Text Classification in Thai language with Convolutional Neural Network

Jethnithi Techasatian

Ratchanan Prasan

Bachelor of Engineering in Software Engineering

International College

King Mongkut's Institute of Technology Ladkrabang

Academic Year 2018

KMITL-2019-IC-B-003-004

Thesis - Academic Year 2018

Bachelor of Engineering in Software Engineering
International College
King Mongkut's Institute of Technology Ladkrabang

Title: Text Classification in Thai language with Convolutional Neural

Authors:

1. Mr. Jethnithi Techasatian Student ID: 58090008
2. Mr. Ratchanan Prasan Student ID: 58090033

Approved for Submission

.....

(Dr. Isara Anantavrasilp)
Advisor

Date ...../...../.....

Acknowledgments

We would like to express our sincere gratitude to Dr.Isara Anantavrasilp, our project advisor, for providing resources with knowledge during the project. During the initial phase, the project seems to be fail and impossible to be achieved without our advisor the goal may not have been achieved. Every comment and suggestion improve the project into the right direction. We would also like to thank Dr.Ukrit Watchareeruetai for letting us use the computer vision laboratory. Lastly, we would like to thanks our family for supporting us and always motivating us to the end.

Abstract

In the generation where the internet plays an important role in the human cycle, the fact that articles are all over our social can be reliable. Articles can be propagated in several way author creativity, the field of study, familiarity, or coincidence. One of the huge challenges for many researchers in Natural language processing (NLP) is how to make a machine acknowledges a specific article. the success of this challenge would lead to efficiently application such as the article's recommendation system which suggest articles based on reader interest and behavior and could be used in many huge platforms.

This paper focuses on Thai language. Compare to English language, Thai language has its own difficulty, for instance, defining word boundaries, separating vowels and etc. This paper observes, tokenizes and categorizes the article in several methods using Convolutional Neural Networks (CNNs). There are 2 types of CNN, 2D Convolutional Neural Network (2D CNN) and 1D Convolutional Neural Network (1D CNN). Both 2D CNNs and 1D CNNs have been tested and visualized in different parameters. 1D CNNs performance is slightly higher compare to 2D CNNs. For feature extract processing, the article's content extracts into a vector with vary size before input into the deep learning model. Therefore, the 1D CNNs used in proposed models.

Table of contents

List of figures

List of tables

Chapter 1
Introduction

1.1 Problem description

One of the most interesting is article classification that is important for other parts in language processing. In order to discover the best classifier, the model needs to be well trained, tested with various parameters. However, Thai language itself is a difficult problem. Therefore, studies for Thai language processing still not widely developed and normally focus only on one task, for example, identify positive, negative, neutral polarities in subjective sentences.

1.2 Objective

In order to indicate the accomplishment of the project. The following goals must be satisfied:

- System can classify articles based on the given characteristics.
- To study the effectiveness of integrating CNN into natural language processing.
- The overall system's accuracy is expected to be more than 60%.
- To compare the performance among parameters including data augmentation, stop word filtering, amount of data and languages.

1.3 Report structure

The proposed research focuses on text tokenization and article classification. The first module splits words from the sequence of words input depends on the morphology, syntactic, and semantic. Then, the module performs to stop words filtering. The result from this module converts into vectors and output into one of the given classes.

The resource for this module is articles and trends. The article used in order to train the model is from Wikipedia, on the other hand, trends are from Google Trends. 20% of the dataset converted into testset in order to measure of the model in term of accuracy.

- Chapter 2 Background Knowledge
- Chapter 3 Related Works
- Chapter 4 Proposed Methods
- Chapter 5 Implementation and Experiments
- Chapter 6 Experimentation
- Chapter 7 Conclusion

Chapter 2
Background Knowledge

In order create a article classifier, it is necessary to understand how the underlying process of Natural language processing work.

This chapter discusses about background knowledge. It introduces various concepts used in this project.

2.1 Natural Language Processing

NLP is a branch of computer science, information engineering, and artificial intelligence that concern how to convey the meaning and understanding between computer language and human language [22].

2.1.1 Morphology Analysis

Morphology analysis is the process that determining the morphemes, the smallest meaningful unit in a language, from which a given word is formed. It must be able to distribute between rules of conventions for writing a language (orthographic rules) and rules of the structure of words (morphological rules). For example, the word "apples" can be separated to "apple" which is the stem, and 's' which is a suffix indicating plurality [22].

2.1.2 Syntactic Analysis

Syntactic analysis is how to determine whether the sentence text string that input in human language. This process will making computer to understand relationships between word and word. The result of this process will contains an explanation of the sentence [22].

2.1.3 Semantic Analysis

Semantic analysis is the process that describes meaning representations are composed for linguistic expressions that related with syntactic structures, phrases, clauses, sentences and paragraphs, and the level of the writing to their language-independent meanings [22].

For example, "I *will read* this at the end of this year.", is simple future tense by using "will read" to determine what will happen in "the end of this year".

2.1.4 Discourse Analysis

Discourse analysis is the process that describes meaning beyond the sentence by using a part of morphology, syntactic, and semantics to finding the meaning of the paragraph or sentences [22].

For example, "I really suffer this work for very long time. Can you give me just one minute to let me take a breath?", the meaning of this sentences is "Can I take a break?". The first sentence encourage the second sentence to be more serious.

2.1.5 Pragmatic Analysis

Pragmatic Analysis is the process of linguistics and semiotics which studies the ways in which context contributes to meaning. Specifically, it's the portion that focuses on taking a structures set of text and figuring out what the actual meaning was [22].

2.2 Data Augmentation

Data Augmentation is a technique used in machine learning field. This method allows to generate the new set of data that combine the content of a base data with appearance of another ones. The newly data considered as dataset since they can be used to pre-train the given neural network to improve the training process accuracy.

2.3 Word Vectors

Word vectors are the vectors of multidimensional continuous floating point numbers that embedded the meaning of a word. Basically, a word vector is a row of real numbers where each point semantically similar words have similar vectors or very nearby from them. Since the word vector is a row of real numbers, the mathematical operations can be applied.

$$king - male + female = queen$$

In other word, the output of vectors of king subtract by vectors of male and plus vector will leave the approximate vector of queen.

The semantics of the word are embedded across the dimensions of the vector, each dimension represents a feature and the word's numerical weight which locate the closeness of its relation to that meaning.

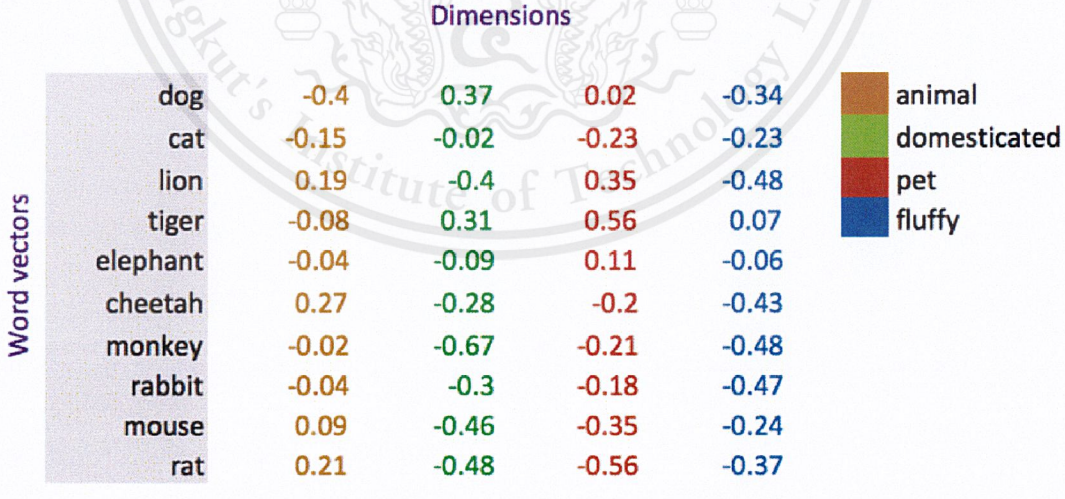

Figure 2.1: Word2Vec Example [16]

From the figure above, illustrate the concept of word vector dimensions. Each word's weight on each dimension represents how closely they relates to the dimension.

2.3.1 Word Embedding

Word embedding is a technique to represent each word as a fixed-size vectors. Word embedding is also a technique to decrease the size of vector space. The vector values depended on the pre-trained weight model such as Word2Vec, FastText and Thai2Vec.

Continuous Bag of Word

Bag of word is a model that commonly used in methods of text classification by count the occurrence of each word. Continuous Bag of Word (CBOW), it works under the principle that if the words that frequently occur nearby a given word, the word in the middle must be acknowledged. For instance, The input to the model could be

$$"The", "cat", 'over", "the', "television"$$

as context text and from these words, be able to predict or generate the center word "$jumped$".

Skip-gram Model

Skip-gram model works under the principle that predicting the context given a word which is a reverse version of continuous bag of word. For example, the input to the model is "$jumped$". The output of the neural network will be
"The", "cat", "$over$", "the", "$television$"

2.4 Convolutional Neural Networks

CNNs [11] are a category of neural networks that have been very famous in areas such as image classification and text classification. In NLP, CNN is made up of several layers. The first layer is convolutional. The second is pooling layer. And the last is fully-connected or dense layer [5]. A vector of word goes through this layer-by-layer and finally gives out a fully-connected layer where all the neurons are connected to each other and the output is processed. The detail of each layer is explained as follow:

2.4.1 Convolutional Layer

Convolutional layer's main task is to identify the features which are a distinctive attribute of the input sentence or article. Split the input to word converts it to vector, to extract features from bare texts which represented as a matrix. These vectors are word embeddings (low-dimensional representations). As you can see in Figure 2.2.

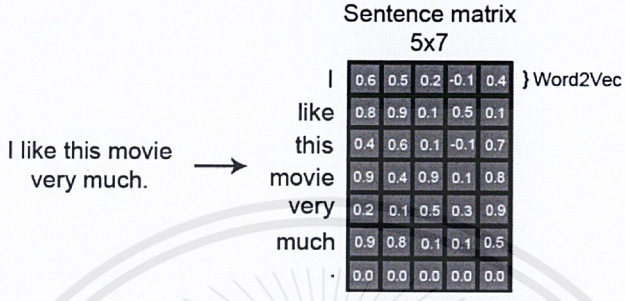

Figure 2.2: 5×7 Matrix [11].

The 5×7 black matrix is known as the convolution layer. Data in side matrix is a data that come from word embeddings process [2.3.1]. And need to be convoluted with a filter before going to the next layer. As you can see in Figure 2.3.

