MINING SOCIAL MEDIA CROWD TRENDS FROM THAI TEXT POSTS AND COMMENTS

Bundit Thanasopon[1], Jirawin Buranapanitkij[2], Ponrudee Netisopakul[3]

[1,2,3]Information Technology Faculty
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand

Email: bundit@it.kmitl.ac.th[1], jirawin9@hotmail.com[2], ponrudee@it.kmitl.ac.th[3]

DOI: https://doi.org/10.22452/mjcs.sp2019no2.6

ABSTRACT

Text mining from social media stream has attracted wide interests from both businesses and academics. Very large numbers of self-posts from crowd sources contains hidden trends, which can be valuable to a business enterprise. Crowd trend mining methods, together with an easily understood visual presentation, are thus in great demand. We present an approach to mining crowd trends from Thai text posts. A Thai language preprocessing module was necessary to transform continuous text into series of words. Our method could then mine general unforeseen crowd trends by using an automatic context extraction technique, tf-idf score and an aggregated opinion score calculated from automatically classified sentiments for each post or comment. The best sentiment classifier was chosen based on extensive experiments on the same data source. These scores were combined into one unified term popularity which was visualized as a word cloud on a web application. A case study used a popular Thai discussion website - Pantip.com - and achieved three interwoven desired goals: (1) extraction of general and unforeseen crowd trends from a Thai discussion website, (2) assigning unified popularity scores to each candidate term and (3) presenting those terms to end users in an easily comprehended form.

Keywords: Thai text, posts and comments, crowd trends, social media, content extraction, sentiment analysis.

1.0 INTRODUCTION

The availability of very large bodies of crowd sourcing self-posted information, from various social media platforms, has created a unique and potential opportunity affecting both individuals and enterprises. A large number of individual self-posts constitute crowd sources that can be analyzed and synthesized to reflect crowd trends. While there are difficulties associated with detecting and extracting crowd trends from an enormous number of posts and comments: Pappas reported that it is in a scale of several thousand posts per second [1]. There are also undeniable business benefits for monitoring relevant crowd trends. For example, firms investing in social listening technology can gain a competitive advantage, over other firms in terms, of adapting and responding to customer needs and wants [2]. In addition, when common crowd interests are identified, the firms can analyze and revise their strategies, either to attract potential new customers or to promote their public image or to create brand royalty.

On the surface, the task of mining social media crowd trend can be classified as an instance of a terminology extraction task, where a collection of social media text posts and comments constitute a domain corpus, and the desired result is a set of relevance terms or concepts. Nevertheless, it is not as simple and straightforward task as one may consider, especially in the Thai language. First, in general, the task is somewhat different from a classical terminology extraction task, in the sense that the text corpus collection from the social media stream, although it has a finite number of posts, in some fixed timeframe, it is changing in real time. Hence, the trend can change very quickly in a very short time. Next, there is no real *domain corpus*, posts and comments are mostly short opinions of individuals. Unlike essay style documents, these short messages may not even contain a prominent or noticeable topic in themselves. Third, there is a significant difference between detection domain specific terminology and a general terminology. While one can predefine seed terms in the former task, there is no such constraints in the latter task. This presents difficulties in detecting *trends* from social media text posts in any language. For Thai language, the last, and the most daunting, task is processing Thai text, as discussed by Netisopakul and Wohlgenannt [3]. In short, the continuous conjugated Thai text causes problems in most knowledge extraction tasks, including terminology extraction. Thai is highly ambiguous in word formation and word role directly affects the determination of word boundaries, as well as terms.

We developed an algorithm to solve these problems, that is, to detect crowd trends from Thai text posts and comments. The methodology includes Thai text preprocessing, automatic content extraction using a weighted scheme on text location, TF-IDF, to calculate text prominences. In addition, sentiments were classified for each trend to identify whether is was favorable. The case study used Pantip.com, one of the most accessible online self-posting and commenting site in Thailand. Alexa.com reports that Pantip.com is fifth ranked in Thailand [4], with approximately 450 millions page views per month and 4.2 million unique users per day. In 2016, it had more than 34 million forum threads and is still growing [5].

