NANYANG TECHNOLOGICAL UNIVERSITY

FINAL YEAR PROJECT REPORT

Keyword and Named Entity Recognition on Emergency Call Texts

Examiner:	**Prof Jagath Chandana Rajapakse**
Supervisor:	**Assoc Prof Chng Eng Siong**
Author:	**Hu Wanyu**
Matric Number:	**U1721068J**

School of Computer Science and Engineering

October 2020

Table of Contents

Abstract

This report summarizes the work that has been done in the Final Year Project on the topic Keyword and Named Entity Recognition on Emergency Call Texts. With the development of Artificial Intelligence, much more attention than ever before has been paid to the idea of AI-Oriented systems that can be used to accomplish tasks. This report utilize one of the methods of Artificial Intelligence (AI), specifically the Named Entity Recognition (NER) technique on Emergency Call Texts. The model used for this project is the Bidirectional Representations from Transformers (BERT) model. 75% of the Emergency Call Texts dataset is obtained online. The remaining 25% is generated using a custom Data Generation Model design using BERT. The dataset is fed into the model and the result is evaluated by confusion matrix and F1 score. The predicted result is visualized using a Web Application design using Python Streamlit package. The named entities will be highlighted based on the result from the model and then presented in the web application.

Keywords:

Named Entity Recognition (NER), Bidirectional Representations from Transformers (BERT) model, Emergency Call Text, F1 Score, Streamlit

Acknowledgements

I would like to express my sincerest gratitude to the following people for their kind guidance towards the successful accomplishment of this Final Year Project

Associate Professor Chng Eng Siong, who is my supervisor. He guided me throughout the whole project. His opinions, suggestions towards the project topics and requirements, and his feedbacks on my research schedules and works have greatly helped me in accomplishing this project.

Mr. Andrew Koh Jin Jie, who is Ph.D candidate, shared with me his understanding on various techniques on Natural Language Processing and Deep Neural Network, such as methods to calculate softmax in a BERT model.

Lastly, I would like to thank my schoolmate, **Mr. Andy** and **Ms Nikole**, for spending time with me to go through their Final Year Project experience.

Thank you all.

Acronyms

AI	Artificial Intelligence
NLP	Natural Language Processing
NER	Named Entity Recognition
BERT	Bidirectional Representations from Transformers
LSTM	Long Short-Term Memory
ELMo	Deep contextualized word representations
CNN	Recurrent Neural Network
F1 Score	F-score or F-measure

List of Figures

List of Tables

Chapter 1

1. Introduction

1.1. Background

Over the past 10 years, the growth of Artificial Intelligent (AI) technology has been staggering. Much more attention than ever before has been paid to the idea of AI-Oriented systems that can be used to accomplish many of those human-oriented tasks. AI technology has drastically helped humans to boost productivity and efficiency [1] in areas such as Healthcare, Transportation, Business, Speech Recognition and Face Detection tasks.

Thanks to this information era, large amount of data is readily available on the internet. Together with increasing computation power of computing machines, fields like Information retrieval, Text mining, Natural Language Processing, Computer Vision, can take the advantage of huge amount of those available data to learn the general pattern of any specific domain by calculating the stochastic probability of the any task. One of the main research areas of the such task is Named Entity Recognition.

Named entities that are used normally includes person name, organization name, location address. This project focuses on named entity recognition on Emergency Call Texts. So another named entity “Medical Condition” is also needed to be recognised. An example of the highlighted named entity in an emergency call text is as follows:

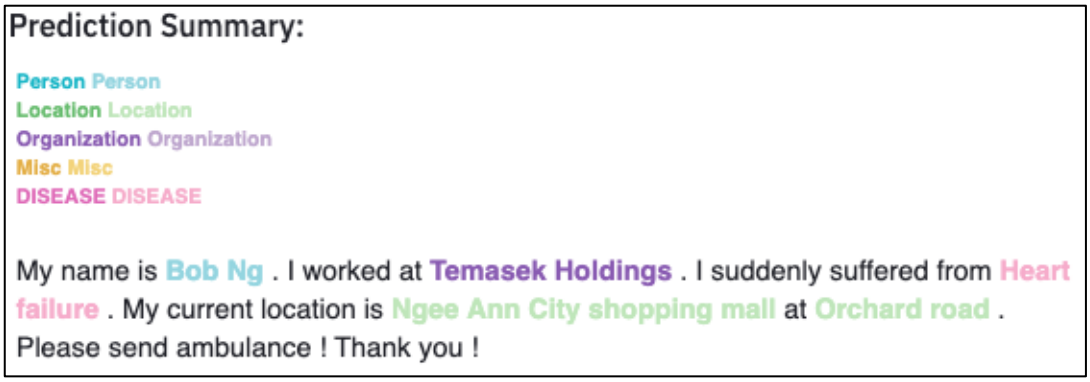


Figure 1An Example of Named Entity Recognition (NER)

The person's name is highlighted in blue; Organization's name is highlighted in purple; Medical condition is highlighted in red and location address is highlighted in green.

1.2. Motivation

NER is domain-specific, which requires to be trained with a large amount of relevant corpus. So far, many NER models are trained using various corpus to solve different domain-specific problems. For example, OntoNotes5 [2], is used to allow the model to classify news content from news articles, However, little research has been done on how to deploy NER on conversational texts from Emergency Calls.

According to the statistics published in Clinical Negligence Team UK, "that potentially around 2,500 lives a year are being lost as a result of delayed ambulance response times and the inability to resuscitate patients at the scene" [2]. Moreover, in Singapore, although "Calls are answered within 10 seconds and appropriate resources are deployed within 80 seconds", the Officers are manually inputting the patient's information into the system [3]. It causes a lot of delay in sending the ambulance to the patients. It is crucial to ensure minimal amounts

of time is spent at the ambulance call centre, especially if the patients are suffering from critical conditions. One life lost is one too many and we must avoid such tragedy happening due to human delay. How can we use AI to improve it?

Therefore, our study aims to extend beyond previous researches and focuses on designing and deploying new domain-specific NER techniques for Emergency Call texts. The output of this project should have two capabilities. Firstly, given a sequence of Emergency Call sentences that was extracted from Speech-to-Text converter, Our model should automatically extract key entities from this Emergency Call texts, such as Person name, Location, Medical condition. It updates the Ambulance call system simultaneously. Nearby ambulance operators and doctors will then be notified immediately with patients' location and diseases. Hopefully, it will allow the patients to be treated in the shortest period, which can save life.

1.3. Hypothesis

In this paper, one hypothesis has been put forward as follows:

There must exist at least one named entity in the user input sentences. In other words, if the user input sentences contain the following named entities, such as Person, Location, Medical Conditions, or Organization, the model is able to identify and extract the named entities correctly. Therefore, the user can save time in typing the keywords in the ambulance call system.

1.4. Assumption

1) One assumption of the project is that the speech-to-text Converter developed by researcher in the lab is capable of converting the spoken emergency call conversations to text with relatively high accuracy. The model is trained on natural spoken language, with standard syntax. It is capable of predicting on the sentences with less error.
2) Second assumption of the project is that the pre-trained model we used in the project is open source and readily available online. The pre-trained model we used in the project is the Bidirectional Representations from Transformers (BERT) model. It is available as an open source project developed by Google AI Language [4] at the point of writing this dissertation.

1.5. Objectives

Performing Named Entity Recognition on Emergency Call Text is difficult because there are no freely available corpora that contains the information of such type. The corpora are either confidential or incomplete. Furthermore, there are no right or wrong solutions to recognize the named entities perfectly from the emergency call texts. Our aim is to achieve the accuracy as high as possible, before overfitting occurs, so that the majority of the named entities can be recognized and extracted correctly from emergency call texts and can be used for ambulance call system.

The objectives have been put forward as follows:

1) Build the Named Entity Recognition model from an existing pre-trained language model, BERT model.
2) Search for and self-generate relevant annotated corpora for training.
3) Followed by training the model with large amounts of relevant corpora.
4) The resulting model is expected to be capable of identifying all the named entities when typing Emergency Call Texts.
5) A User Interface should be established, which can take in the sentences and annotate and display all the relevant entities in the sentence as accurately as possible.
6) The resulting named entities should help the user to identify key information in sentences and react immediately.