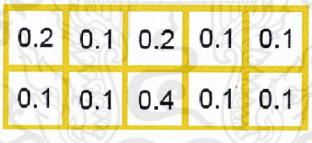

Figure 2.3: The Filter Matrix [11].

Figure 2.3 illustrates the filter that have to be computed with embedded matrix.

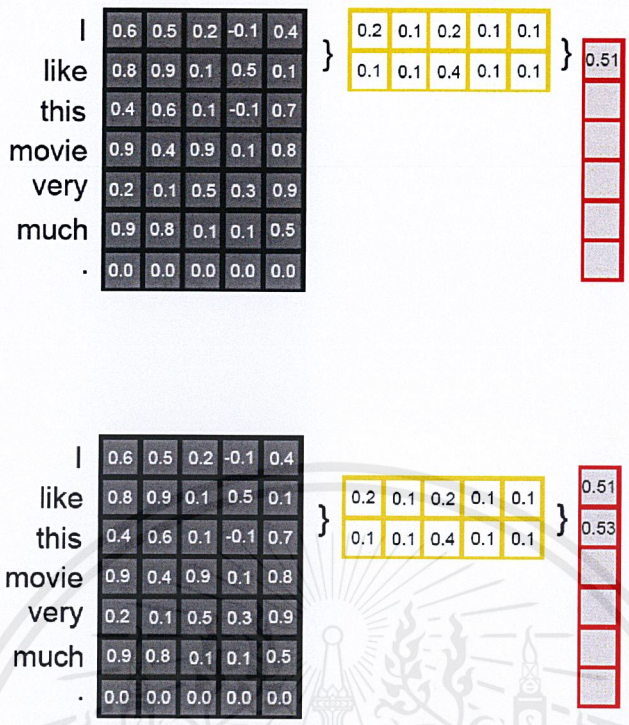

Figure 2.4: The Calculation in Convolutional Layer [11].

Figure 2.4 illustrates the action of the 2-word filter on the sentence matrix. First, the two-word filter, represented by the 2×5 yellow matrix, overlays across the word vectors of "I" and "like". Next, it performs an element-wise product for all its 2×5 elements, and then sum them up and obtain one number ($0.6 \times 0.2 + 0.5 \times 0.1 + \ldots + 0.1 \times 0.1 = 0.51$). 0.51 is recorded as the first element of the output sequence, for this filter. Then, the filter moves down 1 word and overlays across the word vectors of 'like' and 'this' and perform the same operation to get 0.53. Therefore, the output will have the shape of (s-h+1 $\times$ 1), in this case (7-2+1 $\times$ 1).

2.4.2 Normalization Layer (Rectified Linear Unit)

The normalization layer or the activation funtion (rectified linear unit) is done after the convolutional layer which replaces all the negative value inside vector to zero instead. The process adds nonlinearity to the network otherwise the network will have only the linear function. Therefore, it eliminates all unnecessary values to prevent mis-calculating. The ReLU's equation is illustrated in Figure 2.5.

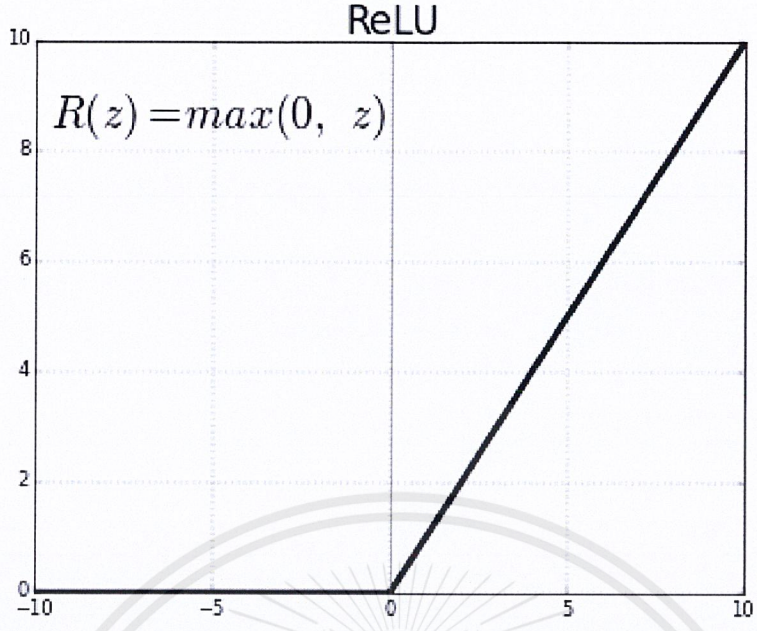

Figure 2.5: ReLU Equation, z is value can be 0 to infinity[12].

The main purpose of the normalization layer is delete dead (negative) nonlinear values because if nonlinear activation functions were not used, the network would be a large linear classifier, and could be simplified by simply multiplying the weight matrices together (accounting for bias). It would not be able to do anything

2.4.3 Pooling Layer

The pooling layer, which is also known as subsampling or downsampling, is applied to the model after the ReLU layer. There are several types of pooling such as max, average, or sum pooling. The process basically takes an input vector and reduces the vector size, however it still keep the important informations. In this case, the type is the 1-max pooling and the size is 1×1, illustrated in Figure 2.6.

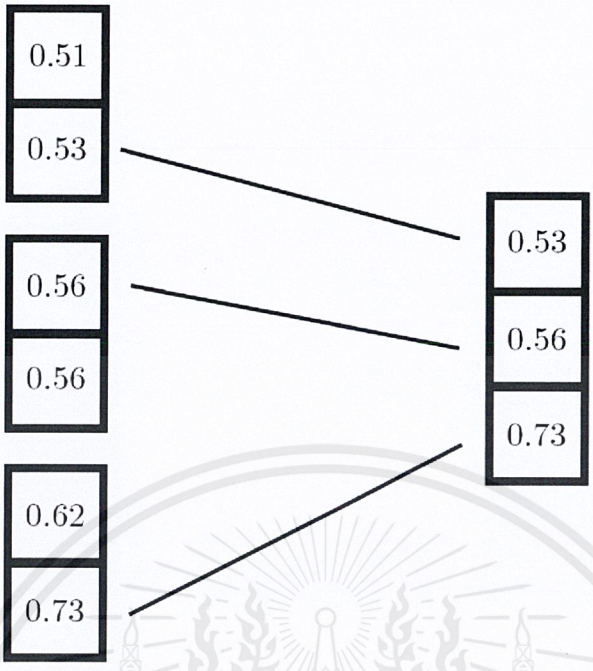

Figure 2.6: The 1-Max Pooling [13].

The target of the pooling layer is helping make CNN layers tolerate distortions. The pooling layer allows features to shift relative to each other resulting in robust matching of features even in the presence of small distortions. Reduces the spatial dimension of the feature map to make the model smaller. And reducing the number of parameters high up the processing hierarchy. This simplifies the overall model complexity.

2.5 Fully Connected Layer

The fully connected layer is one of the important concept of CNN model. Fully connected layers connect every neuron in one layer to every neuron in another layer. The flattened matrix goes through a fully connected layer to classify the input. Each neuron in a neural network computes an output value by applying a specific function to the input values coming from the previous layer. The function that is applied to the input values is determined by a vector of weights and a bias (typically real numbers). Learning, in a neural network, progresses by making iterative adjustments to these biases and weights. As figure 2.7, values at connected lines are weights and circles are neurons.

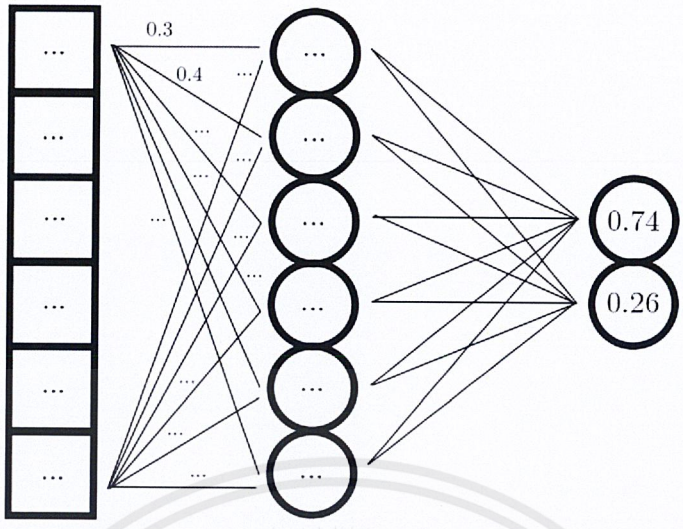

Figure 2.7: The Fully Connected Layer [11].

In the training process, this model normally using Backpropagation(recursively weights update) [23] to updates weights at fully connected layer(s) and values of filters at convolutional layer(s).

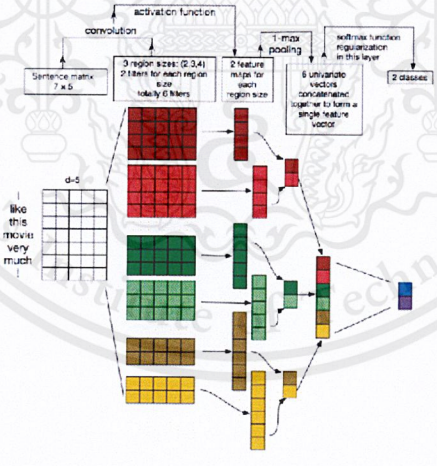

Figure 2.8: Convolutional Neural Networks Model for NLP [11].

Chapter 3
Related Works

This chapter surveys previous work in traditional recommendation without considering social trending. However, each recommendation has its own ways of filtering data.

3.1 Related Works

3.1.1 Extreme Learning Machine in Thai language

This paper proposes by our senior, they proposed three systems for each task in natural language processing. It consists of tokenization, part-of-speech tagging, and named-entity recognition. Similar to our work, the tokenization system is used in data preparation or pre-processing. For the next tasks, part-of-speech tagging, used to perform text structurize using neural networks with backpropagation and extreme learning machine (ELM) as learning algorithms. For named-entity recognition, both neural network and extreme learning machine used as learning algorithms. However, the result shows that neural networks with backpropagation provides a slightly higher accuracy compare to ELM. The resources provided from the 1999 edition of Royal Institute Dictionary as the dictionary which used in the study, while corpora are ORCHID and BEST I corpora from NECTEC. The average accuracy of all system is approximately 90 %.