The outline of this article is as follow. Section 2 reviews relevance existing research. Section 3 presents the architecture of crowd trend detection system with a detailed illustration. Section 4 describes results from the system and section 5 discusses the advantages and disadvantages of our approach, including opportunities for future improvement.

2.0 REVIEW OF EXISTING RESEARCH

2.1 Social Listening

Social listening monitors and collects digital conversations on social networking platforms. The main purpose is to look for mentions of brands, products, competitors, themes, trends and interesting ideas. The collected data are then analyzed to obtain actionable insights [2]. This actionable element of social listening differentiates it from "social media monitoring". The latter mainly concerns monitoring and gathering data. In addition, social media monitoring focuses on metrics, like engagement rate, number of likes and so on, while social listening looks beyond the numbers and highlights the overall mood and trends behind the social media posts [2].

An example of a work on social listening is a study by Cuesta et al. [6], who developed a framework for trend discovery from Twitter data. The framework was centered on a MongoDB database, surrounded by five key plus one additional working components: a miner, a trainer with a tester, a classifier, a sentiment classifier and a report generator. Components were communicated to each other using JSON format. The miner acts as collector of data from Twitter, by continuously *listening* to a Twitter stream, applying filters and storing data in the central MongoDB database. Three components – the trainer, the tester and the classifier work together as a supervised learner to create classification models. While the trainer automatically creates corpora and trains models in the background, the classifier creates manually classified training data through a web interface by presenting each tweet message, one by one, to human collaborators, who decide its sentiment. The tester is then applied via a command line and web interface to assess the suitability of the trained models in real-time. The best classifier is used in the sentiment classifier. The report generator presents both a quantitative statistical analysis report and a sentiment classification report to end-users. Major portions of their paper are devoted to applying sentiment analysis in a case study - Spanish political activity on Twitter, using naïve Bayes classifiers. Their framework has several advantages. First, it stored and exchanged data using an open data formats, such as JSON. In addition, the report was also generated as CVS files. Hence, the collected data and the output can be easily ported to other systems. Second, it used publicly available tools such as NLTK. This enhanced component extensibility. Third, it embedded an ability to collect and process data in parallel and, lastly, it provided a solution to obtain a training data in a human collaborative fashion. However, the classifier only classified tweets into positive or negative classes without considering term popularity.

2.2 Keywords Extraction

A keyword can be a single word or a multi-word term and it often serves as a dense summary for a document or a description of a document [7]. Keywords extraction is an important topic in information retrieval, web page retrieval, summarizing and text mining fields [8] [9]. Keyword extraction deduces *important words or phrases*, usually from a single document [8] [10]; while terminology extraction, given a collection of documents - usually in the same domain, extracts relevant terminology for the domain. Extracted terms are associated to hierarchical concepts and relations, in a domain ontology. While the keyword extraction literature is extensive, there are fewer works on terminology extraction. However, sometimes the names of these two tasks are used interchangeably by researchers.

Several keywords extraction approaches have been suggested previously. A well known method is TF-IDF (Term Frequency-Inverse Document Frequency). The intuition behind this approach is that terms that appear repeatedly in one document, but rarely seen in others, are often a good representative of that document. TF-IDF and its variants have been widely used in concept extraction and satisfactory results reported. For example, Li et al. [11] analyzed

linguistic characteristics of news documents in general. From the analysis, they devised a keyword extraction method for Chinese news documents based on TF-IDF with multi-strategies. An experiment comparing their approach with a baseline method provided good results. Bun and Ishizuka [12] adopted TF-**P**DF, or Term Frequency-**P**roportional Document Frequency, to extract hot topics from archives of newswire sources on the Internet. Essentially, they included the number of news sources, that are reporting on a topic, to improve extraction performance.

