



**CHATBOT APPLICATION FOR INDUSTRIAL LAW**

**SUTTIDECH JITTAWISUTTIKUL**

**MASTER OF ENGINEERING  
IN  
COMPUTER ENGINEERING**

**SCHOOL OF APPLIED DIGITAL TECHNOLOGY  
MAE FAH LUANG UNIVERSITY**

**2024**

**©COPYRIGHT BY MAE FAH LUANG UNIVERSITY**

**CHATBOT APPLICATION FOR INDUSTRIAL LAW**

**SUTTIDECH JITTAWISUTTIKUL**

**THIS THESIS IS A PARTIAL FULFILLMENT OF  
THE REQUIREMENTS FOR THE DEGREE OF  
MASTER OF ENGINEERING  
IN  
COMPUTER ENGINEERING**

**SCHOOL OF APPLIED DIGITAL TECHNOLOGY  
MAE FAH LUANG UNIVERSITY**

**2024**

**©COPYRIGHT BY MAE FAH LUANG UNIVERSITY**



**THESIS APPROVAL**  
**MAE FAH LUANG UNIVERSITY**  
**FOR**

**MASTER OF ENGINEERING IN COMPUTER ENGINEERING**


**Thesis Title:** Chatbot Application for Industrial Law

**Author:** Suttidech Jittawisuttikul

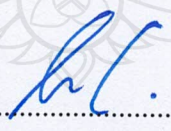
**Examination Committee:**

|   |             |
|---|-------------|
| Associate Professor Punnarumol Temdee, Ph. D.       | Chairperson |
| Associate Professor Roungsan Chaisricharoen, Ph. D. | Member      |
| Chayapol Kamyod, Ph. D.                             | Member      |
| Associate Professor Nattapol Aunsri, Ph. D.         | Member      |
| Associate Professor Rawid Banchuin, Ph. D.          | Member      |

**Advisors:**

  
.....Advisor  
(Associate Professor Roungsan Chaisricharoen, Ph. D.)

**Dean:**

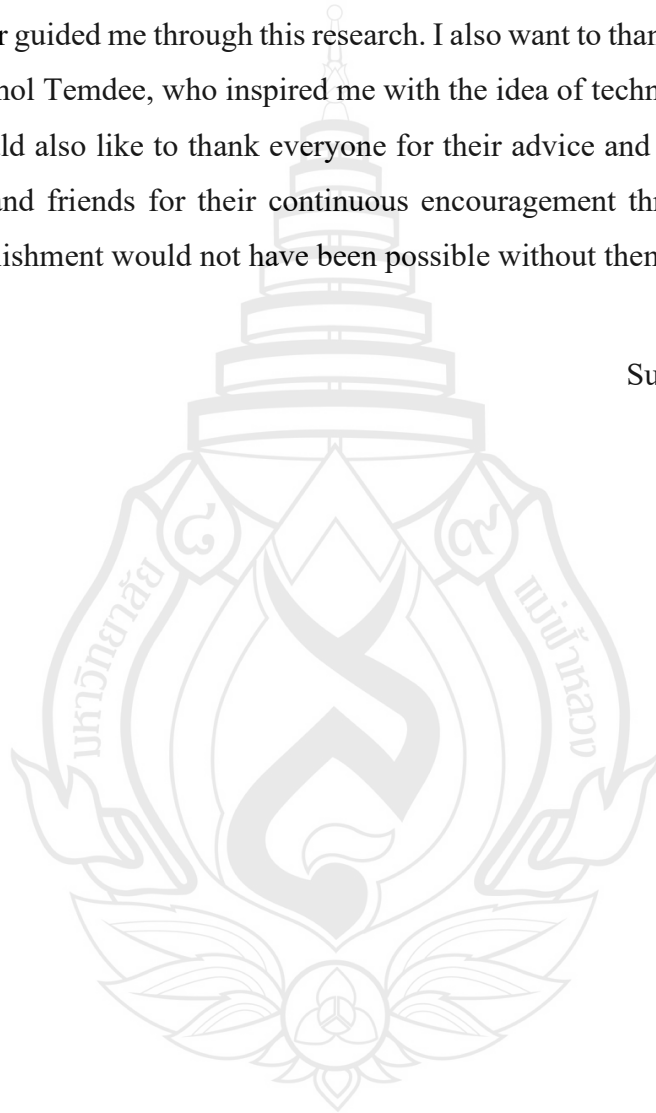
  
.....  
(Assistant Professor Nacha Chondamrongkul, Ph.D.)

## ACKNOWLEDGEMENTS

I am deeply grateful to my instructors and the Mae Fah Luang University faculty for their guidance and support in my dissertation. I especially thank my supervisor, Associate Professor Dr. Rounsang Chaisricharoen, whose insight and knowledge of the subject matter guided me through this research. I also want to thank Associate Professor Dr. Punnarumol Temdee, who inspired me with the idea of technology in my project.

I would also like to thank everyone for their advice and help. I am grateful to our parents and friends for their continuous encouragement throughout this project. This accomplishment would not have been possible without them.

Suttidech Jittawisuttikul





|                     |   |
|---------------------|---|
| <b>Thesis Title</b> | Chatbot Application for Industrial Law              |
| <b>Author</b>       | Suttidech Jittawisuttikul                           |
| <b>Degree</b>       | Master of Engineering (Computer Engineering)        |
| <b>Advisor</b>      | Associate Professor Rounsang Chaisricharoen, Ph. D. |

## **ABSTRACT**

Nowadays, Speech and textual information play a more critical role in human communication than counting face-to-face exchanges. An article in “The New York Times” published that adults today spend more than 8 hours daily on computer screens or mobiles, which is done through web applications such as WhatsApp, Facebook, and Twitter, among others, in speech and text conversations. However, in the study part, there are few online teaching materials compared to other online media. In the present paper, we built a chatbot in the education domain. The proposed chatbot assists in answering questions provided by the users as an additional way to study for students in modern times. The chatbot that we developed has focused on industrial law because most students in Thailand have difficulty using English. Therefore, we have created a chatbot in the Thai language, which currently does not have a chatbot based on the Thai language.

The chatbot uses NLP to build a system that can understand and respond to users’ questions. In developing the chatbot system, we created a chatbot using neural network algorithms to provide services related to Thai industrial law. The main objective is the development of a chatbot application designed to serve as an information provider for industrial laws, with a primary focus on materials presented in the Thai language, aiming to enhance accessibility and efficiency in legal information dissemination. This system is intended to facilitate users in obtaining relevant legal knowledge by leveraging advanced chatbot technology, ensuring that industrial law resources are both easily accessible and comprehensible to Thai-speaking users. By integrating legal content into an interactive and user-friendly platform, the application seeks to bridge the gap between complex legal regulations

and practical understanding, ultimately improving compliance and awareness in industrial sectors.

**Keywords:** Chatbots, NLP, Neural Network

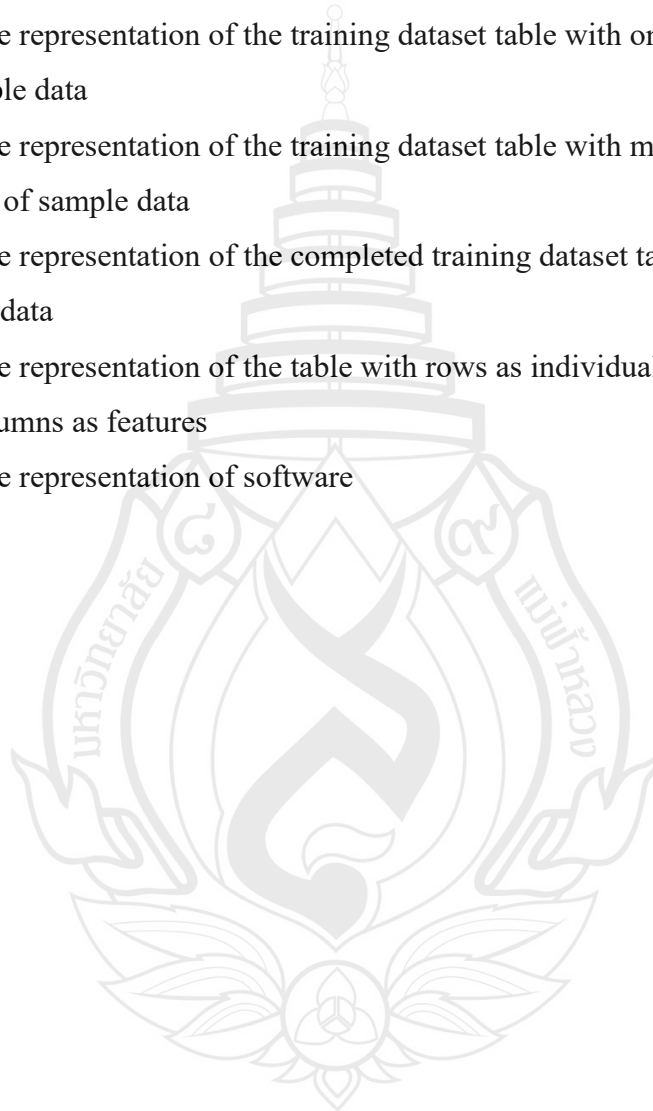


## TABLE OF CONTENTS

| CHAPTER                         | Page      |
|---------------------------------|-----------|
| <b>1 INTRODUCTION</b>           | <b>1</b>  |
| <b>2 LITERATURE REVIEW</b>      | <b>3</b>  |
| 2.1 Background Theory           | 3         |
| 2.2 Literature Purpose          | 6         |
| 2.3 Related Technology          | 7         |
| <b>3 PROPOSED METHODS</b>       | <b>10</b> |
| 3.1 Methodology                 | 10        |
| 3.2 Our Developed Software      | 21        |
| <b>4 EXPERIMENTS AND RESULT</b> | <b>23</b> |
| 4.1 Experiment Condition        | 23        |
| 4.2 Experiment                  | 41        |
| 4.3 Results                     | 44        |
| <b>5 CONCLUSION</b>             | <b>47</b> |
| <b>REFERENCES</b>               | <b>48</b> |
| <b>APPENDIX</b>                 | <b>50</b> |
| <b>CURRICULUM VITAE</b>         | <b>52</b> |

## LIST OF TABLES

| Table   | Page |
|---|------|
| 3.1 The table representation of sample question and answer pairs                                  | 12   |
| 3.2 The table represents the sample of features table   | 16   |
| 3.3 The table representation of the training dataset table with one record of sample data         | 17   |
| 3.4 The table representation of the training dataset table with multiple records of sample data   | 18   |
| 3.5 The table representation of the completed training dataset table with sample data             | 18   |
| 3.6 The table representation of the table with rows as individual samples and columns as features | 19   |
| 4.1 The table representation of software  | 24   |





## LIST OF FIGURES

| Figure   | Page |
|--|------|
| 2.1 The relation of five essential large language models   | 3    |
| 2.2 The neural network model   | 7    |
| 3.1 Extraction text from PDF file using PyPDF2   | 10   |
| 3.2 The sample result from the extracting process from the source file   | 11   |
| 3.3 Using the thaipellcheck to spell check   | 11   |
| 3.4 The sample raw training dataset in JSON format   | 14   |
| 3.5 Extract words from statements using word_tokenize  | 15   |
| 3.6 The prepared result will be used in the next step, the sorted words are the final result of step 2                                       | 15   |
| 3.7 The result of the training for input of the Neural Network classification algorithm  | 19   |
| 3.8 The question set of the training for input of the Neural Network classification algorithm, defined as <i>train_sequences</i>             | 20   |
| 3.9 The answer set of the training for input of the Neural Network classification algorithm, defined as <i>train_labels</i>                  | 20   |
| 3.10 The feature words dataset of the training for the input of the Neural Network classification algorithm, defined as <i>feature_words</i> | 21   |
| 4.1 The developing and testing hardware specification  | 23   |
| 4.2 The source documents of industrial laws from <a href="https://law.industry.go.th">https://law.industry.go.th</a>                         | 27   |
| 4.3 A sample source of industrial law detail in the PDF file   | 27   |
| 4.4 A sample of question-answer pair in JSON form  | 28   |
| 4.5 Pypdf2 package installation in PyCharm CE  | 28   |
| 4.6 The sample source of the industrial laws (pdf file format)   | 29   |
| 4.7 The sample Python code for extracting text from the source file  | 29   |
| 4.8 The sample result from the extracting process from the source file   | 30   |
| 4.9 Thaispellcheck package installation in PyCharm   | 30   |
| 4.10 The sample Python code for spell-checking   | 31   |

## LIST OF FIGURES

| Figure  | Page |
|---|------|
| 4.11 The sample result from the spell-checking process  | 31   |
| 4.12 The sample Python code to get the text from all question statements before extracting words from statements                  | 32   |
| 4.13 The sample Python code to extract words from statements and how to sort them   | 33   |
| 4.14 The result of the word-filtering and sorting process   | 33   |
| 4.15 The lines of code for training the model using the Neural Network classification algorithm                                   | 34   |
| 4.16 The formula to calculate the output of a single neuron in a neural network using the ReLU activation function                | 37   |
| 4.17 The formula for calculating the loss of predicted probabilities in a neural network  | 38   |
| 4.18 The output report for each epoch in the training process   | 40   |
| 4.19 The sample code to predict the answer to the new question by using the trained model   | 41   |
| 4.20 Logging files by testing loop  | 42   |
| 4.21 The sample report of result logging file   | 43   |
| 4.22 The prediction time usage results of the Chatbots  | 43   |
| 4.23 Summary reporting table of training results logging file   | 44   |
| 4.24 The prediction accuracy result, the blue in the graph means the prediction response is corrected, and the white is incorrect | 44   |
| 4.25 The sample source file of Thai industrial laws   | 45   |
| 4.26 The training dataset collection workflow   | 45   |
| 4.27 The industrial laws chatbot application  | 46   |

## CHAPTER 1

### INTRODUCTION

Chatbots, or chat robots, are software agents that simulate human conversations via text or voice. Their main goal is to be like clever humans. They know the answers to most questions and answer them like a human would. To this end, they have created many kinds of machine learning and deep learning methods. These methods will be used to develop software that can think like humans (Yan et al., 2016).