3.1.2 E-Commerce Recommendation Applications

This paper aims to examine how the recommender system help E-commerce sites increase sales and analyze the recommender system at market-leading sites. They create a taxonomy of recommender systems, present the ways the recommendations are presented to consumers. However, this recommendation engine addresses and focuses only the e-commerce industry.[3]

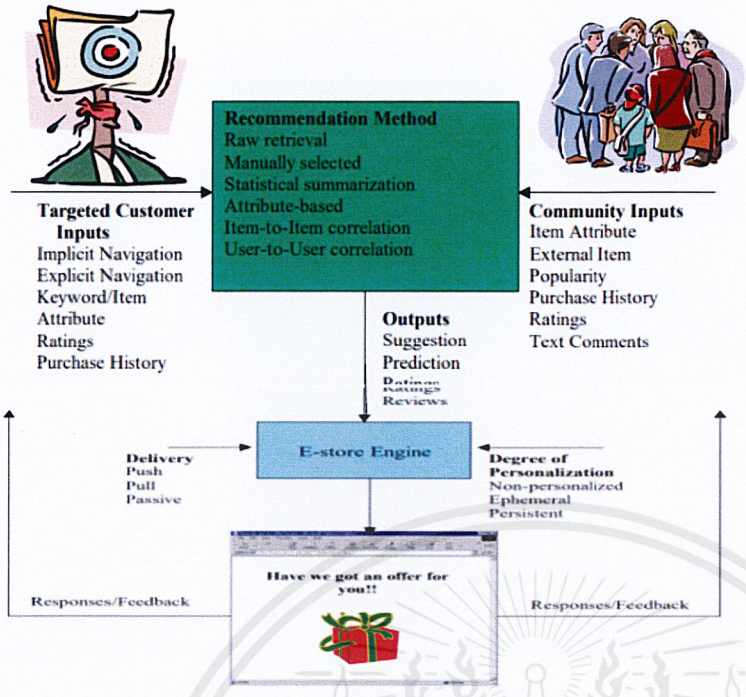

Figure 3.1: E-Commerce Recommendation Model

From Figure 2-1, Recommender applications combine inputs about the customer in question, with those about product and user communities to generate recommendation sites use decisions about personalization level and delivery method to transform these into specific recommendation package: Feedback to these recommendations may generate additional inputs for future recommendation

In order to output recommendations of appropriate item vary in type and quantity. this paper focuses on 2 parameters, targeted customer inputs, and community input. For targeted customer, inputs can be processed into a personalized recommendation. They categorize customers behavior input as implicit navigation input and explicit navigation input. Implicit navigation means inferred from the customer's behavior without the customer's awareness of their use of recommendation processes unlike explicit. Rather than asking customers feedback to provide explicit input, they use targeted customer's purchase history as implicit input. For community inputs, many item attributes such as movie genre and novel categories reflect the consensus of the broader society. They also utilize the community purchase history to provide them implicit community input.

3.1.3 Deep Learning-based Recommendation: Current Issues and Challenges

In this paper is interested in the deep learning based recommender systems which benefit from the deep model for enhancing the management of users, items, contexts, and user-items interconnections to guarantee user satisfaction. It has presented the background of systems as well as the deep learning architecture which is chained with the illustration of the deep learning-based recommender approaches. They have classified these approaches according to two criteria: to the integration way of the deep learning and the dependence level on it. Moreover, they have identified the new challenges and the future directions dealing with the deep learning based recommendation. Finally, they have handled a real case study that implements a deep learning based recommender system which is YouTube as it can be considered as a famous video recommender system driven by the Google Brain team. YouTube integrate the deep neural network to learn everything about viewers' habits and preferences.[4]

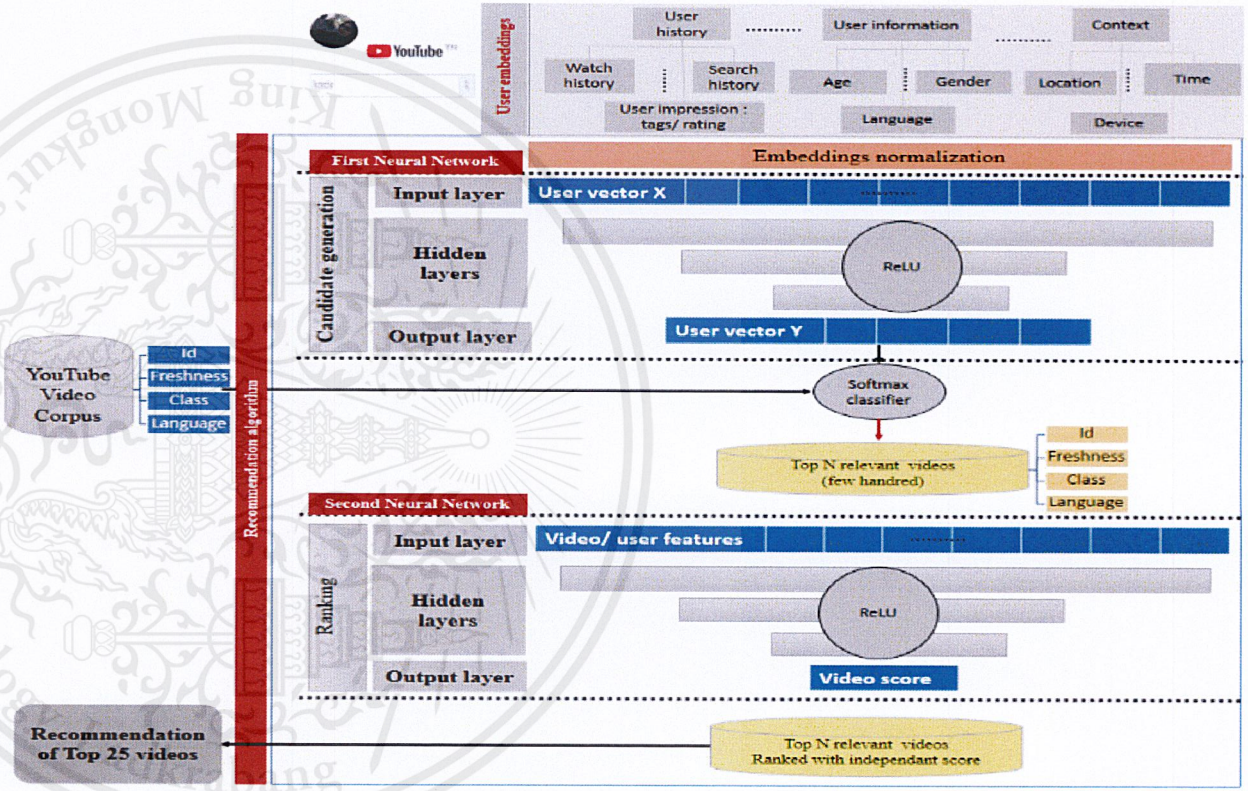

Figure 3.2: Feed forward vs.recurrent neural network architecture.

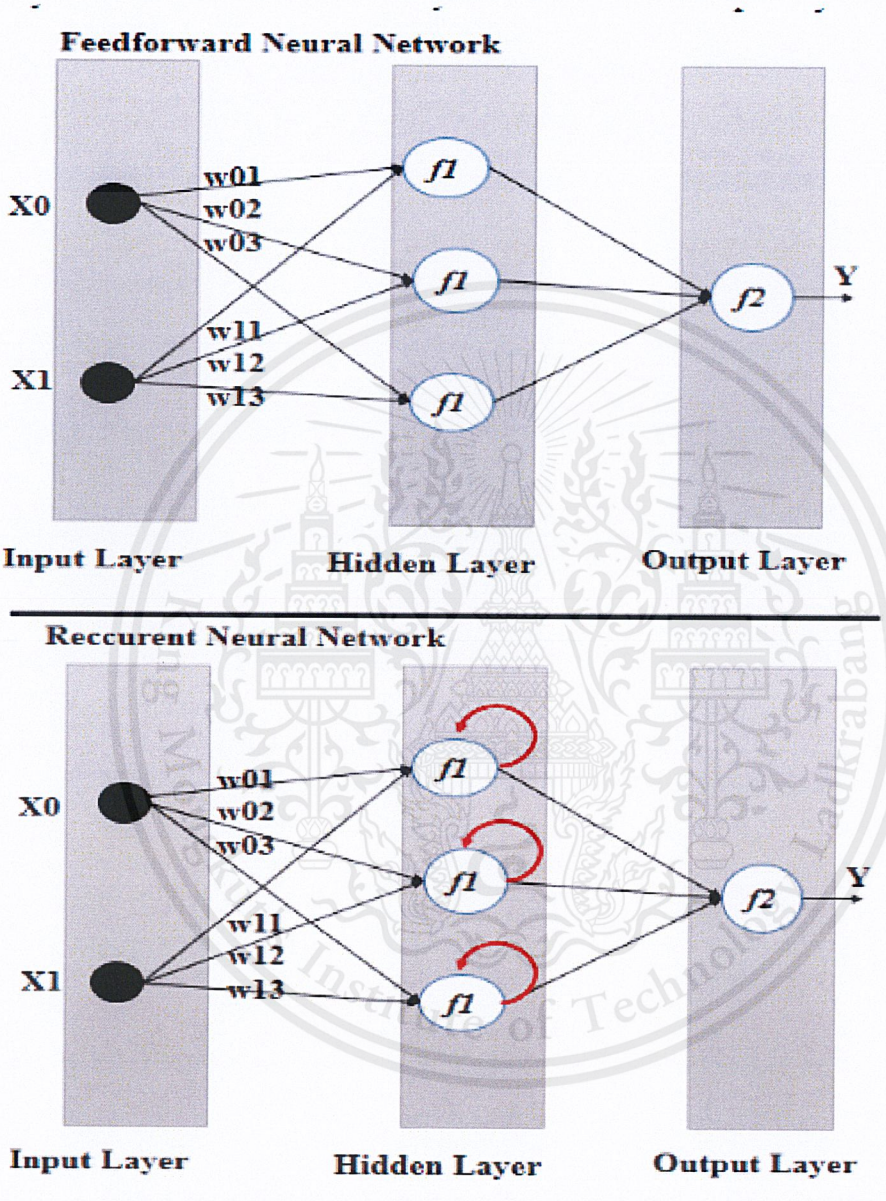

Figure 3.3: Feed forward vs.recurrent neural network architecture.

Chapter 4
Proposed Methods

This chapter describes our approach to constructing training models that capture the form and appearance of a variety of behaviors and trends. In real-time, automatic notify is monitoring and notifying the system that can show trends and notify when authors have behaviors or aptitudes that conform to that trends. Finally, we explain how the model is applied to behaviors and trends.

4.1 Our Approach

In our experiment, we approach to build the model to increase the accuracy of extract behaviors from articles and data from trends. At first, we will provide a basic way to extract data. First, we will tokenize [2] and clean a word because some word that was tokenized is not a use for extraction. Second, we will count and cluster a group of words. Finally, matching between groups from article and groups from an up-coming trend.