The detail can be shown as follows:

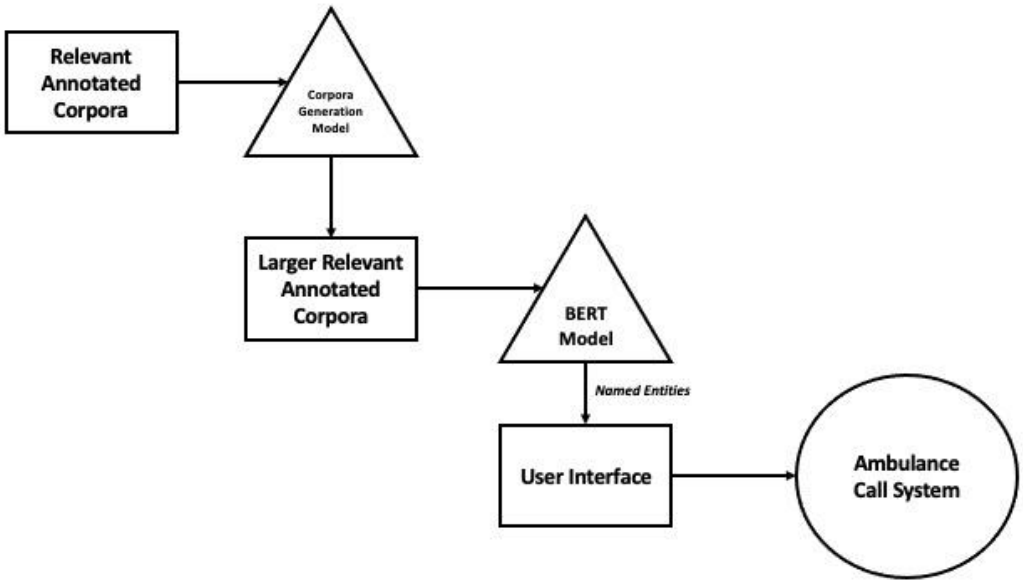


Figure 2 Flow of the Processes

1.6. Major Contribution of Dissertation

1) Proposed Named Entity Recognition method that uses BERT pre-trained model.
2) Built a model by using a BERT pre-trained model and added a fine-tune layer on top for project specific tasks.
3) Generation of training corpora by randomly selecting from existing 30% of the sample annotated corpora, and then work on data cleaning.
4) Trained the model with the corpora and analyze the performance to prevent the model from overfitting.
5) Built a User Interface and server that can host and display the predicted result from the model.
6) Compared and analyzed the predicted result with the target result and find areas for improvements.

1.7. Organization of the Dissertation

The report is organized as follows:

- Chapter 1introduces the background of the characteristics and advantages of Named Entity Recognition and the motivation to work on NER for Emergency Call Text. Followed by hypothesis, assumptions, objectives, main contribution and main structure of the dissertation.
- Chapter 2 shows my researches for this project, which includes the definition of Named Entity Recognition, Tagging Schemes, Current NER systems, and most importantly, the detailed evaluation of the BERT model.
- Chapter 3 describes the implementations of the project, which includes the data generation and cleaning process, model parameters for training.
- Chapter 4 presents the experiment result as well as the constructing of the User Interface.
- Chapter 5 concludes the report by identifying the potential shortfalls of this project and providing a set of solution and possible future recommendations to overcome these problems for future use.

Chapter 2

2. Literature Review

2.1. Named Entity Recognition (NER)

Named Entity Recognition (NER) is one of the main research areas in AI. NER is a subtask of information extraction. It is the process of identifying various types of real-world information, or entities, from unstructured text, such as spoken languages. A Neural Network Language Model is required for NER tasks. The model is trained over a large amount of relevant corpora. The resulting model should have the capabilities of extracting and classifying relevant words or tokens into predefined entities categories, such as Person, Location, Event, Organisations, Datetime and so on. The figure below shows an example of Named Entity Recognition of an article by Spacy [5]:

Figure 3Example of NER by Spacy

In this project, a new entity is created call "Disease" representing the medical condition. It extracts all the injuries, medical conditions from the sentence.

2.2. Tagging Schemes

There are various Tagging schemes available for Named Entity Recognition. They are BIO, BILUO/BIOES. [6]

- **BILUO/BIOES**

  BILUO encodes the Beginning, Inside, Last/Ending token of a multi-token chunk while U/S represent Unit-length chunk. O represents the token belongs to no chunk. The feature below shows an example of a sentence tagged by BILUO scheme.

```
Alex S-PER
is O
going O
with O
Marty B-PER
A. I-PER
Rick E-PER
to O
Los B-LOC
Angeles E-LOC
```

Figure 4 BILUO Tagging

- **BIO**

  Similar to BILUO, BIO encodes the Beginning, Inside token. O represents the token belongs to no chunk.

```
Alex I-PER
is O
going O
to O
Los I-LOC
Angeles I-LOC
in O
California I-LOC
```

Figure 5 BIO Tagging

2.3. Supervised and Unsupervised Learning

In Supervised learning, there are training and testing dataset. The model continuously learns the patterns from the input training dataset, calculate the probabilities, make predictions and validate on test dataset. The hyperparameters are updated after every epoch. The prediction result is compared with the target label. The eventual performance is weighted by accuracy. If the accuracy of the model is satisfactory, the learning will come to a stop.

For Unsupervised learning, there are no target labels given. Model needs to make its own learning and it is normally used for clustering and association tasks.

For this project, supervised learning is used because we are required to recognize the named entities in a sentence and not cluster them.

2.4. Current Named Recognition Techniques

Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) model is a popular model used for language tasks *[7]* . RNN can forward process sequential information element by element. Information from the past elements can be carried forward to the current element and then make decisions based on all the information available.

However, as the sentence gets longer, the information of the very early element is gradually vanished. Due to the vanishing gradients, it is difficult to train RNN to tackle long sentences that contains long term dependencies.

The figure below shows an RNN structure *[8]*. The right part shows information from the previous element is passed to the later time step.

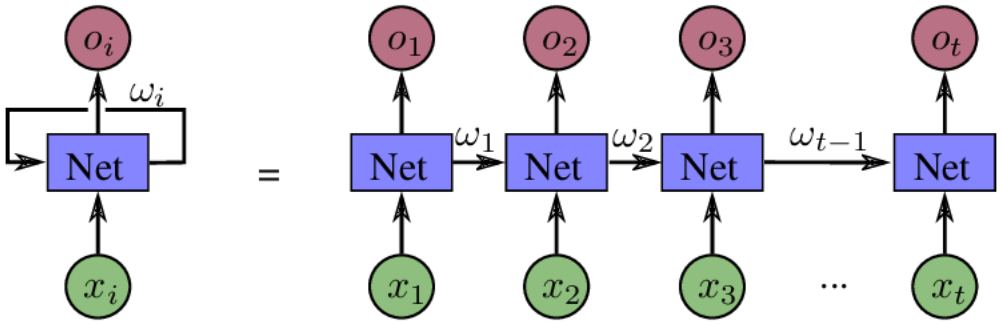


Figure 6 RNN Structure

Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a deep learning system that can learn tasks which require memories that happened millions of discrete time steps earlier *[7]*. It effectively solves the problem of vanishing gradients when tackling with long sentences.