Currently, very little information is available about chatbots and industrial law. Therefore, we want to develop a chatbot system to answer questions about industrial laws. Legal books are time-consuming, and legal information may not be easily accessible to the public. They are also costly, even those written for the public.

Current law, including industrial law, is often compartmentalized, meaning many sections and sub-sections refer to the main title. That makes it very confusing for a user to find relevant search information. A user would have to know what section their pertinent information is in, which makes using knowledge obtained from typical searches ineffective. For example, if we want to search for laws regarding “Industrial Steam Boilers in factories,” We may have to gather various legal texts from the Constitution, Acts, Royal Decrees, and Ministerial Regulations, including announcements, regulations, and requirements could take extensive time and effort to obtain comprehensive information.

We want to develop a superb system to gather and put scattered industrial legal knowledge in one place. This system will provide concise answers to those who want answers about industrial law. Users can ask questions and be provided answers through an AI chatbot system, which is an easily accessible online application. Chatbots are convenient to use and replace the search functions in browsers (Huang, 2010), searching by reading legal books, hiring an attorney, or studying from other online media.

In this project, we used the Neuron Network classifier methods to create a chatbot to assist users in their quest for relevant answers to industrial law (Prayitno et al., 2021).

So, we are motivated to design an automated conversational system, namely a chatbot, for industrial laws. To design the chatbot, we have observed the following step.

1. Data collection: We used information on industrial laws published by the Department of Industrial Works (<https://law.industry.go.th/>) and the Ministry of Industry (Thailand) in an unstructured form and created unique question-and-answer pairs.

2. Data preprocessing

3. Training & Testing

4. Building a Chatbot System

Developing a chatbot for industrial law is challenging due to the time-consuming process of gathering knowledge from books. As a result, the chatbot is trained on limited content and can only answer predefined questions, though users can expand its knowledge over time. Thai language processing presents difficulties due to the lack of spaces between words, requiring advanced segmentation algorithms for accuracy. Errors in word segmentation affect response precision, making English-based chatbots potentially more reliable. Additionally, the chatbot's performance depends on device capability and internet speed, as it operates as a web-based application within a server-client framework, limiting access to online users only.



## CHAPTER 2

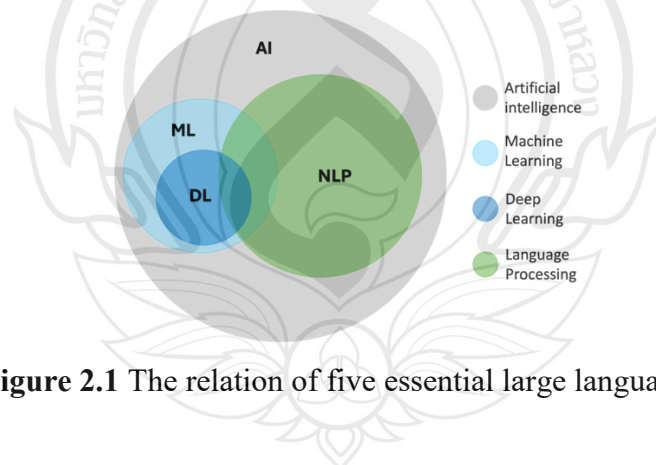
### LITERATURE REVIEW

#### 2.1 Background Theory

Chatbots are software programs that simulate intelligent conversations with humans using rules or artificial intelligence. Users interact with the chatbot through a conversational interface using written or spoken text.

There are two main types of chatbots: rules-based and self-learning. Rule-based chatbots follow predefined rules to determine the correct response to user input. Self-learning or AI-powered chatbots use machine-learning algorithms to understand and respond to user input.

The most common type of self-learning chatbot is a fetch chatbot. This type selects an answer from predefined responses based on the similarity between the user's input and the predetermined answer.



**Figure 2.1** The relation of five essential large language models

Natural Language Processing (NLP) is an essential science in Machine Learning. It is a branch of knowledge from various fields, such as linguistics (Zhou et al., 2020).

Computers to “understand” textual or verbal data, just as humans can. This is not only about understanding the direct meaning of the text but also about the perception of its implications. Author's feelings Language Contextual Differences Include being able to analyze in various ways as well.

NLP makes it easier for humans to communicate and collaborate with machines by allowing them to do so in the natural human language they use daily. NLP offers benefits across many industries and applications (Garg et al., 2021).

1. Automation of repetitive tasks
2. Improved data analysis and insights
3. Enhanced search
4. Content generation

NLP originated in the middle of the 19th century and has been continuously developed until today. In this case, we would like to divide its evolution into three eras.

### **2.1.1 Rule-based Method Era (1950-1990s)**

In the early days, NLP was used as a rule-based method by linguists who specialized in the language structure they were interested in. Write the rules the computer can calculate to find the answer to your desired problem.

### **2.1.2 Machine Learning Era (1990-2010)**

Machine learning (ML) is a branch of computer science that uses data and algorithms to enable AI to imitate how humans learn, gradually improving its accuracy.

In the later era, it was found that writing rules by hand could only respond to simple problems. PC performance Includes knowledge of statistics and machine learning, which has been developed for use in NLP work by importing data so that computers can learn by themselves instead of using language experts.

### **2.1.3 Machine Learning Has Three Main Algorithms**

1. A Decision Process: Machine learning algorithms generally make predictions or classifications. Based on some input data, which can be labeled or unlabeled, your algorithm will estimate a pattern in the data.

2. An Error Function: An error function evaluates the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.

3. A Model Optimization Process: If the model better fits the data points in the training set, weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm repeats this iterative “evaluate and optimize” process, updating weights autonomously until a threshold of accuracy is met.

#### **2.1.4 Deep Learning Era (2010-present)**

The computing power of computers is constantly evolving, making technology highly complex. Deep Learning is replacing Machine Learning, which uses traditional statistical knowledge. It is also widely used in NLP work, such as language modeling and parsing.

Deep learning is a subset of machine learning that uses multilayered neural networks, called deep neural networks, to simulate the complex decision-making power of the human brain.

#### **2.1.5 How Does Deep Learning Work?**

Neural networks, or artificial neural networks, attempt to mimic the human brain by combining data inputs, weights, and bias—all acting as silicon neurons. These elements work together to recognize, classify, and accurately describe objects within the data.

Deep neural networks consist of multiple layers of interconnected nodes, each building on the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation (Yu et al., 2002). A deep neural network’s input and output layers are called *visible* layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

Another process called backpropagation uses algorithms, such as gradient descent, to calculate prediction errors. It then adjusts the function’s weights and biases by moving backward through the layers to train the model. Forward propagation and backpropagation enable a neural network to make predictions and correct errors. Over time, the algorithm becomes gradually more accurate.

Deep learning requires a tremendous amount of computing power. High-performance graphical processing units (GPUs) are ideal because they can handle extensive calculations in multiple cores with copious memory available. Distributed cloud computing might also assist. This level of computing power is necessary to train deep algorithms through deep learning. However, managing multiple GPUs on-premises can create a great demand on internal resources and be incredibly costly to scale. For software requirements, most deep learning apps are coded with one of these three learning frameworks: JAX, PyTorch, or TensorFlow.

## 2.2 Literature Review

By using machine learning algorithms to develop chatbots. You no longer need to define and code new pattern-matching rules. This allows chatbots to be more flexible and no longer dependent on domain-specific knowledge. As noted, AI models can be divided into retrieval-based and general-purpose models. The fetch-based data extraction model is designed to provide a text-based dataset. The algorithm will be able to retrieve the necessary information based on user input. The algorithms used are usually shallow learning algorithms (Holdsworth & Scapicchio, 2024). However, some retrieval models use rules-based algorithms and deep learning models—a predefined set of possible answers (Jothi et al., 2022). The chatbot processes the user's search query based on this input. Will choose one of the answers contained in the set. The knowledge base for this type of model is usually built from a database of question-and-answer pairs. A conversation index is generated from this database. To predict all possible answers according to the prompts and choose them based on the accuracy rate of the prediction function. When a user provides input to a chatbot, the chatbot will treat the input as a query (Hashana et al., 2023).

A fetching model similar to that used for web browsing matches user input to similar data in the chat index. Therefore, the results returned to the user are answers matched to the selected questions listed in the Discussion Index. The main advantage of this model is that it ensures the quality of the answers. Because they are not automatically generated, these models have gained immense popularity with the advent



of Web 2.0 and the rise of text-based information that can be retrieved on social media platforms, forums, and chats. The approach to building the necessary knowledge base is costly, time-consuming, and tedious. It also means that matching user input with correct answers becomes more challenging. Training the system to choose the correct answer takes time and resources.

## 2.3 Related Technology

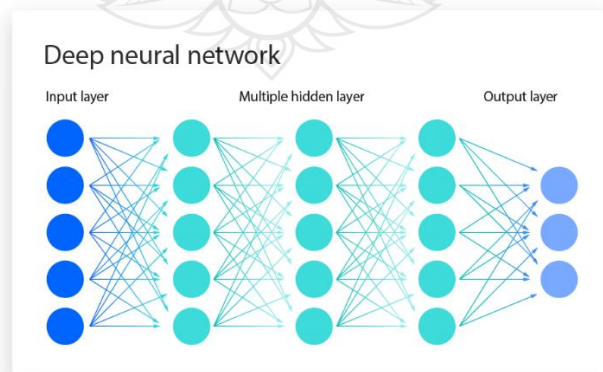
We used the Neural Network classifier techniques to develop an industrial laws AI chatbot system.

### 2.3.1 Neural Network (Multilayer Perceptron Classifier)

A neural network is a machine learning program or model that makes decisions like the human brain. It uses processes that mimic how biological neurons work together to identify phenomena, weigh options, and arrive at conclusions.

Neural networks, also called artificial neural networks or simulated neural networks, are a subset of machine learning and are the backbone of deep learning algorithms. They are called “neural” because they mimic how neurons in the brain signal one another.

Neural networks comprise node layers—an input layer, one or more hidden layers, and an output layer. Each node is an artificial neuron that connects to the next, and each has a weight and threshold value. When one node’s output is above the threshold value, that node is activated and sends its data to the network’s next layer. If it’s below the threshold, no data passes along (Singh & Banerjee, 2019).



**Figure 2.2** The neural network model

### 2.3.2 Types of Neural Networks

Neural networks (IBM, 2024) can be classified into different types, each used for various purposes. While this isn't a comprehensive list, the below types represent the most common.

The classification task in machine learning has been a significant focus of research for decades, with neural networks (NN) playing a pivotal role in achieving state-of-the-art performance across various domains. Neural networks, inspired by the structure and functioning of biological neurons, are powerful tools for learning complex patterns in data. Their ability to adapt to diverse datasets and uncover non-linear relationships has made them an essential choice for classification problems.

The evolution of neural networks can be traced back to the development of the perceptron in the late 1950s. However, their popularity surged with the advent of multi-layer perceptron (MLPs) and the backpropagation algorithm in the 1980s. These advancements enabled neural networks to address the limitations of single-layer networks by learning hierarchical feature representations. Introducing non-linear activation functions, such as the sigmoid and ReLU (Rectified Linear Unit), enhanced their performance by overcoming issues like vanishing gradients and improving convergence rates.

1. Convolutional neural networks (CNNs) are similar to feedforward networks but are usually utilized for image recognition, pattern recognition, and/or computer vision. These networks harness principles from linear algebra, particularly matrix multiplication, to identify patterns within an image.

2. Recurrent neural networks (RNNs) are identified by their feedback loops. These learning algorithms use time series data to predict future outcomes, such as stock market predictions or sales forecasting.

3. Generative Adversarial Networks (GANs): Composed of a generator and a discriminator, GANs pit these two components against each other. The generator creates data while the discriminator assesses its authenticity. This adversarial process results in the generator producing increasingly realistic data, often used for generating images, videos, and audio.

4. Feedforward Neural Networks: This basic type processes data linearly from input to output, without loops. They're commonly used for straightforward tasks like classification and regression.

Several training and optimization techniques have improved the performance of the neural network classifier. Methods for standard adjustment, such as L2 standard adjustment and disconnection from the system, prevent overfitting by adding noise during training. Optimizers such as Adam and RMSprop dynamically adjust the learning rate. Accelerate convergence Batch normalization and data augmentation also significantly improve the stability and robustness of the neural network in classification tasks.

Neural networks have been applied successfully in numerous classification problems, including:

1. Healthcare: Diagnosis of diseases through medical imaging (e.g., X-rays, MRIs) and genetic data classification (Reddy et al., 2023).
2. Finance: Fraud detection and risk assessment using transaction data (Guolin, 2007).
3. Agriculture: Crop disease detection and yield prediction through image and sensor data analysis (Abdullahi, 2014).
4. Natural Language Processing: Sentiment analysis, spam detection, and language translation.
5. Despite its success, neural network classification still faces several challenges. These include the need for large, labeled datasets, the computational cost and interpretation of recent efforts in transfer learning, and integrated learning and artificial intelligence that can explain. Aiming to overcome these limitations, integrating neural networks with hybrid models and quantum computing also opens new avenues for research and applications.

## CHAPTER 3

### PROPOSED METHODS

We use natural language processing in conjunction with Python to develop a chatbot based on the Thai Language, focusing on industrial law.

We have gathered information, questions, answers, and knowledge about industrial law to review and remove some redundant information. Then, we converted the dataset from the text file to a programming language format, such as JSON. After we had changed the format of the question-answer pair data set, we developed a system to prepare the data in the form of an input dataset for the system, using the following step to illustrate the method.

#### 3.1 Methodology

We gathered knowledge about industrial laws and then converted the dataset from the text file to the dataset in a programming language format, such as the JSON format. After the question-answer pair dataset format had been changed, a system was developed to prepare the data as an input dataset for the system using the following steps. Examples of words are in English so that the examples may be widely understood.