4.2 Preprocess Flow

The preprocess is the process that will clean, tokenize, filter and embed Word2Vec all words. The preprocess following these steps:

- Tokenizing: tokenize words from sentences by using tokenizer from PyThaiNLP [14].
- Cleaning: clean words that are other language or misspelling by using RegEX and library from PyThaiNLP.
- Filtering: filter all stop-words and common words(words that can not classify category).

- Counting words: count words to get the most 64 appeared words to use in machine learning process.
- Embedding Word2Vec: embed value from Pre-trained value from Word2Vec by Thai2Vec [15] in PyThaiNLP.

Figure 4.1 is the overview of preprocess flow.

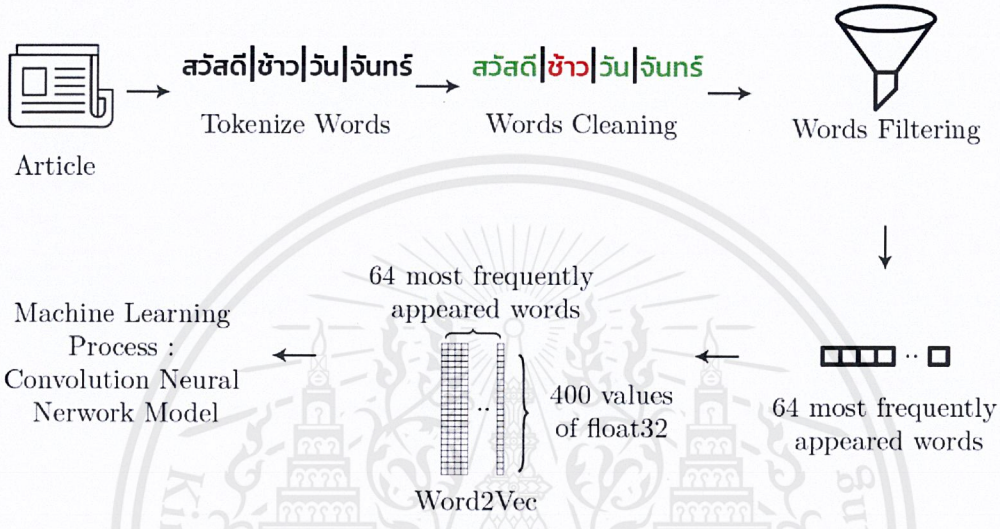

Figure 4.1: Preprocess Flow.

4.3 2D Convolution Neural Network (First Model)

This is the our first CNN model that use to train in machine learning process. Which contains many layers following these list :

1. Input layer: input as 2D using 64 most appeared words which have 400 values from Thai2Vec [15] for each. (64x400)
2. 1st Convolutional 2D layer: filter=32, strides=5x5
3. 1st Max pooling 2D layer: strides=2x2
4. 2nd Convolutional 2D layer: filter=64 strides=2x2
5. 2nd Max pooling 2D layer: strides=2x2
6. Flatten : size=64x16x100=102400
7. 1st Fully Connected 2D layer: units=256
8. Dropout layer: rate=0.5
9. 2nd Fully Connected 2D layer: units=10

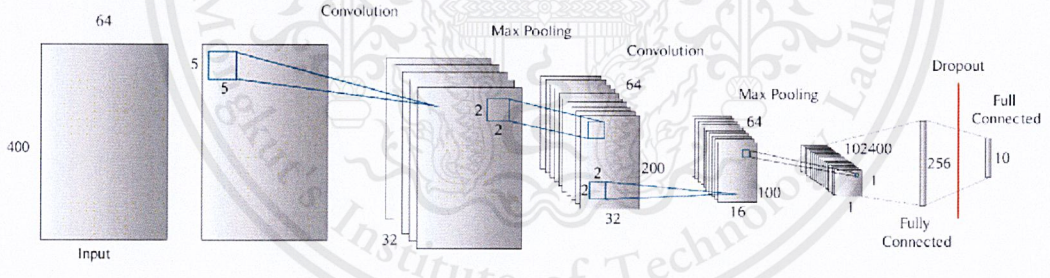

Figure 4.2: 2D Convolution Neural Network

4.4 1D Convolution Neural Network (Second Model)

1. Input layer: input as 1D using 64 most appeared words which have 400 values from Thai2Vec for each. [64(1x400)]
2. 1st Convolutional 1D layer: filter=200, kernel size=5
3. 1st Max pooling 1D layer: pool size=1, strides=2
4. 2nd Convolutional 1D layer: filter=100, kernel size=3
5. 2nd Max pooling 1D layer: pool size=1, strides=2
6. Flatten : size=100x16=1600
7. 1st Fully Connected 1D layer: units=128
8. Dropout layer: rate=0.5
9. 2nd Fully Connected 1D layer: units=10

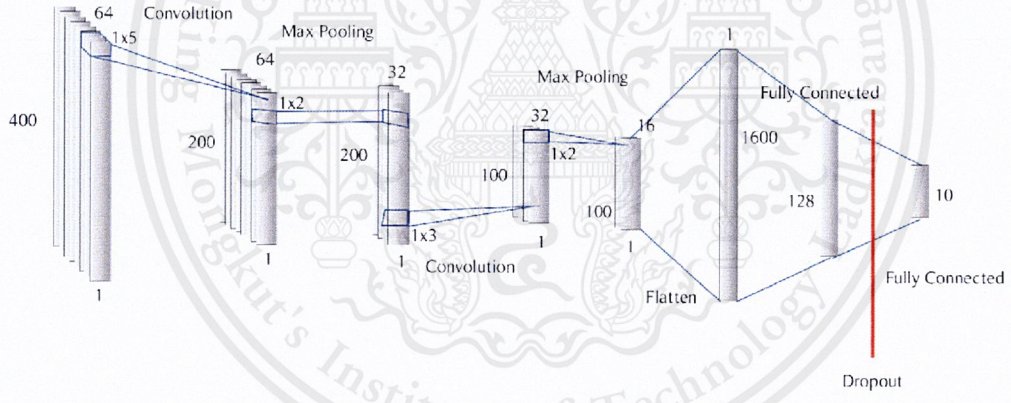

Figure 4.3: 1D Convolution Neural Network

4.5 1D Convolution Neural Network (Third Model)

1. Input layer: input as 1D using 64 most appeared words which have 400 values from Thai2Vec for each. [64(1x400)]
2. 1st Convolutional 1D layer: filter=200, kernel size=4
3. 1st Max pooling 1D layer: pool size=2, strides=2
4. 2nd Convolutional 1D layer: filter=100, kernel size=4
5. 2nd Max pooling 1D layer: pool size=2, strides=2
6. 3rd Convolutional 1D layer: filter=25, kernel size=4
7. 3rd Max pooling 1D layer: pool size=2, strides=2
8. Flatten : size=25x8=200
9. 1st Fully Connected 1D layer: units=200
10. Dropout layer: rate=0.5
11. 2nd Fully Connected 1D layer: units=2

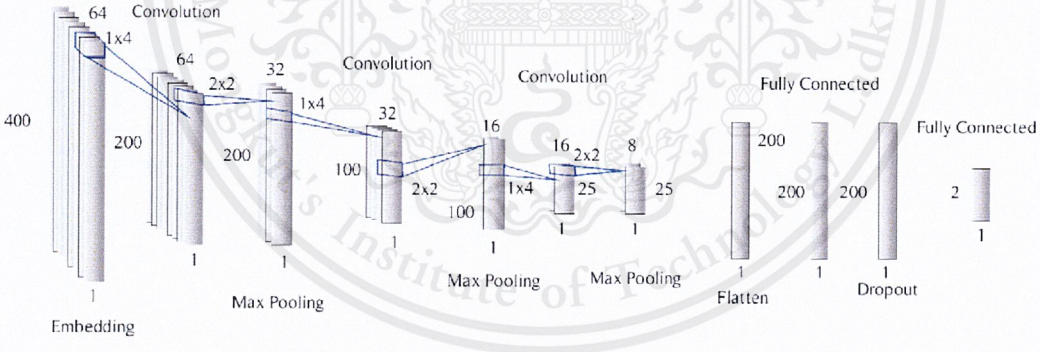

Figure 4.4: 1D Convolution Neural Network 2

4.6 1D Convolution Neural Network (Fourth Model)

1. Input layer: input as 1D using 32 most appeared words which have 400 values from Thai2Vec for each. [32(1x400)]
2. 1st Convolutional 1D layer: filter=200, kernel size=4
3. 1st Max pooling 1D layer: pool size=2, strides=2
4. 2nd Convolutional 1D layer: filter=100, kernel size=4
5. 2nd Max pooling 1D layer: pool size=2, strides=2
6. 3rd Convolutional 1D layer: filter=25, kernel size=4
7. 3rd Max pooling 1D layer: pool size=2, strides=2
8. Flatten : size=25x4=100
9. 1st Fully Connected 1D layer: units=100
10. Dropout layer: rate=0.5
11. 2nd Fully Connected 1D layer: units=2

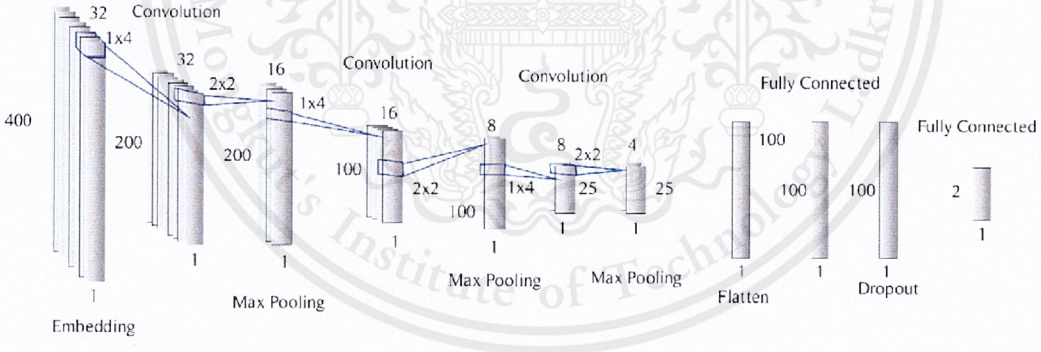

Figure 4.5: 1D Convolution Neural Network input vector size of 400 x 32

4.7 1D Convolution Neural Network (Fifth Model)