The figure below shows an LSTM structure *[9]* :

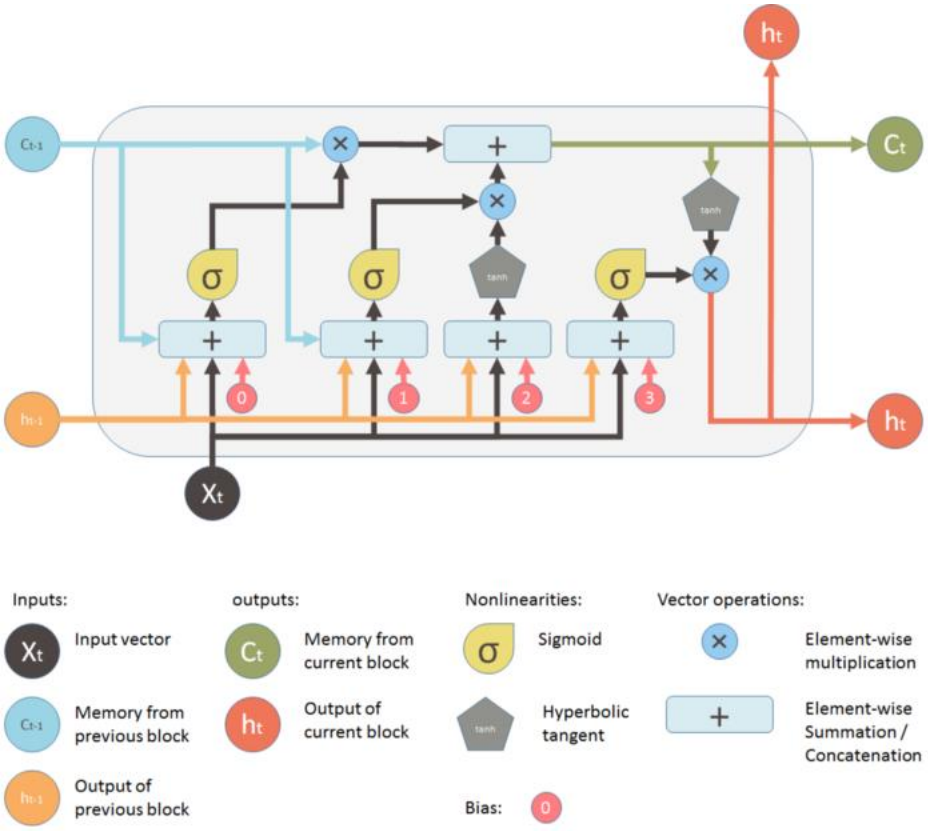


Figure 7 LSTM Structure

2.5. Bidirectional Representations from Transformers (BERT) model

Introduction

Bidirectional Representations from Transformers (BERT) model is an extremely powerful general-purpose model that can be used for any text-based machine learning tasks. It is pretrained on general tasks like language modeling by Google AI Language [1]. The user only requires "fine-tuning" this pre-trained model with the data from specific tasks. As compared to traditional language models, BERT took the advantage of Transformer Architecture, together with its bidirectional technique and Masked Language Modeling task. BERT outperforms most of the state-of-the-art language models.

The flow of the BERT model is as follows:

1) The input data is tokenized into a sequence of tokens
2) The sequence of tokens goes through three embedding layers, namely token embedding layer, segment embedding layer and position embedding layer. The tokens are embedded into high dimensional vector, which equipped with word, sentence and positional information.
3) This vector space input into the neural network to start the fine-tuning process. During fine-tuning, a classification layer that is capable of predicting NER labels is added. BERT trains the vector space together with Masked LM and Next Sentence Prediction to keep the cost function to minimum. At the end of the fine-tuning process, only those hyperparameters for the target labels changes, the remaining hyperparameters remain unchanged.

Tokenization

BERT tokenizer uses Wordpiece tokenizer [10] . When a word is input into Wordpiece tokenizer, the tokenizer tries to search for the best matched token with the highest likelihood given the input word. It takes in the frequency of occurrences to check which is the best match at each iteration and decide based on likelihood of the token. It is the probability of choosing current token as the optimal token given the input word:

$$P(Token|Word)$$

Word Embeddings

BERT uses three layers of embeddings to compute the synaptic input u. They are "Token Embeddings", "Segment Embeddings" and "Position Embeddings". The input embeddings are the sum of the three embedding layers. The figure below shows the BERT Input representation [4]

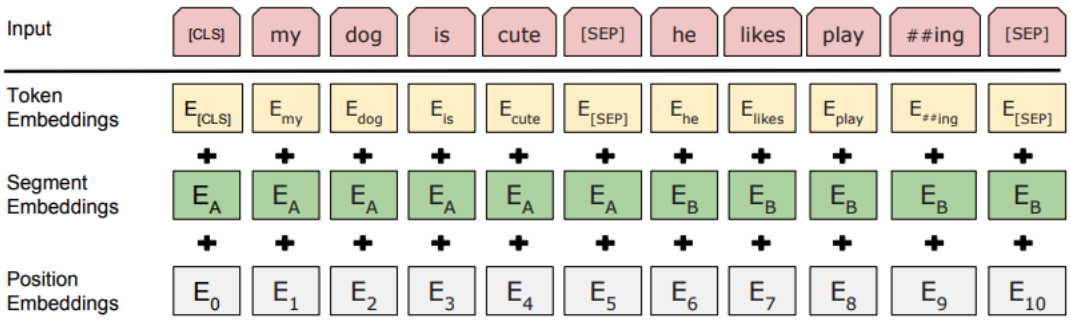


Figure 8 BERT Input representation

Token Embeddings: It vectorizes the input tokens or the words. Every token is presented by a high dimension vector space. A [CLS] reserved token is inserted at the beginning of the first sentence and [SEP] at the end of each of the sentence. Token Embeddings provides a better vector feature on NLP tasks.

Segment Embeddings: It is also referred as sentence embeddings. It is to perform embeddings on whole sentence, or a sequence of sentences separated by [SEP] reserved token. Its purpose is to understand the intention of the sentence.

Position Embeddings: It refers to the position of the input tokens in the input sequence. The position starts from 0 for [CLS] reserved token.

Attention Mechanisms

The attention mechanism was originally introduced to address issues for sequence to sequence (seq2seq) model. Attention mechanism allows the model to process long input sequence, "since only the last hidden state of the encoder RNN is used as the context vector for the decoder. On the other hand, the Attention Mechanism directly addresses this issue as it retains and utilises all the hidden states of the input sequence during the decoding process. It does this by creating a unique mapping between each time step of the decoder output to all the encoder hidden states. [11]" . In short, attention mechanism allows each of the decoder output to selectively choose the specific input from the entire input sequence and assign higher weights to the selected input word. It increases the accuracy of the output.

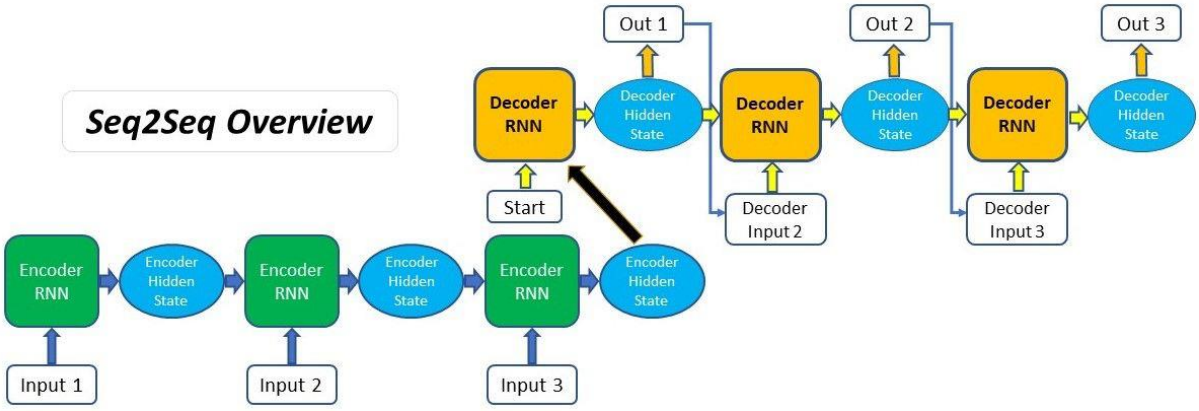


Figure 9 Sequence to Sequence Overview

Transformers in BERT

The figure below shows the how transformers is used in BERT. BERT model replaced RNN in the previous famous ELMo model with Transformer. This is because, as discussed above, Transformer performs much better than RNN.