##### 3.1.1 Step 1. Words collecting

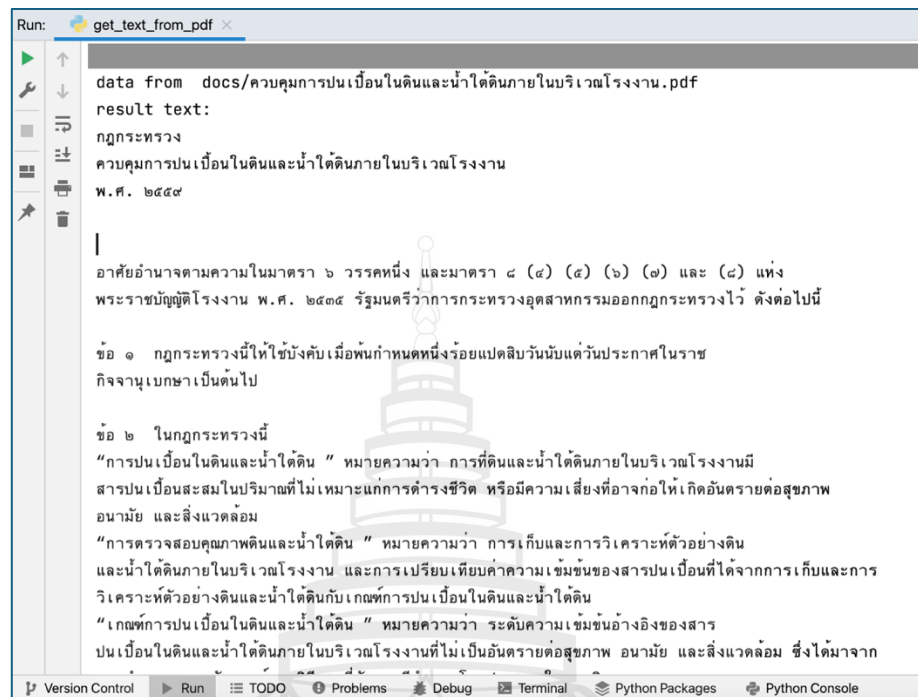
This step is the word processing to read all the words from datasets. We need to collect all the words to create a set of features that is a standard requirement of input data for the deep learning methods we used

1. Get text from pdf using PyPDF2 on PyCharm CE



**Figure 3.1** Extraction text from PDF file using PyPDF2

The sample result of the text extraction process:



**Figure 3.2** The sample result from the extracting process from the source file

The result of the `get_text_from_pdf` is the raw text that we have to cleanse.  
Cleansing.

2. Create questions and answers paired to make the training dataset

We must clean text from PdfReader using the coding below. The reader tools may have an error depending on the source files.

Text cleaning using the `thaispellcheck` library



**Figure 3.3** Using the `thaispellcheck` to spell check

In this step, we use AI to help detect the error of extracting text from a PDF file. We then must rewrite the text manually and use it to create the question-and-answer pairs table.

### 3. Create the question-and-answer pairs table (manually)

**Table 3.1** The table representation of sample question and answer pairs

| question_id | question_text   | answer_id | answer_text  |
|-------------|---|-----------|--|
| 1           | ใบอนุญาตประกอบ<br>กิจการโรงงานมีอายุ<br>เท่าไร?               | 1         | ใบอนุญาตประกอบกิจการโรงงานไม่มีการ<br>กำหนดอายุการใช้งานแบบจำกัด แต่จะมีผล<br>บังคับใช้ต่อเนื่องตราบใดที่ผู้ประกอบการยัง<br>ปฏิบัติตามข้อกำหนดของกฎหมายโรงงาน<br>รวมถึงมาตรการควบคุมสิ่งแวดล้อมและความ<br>ปลอดภัย หากมีการฝ่าฝืนกฎระเบียบ อาจถูก<br>เพิกถอนใบอนุญาตได้   |
| 2           | ผู้ได้รับใบอนุญาตต้อง<br>ติดตั้งเครื่องมือ<br>ควบคุมอะไรบ้าง? | 2         | ผู้ได้รับใบอนุญาตโรงงานต้องติดตั้งเครื่องมือ<br>หรืออุปกรณ์สำหรับควบคุมมลพิษที่อาจเกิดขึ้น<br>เช่น เครื่องบำบัดน้ำเสีย ระบบกำจัดฝุ่นและ<br>ควัน รวมถึงเครื่องมือลดเสียงดัง ทั้งนี้ ขึ้นอยู่<br>กับประเภทของโรงงานและการกำหนดของ<br>พนักงานเจ้าหน้าที่ เพื่อให้มั่นใจว่าการดำเนิน<br>กิจการไม่ก่อผลกระทบต่อสิ่งแวดล้อมและ<br>ชุมชนใกล้เคียง |
| 3           | การเพิกถอน<br>ใบอนุญาตมีเงื่อนไข<br>อะไร?                     | 3         | ใบอนุญาตโรงงานอาจถูกเพิกถอนได้ในกรณีที่ผู้<br>ประกอบกิจการไม่ปฏิบัติตามกฎหมายโรงงาน<br>ฝ่าฝืนคำสั่งของพนักงานเจ้าหน้าที่ หรือไม่<br>แก้ไขปัญหาที่ก่อให้เกิดผลกระทบต่อ<br>สิ่งแวดล้อมและความปลอดภัย ทั้งนี้ การเพิก<br>ถอนจะมีการตรวจสอบข้อเท็จจริงและแจ้งให้<br>ทราบล่วงหน้า เพื่อให้ผู้ประกอบการมีโอกาส<br>แก้ไขปัญหา                     |

Table 3.1 (continued)

| question_id | question_text   | answer_id | answer_text   |
|-------------|---|-----------|---|
| 4           | โรงงานประเภทใด<br>ต้องจัดทำรายงาน<br>ผลกระทบ<br>สิ่งแวดล้อม?              | 4         | โรงงานที่มีลักษณะการดำเนินงานที่ส่งผล<br>กระทบรุนแรงต่อสิ่งแวดล้อม เช่น โรงไฟฟ้า<br>โรงงานที่ใช้วัตถุอันตราย หรือโรงงานขนาดใหญ่ที่มีการปล่อยมลพิษในปริมาณสูง จะต้อง<br>จัดทำรายงานการวิเคราะห์ผลกระทบ<br>สิ่งแวดล้อม เพื่อให้หน่วยงานที่เกี่ยวข้อง<br>ตรวจสอบและอนุมัติก่อนเริ่มดำเนินการ |
| 5           | ใบรับแจ้งของโรงงาน<br>จำพวกที่ 2 มีผล<br>ผูกพันอย่างไร?                   | 5         | ใบรับแจ้งสำหรับโรงงานจำพวกที่ 2 เป็น<br>เอกสารที่แสดงว่าผู้ประกอบการได้แจ้ง<br>รายละเอียดการดำเนินงานต่อพนักงาน<br>เจ้าหน้าที่เรียบร้อยแล้ว ซึ่งใบรับแจ้งนี้ไม่ได้<br>เป็นการอนุญาต แต่เป็นการรับทราบและ<br>ยืนยันว่ากิจการดังกล่าวสามารถดำเนินงานได้<br>ตามเงื่อนไขที่กำหนด              |
| 6           | พนักงานเจ้าหน้าที่มี<br>สิทธิอะไรบ้างในการ<br>ตรวจโรงงาน?                 | 6         | พนักงานเจ้าหน้าที่มีสิทธิเข้าตรวจสอบโรงงาน<br>โดยไม่ต้องแจ้งล่วงหน้า เพื่อตรวจสอบว่า<br>โรงงานปฏิบัติตามข้อกำหนดด้านความ<br>ปลอดภัย การควบคุมมลพิษ และกฎหมายอื่น<br>ๆ ที่เกี่ยวข้อง หากพบความผิดปกติหรือการ<br>ละเมิดกฎ อาจออกคำสั่งให้แก้ไขหรือระงับการ<br>ดำเนินกิจการ                  |
| 7           | การขอเปลี่ยนแปลง<br>เครื่องจักรต้องแจ้ง<br>พนักงานเจ้าหน้าที่<br>หรือไม่? | 7         | หากการเปลี่ยนแปลงเครื่องจักรอาจส่งผลต่อ<br>ความปลอดภัยของโรงงานหรือมีผลกระทบต่อ<br>สิ่งแวดล้อม เช่น การเพิ่มขนาดกำลังผลิตหรือ<br>การใช้เครื่องจักรที่สร้างมลพิษ ผู้ประกอบ<br>กิจการต้องแจ้งพนักงานเจ้าหน้าที่และขอ<br>อนุญาตก่อนดำเนินการเปลี่ยนแปลง                                      |
| 8           | การต่อใบอนุญาตทำ<br>ได้เมื่อใด?   | 8         | ใบอนุญาตประกอบกิจการโรงงานไม่ต้องต่อ<br>อายุ แต่ผู้ประกอบการต้องปฏิบัติตามเงื่อนไขที่<br>ระบุในใบอนุญาตอย่างต่อเนื่อง หากมีการฝ่า<br>ฝืนกฎหมายหรือไม่ปฏิบัติตามคำสั่ง อาจถูก<br>เพิกถอนใบอนุญาตหรือสั่งระงับการดำเนินงาน<br>ได้   |

Table 3.1 (continued)

| question_id | question_text   | answer_id | answer_text  |
|-------------|---|-----------|--|
| 9           | โรงงานที่ตั้งในพื้นที่<br>ชุมชนต้องมี<br>ข้อกำหนดอะไร<br>เพิ่มเติม? | 9         | โรงงานที่ตั้งอยู่ในพื้นที่ชุมชนต้องปฏิบัติตาม<br>มาตรการควบคุมมลพิษอย่างเคร่งครัด เช่น ลด<br>เสียงรบกวน ควบคุมการปล่อยมลพิษทาง<br>อากาศ และบำบัดน้ำเสีย นอกจากนี้ ควรมีการ<br>สื่อสารกับชุมชนใกล้เคียงเพื่อสร้างความเข้าใจ<br>และลดความขัดแย้ง |
| 10          | การขนย้ายวัตถุ<br>อันตรายจากโรงงาน<br>ต้องแจ้งใคร?                  | 10        | การขนย้ายวัตถุอันตรายจากโรงงานต้องแจ้ง<br>พนักงานเจ้าหน้าที่หรือหน่วยงานที่เกี่ยวข้อง<br>เช่น กรมโรงงานอุตสาหกรรม หรือหน่วยงาน<br>ควบคุมการขนส่งวัตถุอันตราย เพื่อให้มั่นใจว่า<br>การขนย้ายดังกล่าวปลอดภัยและปฏิบัติตาม<br>กฎหมาย              |
| n           | ...   | ...       | ...  |

## 4. Transform the question-and-answer to the training set (JSON format)

```

1  training_dataset_3.json
2  "intents": [
3    {
4      "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",
5      "question_id": "1",
6      "patterns": ["ใบอนุญาตประกอบกิจการโรงงานมีอายุเท่าไร?"],
7      "answer_id": "1",
8      "responses": ["ใบอนุญาตประกอบกิจการโรงงานไม่มีการกำหนดอายุการใช้งานแบบจำกัด แต่จ..."],
9    }, {
10     "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",
11     "question_id": "2",
12     "patterns": ["ผู้ได้รับใบอนุญาตต้องติดตั้งเครื่องมือควบคุมอะไรบ้าง?"],
13     "answer_id": "2",
14     "responses": ["ผู้ได้รับใบอนุญาตโรงงานต้องติดตั้งเครื่องมือหรืออุปกรณ์สำหรับควบคุมมลพิษที่อา..."],
15   }, {
16     "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",
17     "question_id": "3",
18     "patterns": ["การเพิกถอนใบอนุญาตมีเงื่อนไขอะไร?"],
19     "answer_id": "3",
20     "responses": ["ใบอนุญาตโรงงานอาจถูกเพิกถอนได้ในกรณีที่ผู้ประกอบการไม่ปฏิบัติตามกฎหมาย..."],
21   }, {
22     "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",

```

Figure 3.4 The sample raw training dataset in JSON format



### 3.1.2 Step 2. Word Filters and Sorting

This step is the first optimization technique after we have the question-and-answer paired dataset as JSON from Step 1. We must collect all words from the dataset. Then, we sort and filter the duplicate words from the input dataset.

1. Get all words from the question-and-answer paired dataset will look like this:

All words from all questions = “ใบอนุญาตประกอบกิจการโรงงานมีอายุเท่าไร? ผู้ได้รับใบอนุญาตต้องติดตั้งเครื่องมือควบคุมอะไรบ้าง? การเพิกถอนใบอนุญาตมีเงื่อนไขอะไร? โรงงานประเภทใดต้องจัดทำรายงานผลกระทบสิ่งแวดล้อม? ใบรับแจ้งของโรงงานจำพวกที่ 2 มีผลผูกพันอย่างไร? พนักงานเจ้าหน้าที่มีสิทธิอะไรบ้างในการตรวจโรงงาน? การขอเปลี่ยนแปลงเครื่องจักรต้องแจ้งพนักงานเจ้าหน้าที่หรือไม่? การต่อใบอนุญาตทำได้เมื่อใด? โรงงานที่ตั้งในพื้นที่ชุมชนต้องมีข้อกำหนดอะไรเพิ่มเติม? การขนย้ายวัตถุอันตรายจากโรงงานต้องแจ้งใคร?...”

2. Extract words from the result of the question-and-answer paired, then sort all and filter out duplicate words.



**Figure 3.5** Extract words from statements using word\_tokenize

```

Run: extract_words_from_text x
/Volumes/Data/PythonProjects/TestCoding/venv/bin/python /Volumes/Data/PythonProjects/Tes
All Words = ['ใบอนุญาต', 'ประกอบ', 'กิจการ', 'โรงงาน', 'มีอายุ', 'เท่าไร', 'ผู้', 'ได้รับ', 'ใบอนุ
Sort all words = ['', '1,2', '1', '2', '2', '2', '2', '3', '3', '7', '8', '8', '9', 'ก
Unique words = ['ขอเขต', 'ปล่อย', 'ปฏิบัติตาม', 'ขนย้าย', 'ขอ', 'ชุมชน', 'หรือ', 'ผู้', 'เพิ่มประสิ
Sorted words = ['', '1,2', '1', '2', '3', '7', '8', '9', 'กฎกระทรวง', 'กฎหมาย', 'กรณี',
  
```

**Figure 3.6** The prepared result will be used in the next step, the sorted words are the final result of step 2

### 3.1.3 Step 3. Stemming Words

This step is the second optimization technique. We must group words with the exact meaning and convert them to one of the forms, such as “get,” “got,” “gotten,” or “getting.” This is a group of words that represent the same meaning. We must convert

all words in this group to “get.” This technique is robust and reduces the size of the dataset well, but it doesn't have adverse effects. However, the stemming words will be affected only in the English language.