1. Input layer: input as 1D using 64 most appeared words which have 300 values from Thai2Vec for each. [64(1x300)]
2. 1st Convolutional 1D layer: filter=150, kernel size=4
3. 1st Max pooling 1D layer: pool size=2, strides=2
4. 2nd Convolutional 1D layer: filter=75, kernel size=4
5. 2nd Max pooling 1D layer: pool size=2, strides=2
6. 3rd Convolutional 1D layer: filter=25, kernel size=4
7. 3rd Max pooling 1D layer: pool size=2, strides=2
8. Flatten : size=25x8=200
9. 1st Fully Connected 1D layer: units=200
10. Dropout layer: rate=0.5
11. 2nd Fully Connected 1D layer: units=2

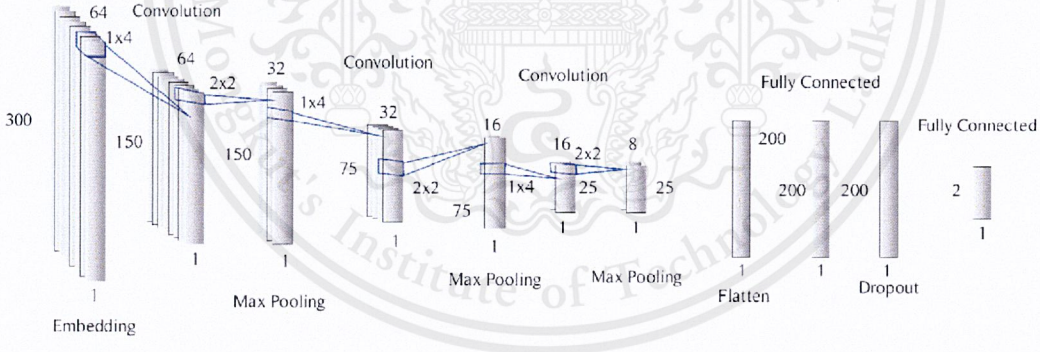

Figure 4.6: 1D Convolution Neural Network (Extracted from English text)

4.8 Neural Network Architecture

4.8.1 Layers

1D Convolution Layer

This layer constructs a convolution kernel that is coordinate with the layer input over a single dimension to produce corresponding outputs.

Dropout Layer

Dropout is a commonly used regularization technique for neural networks where during each iteration of gradient descent, set of neurons dropped randomly depended on the given probability.

Regularization

Regularization is the process of adding information in order to prevent overfitting in the field of data science.

4.8.2 Activation Function

Activation function decides which neurons should be eliminated or activated. In order to introduce non-linearity into the output of a neuron, activation function is required.

ReLu

The Rectified Linear Unit is non-linear classifier which activation function defined as

$$A(x) = max(0, x)$$

where x is the input to a neuron. Also know as ramp function and analogous to half-wave rectification.

Relu is the common activation function for CNN model, normally use in computer vision and sound process. Relu is effective in accuracy with many models and make machine learning process faster than many other activation function that is our reason to use Relu as activation function in our model.

4.9 Classification

The classification applied by deep learning, the model based on word vector, in order to construct a classifier to classify contexts.

4.9.1 Model Training

The neural network is fed with two individuals types of feature each with similar parameters to ensure the highest possible accuracy. The format of data is a substance of articles in the form of word vector with a size of 1 x 400. Those data are used to generate a relating classification model.

4.9.2 Predictive Model

["การเมือง","การศึกษา","กีฬา","ดนตรี","พืช","ภาพยนตร์","ภาษา","ศาสนา","สัตว์","อาหาร","สถานที่"]

Figure 4.7: Categories

Each input will be classified into their own categories consist of 10 difference classes. Each model is made up of number of weights(value to decides the result of the prediction), works by analyzing historical and current data and generating a model to predict output in the future.

Chapter 5
Implementation and Development

This section describes the software specifications and data acquisition for machine learning of this thesis. In order to create a classier in deep learning, the model required a number of datasets, the dataset must be tagged or categorized.

5.1 Dataset

The resource of the dataset is articles. The acquisition process provided by Wikipedia. They are not able to be categorized and vectorized by computer logic since the propagation of them are not from the machines but acknowledged phrase or vocabulary from humans. Each article required an amount of post or data to ensure that the machine acknowledged them as much as possible. The particular part of the dataset is converted from RSS feed as XML from Google.

5.1.1 Property

Since the experiment must be conducted on both Thai and English, each dataset required for each language. Wikipedia provided sub-categories for each article, in order to categories them, the proposed classes must cover the sub-category. Python script made to performed this job.

Thai Dataset

Amount of data: 30,000 articles **Amount of words:** Average 100 words per article **Amount of classes:** 10

English Dataset

Amount of data: 3,600 articles. **Amount of words:** Average 150 words per article **Amount of classes:** 9

CPU	Intel Core i5-9600K
RAM	16GB DDR4
GPU	NVIDIA GeForce GTX 970 4GB
SDD	180 GB
Window Kernel	Hybrid (Windows NT)

Table 5.1: Computer Specification

Python	3.6.0	Pythainlp	1.8
Django	2.1.2	Thai2Vec	0.2
Requests	2.20.1	Keras	2.2.4

Table 5.2: Versions of the software libraries

5.2 Setup

The experiments will be conducted on a custom built desktop computer running Window for the specification can be found in table 5.1 and the versions of the software libraries used in table 5.2.

5.3 Preprocessing

The input to our model is text from articles. Before text can be used as input to the model it must first be preprocessed to enhance the information appropriate for the feature that the model will classify.

5.3.1 The Data Cleaning

The data cleaning is the one important part of preprocessing process. This process can enhances the feature extraction result and improves speed of learning process. Our model cleaning the data separately from articles and social medias because the format and level language. The data from social media will have only text but the data from articles can have some media example images. And this is our order to clean the data before next step.

Text From Articles

Step 1 : Delete medias.

Step 2 : Delete symbols and escape characters.

Step 3 : Delete another language text using regular expression.

Step 4 : Delete words that appear in every category.

5.3.2 Text Tokenizing

After the cleaning process, our next process is Tokenizing text to use in Natural Language Process. The tokenization is must depends on morphology, syntatic, and semantic. It will be a hard part to tokenize it but there are some library to tokenize and all around the NLP in Thai language called PyThaiNLP [14] library. After the tokenize process, Figure 5.1 is the result of this process.

"ฉันรักภาษาไทย เพราะภาษาไทยเป็นภาษาแม่ของฉัน" ⟶ ["ฉัน", "รัก", "ภาษาไทย", ... ,"ของ", "ฉัน"]

Figure 5.1: Tokenization for Thai language [14].

5.3.3 Stop Words Filtering

After receive words from article, unclean words need to be filtered stop words such as

เป็น คือ และ หรือ

Figure 5.2: Stop words

Our way to do it is collect all words from every category, pack it to one file and filter all intersect words from every category. But maybe useful words can intersect in every category, frequency was used in this process to select what word should filter. Assume stop words are high frequency and useful words are low frequency. The process can set limit of frequency. If the frequency of word is more than the limit. The word will be filtered and vice versa.

Algorithm 1 Intersection (words that appear in every category)

```
1:  procedure Intersection(categories,category words,limit)
2:      temp words ← An empty array.
3:      intersected words ← An empty array for intersected words.
4:      for each category in categories do
5:          for each word in category do
6:              if word count > limit then
7:                  Append word into temp words
8:          if length of intersected words = 0 then
9:              intersected words ← temp words
10:         else intersected words ← temp words intersect with intersected words
11:     return intersected words
```

Algorithm 2 Stop Words Filtering

```
1:  procedure Filtering( words,intersected words)
2:      for each word in words do
3:          if word is in intersected words then
4:              remove word in words
5:      return words
```

5.4 Feature Extract Process

5.4.1 Word2Vec in Thai Language

Before input the text to the deep learning model, text must be inside matrix or vector. Our model is using Word2Vec to be a feature extractor and a word embedder to embeds text to vectors. And inside the PyThaiNLP already has this function called Thai2Vec [15] that has to use the same tokenizer(PyThaiNLP) to use it. Figure 5.3 is the example of Thai2Vec result.

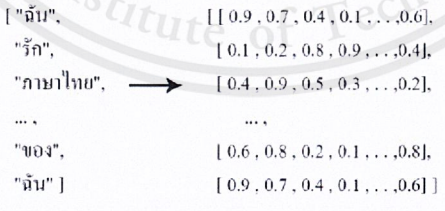

Figure 5.3: Word2Vec For Thai Language [15].

5.4.2 Word2Vec in English Language

As mentioned above, text must be inside matrix or vector to be as data featured. In English model that implement for comparing with the Thai model is using Word2Vec

to be a feature extractor and a word embedder to embeds text to vectors. The Google's pre-trained Word2Vec model [19] in Gensim is a pre-trained data that can download for free and contain Word2Vec that needed.

5.5 Data Augmentation For Text Data

The main problem of deep learning is require large data for training model. Thai language data is much less than English language. Data augmentation come to use when data is not enough, this can generate data for train.

Algorithm 3 Data Augmentation

```
1: procedure Data Augmentation(filtered words,number of each duplicates words)
2:     data augmented words ← An empty array for data augmentation words.
3:     for each word in filtered words do
4:         magic number ← sum of filtered words count time random number .
5:         Append word,magic number into data augmented words
6:     return data augmented words
```

```
l = sorted((random.random() * (x[1]/len_allcounts), x[0]) for x in l)
```

5.6 Natural Language Toolkit

Natural Language Toolkit (NLTK) is a Python library and program to work with human language data. It provides many resources and function lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries[20]. It is one of the widest libraries of human language data in Python. It proving easy interface and easy to understand how to use. Many of function is useful and high accuracy and that is an important tool to use in some of our parts.

5.7 T-Distributed Stochastic Neighbor Embedding

T-Distributed Stochastic Neighbor Embedding (t-SNE) is a visualization technique that is specifically for the of the high dimensional dataset. Our project using t-SNE to visualize the dataset in order to identify errors on our dataset with our model.

Chapter 6
Experimentation

This section describes the software specifications and data acquisition for machine learning of this thesis, which is then followed by a section containing the expected experimental results.

6.1 Experiments and Results

6.1.1 Experiment 1

Objective

At very early of this project, our approach is using a simple CNN model in Thai language. The target of the first model is proving and finding the important parameter to predict classify 10 classes. Parameters are Filtered Stop Words, Data Augmented, and Amount of Data.

Experiment Setup

The experiment is set up as follow:

Dataset: This experimentation uses datasets from Wikipedia.

Parameters: There are 3 parameters. First, stop words filtering to find stop words are affected with accuracy. Second, Data Augmentation. The last parameter is the Amount of Data.

Result

After several experiments, it could be concluded as the followings:

Table 6.1: The Experiment Result Table of Early of the Project.

