and powerful which was approved by many DNN for NLP users and researchers, therefore, it deserves good research.

4.1 Working environment

The whole experiment was conducted on my laptop, here are the computer configuration and other essential environments:

GPU: GeForce GTX 1070 (the laptop version 8GB)

Main programming language: Python 3.6

Main framework or model: PyTorch and pre-trained -BERT-base

Dataset for the BERT and BiLSTM and word embedding file for BiLSTM:

CoNLL 2003 and Glove.6b.100

4.2 A brief introduction about the mutually used dataset and pre-processing of the data for the BERT

The CoNLL 2003 was deliberately designed for NER tasks, and it is very renowned and widely used for the English NER. There are three files for every targeted

language: training file, validation file, and test file and there is a file for the unannotated data which is huge, for the experiment which we did we only used these three files. We used the training file for the process of the training of the neural networks and the validation file was used for the process of the tuning of the parameters (e.g.: weights, learning rate) of the networks, and finally, the test file was used for testing the best parameters of the model which we got from the above two processes to check if the model with these parameters reaches our expectation or not.

For the English corpus, the resource of it is called Reuters Corpus. The components of this resource are Reuters news stories which were taken from August 1996 to August 1997. For the training file and validation file, the texts were from the end of August of 1996. And, for the texts for the file of testing, the date of the resource texts is from December of 1996 [40].

Also, the contents of these three files consist of 14987 sentences for training file, 3466 sentences for validation file, and 3684 sentences for the test file. Here is the overview of the total number of different labels in different files:

English data	LOC	MISC	ORG	PER
Training set	7140	3438	6321	6600
Development set	1837	922	1341	1842
Test set	1668	702	1661	1617

Fig17[40]: MISC means miscellaneous, PER means person, ORG means organization, and LOC means location.

For BILSTM, the dataset does not need to be further prepared [38], but for BERT the pre-processing of the data is needed. For each input sentence, there are two additional tokens added: [CLS] at the beginning of the input and [SEP] at the end of the sentence.



Fig18: The simple example of the correct form of the BERT input. [38]

4.3 The results of the experiment:

The results of the experiment are shown in table 1:

	Precision	Recall	F1 score
BERT	89.60%	91.54%	90.55%
BILSTM-CNN-CRF	91.02%	90.49%	90.76%

Table1: the results of the comparison between these two techniques.

More information about each label for BERT are shown in table 2:

	Precision	Recall	F1 score	Support
MISC	77.08%	81.91%	79.42%	702
PER	95.24%	95.24%	95.24%	1617
LOC	91.97%	93.35%	92.96%	1668
ORG	87.04%	90.19%	88.59%	1661

Table 2: The percentage for every label

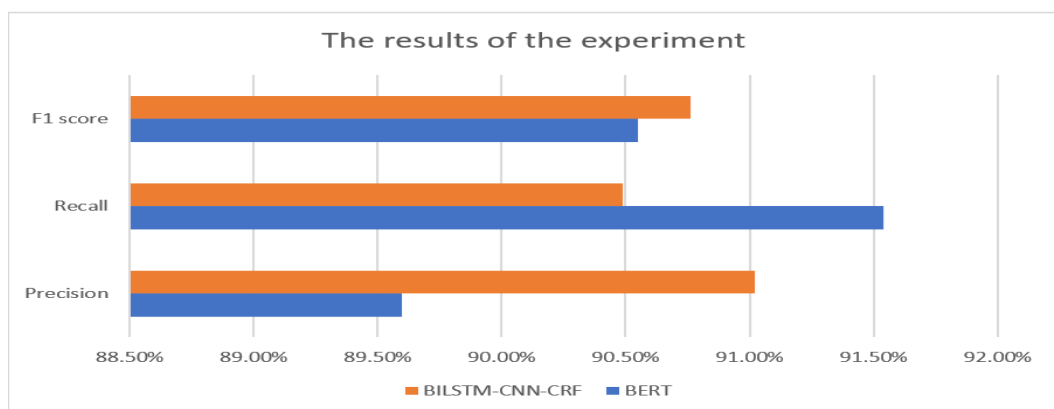


Figure19: The results in the form of a chart

The metrics we used for comparison are called Macro-average, because we want to check the overall performance for the whole model and the variance of the size between the class of the test file is not huge, except the MISC class.