#### 3.1.4 Step 4. Transforms the Stemmed Words to the Features Table

This step depends on the dataset with more unique words. We have more features for processing deep learning methods. We must sort all the words and create a features table containing all the word indexes by sorting the words in ascending.

**Table 3.2** The table represents the sample of features table

| กฎกระทรวง | กฎหมาย | กรณี | ใบอนุญาต | อายุ | ... |
|-----------|--------|------|----------|------|-----|
| ...       | ...    | ...  | ...      | ...  | ... |

#### 3.1.5 Step 5. Transform the question sentence to a features table

This step is essential and takes time. It depends on the hardware used because we must do all the steps above for every question sentence. After getting the feature table, we need to put each question into the feature table by following these steps

1. Words tokenization: We must split all words from a sentence in this step.

For example:

The question is: “ใบอนุญาตประกอบกิจการโรงงานมีอายุเท่าไร?”

The result is: [‘ใบอนุญาต’, ‘ประกอบ’, ‘กิจการ’, ‘โรงงาน’, ‘มีอายุ’, ‘เท่าไร’]

2. Sorting words: we must sort the tokenized words before sending them to the next step.

For example:

The tokenized words is: [‘ใบอนุญาต’, ‘ประกอบ’, ‘กิจการ’, ‘โรงงาน’, ‘มีอายุ’, ‘เท่าไร’]

Then, we sort them by using the collate function:

*sorted\_words = collate(words\_of\_sentence)*

The result is: [‘กิจการ’, ‘เท่าไร’, ‘ใบอนุญาต’, ‘ประกอบ’, ‘มีอายุ’, ‘โรงงาน’]

3. Bag of word vectors: After sorting words, we have to define the vector of each word by using the feature table from Table 3.2. In this step, we do a sub-step by following:

1) Create the temp vector record with dimensions equal to the number of all words in the feature table by coding like this.

For example: `bag_of_word = np.zeros(len(all_words), dtype=np.int32)`

The result is: `bag_of_word = [0 ... 0]`

2) Define vector value by comparing the words of Step 5.2 to all the words in the feature table by the following pseudocode:

Function `do_bag_of_words()`

For `inx = 0 To len(all_words)`:

    If found `all_words[inx]` in the result of step 5.2

`bag_of_word[inx] = 1`

    Else

`bag_of_word[inx] = 0`

    End If

End For

From the pseudocode above, we will get a result that looks like

`bag_of_word = [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 ... 0]`;

The result will differ depending on the question sentence.

3) Input the result of 5.3.2 to the training dataset table in Table 3.3. It will look like the following table:

**Table 3.3** The table representation of the training dataset table with one record of sample data

| Question ID | The feature word index |        |     |        |      |     |
|-------------|------------------------|--------|-----|--------|------|-----|
|             | กฎกระทรวง              | กฎหมาย | ... | กิจการ | อายุ | ... |
| 1           | 0                      | 0      | 0   | 1      | 1    | ... |

4) Repeat steps 1 to 5 until all questions are completed. The result of this step will look like the following table:

**Table 3.4** The table representation of the training dataset table with multiple records of sample data

| Question ID | The feature word index |        |     |        |      |     |
|-------------|------------------------|--------|-----|--------|------|-----|
|             | กฎกระทรวง              | กฎหมาย | ... | กิจการ | อายุ | ... |
| 1           | 0                      | 0      | 0   | 1      | 1    | ... |
| 2           | 0                      | 0      | 1   | 0      | 0    | ... |
| 3           | 0                      | 0      | 0   | 0      | 1    | ... |
| ...         | ...                    | ...    | ... | ...    | ...  | ... |
| n           | ...                    | ...    | ... | ...    | ...  | ... |

5) Modify the training dataset table by adding the answer column to the latest column and defining the answer ID of each question. The result will look like the following table:

**Table 3.5** The table representation of the completed training dataset table with sample data

| Question ID | The feature word index |        |     |        |      |     | Answer ID |
|-------------|------------------------|--------|-----|--------|------|-----|-----------|
|             | กฎกระทรวง              | กฎหมาย | ... | กิจการ | อายุ | ... |           |
| 1           | 0                      | 0      | 0   | 1      | 1    | ... | 1         |
| 2           | 0                      | 0      | 1   | 0      | 0    | ... | 2         |
| 3           | 0                      | 0      | 0   | 0      | 1    | ... | 3         |
| ...         | ...                    | ...    | ... | ...    | ...  | ... | ...       |
| n           | ...                    | ...    | ... | ...    | ...  | ... | ...       |

### 3.1.6 Step 6. Training the Chatbots

In this step, we train the chatbot we created using the “Neural Network (Multilayer Perceptron classifier)” using the training dataset from the completed training dataset table of Step 5.

After the training dataset, we must create the chatbot model by following the steps and sample Python code.

Step 6.1: Create the input data from the completed training dataset table in Table 3.5

The `input_training_dataset_list` will look like the following figure:

| input_training_dataset_list : | Unnamed: 0 | 0   | 1   | 2   | 3   | 4   | ... | 217 | 218 | 219 | 220 | 221 | target |
|-------------------------------|------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--------|
| 0                             | 0          | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| 1                             | 1          | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0      |
| 2                             | 2          | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| 3                             | 3          | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| 4                             | 4          | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| ...                           | ...        | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ...    |
| 95                            | 95         | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| 96                            | 96         | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| 97                            | 97         | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| 98                            | 98         | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |
| 99                            | 99         | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0      |

**Figure 3.7** The result of the training for input of the Neural Network classification algorithm

The training process's input will consist solely of a number as we convert the raw data into the input format for the Neural Network classification algorithm.

For a better understanding, the input and target of the Neural Network classification algorithm are needed. Tabular Data (Structured Data)

1. Example Application: Predicting customer churn, stock prices, or agricultural yield.
2. Format: A table with rows as individual samples and columns as features.

**Table 3.6** The table representation of the table with rows as individual samples and columns as features

| Feature1 | Feature2 | Feature3 | Feature4 | Feature5 |
|----------|----------|----------|----------|----------|
| 2.3      | 1.5      | 0.7      | 8.0      | 1        |
| 1.8      | 2.2      | 0.9      | 7.5      | 0        |
| 2.5      | 1.3      | 0.8      | 8.1      | 1        |

3. Input to Neural Network: A matrix of features (e.g.,  $X = [[2.3, 1.5, 0.7, 8.0], [1.8, 2.2, 0.9, 7.5], \dots]$ )

4. Target/Label: The last column (e.g.,  $y = [1, 0, 1, \dots]$ ).

We need to prepare three datasets in the training process:

1. Question set: the list of questions we transformed into an input format of the Neural Network classification algorithm. The sample of the question set will look like the following figure.

|                     |   |   |   |   |   |     |
|---------------------|---|---|---|---|---|-----|
| train_sequences: [[ | 0 | 0 | 0 | 0 | 0 | 15] |
| [                   | 0 | 0 | 0 | 0 | 0 | 6]  |
| [                   | 0 | 0 | 0 | 0 | 0 | 16] |
| [                   | 0 | 0 | 0 | 0 | 0 | 17] |
| [                   | 0 | 0 | 0 | 7 | 1 | 8]  |
| [                   | 0 | 0 | 0 | 0 | 0 | 9]  |

**Figure 3.8** The question set of the training for input of the Neural Network classification algorithm, defined as train\_sequences

2. Answer set: The answer set is the list of answers from the raw questions and answers paired with the dataset from Table 3.1. The sample of the answers set will look like the following figure.

```
train_labels: ['ใบอนุญาตประกอบกิจการโรงงานไม่มีการกำหนดอายุการใช้งานแบบจำกัด แต่จะมีผลบังคับใช้ต่อ
'ผู้ได้รับใบอนุญาตโรงงานต้องติดตั้งเครื่องมือหรืออุปกรณ์สำหรับควบคุมมลพิษที่อาจเกิดขึ้น เช่น เครื่องบำบัดน้ำเสีย
'ใบอนุญาตโรงงานอาจถูกเพิกถอนได้ในกรณีที่ผู้ประกอบการไม่ปฏิบัติตามกฎหมายโรงงาน ผ่านคำสั่งของพนักงาน
'โรงงานที่มีลักษณะการดำเนินงานที่ส่งผลกระทบต่อสิ่งแวดล้อม เช่น โรงไฟฟ้า โรงงานที่ใช้วัตถุอันตราย ห
'ใบรับแจ้งสำหรับโรงงานจำพวกที่ 2 เป็นเอกสารที่แสดงว่าผู้ประกอบการได้แจ้งรายละเอียดการดำเนินงานต่อ
'พนักงานเจ้าหน้าที่มีสิทธิเข้าตรวจสอบโรงงานโดยไม่ต้องแจ้งล่วงหน้า เพื่อตรวจสอบว่าโรงงานปฏิบัติตามข้อกำหนด
'หากการเปลี่ยนแปลงเครื่องจักรอาจส่งผลกระทบต่อความปลอดภัยของโรงงานหรือมีผลกระทบต่อสิ่งแวดล้อม เช่น การเพิ่ม
'ใบอนุญาตประกอบกิจการโรงงานไม่ต้องต่ออายุ แต่ผู้ประกอบการต้องปฏิบัติตามเงื่อนไขที่ระบุในใบอนุญาตอย่างต่อ
'โรงงานที่ตั้งอยู่ในพื้นที่ชุมชนต้องปฏิบัติตามมาตรการควบคุมมลพิษอย่างเคร่งครัด เช่น ลดเสียงรบกวน ควบคุมการ
'การขนย้ายวัตถุอันตรายจากโรงงานต้องแจ้งพนักงานเจ้าหน้าที่หรือหน่วยงานที่เกี่ยวข้อง เช่น กรมโรงงานอุตสาหกรรม
'โรงงานต้องติดตั้งอุปกรณ์ลดเสียงรบกวน เช่น ผนังกันเสียง หรือเครื่องเก็บเสียง และดำเนินการลดระดับเ
'เครื่องจักรที่มีกำลังผลิตสูงต้องได้รับการตรวจสอบและบำรุงรักษาอย่างสม่ำเสมอ เพื่อป้องกันอุบัติเหตุและผลกระทบ
'โรงงานต้องมีระบบบำบัดน้ำเสียที่ได้มาตรฐานก่อนปล่อยออกสู่สิ่งแวดล้อม และต้องมีการตรวจสอบคุณภาพน้ำเสียเป็
'โรงงานที่เกี่ยวข้องกับวัตถุอันตราย เช่น การผลิตสารเคมี การเก็บรักษาสารไวไฟ หรือโรงงานที่อาจก่อกมลพิษใน
```

**Figure 3.9** The answer set of the training for input of the Neural Network classification algorithm, defined as train\_labels

3. Feature words: The feature words are all the words from all question statements in the training dataset we prepared in Step 2. The sample of the feature words will look like the following figure.

```

feature_words:
['(EIA)', '1', '12', '2', '2535', '3', '7', '8', '9']
['กฎหมาย', 'กรณี', 'กระบวนการผลิต', 'ก่อน', 'กับ', 'การ', 'การควบคุม', 'การแบ่งประเภท',
['เกณฑ์', 'เกิดขึ้น', 'ใกล้', 'ขนย้าย', 'ขนาด', 'ขยะ', 'ขยาย', 'ข้อกำหนด', 'ข้อ', 'ขอ',
['ควบคุม', 'ควร', 'ความปลอดภัย', 'ความหมาย', 'คำนึงถึง', 'คำสั่ง', 'คือ', 'คุณภาพ', 'คุณ',
['จัดทำ', 'จาก', 'จำพวก', 'แจ้ง', 'ชนิด', 'ช่วย', 'ชั่วคราว', 'ชุมชน', 'ใช้', 'ด้าน', 'ด',
['ตรวจสอบ', 'ต้อง', 'ต้องห้าม', 'ต่อ', 'ต่ออายุ', 'ตั้ง', 'ตามกฎหมาย', 'ตาม', 'ติดตั้ง', 'เ',
['น้ำเสีย', 'ในประเทศ', 'ใน', 'บ้าง', 'บุคคลธรรมดา', 'แบ่ง', 'แบบ', 'ใบรับ', 'ใบอนุญาต',
['ผล', 'ผ่าน', 'ผิดกฎหมาย', 'ผูกพัน', 'ผู้ตรวจสอบ', 'ผู้', 'ผู้รับผิดชอบ', 'ผู้อื่น', 'พนักงานเจ้า',
['เพิ่มประสิทธิภาพ', 'เพื่อ', 'มลพิษทางอากาศ', 'มลพิษ', 'มาตรการ', 'มาตรฐาน', 'มาตรา', 'เ',
['ไม่ต้อง', 'ไม่', 'ยกเลิก', 'ยื่นคำขอ', 'ยื่น', 'ร้องเรียน', 'ระบบ', 'ระบายอากาศ', 'รับรอง',
['เลิกกิจการ', 'และ', 'วัตถุติด', 'วัตถุประสงค์', 'วัตถุ', 'วันก่อน', 'วิเคราะห์', 'วิธี', 'เวลา',
['สูญหาย', 'เสียงดัง', 'เสียงที่', 'หน่วยงาน', 'หน้าที่', 'หมดอายุ', 'หมายถึง', 'หมุนเวียน', 'เ',
len feature_words:245

```

**Figure 3.10** The feature words dataset of the training for the input of the Neural Network classification algorithm, defined as feature\_words

After we had already prepared the input dataset of the training process, we had to create the model by using the Neural Network classification algorithm to build the chatbot software. For more details and understanding, we will explain how to build the chatbot in Chapter 4.