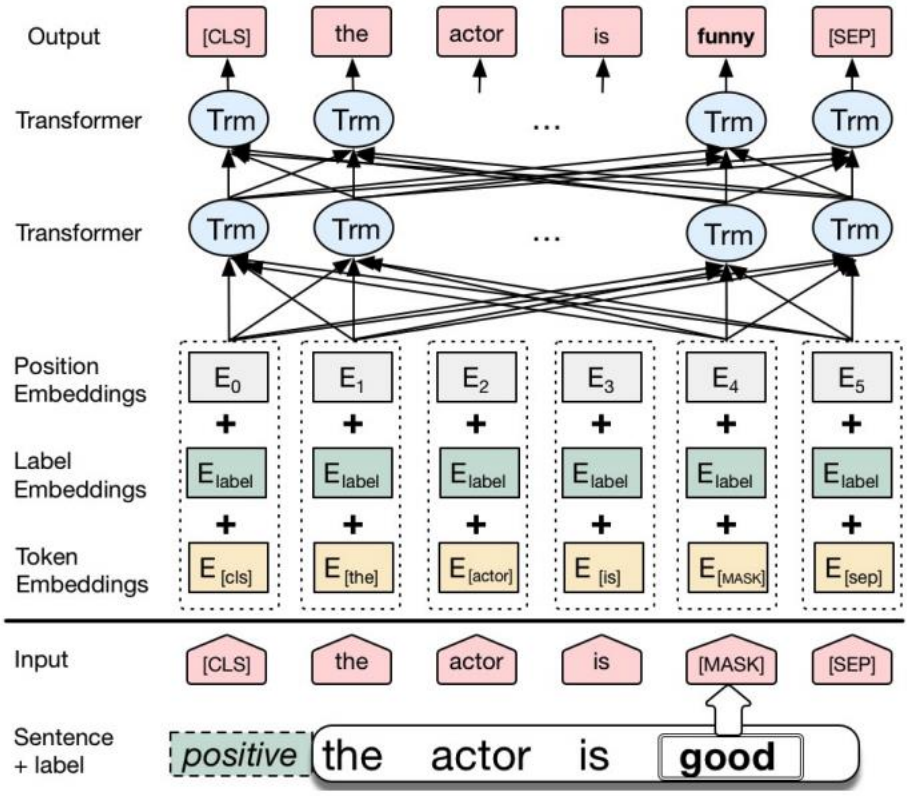


Figure 10 Transformers in BERT

Masked LM (MLM)

In Masked LM (MLM), BERT masks out 15% of the WordPiece. 80% of the masked WordPiece is replaced with a [MASK] token, 10% with a random token and the remaining 10% leaves unchanged. The BERT model then predicts the masked words and compare the predicted results with the actual words.

BERT trains on the 10% random token and the 10% original word so that it can learn what may be the correct words for the missing words. The figure below shows an example of training on masked words [12] :

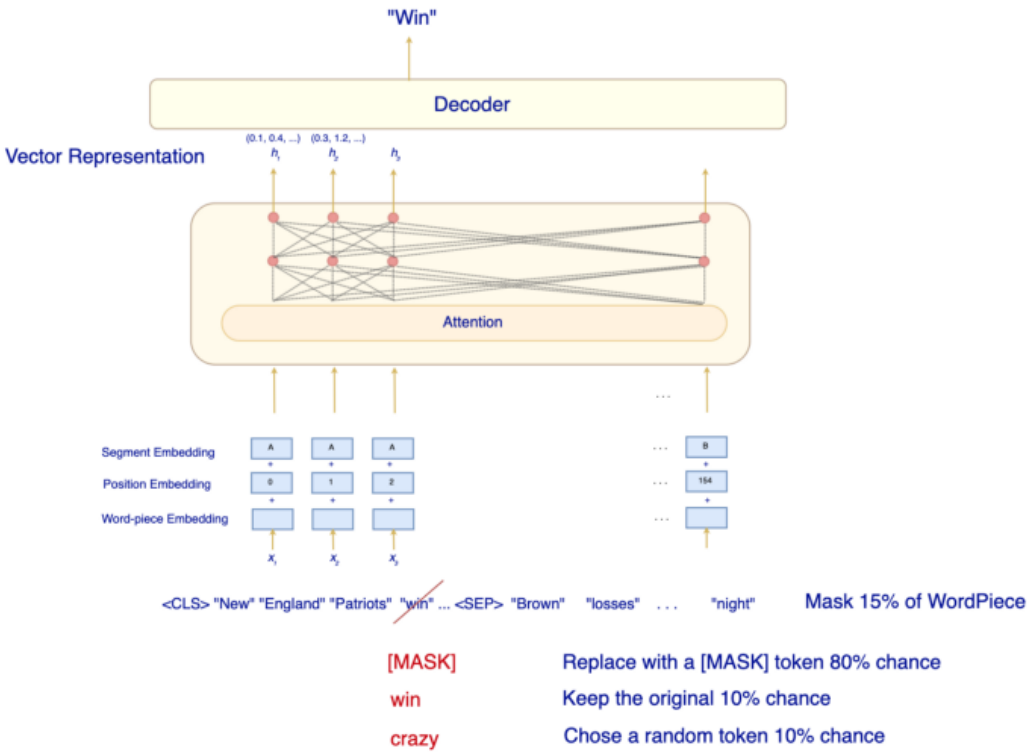


Figure 11 Masked LM (MLM)

Chapter 3

3. System Design and Implementation

This chapter provides an overview of the System for the named entity recognition on Emergency Call Text. It would include the generation of new data from currently available dataset using BERT. Followed by the selection and cleaning of the dataset. Finally, the data is input to the BERT Model for training and fine-tuning.

3.1. System Architecture

The system architecture of the project is a client and server model. The client interacts with the named entity recognition model in the server. When an input text, such as sentences, is provided by the client from the web app, the input text is encoded by Transformers encoder, apply to the model, and then decoded to produce the predictions for the task by Transformers Decoders. The named entities will be highlighted by Named Entity Highlighter and displayed to the client. The figure below shows the system architecture of this project.

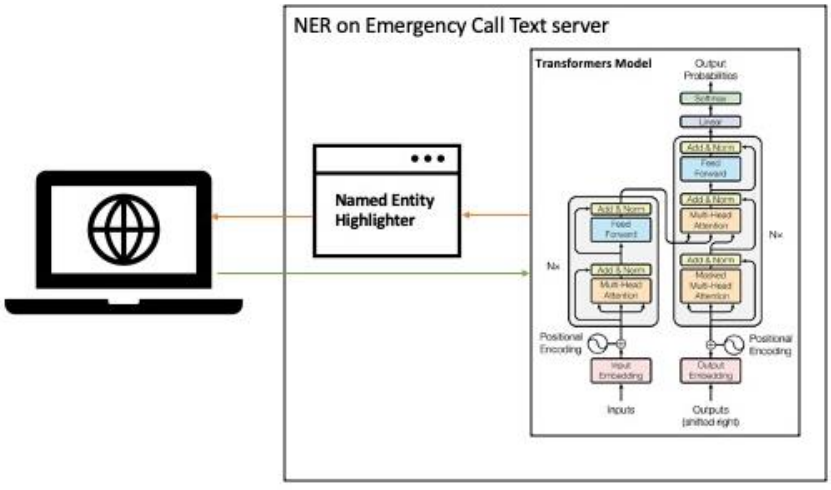


Figure 12 System Architecture of this project

3.2. Dataset

The model is trained using the combination of three dataset. They are CONLL2003, WNUT 17 (Emerging Entities'17) and NCBI disease corpus + i2b2/VA dataset.

The CONLL2003 is a well-known dataset that contains commonly used words for 4 entities. It is used because of its high accuracy and F1 score during various language task challenges.

But there is one issue in using a popular dataset: the model tends to achieve a high accuracy and F1 score when it is trained on well-known dataset. However, when it comes to rarer or previously unseen emerging entities, the model tends to fail on prediction, making their performance scores down. This leaves the model to be less capable of handling Named Entity Recognition in new environment.

Nowadays the named entities are emerging continuously. To tackle this issue, one way is to continuously train the model with the latest data and allow the model to learn and update the weights and biases parameters. However, this method is expensive. The new dataset has to be cleaned, annotated, fine-tuned, trained and evaluated.

Another method is to select a more generalized, or emerging dataset which can train the model to be less sensitive to new changes to the entities in the dataset. WNUT17 is such dataset we select to make the model less sensitive to new information and is more generalized to unseen data.

Finally, NCBI disease corpus + i2b2/VA is used for training the model to learn different types of injuries and medical conditions. NCBI disease corpus + i2b2/VA dataset comprises the disease-related and clinical-related data of over forty thousand patients. It includes

information such as medical conditions, medications, procedures, caregiver notes, medical report. The dataset is freely available and annotated online.