In addition, several interesting supervised and unsupervised approaches have been suggested. Krulwich and Burkey [13] used a set of heuristic rules to extract significant phrases from user classified documents, then applied decision tree to obtain important keywords and to transform these keywords into a new query search string. Ramirez and Mattmann [14] described an Automatic Concept Extraction (ACE) system focusing on keyword extraction from web pages. ACE used Term Frequency (TF) and HTML scores to extract keywords from web pages. However, both TF and HTML scores tended to emphasize single string concepts, while a concept may be comprised of more than a single string. Consequently, they also observed tokens around the identified keywords and combined the tokens with the keywords to provide additional meaning. Building upon Ramirez and Mattmann's ACE, Zhang et al. [15] added several improvements (i.e., tokenization, tag and sentence boundary, combined TF scorer, emphasis scorer and Porter stemming) and applied their method in an e-commerce context, i.e., product matching and topic-based opinion mining. Experiments showed that their enhancements significantly improved the performance of ACE. Similar to our study, O'Conner et al. [16] developed TweetMotif – a topic extraction system based on syntactic filtering, language modeling, near-duplicate detection, and set cover heuristics. It was used to wear down rumors, detect scams, summarize sentiment, and track political protests in real-time. Witten et al. [17] used naïve Bayes learning techniques in KEA (automatic keyphrase extraction) to build an extraction model from known keywords.

Netisopakul and Wohlgenannt noted that there are surprisingly few works on terminology extraction from Thai documents [3]. The work by Imsombut and Kawtrakul [18] automatically built an ontology in the agriculture domain by extracting terms and their relations from agricultural documents: they employed shallow parsing techniques, that is, lexicon-syntactic patterns. When ambiguity surfaced, candidate terms were chosen based on maximum likelihood of four lexical features and a co-occurrence feature. In addition, each feature could have a different weight, determined by calculating information gain obtained from partition the examples according to the feature.

Interestingly, the four lexical features proposed by Imsombut and Kawtrakul [18] are: (1) whether the head word of a candidate term was compatible with the head word of the related term or not. If so, then the two terms are related. (2) Whether a hypernym of a candidate term belonge to the same NE class as the related term or not. (3) Property lists, such as colors, shapes and appetences, were used to associate properties to concepts. (4) Whether the candidate term was the topic term of the document or the paragraph or not. The last feature is the co-occurrence feature. It measured co-occurrence chances of the candidate and the related term, with experiments to show the highest precision with χ^2-test.

2.3 Sentiment Classification

Prior research on sentiment classification has emphasized classifying text sentiment at three levels: (1) document level, (2) sentence level and (3) entity and aspect level. Here, we classified text sentiment at a document level. This section therefore focuses on document level sentiment classification. Most methods analyzing at this level are based on supervised learning techniques, while some have suggested unsupervised methods [19].

Several supervised techniques, such as naïve Bayesian, Neural Network and SVM, have often been used to classify sentiment at both sentence and document levels. Prior research often obtained training and testing data from product reviews, since each review usually already had a rating assigned by the reviewer. Pang et al. [20] used movie reviews as data and discovered that naïve Bayes, maximum entropy classification and SVM perform better than human-produced baselines. The baselines were based on the frequency of positive and negative words identified by two graduate students as a good indicator for text sentiment classification in movie reviews. The study suggested that SVM classification model, using unigrams as features, produced the best accuracy rate at 82.9%. Similar to Pang et al. [20], Boiy and Moens [21] used the three supervised techniques to analyze text sentiment in multilingual Web texts. They achieved an accuracy rate of 83% for English, using unigrams as features. However, the accuracy of classifying Dutch texts was 70% and French was 68%. Consistent with most research on machine learning applications, the sentiment classifier's main objectives was to develop an effective set of features. Some of the

features were: (1) word n-grams and their frequency counts; (2) parts of speech (especially adjectives); (3) opinion-bearing words or phrases; (4) negations; and (5) word dependency generated from dependency trees [19].

A good portion of existing works also employed unsupervised learning approaches to sentiment classification. Opinion words or phrases are commonly used in unsupervised learning methods. For example, Kraichit et al [22] used a lexicon-based method for sentiment analysis, which included dictionaries of words and their semantic scores augmented with negations and intensifiers. They found that their performance was consistent across different domains and on unseen texts, which seemed to overcome the main problem of supervised learning methods. In the Thai context, Kraichit and colleages [23] developed a web-application to assess a Facebook page credibility, based on sentiment of comments of the page, together with key statistics such as engagement rate, continuity, etc. The system obtains sentiment scores of sentiment-bearing words from the S-Sense [24] database to assign a sentiment score of each word in their dictionary. Then, a z-score approach is adopted to compute the sentence-level sentiment scores. However, the lexicon-based approach is not without weaknesses since its key process – i.e. dictionary generation – was sometimes unreliable and could not keep up with the ever-changing linguistic trends on social media [25].