	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**
CNN Model	2D CNN	1D CNN	1D CNN	1D CNN
Language	Thai	Thai	Thai	Thai
Model	Model 1	Model 2	Model 2	Model 2
Filtered Stop Words	Yes	Yes	Yes	Yes
Data Augmented	No	No	Yes	Yes
Amount of Data (Article)	3000	3000	20000	100000
Best Accuracy (Percent)	10%	10%	10%	10%

As the result table, the accuracy from the model that filtered stop words predict only 1 class, maybe because of the model is not fit with classification yet. That makes this experiment fails but that made us know what is the point of failure in the model, the dataset is the failure point in this experiment because after the experiment.

6.1.2 Experiment 2

Objective

After the first experiment, the target of the second model at section 4.4 is proving that the dataset is not clean enough by using the t-SNE to visualize the dataset.

Experiment Setup

The experiment is set up as follow:

Dataset: This experimentation uses datasets from the first experiment.

Parameters: There is only one parameters, the augmentation process.

Result

After several visualizes, it could be concluded as the followings:

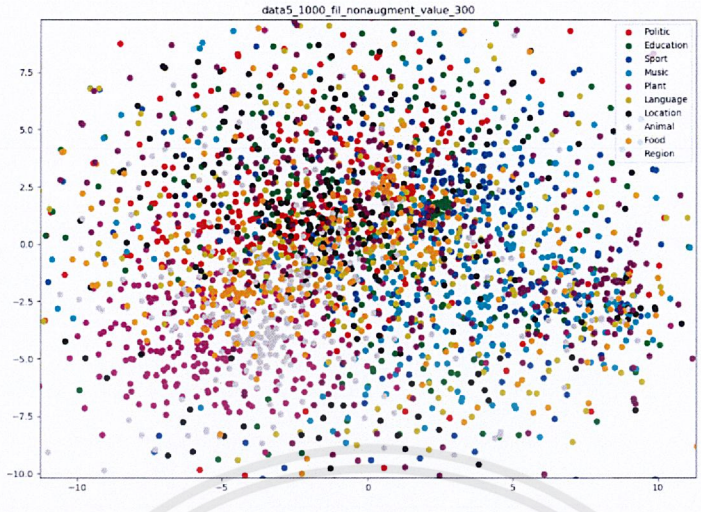

Figure 6.1: The dataset filtered but not augmented process by t-SNE

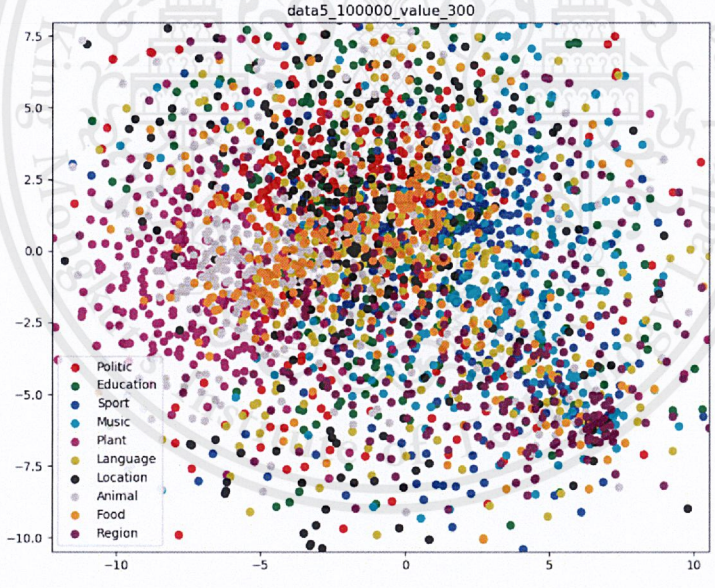

Figure 6.2: The dataset with filtered and augmented process by t-SNE

From results that can be assumed the dataset is clean enough. The next plan is to implement cleaning processes for cleaning the dataset from Wikipedia and reduce the result of the model from 10 classes to 2 classes.

6.1.3 Experiment 3

Objective

In this experiment, targets are experiment new CNN models at section 4.5, 4.6, 4.7, improve the pre-processes, and using the English dataset from Wikipedia to observe the result and parameters. For English dataset, it will prove that our model did not wrong or error, an open source machine learning platform. In the tokenize and Word2Vec process, we use Google pre-trained model with Gensim to do all process.

Experiment Setup

The experiment is set up as follow:

Dataset: This experimentation uses 2 datasets from the first experiment and the English dataset.

Parameters: There 3 parameters, The English dataset is the first parameters. Second, cleaning processes. The last one, the new CNN model that use to train.

Result

After several experiments, it could be concluded as the followings:

Table 6.2: Third Experimentation Result Table.

	Subject 5	**Subject 6**	**Subject 7**	**Subject 8**
CNN Model	1D CNN	1D CNN	1D CNN	1D CNN
Language	Thai	Thai	English	English
Model	Model 3	Model 4	Model 5	Model 5
Amount of class	2	2	2	9
Filtered Stop Words	Yes	Yes	Yes	Yes
Data Augmented	Yes	Yes	Yes	Yes
Cleaning Process	New	New	NLTK[20]	NLTK
Amount of Data (Article)	2000	2000	1000	4500
Epoch	1000	1000	1000	10000
Best Accuracy (Percent)	88.5%	91.5%	96%	89.6%
Average Accuracy (Percent)	63.7%	71.5%	77.4%	87.4%
Average Loss	0.6352	0.5717	0.6452	0.5690

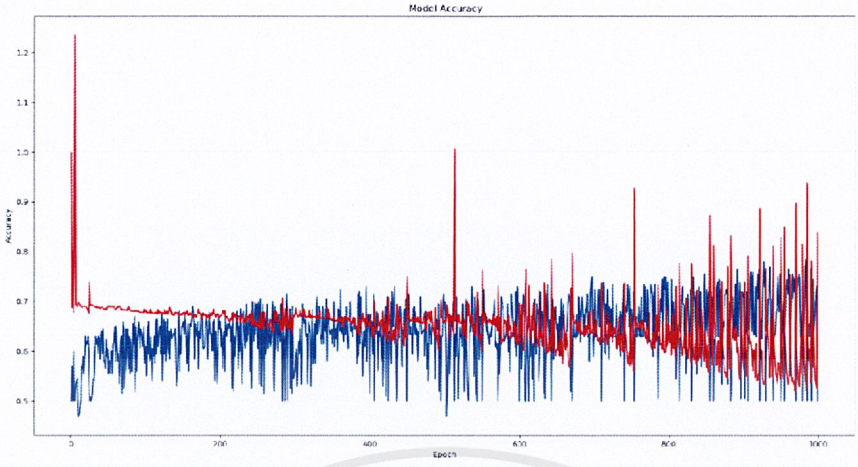

Figure 6.3: The result of the subject 5 between Accuracy(Blue) and Loss(Red)

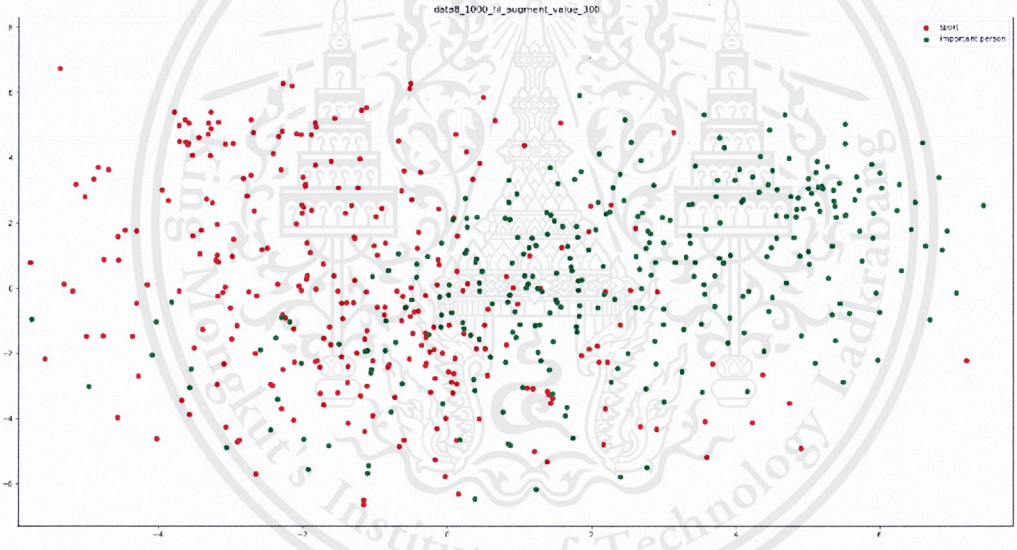

Figure 6.4: The subject 5 visualized image

The result seem unstable and the visualized image show us that it can predict but there are many of the data are in the other pattern in each other that make model can not predict and train efficiency.

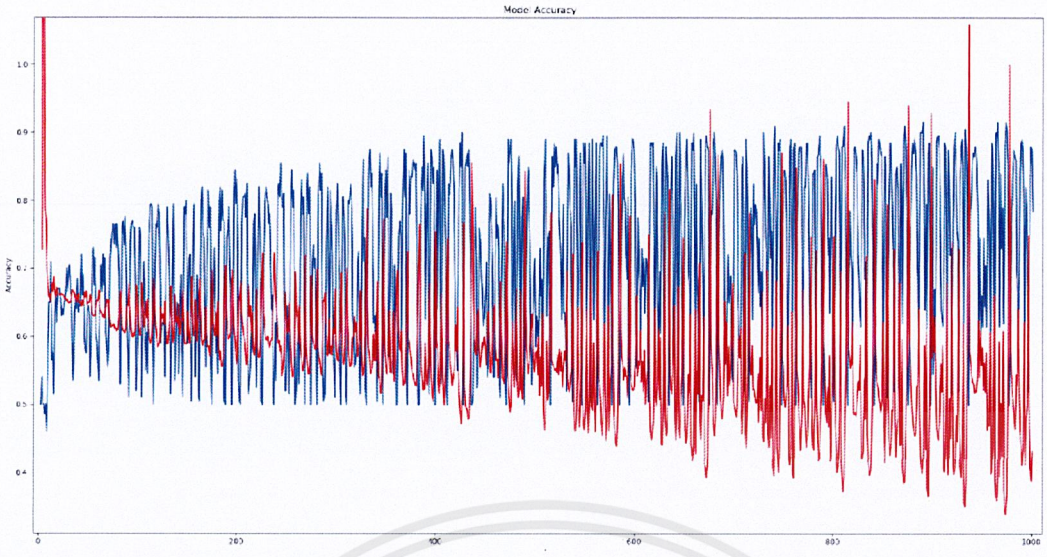

Figure 6.5: The result of the subject 6 between Accuracy(Blue) and Loss(Red)