From the above two tables and the chart, it shows that the overall performance of the BILSTM-CNN-CRF is slightly better than the pure BERT. And the performance of the BERT over each label shows that the metrics about the MISC are quite low but the PER is very high. The possible reason is that the total number of words or phrases with the MISC labels in these three files is way much less than others. The overall performance of the BERT-base matches our expectations and nearly the same as the results of the original paper [38] (not totally the same, our results are a little bit lower. For the results of the reproduction of the experiment, the minor difference is always possible). In the above chart, the F1 score of the BILSTM-CNN-CRF is slightly higher than the pure BERT-base (only 0.22%), but the structure of BILSTM-CNN-CRF is way much more complicated than BERT, and BERT can also be used with CRF or BILSTM-CRF in order to get a remarkable accuracy.

The formulae of the Macro-average for Precision, Recall, and F1 score are:

$$\mathbf{MacroP} = \sum_{n=1}^i \frac{\mathbf{TP}_n}{\mathbf{TP}_n + \mathbf{FP}_n}$$

$$\mathbf{MacroR} = \sum_{n=1}^i \frac{\mathbf{TP}_n}{\mathbf{TP}_n + \mathbf{FN}_n}$$

$$\mathbf{F} = 2 \times \frac{\mathbf{MacroP} \times \mathbf{MacroR}}{\mathbf{MacroP} + \mathbf{MacroR}}$$

The TP, FN, FP represent: True positive, False-negative, and False positive.

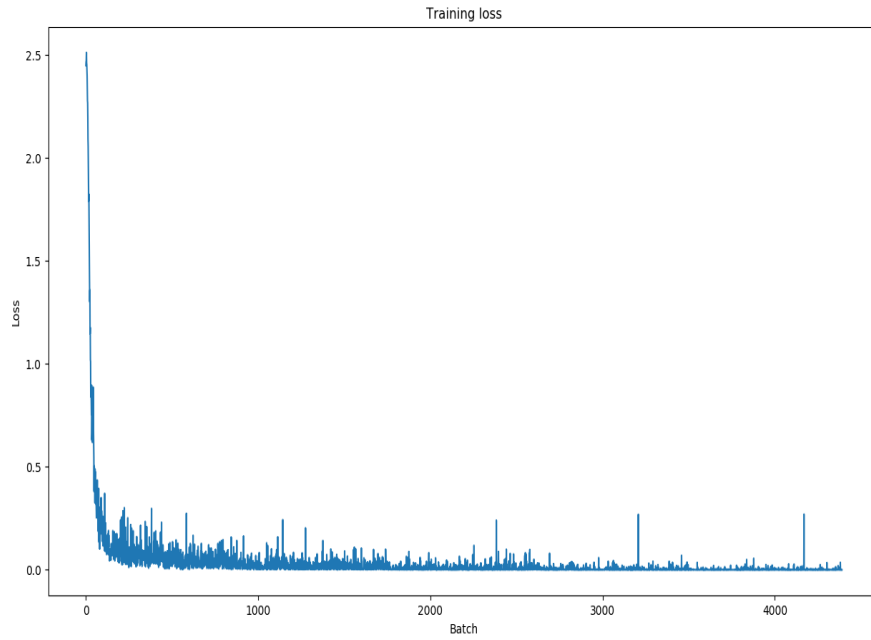


Figure20: The training loss of the BERT of the Conll2003 dataset

In the experiment, we set the total epochs to 5 which is suggested in [38] for the training process, and the total batches are more than 4000. From the above figure, it shows a nice and smooth curve of the training loss, the loss dropped extremely fast which indicates that the BERT is really powerful in dealing with the NER tasks. Also, the technique which is called dropout is used in order to avoid the overfitting issue, and the results are favourable. For the BERT, we used Adam as the optimizer, because it is fast and can save time in training, even though, the total time consumed in training was around 4 hours.

4.4 Limitation of the BERT experiment

The biggest limitation is the configuration of the computer. For BERT, there are two versions of it, the first one is the one we used for our experiment and the other one is called BERT-large. For the base version: hidden size is 768, 12 layers, 12 heads of the self-attention and the total number of parameters is 110 million, and for the large version, the number is: 1024, 24, 16, and 340 million [38][42]. Therefore, the large version requires a powerful GPU to train. From the homepage

of the Google research on GitHub [41] it clearly indicates that currently, it is not doable for the reproduction of the results if the RAM size of the GPU is between 12GB to 16GB (Our GPU has only 8GB), the reason is the memory is too small for the batch size. Therefore, we cannot use the large version in the experiment, which is definitely more powerful and better results can be generated.