1) CONLL2003 [13]

CONLL2003 is a well-known English and German language dataset. For the project, only English language dataset is selected. It is used because of its high accuracy and F1 scores in competitions. It consists of eight files. There are 1 training file, 1 development file, 1 test file and 1 unannotated data file for English language. It consists named entities of types person, locations, organizations and miscellaneous. The data file contains one word per line. The word is followed by Part of Speech (POS) Tagging, Chunk Tag and named entity tag, each separated by a space. An empty line representing sentence boundary. The Named Entity schemes is Beginning, Inside, Outside (BIO). The corpus includes: -

Metric			Training set	Development set	Test set
Articles			946	216	231
Sentences			14987	3466	3684
Tokens			203621	51362	46435
Entities	Total		23499	5942	5648
	Tags	Person	6600	1842	1617
		Location	7140	1837	1668
		Organization	6321	1341	1661
		Miscellaneous	3438	922	702

Table 1 CONLL2003 Dataset

An example of the sentence is:

U.N.	NNP	I-NP	I-ORG
official	NN	I-NP	O
Ekeus	NNP	I-NP	I-PER
heads	VBZ	I-VP	O
for	IN	I-PP	O
Baghdad	NNP	I-NP	I-LOC
.	.	O	O

2) **WNUT 17** [14]

WNUT 17 (Emerging Entities'17) is an English language dataset that contained mostly rare and novel entities of types person, location, Product, Corporation, Creative Work, Group. It composed of Informal text, such as Tweets text, YouTube comments, StackExchange responses, Reddit comments. The corpus:

1) Training Data: 1975 entity phrases, total 62730 tokens
2) Dev and Test Data: -

Metric			Dev	Test
Documents			1008	1287
Tokens			15734	23394
Entities	Total		835	1070
	Types	Person	470	429
		Location	74	150
		Corporation	34	66
		Product	114	127
		Creative Work	104	142
		Group	39	165

Table 2 WNUT 17 Dataset

This dataset provides our model the capability of classifying the novel, emerging and singleton named entities in user generated text in a noisy text environment, such as spoken language. Since the project focus is on named entities on emergency call text, the model has to be trained to be less sensitive to unstructured spoken language.

3) NCBI disease corpus + i2b2/VA

- NCBI Disease corpus [15]

  NCBI Disease corpus is used for disease related knowledge extraction. The corpus is annotated by 12 annotators on 793 PubMed abstracts.

- I2b2/VA [16]

  I2b2/VA is a collection of discharge summaries. The dataset is annotated by three kinds of entities, problem, treatment and test.

The following table shows the breakdown of the two corpora:-

Dataset	Corpus	Training set	Testing set
Disease NER NCBI Disease	Sentences	5661	961
	Disease	5148	961
	Unique Words	8270	
Clinical NER i2b2/VA	sentences	8453	14529
	problem	7072	12592
	treatment	2841	9344
	test	4606	9225
	Unique Words	13000	

Table 3 NCBI disease corpus + i2b2/VA

3.3. Data Generation (Data Augmentation)

For the training of the named entity recognition tasks, the model requires very large dataset to prevent the results from overfitting. Overfitting is one of the problems that could occur during training of a neural network model. After training the model for many epochs using a small set of data, the model learns to respond to the training data correctly, but not capable to generalize to the novel inputs in the test dataset. The training error of the model may become very small at the expense of high test error. The reason of overfitting may be due to insufficient training data supplied, or too many parameters to learn. The relationship between test error and train error is shown in figure below [17].

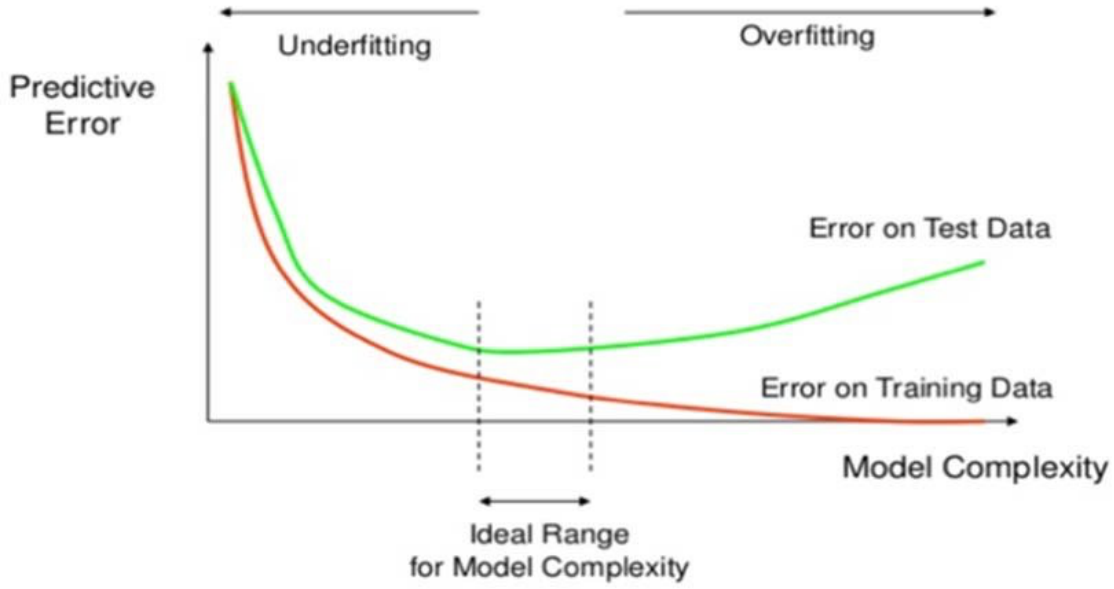


Figure 13 Relationship between Model Complexity and Predictive Error

From the Figure below [18], the test error decreases, meaning the model performs better, as the training dataset increases.

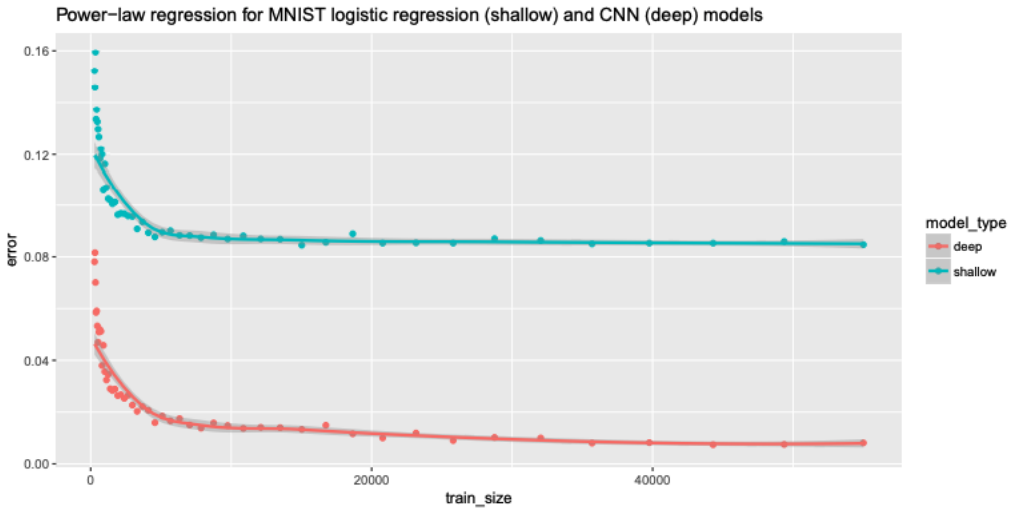


Figure 14 Relationship between Test Error and Train Error

Therefore, we need large amount of training data to avoid overfitting. However, labeled training data is scarce. To obtain sufficiently large training data, there are three methods to obtain the training data. The first one is to find the already annotated Emergency Call Text corpus online. But there are very few corpora available, such as NCBI disease corpus, National NLP Clinical Challenges (n2c2 formerly i2b2) corpus. Another method is by finding actual Emergency Call Texts online and annotate the BIO Taggings manually. However, this option is not feasible as most of the dataset are either government data which is confidential or need to pay for access. The last method is to generate the dataset using a portion of the existing data by data augmentation.