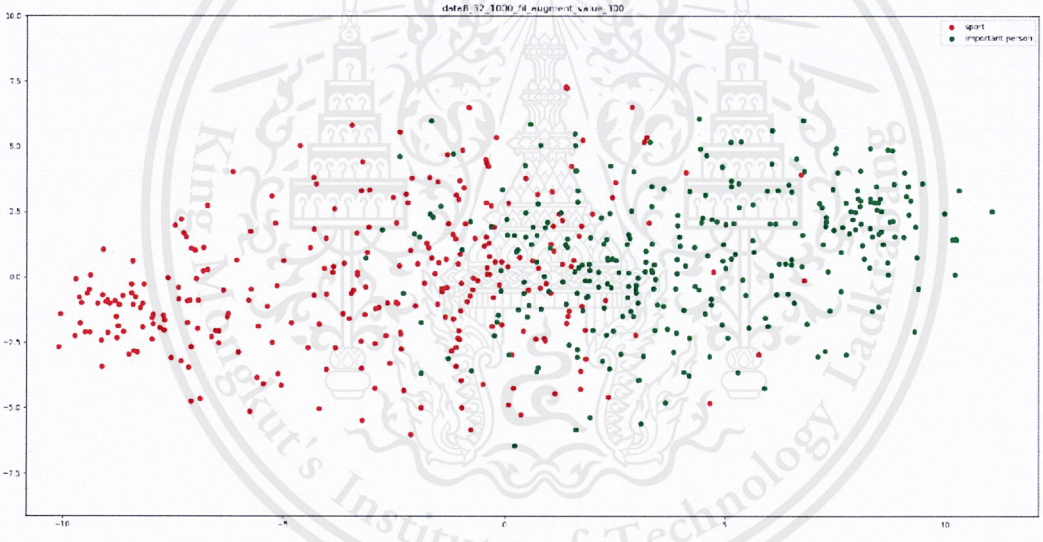

Figure 6.6: The subject 6 visualized image

The result of 32 most appeared most model look like the same as the previous model, unstable and the visualized image show us that it can predict but there are not too many data cross to other class data, seem like they can pattern but in the middle of the image show us, it is still not patternable and hard to predict and train with this model.

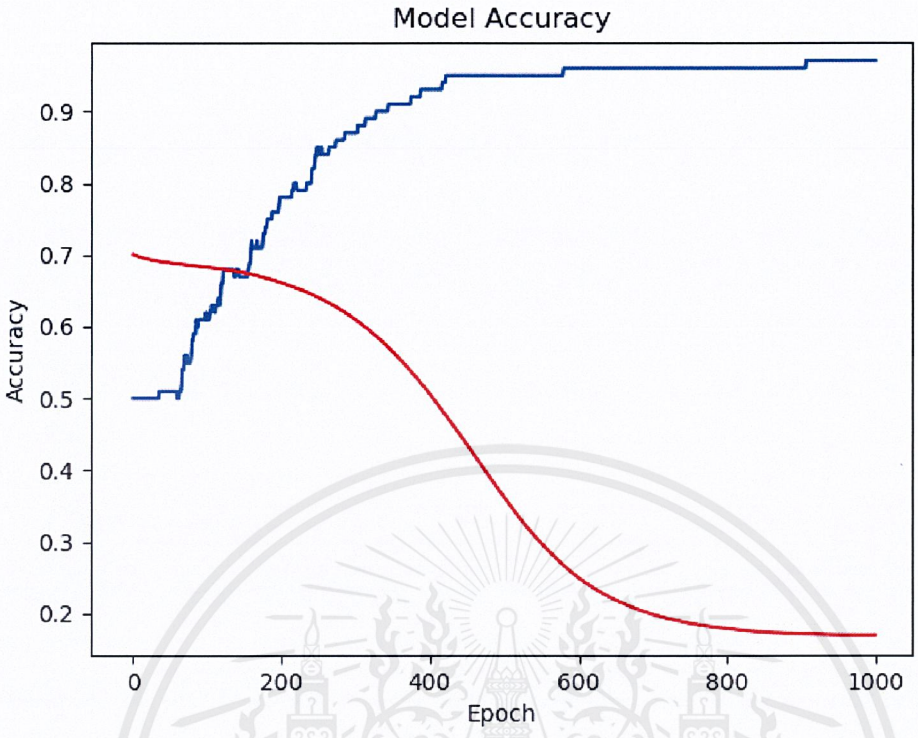

Figure 6.7: The result of the subject 7 between Accuracy(Blue) and Loss(Red)

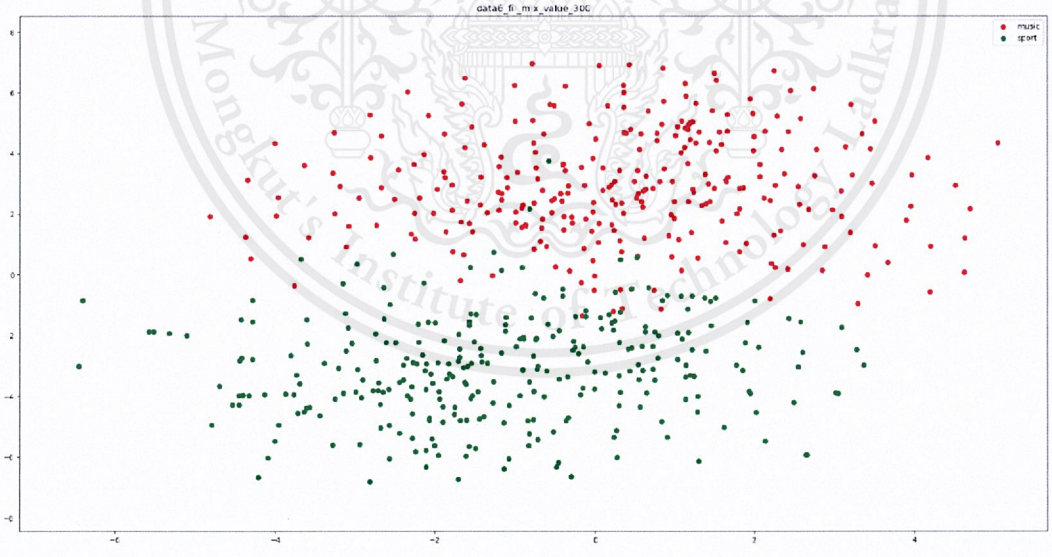

Figure 6.8: The subject 7 visualized image

In English model, the training graph and visualized image are better than all Thai model. Stable and predictable, in the visualized graph, it look like there is a line cross in

the middle of the image. It show us that this model is patternable. Maybe it is because Thai language training dataset are unclean enough.

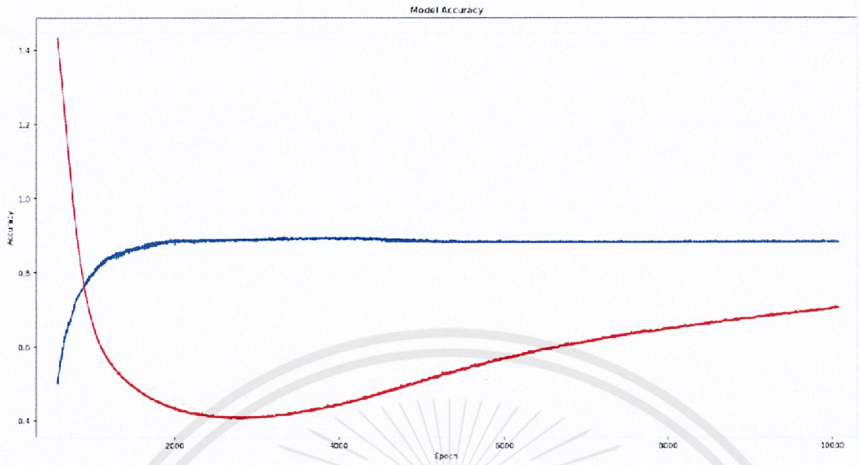

Figure 6.9: The result of the subject 8 between Accuracy(Blue) and Loss(Red)

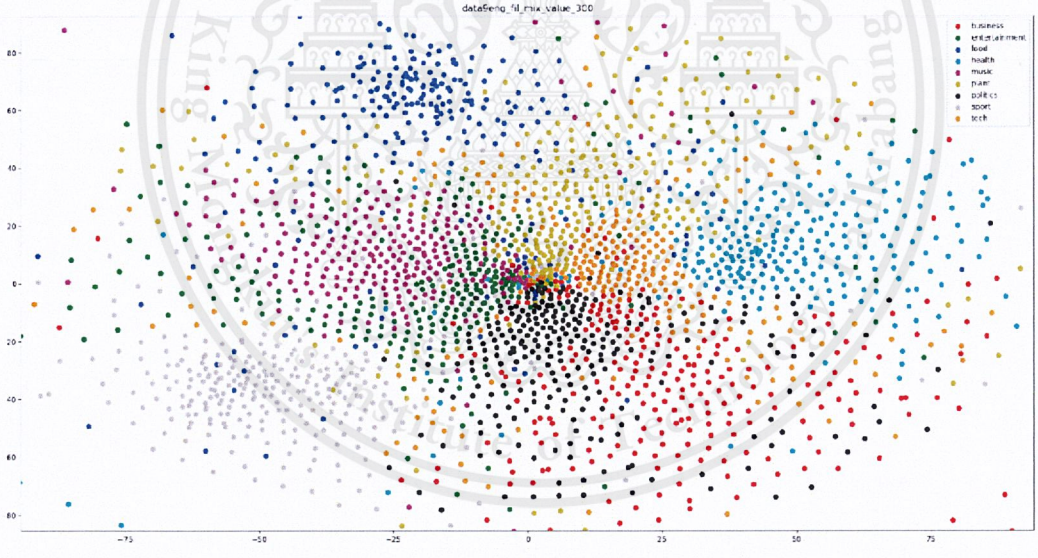

Figure 6.10: The subject 8 visualized image

Similar with the previous model, in the visualized graph, data that pointed in image seem patternable and were clustered in each class. Only a few class is hard to pattern but it is better than all Thai language model. It emphasize us, it is because Thai language training dataset are unclean enough.

6.2 Conclusion

The shown visualized images show that characteristics were able to be captured when English texts extracted from English word embedder, while the others were still unstable.

In the first and second experiment show us how non-classifiable data in visualize figure. Every data that we use in any categories were pointed in middle of the graph and non-pattern. At first, we thought it is because the data is cleaning process at preprocessing process but it is not.