Conclusion and future research

The NER is very important among the NLP tasks. Our experiment proved that BERT is powerful which can change the history of the NLP, also, the success of it makes us confident that the pre-trained model of multiple purposes will be more prevailing in the near future. Although the results of our experiment over the pure BERT is lower than BILSTM-CNN-CRF, by the limitation of our computer, only the base version was used, but the structure of BERT is much less complicated and easy to build up to cope with other tasks if we compare it with the other testing subject. More importantly, BERT can also work with the CRF or BILSTM-CRF which can be called BERT-CRF and BERT-BILSTM-CRF, these two can produce far better results without doubts. But the problem is conspicuous, it requires users to have GPUs with large RAM to train it, the optimization of it can be a future research direction.

And, in October 2019 [42], the Google team once again made another record in the history of the NLP, they developed a lite version of BERT which is called ALBERT, in their experiment they used ALBERT-xlarge (contains 60 million parameters) and compared it with BERT-xlarge (with 1270 million parameters). The team found that the bigger the model is and the more parameters it has does not mean the better performance it might give. In the experiment the RACE (**Re**Ading **Com**prehension **Ex**amination) dataset was used, and the comparison results between these two models are 82.3% and 54.3%. Therefore, it shows a way for the future development of the pre-trained model which means they should be

built in a compressed way, and many methods that have already been used in the ALBERT might be applied. By following the notions of it, we may do not have to rely on the size of parameters only and BERT applications for cell phones may be a probable direction of future researches.

Reference

1. D. Khurana, A. Koli, K. Khatter, S. Singh, 2017, *Natural Language Processing: State of The Art, Current Trends and Challenges*, arXiv:1708.05148
2. KS. Jones, *Natural language processing: a historical review*, 2001, <https://www.cl.cam.ac.uk/archive/ksj21/histdw4.pdf>, University of Cambridge
3. R. Kibble, *Introduction to natural language processing*, <https://london.ac.uk/sites/default/files/study-guides/introduction-to-natural-language-processing.pdf>, University of London
4. R. Bertram, 1969, *SIR: A Computer Program for Semantic Information Retrieval*, <https://dspace.mit.edu/handle/1721.1/6904>, MIT
5. S. Scalise, E. Guevara, *The Lexicalist Approach to Word-Formation and the Notion of the Lexicon*, 2005, *Handbook of Word-Formation. Studies in Natural Language and Linguistic Theory*, vol 64, pp147-187. Springer, Dordrecht
6. P. Goyal, S. Pandey, K. Jain, *Deep Learning for Natural Language Processing- Creating Neural Networks with Python*, 2018, Apress, Berkeley, CA
7. J. Brownlee, *Deep Learning for Natural Language Processing- Develop Deep Learning Models for Natural Language in Python*, http://ling.snu.ac.kr/class/AI_Agent/deep_learning_for_nlp.pdf
8. K. Verspoor, K.B. Cohen, 2013, *Natural Language Processing*, *Encyclopedia of Systems Biology*. Springer, New York
9. T. Young, D. Hazarika, S. Poria, E. Cambria, 2017, *Recent Trends in Deep Learning Based Natural Language Processing*, arXiv:1708.02709
10. C. Dozier, R. Kondadadi, M. Light, A. Vachher, S. Veeramachaneni, R. Wudali, 2010, *Named Entity Recognition and Resolution in Legal Text*, *Semantic Processing of Legal Texts. Lecture Notes in Computer Science*, vol 6036, pp 27-43, Springer, Berlin, Heidelberg