Data augmentation can help to increase the data size by replacing the words with their synonyms based on a data container, such as dictionaries. For this project, as suggested by Varun Kumar in his paper “Data Augmentation using Pre-trained Transformer Models” [19],

I have used 30% of the existing dataset as sample input to augment small and labelled text datasets by using pre-trained transformers

I have built a DataAugmentator class with a transformer model. In the generate function, I firstly randomly select a token in a sentence, followed by creating a mask on this picked token. I then fill this masked token randomly by a synonym using BERT. A new sentence is created. To avoid the model from overfitting on training similar parameters, I created 5000 new sentences. The figure below shows an example of the original sentence and newly generated sentences.

Figure 15 Original Sentence and Newly Generated Sentences

3.4. Data Cleaning

Tagging Schemes

The corpus is required to be tagged before performing training. The tagging schemes, BIO (Beginning, Inside, Outside) is used in this project. BIO is a common tagging format used for named entity recognition. Beginning refers to the beginning of a named entity; Inside refers to the middle of the entity; Outside indicates the token belongs to no chunk [6]. The figure shows an example of BIO scheme:

```
Third-stringer  O
Kerwin  B-PER
Bell  I-PER
,  O
a  O
1988  B-TIME
draft  O
choice  O
of  O
the  O
Miami  B-ORG
Dolphins  I-ORG
,  O
made  O
his  O
NFL  B-ORG
debut  O
.  O
```

Figure 16 BIO scheme

Padding/Trimming the sentences

Since each of the sentence in the dataset is of different length, we have to pad/trim the sentence to keep all the sentences to be the same length. Since the maximal sentence length for BERT model is 512. Those sentences with length less than 512 are padded. Those sentences with length more than 512, the extra portion is trimmed.

3.5. BERT Model

We used "**bert-base-cased**" pre-trained model for our project. The "bert-base-cased" pre-trained model comprised of 12-layer, 768-hidden, 12-heads, 110M parameters. I used "bert-base-cased" pre-trained model because of its training result is accurate enough for the project. The computation time and power requirements are relatively low. I have tried "bert-large-cased" pre-trained model. The training time of this model is tripled on my laptop. Therefore, "bert-base-cased" model is the optimal pre-trained model we chose.

Chapter 4

4. Experiment and Implementation

4.1. Experiment

The model is trained using Pytorch and “bert-base-cased” pre-trained model. The model is trained on Laptop for 5 Epochs. The table below shows the hyperparameters used to train the model:

Trained Model Hyperparameters Details	
no epochs	5
batch size	32
max len WordPiece	75
attention probs dropout prob	0.1
hidden act	gelu
hidden dropout prob	0.1
hidden size	768
initializer range	0.02
intermediate size	3072
layer norm eps	1.00E-12
max position embeddings	512
model type	bert
num attention heads	12
num hidden layers	12
num labels	11

output attentions	FALSE
output hidden states	FALSE
pad token id	0
pruned heads	{}
torchscript	FALSE
type vocab size	2
vocab size	28996

Table 4 Trained Model Hyperparameters

4.2. Results

The result of the of the experiment is shown in the table below. The model is evaluated using a confusion matrix during training. The result is analyzed by F1 Score.

F1 score [20] is the weighted average of the precision and recall. F1 score is between 1 and 0, where 1 represents the best value and 0 indicates the worst score. It is related by a formula below:

$$\boldsymbol{F1 = 2 * \frac{Precision * Recall}{Precision + Recall}}$$

The Precision, Recall and F1-Score are shown in the figure below:

	Precision	**Recall**	**F1-Score**
ORGANIZATION	0.89	0.91	0.90
LOCATION	0.93	0.95	0.94
PERSON	0.94	0.87	0.90
DISEASE	0.85	0.89	0.87
MISC	0.86	0.88	0.87
AVERAGE	0.9	0.91	0.91

Table 5 The Precision, Recall and F1-Score of the Trained Model

When the loss function is at the optimal, the **training accuracy** is **0.992** and **validation accuracy** is **0.954**. The model is carefully trained by using **dropout probability** of **0.1** to avoid overfitting.

	Accuracy	**Loss**	**F1 Score**
Training	0.992089279	0.025385366	0.91
Validation	0.954151639	0.191778849	0.91

Table 6 Accuracy, Loss and F1 Score for the Trained Model

Overfitting

For a deep neural network with a large number of parameters, such as weights and biases, overfittings may occur when

1) Some of the neurons learn the similar data points information repeatedly. The network learns to respond correctly to the training inputs but fails to generalize to the novel input in the test data.
2) Some of the weights attain large values to reduce training error at the cost of test error. It reduces the ability to generalizing the novel data.
3) The model has too many parameters to learn
4) Training Data is insufficient, and the network learn the training patterns too many times.

Methods to overcome overfitting includes Early stopping, L2 Regularization of weights, as well as Dropouts.

In this project, I used Dropouts with a probability of 0.1. Dropout tackles the problem of overfitting by randomly drop neurons and its connections from the network during training. The figure below shows an example of the neural network before and after applying Dropouts [21].

In the probability of Dropout probability is 0.1, meaning each neuron in the network can present to the next layer with 10%. Its weight is also multiplied by 10% as well. Therefore, there is no difference in the final testing time as compared to the expected output at the training time. The gradients of each parameter are calculated based on average gradients of training cases.

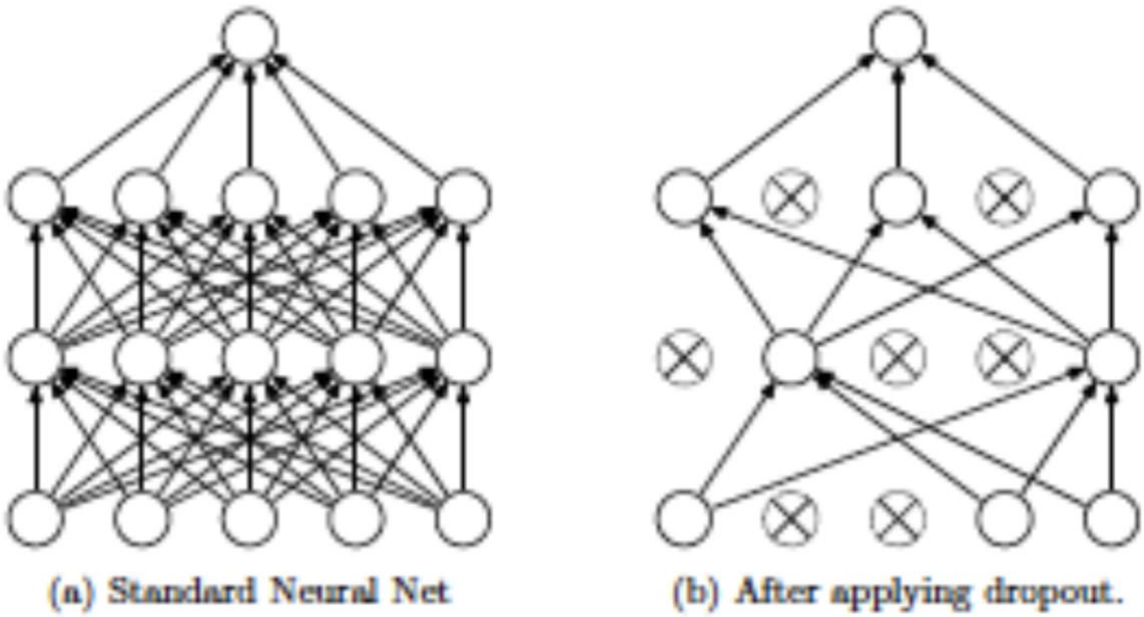


Figure 17 Neuron Network with Dropouts vs without Dropouts

Confusion Matrix

The feature below shows the confusion matrix. The Confusion matrix helps to visualize the performance of model. Each row of the matrix represents the predicted class while each column represents the actual class. It makes the us easy to check whether the system confuses the two.

For example, for the entity B-DISEASE, there are 1582 label identified correctly, while 286 labels are recognized as O entity. The matrix helps us to identify which label or named entity is more likely to be recognized wrongly.