After training, the model's weights are updated based on the data. We can use the model to predict or evaluate its performance on new data.

### 3.1.7 Step 7. Testing Prediction

In this step, we created a simulated system for automatic testing because we had to test the Chatbots for a long time, such as 100, 200, and 500 runtimes, to collect the prediction results and save them to log files.

### 3.1.8 Step 8. Building Chatbot Software

Finally, we created a chatbot system based on the Neuron Network to provide the chatbot application designed to serve as an information provider for industrial laws on the web browser.

## 3.2 Our Developed Software

### 3.2.1 The Industrial Laws AI Chatbot

We developed a chatbot based on a Neural Network (Multilayer Perceptron classifier) for compassion finding that is appropriate for our input datasets.

### **3.2.2 Data Preparation System**

Preprocessing is applied to the input text to standardize it according to the system's requirements. It is based on programming syntax and Input optimization.

### **3.2.3 Prediction and Simulation Results Collection and Reporting**

We have developed a simulator system to keep the results from a prediction by the chatbot prediction and generate the results reports, such as log files for re-checking the response manually

We check the accuracy by randomly selecting the questions from the question dataset list in Step 1. We create questions and answers paired. Before we send the question to predict the answer from the model, we define the actual answer to the question and compare it with the expected answer; if the actual answer equals the predicted answer, we make the correct flag for that prediction round and repeat until end of 100 rounds, to proofing the prediction of the model is acceptable.

### **3.2.4 Artificial Intelligence Chatbots**

The AI model is based on a machine-learning algorithm that allows it to learn from the existing database of human conversations. To do so, they must be trained through machine learning algorithms that can be trained. Model using training datasets by using machine learning algorithms. There is no longer any need to define and code. It just needs an input dataset in the required form so we can provide more knowledge by giving them more input, such as more questions and answers.



## CHAPTER 4

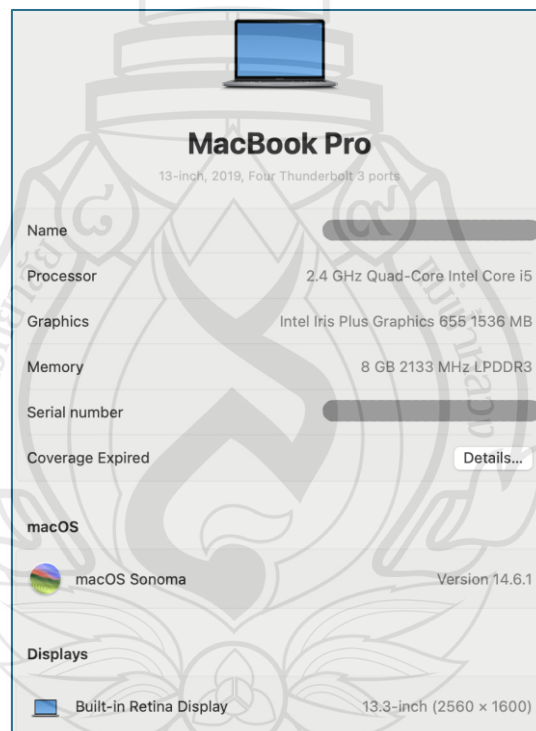
### EXPERIMENTS AND RESULT

#### 4.1 Experiment Condition

##### 4.1.1 Experiment Environment

###### 1. Hardware

We use the MacBook Pro 2019 with hardware specifications, as shown in the figure below.



**Figure 4.1** The developing and testing hardware specification

## 2. Software

**Table 4.1** The table representation of software

| Tool Name | Description   | Version  | License             |
|-----------|---|----------|---------------------|
| MacOS     | macOS is the operating system designed and developed by Apple Inc. for their Mac computers. It's known for its user-friendly interface, stability, and seamless integration with Apple's hardware and ecosystem of devices.   | 14.6.1   | Included in MacBook |
| PyCharm   | PyCharm is a powerful Integrated Development Environment (IDE) for Python, developed by JetBrains. It is one of the most popular IDEs for Python development, offering code completion, debugging, testing, and integration with frameworks like Django, Flask, TensorFlow, and PyTorch.          | 2021.3.3 | Opensource          |
| Python    | Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. It is widely used in various fields, including web development, data science, artificial intelligence, machine learning, automation, and more.                                   | 3.8      | Opensource          |
| SQLite    | SQLite is a lightweight, embedded, serverless database management system that is widely used in applications that need a simple, fast, and reliable database. Unlike other databases like MySQL or PostgreSQL, SQLite does not require a separate server—it stores data in a single file on disk. | 3.0      | Opensource          |

**Table 4.1** (continued)

| <b>Tool Name</b> | <b>Description</b>  | <b>Version</b> | <b>License</b> |
|------------------|---|----------------|----------------|
| Scikit-Learn     | Scikit-Learn is a machine learning library for Python, built on NumPy, SciPy, and Matplotlib. It provides simple and efficient tools for data mining, machine learning, and statistical modeling, making it one of the most popular ML libraries.             | 1.3            | Opensource     |
| Tensorflow       | Open-source machine learning (ML) framework for training and deploying models efficiently using GPUs and TPUs.  | 2.16.2         | Opensource     |
| PyTorch          | Open-source deep learning framework developed by Facebook (Meta). It is widely used in machine learning (ML) and artificial intelligence (AI) research because of its flexibility and ease of use.  | 2.3.0          | Opensource     |
| Keras            | A High-Level API for Deep Learning built on top of TensorFlow. It is designed to be simple, user-friendly, and modular, making it easy to build and train neural networks.  | 3.8.0          | Opensource     |
| Pandas           | Pandas is a powerful and flexible data analysis library for Python. It provides data structures and functions to manipulate, analyze, and visualize structured data efficiently. It is widely used in data science, machine learning, and financial analysis. | 1.4.2          | Opensource     |
| PyTesseract      | PyTesseract is a Python wrapper for Tesseract-OCR, an open-source Optical Character Recognition (OCR) engine. It is used to extract text from images, making it useful for document scanning, CAPTCHA solving, and image-based text processing.               | 0.1.6          | Opensource     |

**Table 4.1** (continued)

| <b>Tool Name</b> | <b>Description</b>  | <b>Version</b> | <b>License</b> |
|------------------|---|----------------|----------------|
| Matplotlib       | Matplotlib is a comprehensive plotting library for Python, widely used for creating static, animated, and interactive visualizations in a variety of formats. It is highly customizable and provides a wide range of tools for visualizing data in 2D and 3D. | 3.1.1          | Opensource     |

#### **4.1.2 Experiment Condition**

To test the best technique, we decided to create a chatbot in Python for deep learning classification algorithm:

##### **Neural Network (Multilayer Perceptron Classification)**

This experiment requires writing a program to test the prediction of the answer to the same question. We create 156 question-answer pairs and use these pairs to create a test dataset to train the created AI chatbot. Then, we run the test by running the program by randomly selecting questions from the 156 questions for 1,000 iterations. We run the test to get the best results by randomly selecting a question once. We will simultaneously send the question to all four bots to collect data, the time it takes to predict the answer, and the answer obtained to analyze. Therefore, after the bots get the answer from the prediction, we also need to check whether the answer is correct or not to explore the accuracy of the prediction of the answer because we want to choose bots that take the time to predict the answer and the accuracy of the answer that is obtained as criteria for consideration.

#### **4.1.3 Prepare a Question-Answer Dataset**

We used information on industrial laws published by the Department of Industrial Works (<https://law.industry.go.th/>) and the Ministry of Industry (Thailand) to create 156 questions and answers for a training dataset.



```

{
  "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",
  "question_id": 1,
  "question_text": "ใบอนุญาตประกอบกิจการโรงงานมีอายุเท่าไร?",
  "answer_id": 1,
  "answer_text": "ใบอนุญาตประกอบกิจการโรงงานไม่มีการกำหนดอายุการใช้งานแบบจำกัด แต่จะมีผลบังคับใช้",
},
{
  "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",
  "question_id": 2,
  "question_text": "ผู้ได้รับใบอนุญาตต้องติดตั้งเครื่องมือควบคุมอะไรบ้าง?",
  "answer_id": 2,
  "answer_text": "ผู้ได้รับใบอนุญาตโรงงานต้องติดตั้งเครื่องมือหรืออุปกรณ์สำหรับควบคุมมลพิษที่อาจเกิดขึ้น เช่น
},
{
  "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",
  "question_id": 3,
  "question_text": "การเพิกถอนใบอนุญาตมีเงื่อนไขอะไร?",
  "answer_id": 3,
  "answer_text": "ใบอนุญาตโรงงานอาจถูกเพิกถอนได้ในกรณีที่ผู้ประกอบการไม่ปฏิบัติตามกฎหมายโรงงาน ผ่า
},
{
  "tag": "พระราชบัญญัติ โรงงาน พ.ศ. 2535",
  "question_id": 4,
  "question_text": "โรงงานประเภทใดต้องจัดทำรายงานผลกระทบสิ่งแวดล้อม?",
  "answer_id": 4,
  "answer_text": "โรงงานที่มีลักษณะการดำเนินงานที่ส่งผลกระทบต่อสิ่งแวดล้อมต้องสิ่งแวดล้อม เช่น โรงไฟฟ้า โรงงาน
},

```

Figure 4.4 A sample of question-answer pair in JSON form

#### 4.1.4 How Do We Do the Word-collecting Process?

We extract the text from the pdf file by coding in Python by following the steps below.

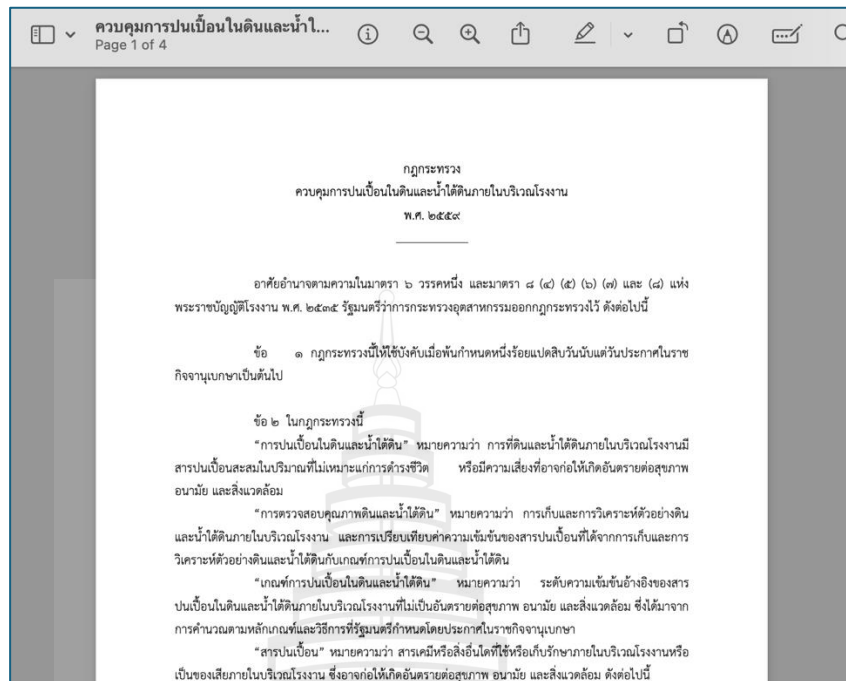
1. Get text from pdf using PyCharm
  - 1) Installation of PyPDF2 in the PyCharm

In the terminal window command: *pip install pypdf2*



Figure 4.5 Pypdf2 package installation in PyCharm CE

- 2) Example Python code for extracting text from PDF file  
Preparing the source file (PDF file)



**Figure 4.6** The sample source of the industrial laws (pdf file format)

The sample Python code function to extract text from pdf file:

```
from PyPDF2 import PdfReader

# --the extracting text function using PdfReader single file
def get_single_pdf_text(pdf):
    text = ""
    pdf_reader = PdfReader(pdf)
    for page in pdf_reader.pages:
        text += page.extract_text()
    return text

# --the sample pdf file
pdf_path = 'docs/ควบคุมการปนเปื้อนในดินและน้ำใต้ดินภายในบริเวณโรงงาน.pdf'

# calling extracting function
print(f'data from ', pdf_path)
read_text = get_single_pdf_text(pdf_path)

# display result (raw text)
print('result text:')
print(read_text)
```

**Figure 4.7** The sample Python code for extracting text from the source file

The result:

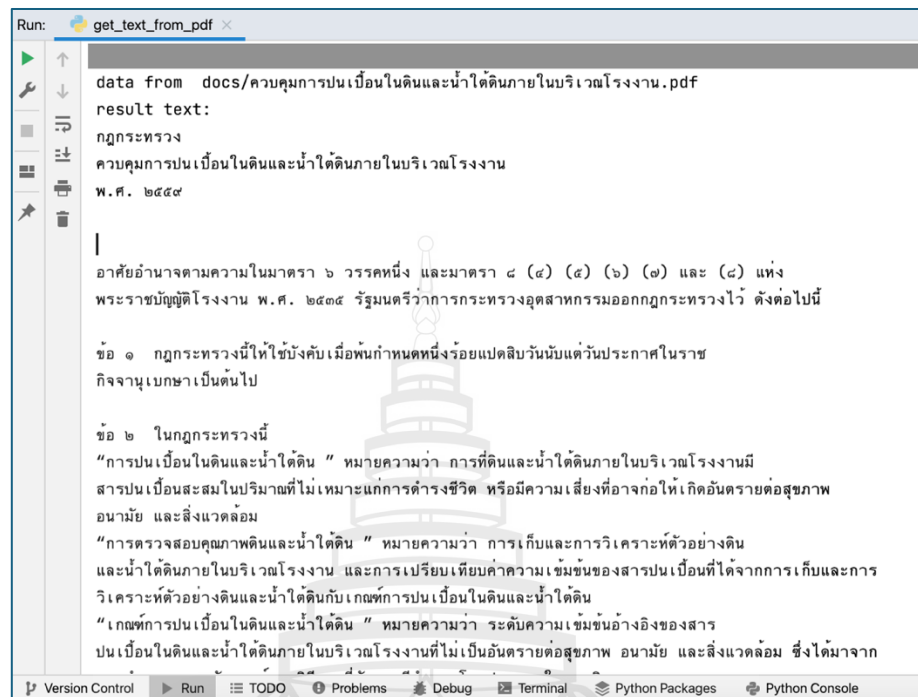


Figure 4.8 The sample result from the extracting process from the source file

## 2. Clean text from the result of the extracting process.

1) Install the thaisspellcheck library and use it to clean text from the result of the extracting process:

In the terminal window command: *pip install thaisspellcheck*



Figure 4.9 Thaisspellcheck package installation in PyCharm

## 2) Coding Python to detect missing spelling words:

The sample Python code to detect missing spelling word



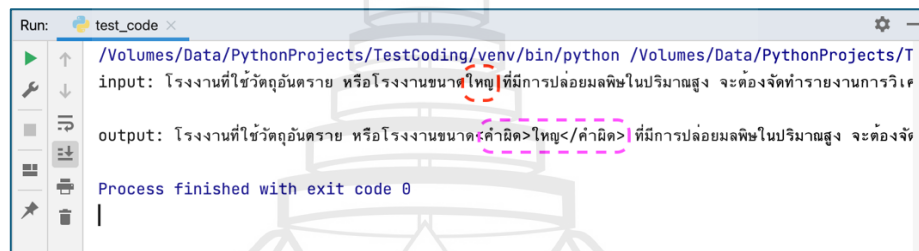
```
import thaispellcheck

# the sample input text that's we want spelling check
text = "โรงงานที่ใช้วัตถุดิบทราย หรือโรงงานขนาดใหญ่ ที่มีการปล่อยมลพิษในปริมาณสูง " \
       "จะต้องจัดทำรายงานการวิเคราะห์ผลกระทบสิ่งแวดล้อม เพื่อให้หน่วยงานที่เกี่ยวข้องตรวจสอบและอนุมัติก่อนเริ่ม"

print('input:',text)
# execute thaispellcheck to check
checked_sentence = thaispellcheck.check(text)
|
print('output:',checked_sentence)
```

**Figure 4.10** The sample Python code for spell-checking

The result from the thaispellcheck function



```
Run: test_code x
/Volumes/Data/PythonProjects/TestCoding/venv/bin/python /Volumes/Data/PythonProjects/T
input: โรงงานที่ใช้วัตถุดิบทราย หรือโรงงานขนาดใหญ่ ที่มีการปล่อยมลพิษในปริมาณสูง จะต้องจัดทำรายงานการวิเคราะห์
output: โรงงานที่ใช้วัตถุดิบทราย หรือโรงงานขนาดใหญ่<คำผิด>ที่มีการปล่อยมลพิษในปริมาณสูง จะต้องจัดทำ
Process finished with exit code 0
```

**Figure 4.11** The sample result from the spell-checking process

After we did this step, we used the cleaned data to create the question-and-answer pairs table (manually) in Table 3.1

### 3. Word-filtering and sorting process.

We fill the duplicated words out and sort the text from the pdf file by coding in Python by following steps:

1) Install the thaispellcheck library and use it to clean text from the result of the extracting process:

In the terminal window command: pip install thaispellcheck

```
import pandas as pd

# The sample training dataset file in JSON format
jsonFilePath = 'docs/training_dataset_3.json'
# Real data from file
data_in_text = pd.read_json(jsonFilePath)
# Transform to Dataframe
df = pd.DataFrame(data_in_text)
# collection text from training dataset
all_text = ""
for index, row in df.iterrows():
    question_text = row[0]['question_text']
    all_text += question_text
# show result
print('All words from all questions = ', all_text)
```

**Figure 4.12** The sample Python code to get the text from all question statements before extracting words from statements

The result of Figure 4.11 will look like this:

All words from all questions = “ใบอนุญาตประกอบกิจการโรงงานมีอายุเท่าไร? ผู้ได้รับใบอนุญาตต้องติดตั้งเครื่องมือควบคุมอะไรบ้าง? การเพิกถอนใบอนุญาตมีเงื่อนไขอะไร? โรงงานประเภทใดต้องจัดทำรายงานผลกระทบสิ่งแวดล้อม? ใบรับแจ้งของโรงงานจำพวกที่ 2 มีผลผูกพันอย่างไร? พนักงานเจ้าหน้าที่มีสิทธิอะไรบ้างในการตรวจโรงงาน? การขอเปลี่ยนแปลงเครื่องจักรต้องแจ้งพนักงานเจ้าหน้าที่หรือไม่? การต่อใบอนุญาตทำได้เมื่อใด? โรงงานที่ตั้งในพื้นที่ชุมชนต้องมีข้อกำหนดอะไรเพิ่มเติม? การขนย้ายวัตถุดิบอันตรายจากโรงงานต้องแจ้งใคร?...”

2) Extract words from the result of Figure 4.11, then sort all and filter out duplicate words.

We use `word_tokenize` from `pythainlp` to extract words from statements.

```

from pythainlp import word_tokenize
from pythainlp.util import collate

text = "ใบอนุญาตประกอบกิจการโรงงานมีอายุเท่าไร?ผู้ได้รับใบอนุญาตต้องติดตั้งเครื่องมือควบคุมอะไรบ้าง?การเพิกถอนใ  

"โรงงานประเภทใดต้องจัดทำรายงานผลกระทบสิ่งแวดล้อม?ใบรับแจ้งของโรงงานจำพวกที่ 2 มีผลผูกพันอย่างไร?"  

"พนักงานเจ้าหน้าที่มีสิทธิอะไรบ้างในการตรวจโรงงาน?การขอเปลี่ยนแปลงเครื่องจักรต้องแจ้งพนักงานเจ้าหน้าที่หรื  

"การต่อใบอนุญาตทำได้เมื่อใด?โรงงานที่ตั้งในพื้นที่ชุมชนต้องมีข้อกำหนดอะไรเพิ่มเติม?การขนย้ายวัตถุอันตรายจากโ  

"โรงงานต้องมีการลดเสียงดังอย่างไร?ข้อกำหนดสำหรับการใช้เครื่องจักรที่มีกำลังผลิตสูงคืออะไร?การจัดการ  

"โรงงานประเภทใดที่ต้องได้รับอนุญาตพิเศษ?โรงงานต้องจัดการของเสียที่ไม่สามารถรีไซเคิลได้อย่างไร?หากพบไร  

"การเปลี่ยนแปลงพื้นที่โรงงานต้องแจ้งอะไรบ้าง?โรงงานต้องมีแผนป้องกันอัคคีภัยอย่างไร?โรงงานที่ใช้พลังงานหมุน  

"การตรวจสอบโรงงานโดยพนักงานเจ้าหน้าที่ต้องทำเมื่อใด?โรงงานต้องจัดการมลพิษทางอากาศอย่างไร?การจัดเก้

# Remove unneeded words or characters
text = text.replace('?', '').replace(' ', '')
words = word_tokenize(text)
print('All Words = ', words)

words = collate(words)
print('Sort all words = ', words)
# Remove Duplicate words
unique_words = list(set(words))
print('Unique words = ', unique_words)
# Sort the word for create the feature Datatable
sorted_words = collate(unique_words)
print('Sorted words = ', sorted_words)

```

**Figure 4.13** The sample Python code to extract words from statements and how to sort them

```

Run: extract_words_from_text x
/Volumes/Data/PythonProjects/TestCoding/venv/bin/python /Volumes/Data/PythonProjects/Tes
All Words = ['ใบอนุญาต', 'ประกอบ', 'กิจการ', 'โรงงาน', 'มีอายุ', 'เท่าไร', 'ผู้', 'ได้รับ', 'ใบอนุ  

Sort all words = ['', '1,2', '1', '2', '2', '2', '2', '3', '3', '7', '8', '8', '9', 'ก  

Unique words = ['ขอเขต', 'ปล่อย', 'ปฏิบัติตาม', 'ขนย้าย', 'ขอ', 'ชุมชน', 'หรือ', 'ผู้', 'เพิ่มประสิ  

Sorted words = ['', '1,2', '1', '2', '3', '7', '8', '9', 'กฎกระทรวง', 'กฎหมาย', 'กรณี',

```

**Figure 4.14** The result of the word-filtering and sorting process

### 4.1.5 Training the Chatbot

After we have prepared the training dataset, we train the model by using the Neural Network classification algorithm to provide more details and understanding of how to build and code the model, and we will explain each line of code as follows:

```

import pandas as pd
import keras
import utils.sentence_utils as util

def available

model = keras.models.Sequential()
model.add(keras.layers.Embedding(len(feature_words) + 1, 100, input_length=train_sequences.shape[1]))
model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(64, activation='relu'))
model.add(keras.layers.Dense(len(train_labels), activation='softmax'))
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model.fit(train_sequences, encoded_labels, epochs=50)

```

**Figure 4.15** The lines of code for training the model using the Neural Network classification algorithm

To understand, we explain the code line by line. For example, the “Line of: `model = keras.models.Sequential()`” means we will explain what the line of code is doing and also explain each parameter in the line of code.

The expansion of code from Figure 4.13:

**Line of: `model = keras.models.Sequential()`:**

We define a valuable model as a `keras.models.Sequential()`. The `keras.models.Sequential()` is a class in Keras used to build a Sequential model, one of the simplest types of Neural Network Models. It is a linear stack of layers where each layer has exactly one input tensor and one output tensor, meaning the layers are added one after the other in a sequence.

**Key Features of Sequential:**

1. Layer-by-Layer Construction:

We build the model by adding layers sequentially using `model.add()`. The data flows from the first layer to the last layer in a forward direction.

2. Ease of Use:

It's simple to set up and works well for models with a straightforward architecture, where the data flows linearly from one layer to the next.

3. Limitations:

It does not support models with multiple inputs, multiple outputs, or layers with complex connections, such as shared layers or skip connections. We would use the Functional API or the Model subclassing approach for such models.

Line of: `model.add(keras.layers.Embedding(len(feature_words) + 1, 100, input_length=train_sequences.shape(1)))`:

In the above line of code, we want to add embedding the feature words to the layers of the model

The embedding layer represents words (feature\_words, or we can call the tokens) as dense vectors in a continuous space. Instead of treating words as discrete entities, embeddings map them to a high-dimensional vector space where semantically similar words are closer together.

In this line of code, the input is Integer-encoded words (sequences of token indices).

The output is dense vector representations (embeddings) of those words.

This layer is advantageous in Natural Language Processing (NLP) tasks, such as text classification, sentiment analysis, machine translation, etc.

The parameters in the code:

1. `len(feature_words) + 1`

This is the vocabulary size, i.e., the number of unique tokens (words) in our dataset.

feature\_words likely contains the set of unique words/tokens, and +1 accounts for an extra padding token or unknown token.

Example: If there are 10,000 unique words, the vocabulary size will be 10,001.

2. `100`

This is the embedding dimension or the size of the dense vector representation for each word.

Each word will be mapped to a vector of length 100. For example:

The word “apple” → [0.12, -0.34, ..., 0.56] (100 values).

We can tune this value based on our dataset and computational resources. Typical choices are 50, 100, 300, etc.

3. `input_length=train_sequences.shape(1)`

This specifies the length of the input sequences, i.e., the number of tokens in each sequence (fixed padding/truncation).

The `train_sequences` is likely a 2D array, with each row containing a sequence of token indices. `train_sequences.shape[1]` gives the number of tokens in each sequence.

**Line of: `model.add(keras.layers.Flatten())`:**

The line `model.add(keras.layers.Flatten())` adds a Flatten layer to our model in Keras. This layer transforms a multidimensional input (such as a 2D or 3D tensor) into a 1D vector. It is typically used when transitioning from convolutional or other feature extraction layers to fully connected (dense) layers.

The input of this line of code is A multi-dimensional tensor (e.g., from a convolutional or pooling layer). The output is a 1D vector, which preserves the batch size while collapsing all other dimensions into a single one. In neural networks, convolutional and pooling layers often output multi-dimensional arrays (e.g., feature maps). However, fully connected (Dense) layers expect inputs from 1D vectors. The Flatten layer bridges this gap by reshaping the data.

For example:

We can flatten the 3D feature maps (14, 14, 32) into a 1D vector of shape by using a multiplier ( $14 * 14 * 32 = 6272$ ), which means we have new data in a 1D vector as [6272]

**Line of: `model.add(keras.layers.Dense(64, activation='relu'))`:**

In this line of code, we want to add the dense layers into model layers. The dense layer is a fully connected layer, meaning:

Every neuron in the current layer is connected to every neuron in the previous layer. It's one of the most commonly used layers in feedforward neural networks.

The parameters in the code:

1. 64:

This specifies the number of neurons in the dense layer.

In this case, the layer has 64 neurons.

2. `activation='relu'`:

1) Specifies the activation function applied to the neurons' output.

- 2) ReLU (Rectified Linear Unit) is short for  $f(x) = \max(0, x)$ .
- 3) ReLU introduces nonlinearity to the model, allowing it to learn more complex patterns. It is widely used because it helps prevent the vanishing gradient problem and improves training efficiency.

### How It Works in the Model

**When data is passed through this layer:**

**Input Dimension:**

If the previous layer (or input data) has  $n$  features, each input is multiplied by the corresponding weight for each neuron. This means there are  $n \times 64$  weights in this layer (plus biases for each neuron).

For example:

If the input has 10 features, there will be  $10 \times 64 = 640$  weights + 64 biases.

**Output Dimension:**

The layer outputs a vector with 64 elements (one for each neuron).

Each output value is computed as:

$$y_i = \text{ReLU} \left( \sum_j (w_j \cdot x_j) + b_i \right)$$

where  $w_j$  are the weights,  $x_j$  is the input, and  $b_i$  is the bias.

**Figure 4.16** The formula to calculate the output of a single neuron in a neural network using the ReLU activation function

The ReLU (Rectified Linear Unit) is one of the most commonly used activation functions in neural networks, especially in deep learning models. Its purpose is to introduce non-linearity into the model, allowing the network to learn complex patterns in the data.

**Line of: `model.add(keras.layers.Dense(len(train_labels), activation='softmax')):`  
`model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',  
metrics=['accuracy']):`**

In this line is the step configures how the model will be trained, including the optimization algorithm, loss function, and evaluation metrics.