11. *Information Extraction and Named Entity Recognition*, [https://web.stanford.edu/class/cs124/lec/Information Extraction and Named Entity Recognition.pdf](https://web.stanford.edu/class/cs124/lec/Information%20Extraction%20and%20Named%20Entity%20Recognition.pdf), Stanford University
12. M. Konkol, 2015, *Named Entity Recognition*, <https://otik.uk.zcu.cz/bitstream/11025/23711/1/Konkol-PhDThesis.pdf>, University of West Bohemia
13. D. Nadeau, S. Sekine, 2007, *A survey of named entity recognition and classification*, *Linguisticæ Investigationes*, Volume 30, Issue 1, pp 3-26, John Benjamins Publishing Company
14. L. Zhang, S. Wang, B. Liu, 2018, *Deep Learning for Sentiment Analysis: A Survey*, arXiv:1801.07883v2
15. M. Mohammadi, R. Munda, R. Socher, *Deep Learning for NLP*, 2015, https://cs224d.stanford.edu/lecture_notes/LectureNotes4.pdf, Stanford University
16. G. E. Hinton, R. R. Salakhutdinov, 2006, *Reducing the Dimensionality of Data with Neural Networks*, vol 313, Issue 5786, pp 504-507, Science
17. A. Krizhevsky, I. Sutskever, G. E. Hinton, 2012, *ImageNet Classification with Deep Convolutional Neural Networks*, *Advances in Neural Information Processing Systems* 25, NIPS 2012
18. Y. Bengio, A. Courville, P. Vincent, 2012, *Representation Learning: A Review and New Perspectives*, arXiv:1206.5538
19. G. E. Hinton, S. Osindero, Y. Teh, 2006, *A Fast Learning Algorithm for Deep Belief Nets*, *Neural Computation*, vol 18, Issue 7, pp 1527-1554, MITP
20. R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts, 2013, *Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank*, https://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf, Stanford University
21. A. Bordes, X. Glorot, J. Weston, Y. Bengio, 2012, *Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing*, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, vol 22, pp 127-135. PMLR
22. Y. Goldberg, 2017, *Neural Network Methods for Natural Language Processing*, Morgan & Claypool Publishers
23. D. Erhan, A. Courville, P. Manzagol, P. Vincent, Y. Bengio, 2010, *Why Does Unsupervised Pre-training Help Deep Learning?*, *Journal of Machine Learning Research*, vol 11, pp 625-660, JMLR

24. J. Devlin, *Bidirectional Encoder Representations from Transformers*, <https://nlp.stanford.edu/seminar/details/jdevlin.pdf>, Stanford University
25. W. Rawat, Z. Wang, 2017, *Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review*, Neural Computation, vol 29, Issue 9, pp 2352-2449, MIT
26. E. David, N. S. Netanyahu, 2017, *DeepPainter: Painter Classification Using Deep Convolutional Autoencoders*, arXiv:1711.08763v1
27. K. O'Shea, R. Nash, 2015, *An Introduction to Convolutional Neural Networks*, arXiv:1511.08458
28. T. Young, D. Hazarika, S. Poria, E. Cambria, 2017, *Recent Trends in Deep Learning Based Natural Language Processing*, arXiv:1708.02709
29. R. Collobert, J. Weston, 2008, *A unified architecture for natural language processing: deep neural networks with multitask learning*, Proceedings of the 25th international conference on Machine learning, pp 160-167, ICML
30. Y. Zhang, B. Wallace, 2015, *A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification*, arXiv:1510.03820
31. S. Poria, E. Cambria, A. Gelbukh, 2016, *Aspect extraction for opinion mining with a deep convolutional neural network*, Knowledge-Based Systems, vol 108, pp 42-49, Elsevier
32. A. Kirillov, D. Schlesinger, W. Forkel, A. Zelenin, S. Zheng, P. Torr, C. Rother, 2015, *Efficient Likelihood Learning of a Generic CNN-CRF Model for Semantic Segmentation*, arXiv:1511.05067
33. A. T. Mohan, D. V. Gaitonde, 2018, *A Deep Learning based Approach to Reduced Order Modeling for Turbulent Flow Control using LSTM Neural Networks*, arXiv:1804.09269
34. X. Ma, E. Hovy, 2016, *End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF*, arXiv:1603.01354
35. Z. Huang, W. Xu, K. Yu, 2015, *Bidirectional LSTM-CRF Models for Sequence Tagging*, arXiv:1508.01991
36. E. F. T. K. Sang, J. Veenstra, 1999, *Representing text chunks*, Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics, pp 173-179, EACL
37. G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, C. Dyer, 2016, *Neural Architectures for Named Entity Recognition*, arXiv:1603.01360

- 38. J. Devlin, M. Chang, K. Lee, K. Toutanova, 2018, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, arXiv:1810.04805
- 39. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, 2017, *Attention Is All You Need*, arXiv:1706.03762
- 40. E. F. T. K. Sang, F. D. Meulder, 2003, *Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition*, arXiv:cs/0306050
- 41. <https://github.com/google-research/bert>
- 42. Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, 2019, *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*, arXiv:1909.11942