		Actual Class										
		B-DISEASE	**B-LOC**	**B-MISC**	**B-ORG**	**B-PER**	**I-DISEASE**	**I-LOC**	**I-MISC**	**I-ORG**	**I-PER**	**O**
Predicted Class	**B-DISEASE**	1582	0	0	0	0	80	0	0	0	0	86
	B-LOC	0	3231	11	51	11	0	11	0	1	1	44
	B-MISC	0	60	1063	115	24	0	0	25	1	0	296
	B-ORG	0	55	45	2443	21	0	2	3	22	1	83
	B-PER	0	21	24	124	3400	0	0	0	0	30	368
	I-DISEASE	0	113	0	0	0	1419	0	0	0	0	130
	I-LOC	0	27	0	0	0	0	328	2	15	0	21
	I-MISC	0	3	25	2	12	0	17	381	37	12	259
	I-ORG	0	6	0	15	0	0	6	19	976	0	57
	I-PER	0	0	0	6	11	0	0	12	25	2958	109
	O	286	44	116	97	58	223	12	65	39	7	83710

Table 7 Confusion Matrix of the Trained Model

4.3. Interface Design

The Interface of the keyword and named entity recognition on Emergency Call text was hosted on a web application. It was developed using Streamlit package in the Python library. Streamlit is an open-source Python library for a building web application. For this project, it is hosted locally at localhost:8501.

The web app allows the user to select different checkpoints. “Model checkpoint” dropdown list contains one of the past models with highest accuracy from each training. The users can select which is the model they want to use. This feature allow the user to test on the model that best fit their language task.

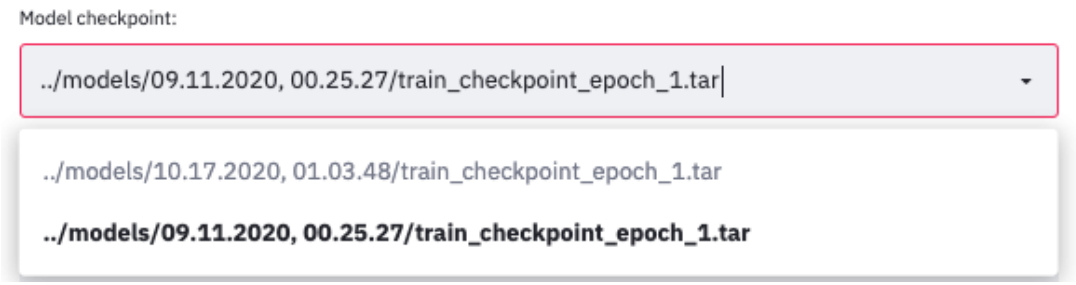


Figure 18 User Interface Model Checkpoint Selection

The web app allows the user to type their Emergency Call Text sentences into the Text box. The predication result will be highlighted in different colors indicating various entities. The colors of different entities are:

Person Location Organization Misc DISEASE

The figures below shows the input box as well as the prediction result box:-

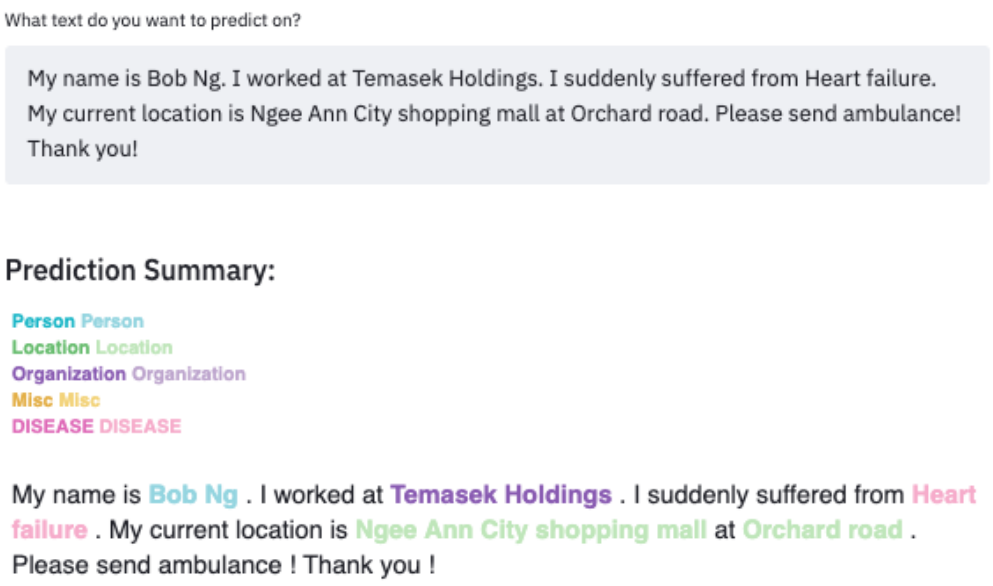
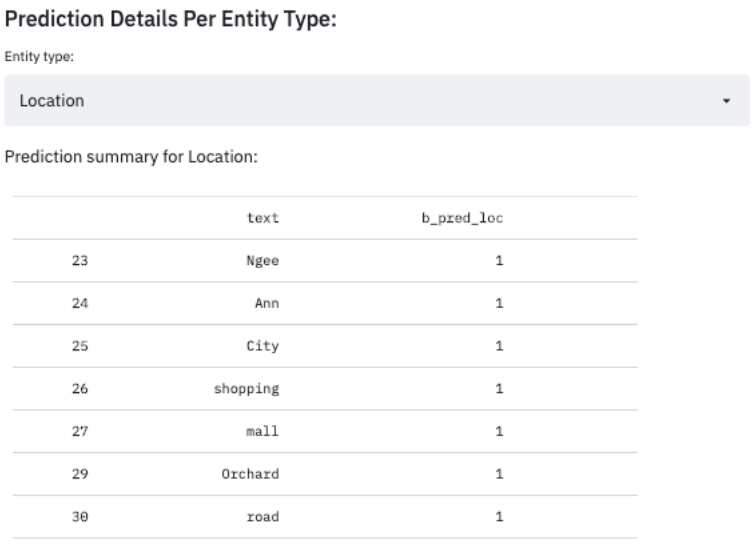


Figure 19 User Interface Input and Result

The user can also check the prediction details by Entity types. The figure below shows all the words that are identified as location by the BERT model.



	text	b_pred_loc
23	Ngee	1
24	Ann	1
25	City	1
26	shopping	1
27	mall	1
29	Orchard	1
30	road	1

Figure 20 User Interface Named Entities by categories

The figure below shows the entire interface:

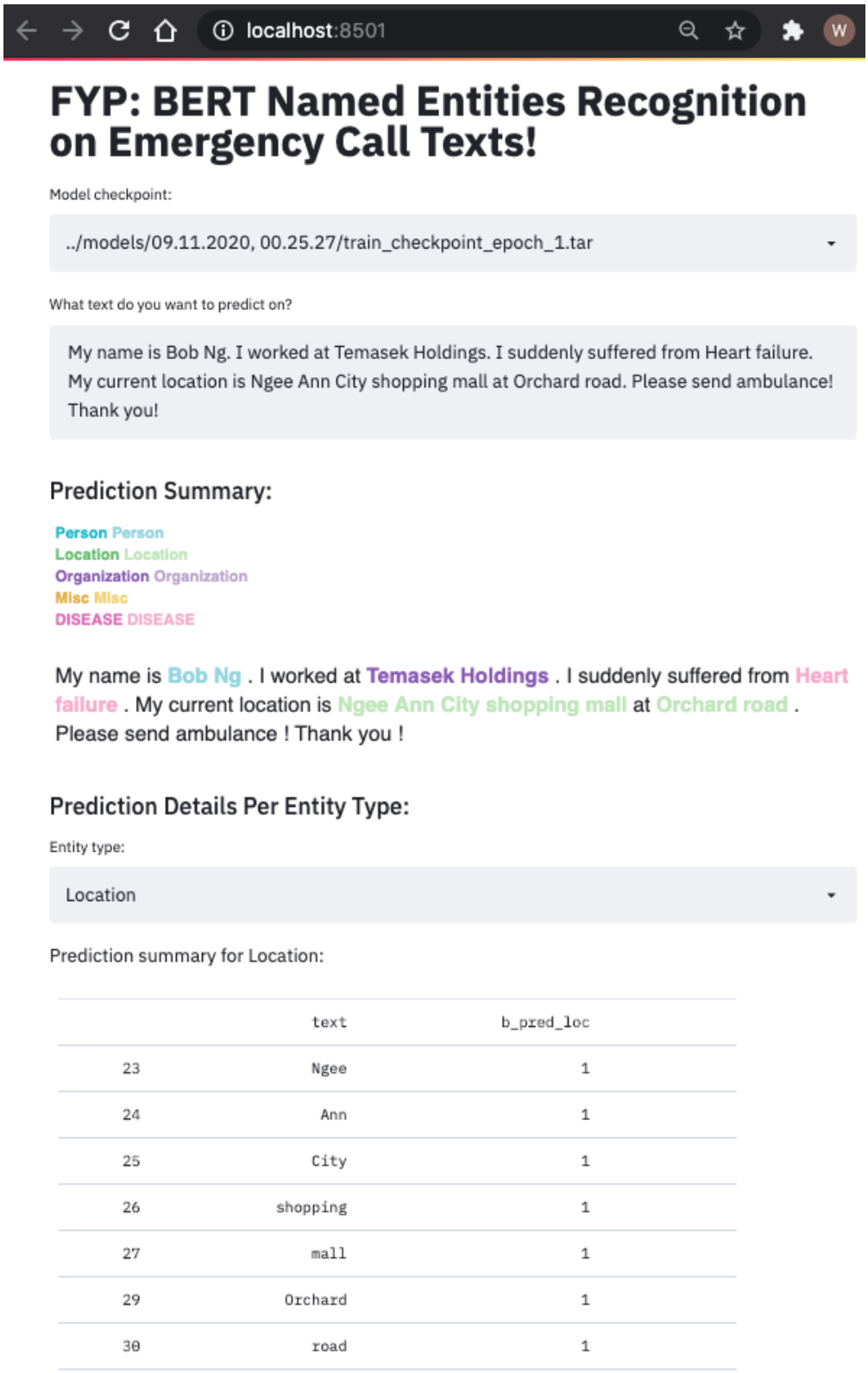


Figure 21 Entire User Interface

Chapter 5

5. Conclusions and future work

5.1. Conclusion

This project has successfully designed, built, developed and implemented. The result of the project can be used to identify the named entities and keywords of Emergency Call Texts. The identified named entities are visualized in a web application.

The dataset used for training are 75% from online, with the combination of three dataset, namely general NER dataset CONLL2003, the Emerging Entities WNUT 17 as well as the medical condition corpora NCBI disease corpus + i2b2/VA. The remaining 25% of the dataset are generated manually using Data Generation BERT Model. All the dataset are combined and splitted for training, validation and testing.

All the data are annotated using BIO Tagging Scheme and then fed into the BERT model. The BERT model is trained and deployed in the web application, which was designed using Python Streamlit package.

The detailed description of demonstration of the interface is provided in the paper. Because as compared to running the code from command prompt, a user interface can help the user to visualize and observe the results easily. The users are able to get the named entities tagged after they input their sentences into the input box. Their input will be encoded and decoded by the BERT Model at the backend and return results with all the named entities highlighted.

The interface also allows the user to compare different model by selecting other checkpoints in the user interface.

The result of the training result is quite accurate. It achieved an accuracy of 0.992 and F1 score of 0.91 before overfitting. This result means that the model can predict the majority of the named entities in user input sentences.

Generally speaking, the model in my dissertation is able to recognize the named entities correctly and the performance is acceptable. Hopefully, it can be used, with some improvements and modifications, in real system and allow the patients to be treated in the shortest period, which can save life.

5.2. Recommendations in Future Work

The performance of the model can be further improved. Here are some of the recommendations that may enhance the accuracy and precision of the model:

1) The model can be trained with more corpora, which includes more specific entities, such as block number and floor number of a location.
2) The training time need to be optimized. Current training time for training 5 epoch without callbacks is more than 10 hours. The project training time can be optimized by adjusting the batch size, number of layers, type of pre-trained model.
3) The model can be re-trained with uncased words. Currently, the model can identify those all the cased named entities and some of the uncased word. Since text in the future comes from speech-to-text converter, it is better to allow the model to be capable of recognize all the uncased named entities as well. As a result, the performance of the model can benefit a lot from this adjustment.

References

[1] K. Gardner, “How AI is Helping Efficiency Improve,” 24 October 2019. [Online]. Available: https://towardsdatascience.com/how-ai-is-helping-efficiency-improve-98d0171a23e2. [Accessed 24 February 2020].

[2] R. Hodgetts, “DEATHS DUE TO AMBULANCE DELAYS,” 25 February 2014. [Online]. Available: https://www.clinicalnegligenceteam.co.uk/blog/deaths-due-to-ambulance-delays/. [Accessed 24 February 2020].

[3] T. M. Tan, “Answering the (995) call with speed and efficiency,” 25 December 2017. [Online]. Available: https://www.straitstimes.com/singapore/answering-the-995-call-with-speed-and-efficiency. [Accessed 24 February 2020].

[4] M. W. Chang, J. Devlin, K. Lee and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” 24 May 2019. [Online]. Available: https://arxiv.org/abs/1810.04805. [Accessed October 2020].

[5] S. LI, “Named Entity Recognition with NLTK and SpaCy,” 17 AUGUST 2018. [Online]. Available: https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da. [Accessed AUGUST 2020].

[6] “Inside–outside–beginning (tagging),” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Inside%E2%80%93outside%E2%80%93beginning_(tagging). [Accessed October 2020].

[7] “Recurrent neural network,” Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Recurrent_neural_network. [Accessed October 2020].

[8] L. Tai and M. Liu, “Deep-learning in Mobile Robotics - from Perception to Control Systems: A Survey on Why and Why not,” December 2016. [Online]. Available: https://www.researchgate.net/publication/311805526_Deep-learning_in_Mobile_Robotics_-_from_Perception_to_Control_Systems_A_Survey_on_Why_and_Why_not. [Accessed October 2020].

[9] S. Yan, “Understanding LSTM and its diagrams,” 14 MARCH 2016. [Online]. Available: https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714. [Accessed OCTOBER 2020].

[10] M. S. Z. C. Q. V. L. M. N. W. M. M. K. Y. C. Q. G. K. M. J. K. A. S. M. J. X. L. Ł. K. S. G. Y. K. T. K. H. Yonghui Wu, “Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation,” 8 October 2016. [Online]. Available: https://arxiv.org/pdf/1609.08144.pdf. [Accessed September 2020].

[11] G. Loye, “Attention Mechanism,” 15 SEPTEMBER 2019. [Online]. Available: https://blog.floydhub.com/attention-mechanism/. [Accessed SEPTEMBER 2020].

[12] J. Hui, “NLP — BERT & Transformer,” 5 November 2019. [Online]. Available: https://medium.com/@jonathan_hui/nlp-bert-transformer-7f0ac397f524. [Accessed August 2020].

[13] E. F. T. K. Sang and F. D. Meulder, “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”.

[14] L. Derczynski, E. Nichols, M. v. Erp and N. Limsopatham, “Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition”.

[15] Z. Lu, “The NCBI Disease Corpus,” [Online]. Available: https://www.ncbi.nlm.nih.gov/research/bionlp/Data/disease/. [Accessed October 2019].

[16] Ö. Uzuner, B. R. South, S. Shen and S. L. DuVall, “2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text,” September 2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3168320/. [Accessed October 2019].

[17] Hayder Naser Khraibet Al-Behadili, Ku Ruhana Ku-Mahamud and Rafid Sagban , “Rule pruning techniques in the ant-miner classification algorithm and its variants: A review,” Research Gate.

[18] M. Johnson, “web.science.mq.edu.au,” 4 September 2017. [Online]. Available: http://web.science.mq.edu.au/~mjohnson/papers/Johnson17Power-talk.pdf. [Accessed 2020].

[19] Varun Kumar, Ashutosh Choudhary and Eunah Cho, “Data Augmentation using Pre-trained Transformer Models,” 4 March 2020. [Online]. Available: https://arxiv.org/pdf/2003.02245.pdf. [Accessed September 2020].

[20] “sklearn.metrics.f1_score,” Scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html. [Accessed October 2020].

[21] Prof Jagath Chandana Rajapakse, “Model selection and overfitting,” Singapore, 2020.

[22] T. Sterbak, “Data augmentation with transformer models for named entity recognition,” 23 August 2020. [Online]. Available: https://www.depends-on-the-definition.com/data-augmentation-with-transformers/. [Accessed September 2020].