Key Components:

1. optimizer='adam':

Adam (Adaptive Moment Estimation) is a widely used optimization algorithm.

It combines the benefits of momentum-based gradient descent and adaptive learning rates to optimize the model efficiently.

Adam is robust and works well in most cases without much hyperparameter tuning.

2. loss='sparse\_categorical\_crossentropy':

1) This loss function evaluates the difference between the predicted probabilities and the actual labels during training.

2) sparse\_categorical\_crossentropy is used when:

The labels are provided as integers (e.g., [0, 1, 2]) instead of one-hot encoded vectors (like [1, 0, 0] for class 0).

3) The function calculates the cross-entropy loss:

$$\text{loss} = -\frac{1}{N} \sum_{i=1}^N \log(p_{i,y_i})$$

Where:

- $N$  is the number of samples.
- $p_{i,y_i}$  is the predicted probability for the true class  $y_i$ .

**Figure 4.17** The formula for calculating the loss of predicted probabilities in a neural network

3. metrics=['accuracy']:

4. Accuracy is used to evaluate the performance of the model.

5. Keras calculates how often the model's predictions match the true labels during training and validation.



Purpose of this line of code:

The dense layer with a softmax activation ensures the output is a probability distribution over all possible classes.

The sparse categorical cross-entropy loss function evaluates how well the predicted probabilities align with the actual labels.

The Adam optimizer efficiently updates the model's weights and accurately gives feedback on how well the model performs during training.

**Line of: `model.fit(train_sequences, encoded_labels, epochs=50):`**

In this line, the `model.fit()` is the function that trains the model using the given input data (`train_sequences`) and corresponding target labels (`encoded_labels`). It adjusts the model's weights to minimize the loss function defined during the `model.compile()`. The parameters `encoded_labels` is

Components of the Code:

1. `train_sequences`:

- 1) This is the input data to the model.
- 2) It represents the features used for training the neural network.
- 3) Example:
  - a. If we train a text classification model, `train_sequences` might be tokenized and padded text sequences.
  - b. For image classification, it could be arrays of pixel values.

2. `encoded_labels`:

- 1) These are the target labels corresponding to the input data.
- 2) The labels should match the type of loss function used:
  - a. If we use `sparse_categorical_crossentropy`, the labels should be integers (e.g., [0, 1, 2] for three classes).
  - b. If we use `categorical_crossentropy`, the labels should be one-hot encoded (e.g., [1, 0, 0] for class 0).

3) Example:

For a classification task with three classes, `encoded_labels` might look like [0, 1, 2, 0, 2, 1].

### 3. epochs=50:

1) Epochs represent the number of complete passes through the training data.

2) During each epoch, the model processes the entire training dataset once.

#### 3) Example:

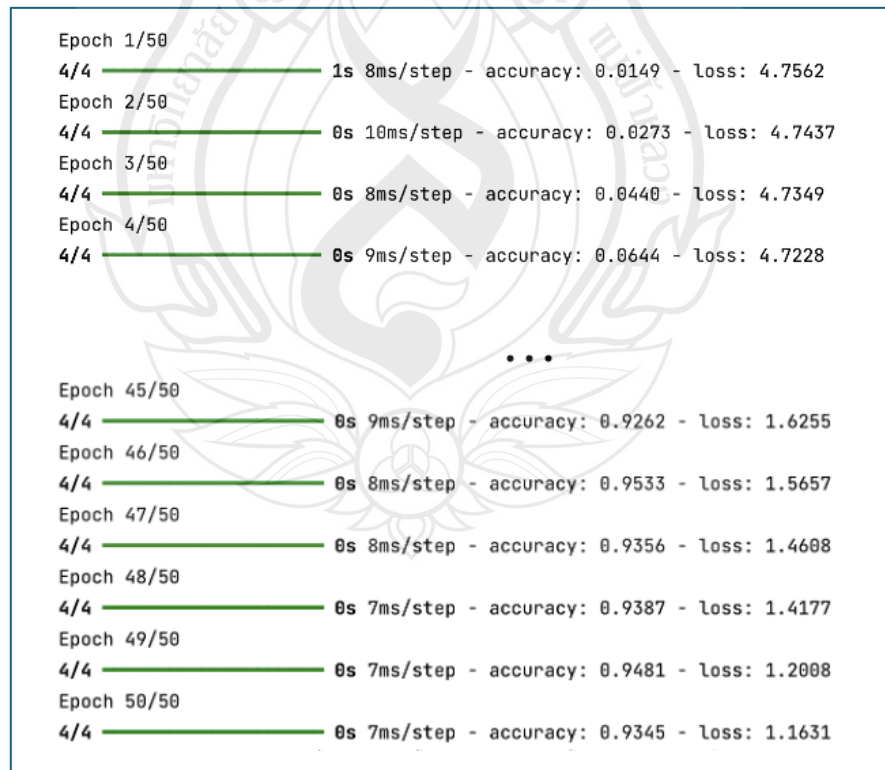
If train\_sequences has 1000 samples and epochs=50, the model will process the dataset 50 times.

4) Increasing the number of epochs allows the model to learn better but risks overfitting if too many epochs are used.

### Key Outputs of model.fit()

#### Training Process:

The following figure shows the loss and metrics (like accuracy) printed in the training process for each epoch.



**Figure 4.18** The output report for each epoch in the training process

Trained Model:

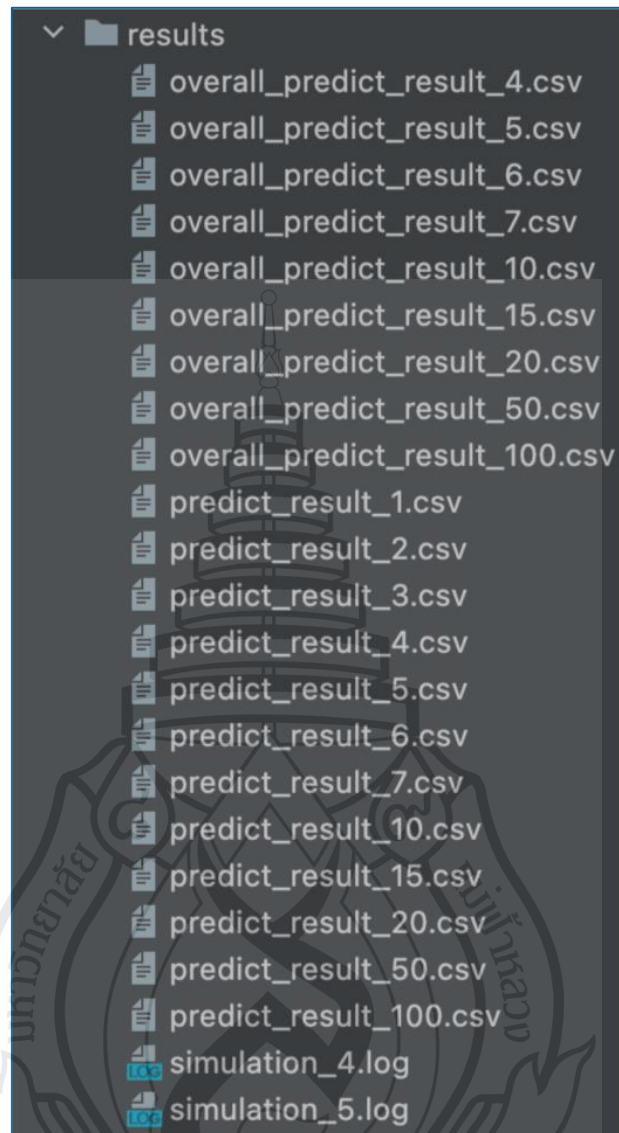
After training, the model's weights are updated based on the data. We can use the model to predict or evaluate its performance on new data. We can code as shown in the following figure.

```
def predict_response(text):
    sequence = tokenizer.texts_to_sequences([text])
    sequence = keras.preprocessing.sequence.pad_sequences(sequence, maxlen=train_sequences.shape[1])
    prediction = model.predict(sequence)
    predicted_label = np.argmax(prediction)
    response = label_encoder.inverse_transform([predicted_label])[0]
    return response
```

**Figure 4.19** The sample code to predict the answer to the new question by using the trained model

## 4.2 Experiment

In running the program test, since system testing takes quite a long time because it uses heavy computer processing, we use the maximum test rounds at 10,000 prediction rounds that depend on our hardware, each round sending the same question to each bot one by one (to fix them to predict the answer with same question nearest time and same environment) store the result in a log file) and keep the results of the running time and accuracy answer rate in log files, as shown in the figure below.



**Figure 4.20** Logging files by testing loop

After running the test, we write a program to analyze the results of the chatbot run and display them in an easy-to-understand format, such as a summary table and several graphs, to see the results and make them easily understood.

```

*****
=====> Predict simulation <=====
Simulate Date: 2025-01-23 11:12:18.966310
runtimes: 10000
samples: 156
features: 257

=====> Training Results <=====
Method | prediction time(Avg. in ms) | Accuracy(Avg)
-----|-----|-----
NN:Neural Network Classifier 297.024 100
-----|-----|-----

=====> Testing Results <=====
No. of testing samples: 156 with 257 features

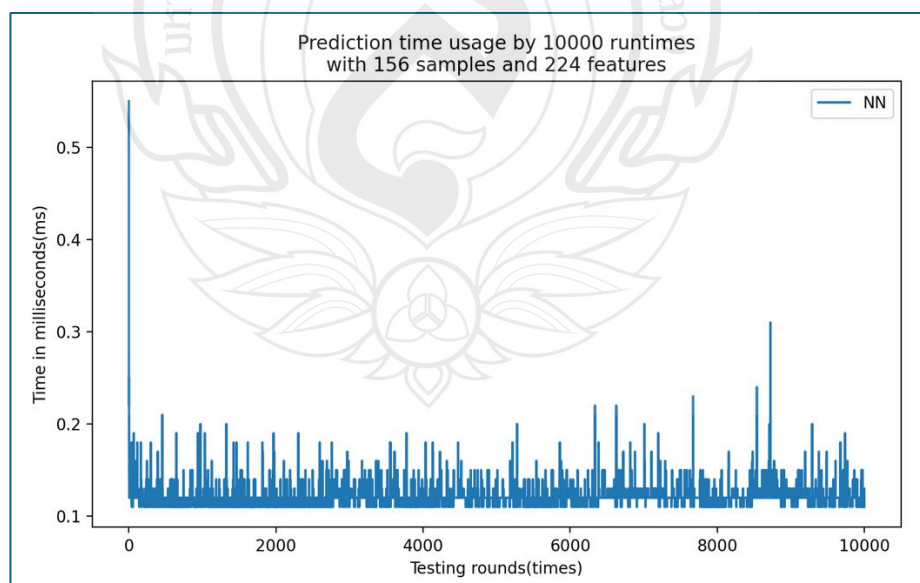
Method | prediction time(Avg. in ms) | Accuracy(Avg)
-----|-----|-----
NN      0.336 100
-----|-----|-----

Simulate Processing time: 3779.072 ms

=====> Predict simulation successful <=====
*****

```

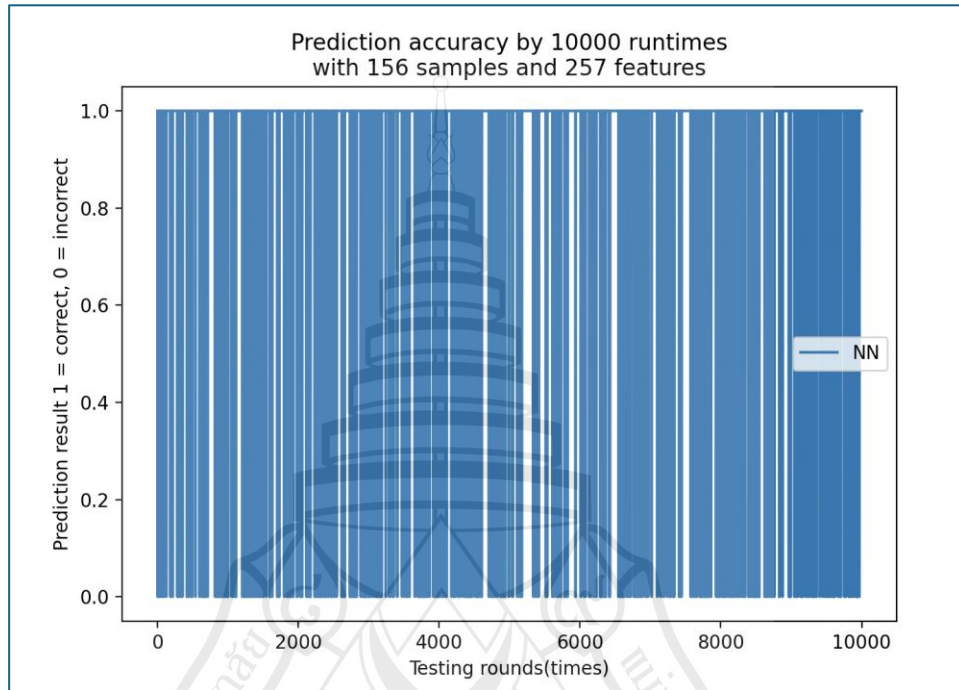
**Figure 4.21** The sample report of result logging file



**Figure 4.22** The prediction time usage results of the Chatbots

| Simples | Features(Tokens) | Training Time Usage(ms) | Accuracy Rate(%) | Loss Rate(%) |
|---------|------------------|-------------------------|------------------|--------------|
| 156     | 257              | 2688.519                | 91.42            | 1.56         |

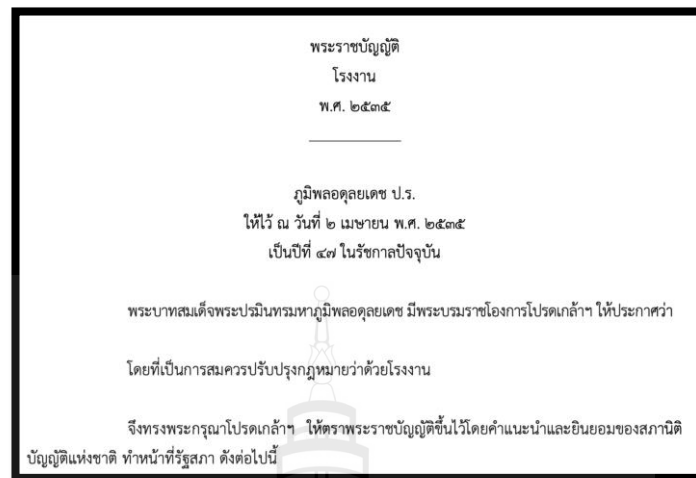
**Figure 4.23** Summary reporting table of training results logging file



**Figure 4.24** The prediction accuracy result, the blue in the graph means the prediction response is corrected, and the white is incorrect

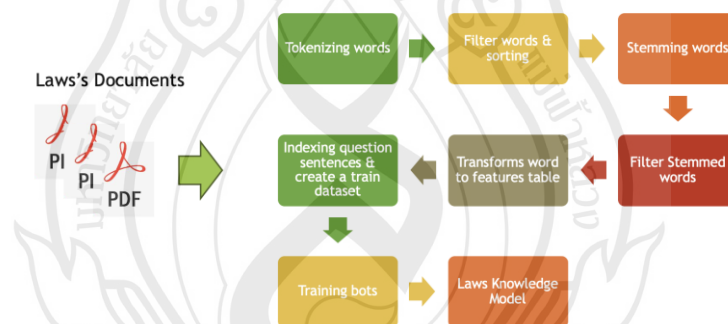
### 4.3 Results

We use the selection of Thai Industrial Laws from the Office of the Council of State by downloading documents such as various acts in pdf file format, as shown in the picture below.