After that, we target the next experiment to see what if the training dataset is English language, can the model predict with efficiency. And reduce categories from 10 to 2, to make sure that the training process did not wrong. The result show us that English model is more efficient and stable than Thai model.

6.3 Application

The web application built in order to display the result of this experiment. The application consists of 2 main system article classification and trending analyses. The application provides file uploader and text editor for the user to input the article, the input will be classified into the given classes using proposed models. For trending analyzed, the system provided the analyzed trending data, the resource of trends received from Google Trend's RSS feed. They parsed and saved into the system and classified into the given classes using proposed models.

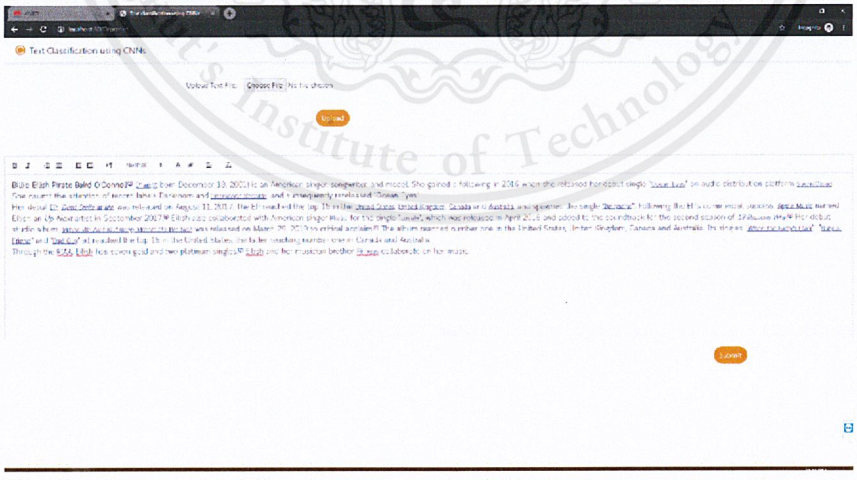

Figure 6.11: Web Application Screenshot 1

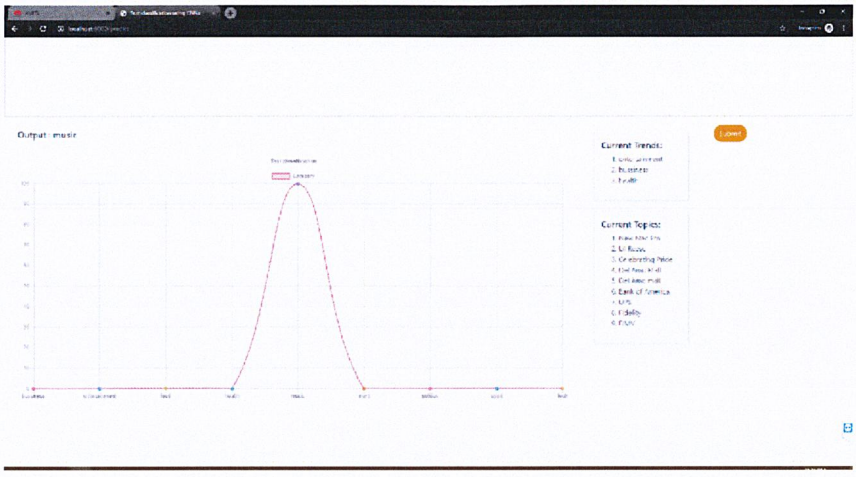

Figure 6.12: Web Application Screenshot 2

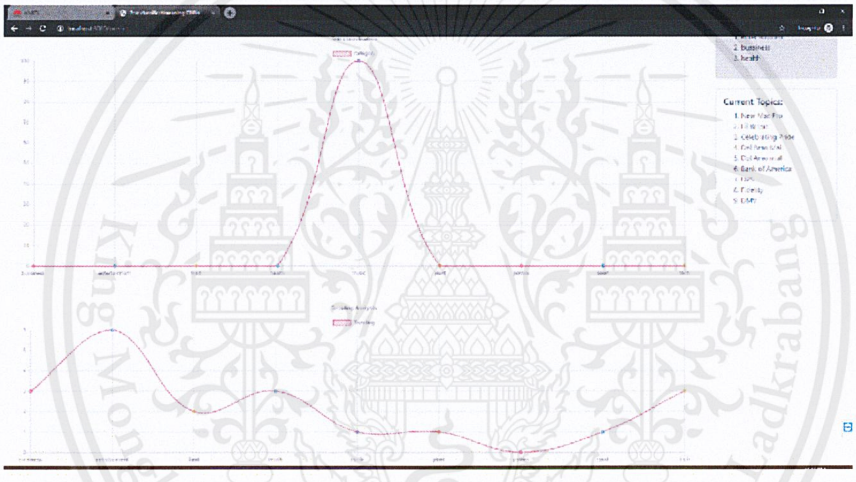

Figure 6.13: Web Application Screenshot 3

Chapter 7
Conclusion

This thesis proposes a text classification system that focuses on Thai processing. Furthermore, this thesis mainly targets on the classification model which consist of 4 convolutional neural networks(CNN) before connected in fully-connected layer. Each CNN classifies a single feature of the article's content. There is only one feature which is an article's content which is string data-type, however, the system performs a word embedding which extracts feature of article's content into a vector with various sizes. The different parameters also affect performance measurement. Parameters are including stop word filtering, data augmentation, amount of dataset and language. These parameters were tested and compared to find the one that produced the highest accuracy possible. Those results are recorded in Chapter 6.

7.1 Future work

To be concluded, this thesis archived indicated objectives. We successfully create a text classifier and used in real-world application using deep learning. However, the accuracy of Thai processing has dramatically fluctuated and the loss value as well. In order to produce the appropriate model, the amount of appropriate dataset is required. In the future, the dataset must be provided and categorized by a cooperated company or organization.

Bibliography

[1] O. Gabry "Requirements Engineering Introduction", 2016. [Online]. Available: https://
medium.com/omarelgabrys-blog/requirements-engineering-introduction-part-1-6d49001526d3 [Accessed: 08-March-2018].

[2] Cambridge University Press "Tokenization", 2009. [Online]. Available: https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html [Accessed: 08-October-2018].

[3] Ben Schafer, Joseph A. Konstan, John Riedl "E-Commerce Recommendation Applications", 2000. [Online]. Available: https://www.researchgate.net/profile/Ben_Schafer/publication/2620988_E-Commerce_Recommendation_Applications/links/02e7e5356860045eaf000000/E-Commerce-Recommendation-Applications.pdf?origin=publication_detail [Accessed: 08-October-2018].

[4] Rim Fakhfakh, Anis Ben Ammar, Chokri Ben Amar "Deep Learning-based Recommendation: Current Issues and Challenges", 2017. [Online]. Available: http://thesai.org/Downloads/Volume8No12/Paper_9-Deep_Learning_based_Recommendation.pdf [Accessed: 08-October-2018].

[5] Denny Britz, "Understanding Convolutional Neural Networks for NLP", 2015. [Online]. Avzailable: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/ [Accessed: 11-November-2018].

[6] "Twitter", 2017. [Online]. Available: https://en.wikipedia.org/wiki/Twitter [Accessed: 11-November-2018].

[7] Maddy Osman "How to Use Hashtags on Every Social Media Network", 2017. [Online]. Available: https://sproutsocial.com/insights/how-to-use-hashtags/ [Accessed: 11-November-2018].

[8] The Editors of Encyclopaedia Britannica "Wikipedia", 2017. [Online]. Available: https://www.britannica.com/topic/Wikipedia [Accessed: 11-November-2018].

[9] Quang Nguyen, "Detecting Experience Revealing Sentences in Product Reviews", 2012. [Online]. Available: https://esc.fnwi.uva.nl/thesis/centraal/files/f1006996408.pdf [Accessed: 11-November-2018].

[10] The Anh Le, "An Exploration of Word2Vec Algorithm", 2016. [Online]. Available: https://digital.library.unt.edu/ark/67531/metadc849728/m2/1/high_res_d/LE-THESIS-2016.pdf [Accessed: 11-November-2018].

[11] Joshua Kim, "Understanding how Convolutional Neural Network (CNN) perform text classification with word embeddings", 2007. [Online]. Available: http://www.joshuakim.io/understanding-how-convolutional-neural-network-cnn-perform-text-classification-with-word-embeddings/ [Accessed: 08-October-2018].

[12] "Neural networks". [Online]. Available: https://ml4a.github.io/ml4a/neural_networks/ [Accessed: 11-November-2018].

[13] A. Karpathy, "Convolutional Neural Networks for Visual Recognition" [Online]. Available: http://cs231n.github.io/convolutional-networks/ [Accessed: 11-November-2018].

[14] Wannaphong Phatthiyaphaibun, "PyThaiNLP" [Online]. Available: https://github.com/PyThaiNLP/pythainlp [Accessed: 11-November-2018].

[15] Charin Polpanumas, "thai2vec" [Online]. Available: https://github.com/cstorm125/thai2vec [Accessed: 11-November-2018].

[16] Alex Minnaar, "The Continuous Bag-of-Words Model", 2015 [Online]. Available: http://mccormickml.com/assets/word2vec/Alex_Minnaar_Word2Vec_Tutorial_Part_II_The_Continuous_Bag-of-Words_Model.pdf [Accessed: 11-November-2018].

[17] Jayesh Bapu Ahire, "Introduction to Word Vectors", 2018 [Online]. Available: https://medium.com/@jayeshbahire/introduction-to-word-vectors-ea1d4e4b84bf [Accessed: 18-March-2019].

[18] Laurens van der Maaten, "t-SNE", 2019 [Online]. Available: https://lvdmaaten.github.io/tsne/ [Accessed: 20-March-2019]

[19] Chris McCormick, "Google's trained Word2Vec model in Python", 2016 [Online]. Available: http://mccormickml.com/2016/04/12/googles-pretrained-word2vec-model-in-python/ [Accessed: 31-March-2019].

[20] "Natural Language Toolkit", 2019 [Online]. Available: https://www.nltk.org/ [Accessed: 31-March-2019]

[21] Madeleine Bates, "Models of natural language understanding", 1995 [Online]. Available: https://www.pnas.org/content/pnas/92/22/9977.full.pdf [Accessed: 25-June-2019]

[22] Francois Chaubard, Rohit, Mundra, Richard Socher , "Deep Learning for NLP", 2016 [Online]. Available: https://cs224d.stanford.edu/lecture_notes/notes1.pdf [Accessed: 27-June-2019]

[23] Saurabh, "Backpropagation – Algorithm For Training A Neural Network", 2019 [Online]. Available: https://www.edureka.co/blog/backpropagation/ [Accessed: 27-June-2019]