**Figure 4.25** The sample source file of Thai industrial laws

The initial data we need to use to teach the chatbot is a PDF file. Some additional steps in preparing the data are to make it convenient for users, so we have developed a system to support direct file import according to the steps in the figure.



**Figure 4.26** The training dataset collection workflow

For user-friendly usability, we have designed a chatbot via a web application form to provide a quick and easy-to-use solution for answering questions and insights into Thai industrial laws that can be easily set up and used anywhere, anytime.



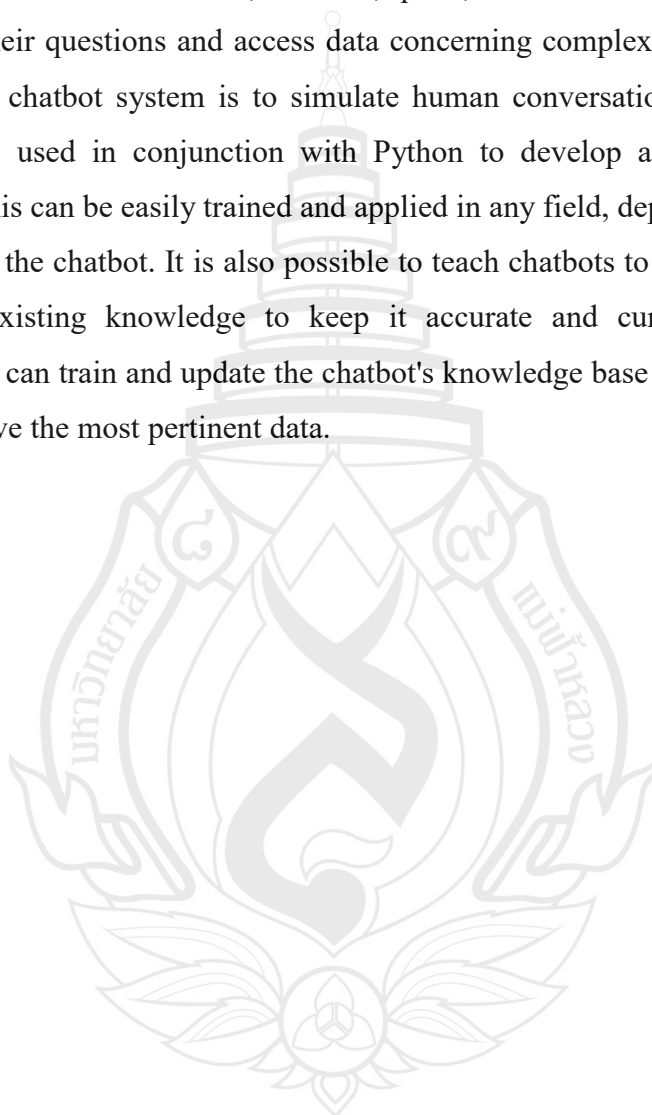
**Figure 4.27** The industrial laws chatbot application



## CHAPTER 5

### CONCLUSION

Chatbots are an effective, efficient, quick, and accurate way for users to seek answers to their questions and access data concerning complex Industrial Laws. The purpose of a chatbot system is to simulate human conversations. Natural language processing is used in conjunction with Python to develop a chatbot. Simple but automated, this can be easily trained and applied in any field, depending on the dataset used to teach the chatbot. It is also possible to teach chatbots to learn new knowledge or modify existing knowledge to keep it accurate and current. The trainer or administrator can train and update the chatbot's knowledge base anytime, enabling the chatbot to have the most pertinent data.



## REFERENCES

- Abdullahi, H. S., Sheriff, R. E., & Mahieddine, F. (2017). Convolution neural network in precision agriculture for plant image recognition and classification. In *2017 Seventh International Conference on Innovative Computing Technology (INTECH)* (pp. 1-3). IEEE.  
<http://doi.org/10.1109/INTECH.2017.8102436>
- Garg, R., Riya, R., Thakur, S., Tyagi, N., Basha, K. N., Vij, D., . . . Sodhi, G. S. (2021). NLP based chatbot for multiple restaurants. In *2021 10th International Conference on System Modeling & Advancement in Research Trends (SMART)* (pp. 439–443). IEEE.  
<http://doi.org/10.1109/smart52563.2021.9676218>
- Guolin, D. (2007). Research on risk evaluation model of project financing based on neural network. In *2007 IEEE International Conference on Grey Systems and Intelligent Services* (pp. 1072-1076). IEEE.  
<http://doi.org/10.1109/GSIS.2007.4443437>
- Hashana, A. M. J., Brundha, P., Ayoobkhan, M. U. A., & Fazila, S. (2023). Deep learning in CHATGPT - A survey. In *2023 7th International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 1001–1005). IEEE.  
<http://doi.org/10.1109/icoei56765.2023.10125852>
- Holdsworth, J., & Scapicchio, M. (2024). *What is deep learning?*.  
<https://www.ibm.com/se-en/topics/deep-learning>
- Huang, Y. (2010). Study of the college network aided teaching platform. In *2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*. IEEE. <https://doi.org/10.1109/icacte.2010.5579856>
- IBM. (2024). *What is a neural network?*. <https://www.ibm.com/topics/neural-networks>

- Jothi, J. N., Poongodi, S., Chinnammal, V., Kannagi, L., Panneerselvam, M., . . . Prabu, R. T. (2022). AI based humanoid chatbot for medical application. In *2022 3rd International Conference on Smart Electronics and Communication (ICOSEC)* (pp. 1135–1140). IEEE.  
<https://doi.org/10.1109/icosec54921.2022.9951910>
- Prayitno, P. I., Leksono, R. P. P., Chai, F., Aldy, R., & Budiharto, W. (2021). Health chatbot using natural language processing for disease prediction and treatment. In *2021 1st International Conference on Computer Science and Artificial Intelligence (ICCSAI)* (pp. 62–67). IEEE.  
<https://doi.org/10.1109/iccsai53272.2021.9609784>
- Reddy, V. V., Pavan Kumar, P. V., & Suvana Vani, K. (2023). Lung cancer stage classification utilizing k-Nearest Neighbors (k-NN) and Convolutional Neural Networks (CNN). In *2023 2nd International Conference on Futuristic Technologies (INCOFT)* (pp. 1-7). IEEE.  
<https://doi.org/10.1109/INCOFT60753.2023.10425683>
- Singh, J., & Banerjee, R. (2019). A study on single and multi-layer perceptron neural network. In *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)* (pp. 35-40). IEEE.  
<http://doi.org/10.1109/ICCMC.2019.8819775>
- Yan, R., Song, Y., & Wu, H. (2016). Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 55-64). Association for Computing Machinery.  
<https://doi.org/10.1145/2911451.2911542>
- Yu, X., Efe, M. O., & Kaynak, O. (2002). A general backpropagation algorithm for feedforward neural networks learning. *IEEE Transactions on Neural Networks*, 13(1), 251-254. <https://doi.org/10.1109/72.977323>
- Zhou, L., Gao, J., Li, D., & Shum, H.-Y. (2020). The design and implementa Xiaoice, an empathetic social chatbot. *Computational Linguistics*, 46(1), 53-93. [https://doi.org/10.1162/coli\\_a\\_00368](https://doi.org/10.1162/coli_a_00368)

## APPENDIX

### THE SOURCE DOCUMENT OF INDUSTRIAL LAWS

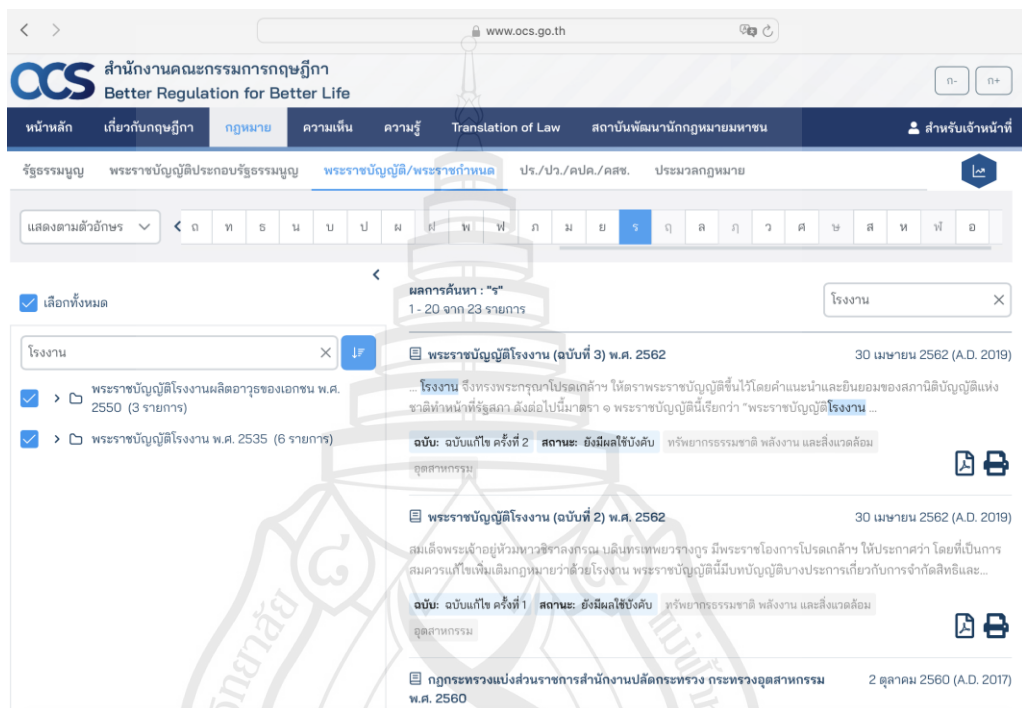
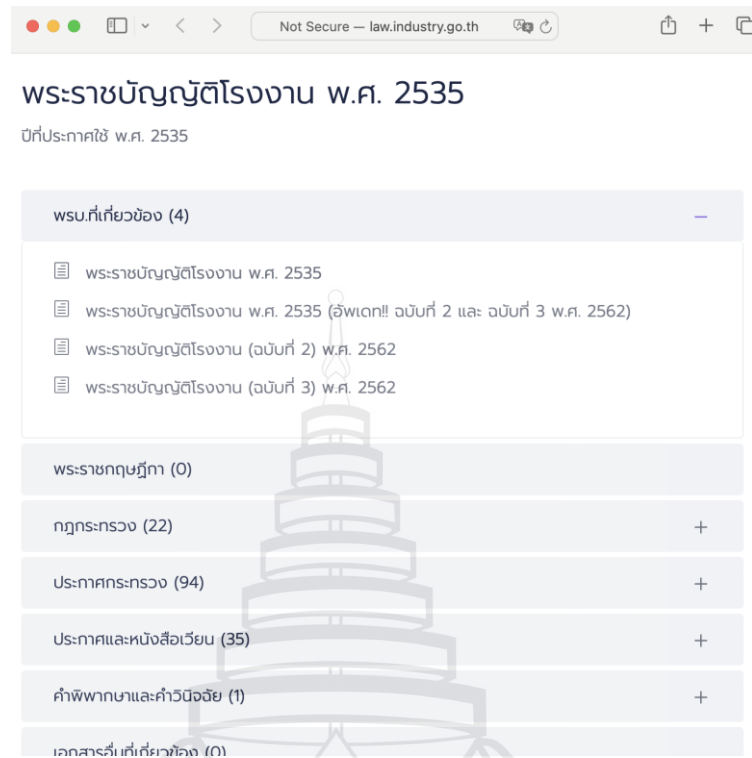


Figure A1 The Office of The Council of State (www.osc.go.th)



Figure A2 Department of Industrial Works (https://www.diw.go.th/)



**Figure A3** The Source file (<http://law.industry.go.th>)

## CURRICULUM VITAE

**NAME**

Suttidech Jittawisuttikul

**EDUCATIONAL BACKGROUND**

2004

Bachelor's Degree of Engineering and  
Technology  
Computer Science  
Mae Fah Luang University

**WORK EXPERIENCE**

2013 – Currently

Senior Programmer  
Relation Soft Company Limited

2012 – 2013

Senior Programmer  
Kiatnakin Phatra Bank Public Company  
Limited

2011 – 2012

Programmer  
Home Product Center Public Company  
Limited

2008 – 2011

Programmer  
Crystal Software Group Public Company  
Limited

2005 – 2008

Programmer  
Abstract Computer Company Limited

2004 – 2005

Programmer  
Western University