Another work by a research group at NECTEC [26] called S-Sense or social media sensing is more related to our work. There are four main components. A text collecting and processing component crawls and collects contents from social media websites, then processes basic Thai text operations such as sentence segmentation, tokenization and term lemmatization. A UREKA component has two duties; (1) it extracts, from a given piece of text, key feature terms or phrases and (2) filters and classifies the given text into a topic. A main component, called S-Sense, contains several language analysis and sentiment analysis modules and a user-interface module with emoticon visualization, interactive dashboard and word cloud. The last important component is a language resource and a tagging tool to enhance the system ability. The author of this work claims the work's strength as "The framework is designed for easy adaptation to businesses in different domains" and gives potential applications such as brand monitoring, campaign monitoring, competitive analysis and employee engagement. One interesting feature of this system is an intention analysis module, inside the S-Sense component, which classifies a given short text into four classes of intentions. - announcement, request, question and sentiment. This seems consistent with many research observation that about half of social media posts contain neither positive nor negative sentiments, but are neutral.

3.0 OUR APPROACH

Our approach to crowd trend detection involves two key steps. The first step involves extracting keywords from individual forum threads. The objective is to identify words or phrases, that represent key topics or ideas of a forum thread. The aggregation of those keywords would reflect the current popular trends online. Additionally, we believe that extracting topics or keywords and assessing their popularity, based on how frequently people are mentioning those topics, may not give the full picture of online crowd trends. The other key dimension of popularity is how people are talking about the topics, thus sentiment classification. We generally observed that crowd trends were likely attract high levels of attention of an online crowd. In addition, online people often expressed their opinions and attitudes towards hot topics and did not just simply mention it, i.e. were neutral. Therefore, topics or keywords that were discussed with extreme positive or negative sentiment had a higher possibility of becoming crowd trends.

Before we can start extracting crowd trends, some necessary steps of text processing must be implemented. We cleaned the collected text by removing special characters and emoticons. Further, since Thai is an unsegmented language, we segmented the text. A dictionary-based approach was used. Essentially, the approach involved scanning through series of input characters and matching them against words from a dictionary. We used LexTo [27] together with the LEXiTRON dictionary [28], which contains more than 100,000 Thai words, for this task. Stop words were also dropped. The LexTo library adopts a longest matching technique to word segmentation. The processed data then entered the keyword extraction procedure.

3.1 Keyword Extraction

The first step to mining crowd trends involved extracting keywords or key phases from online posts. As you may imagine, millions of posts are emerging on the Internet every second. To perform keywords extraction, we therefore used two of the key elements of the Ramirez and Mattmann's ACE [14], which are Term Frequency and HTML scores. Both arguably require no document corpus and relatively less processing power. Moreover, ACE is specifically engineered for extracting keywords or concepts from webpages.

3.1.1 Term Frequency (TF)

Following Ramirez and Mattmann, [14], TF can be defined as the number of times that a term (*t*) appears in a set of one or more documents. In our case study, we restricted the computation of TF to a single discussion thread and computed TF from:

$$TF_a(t_i) = \frac{Occurence(t_i)}{Max(Occutance(t_1),Occurance(t_2),\ldots,Occurance(t_n))} \quad (1)$$

3.1.2 HTML Scorer

The HTML scorer assumes that there are several HTML tags that emphasize the main concept of a web page more often than others. Words within those tags are more likely to be keywords. We therefore gave higher weight to words in those tags, for example, header (<h1>) or bold (<b>) tags. Examples of HTML tags and their weights, taken from [15], are listed in Table 1. In addition, we also assumed that the location of each word on an HTML page is important in the Pantip.com context. Some locations, such as thread owner posts, were likely to contain keywords. We discussed the nature of the Pantip.com website and ultimately allocated weights to locations. We ran an experiment (discussed in Section 4.2) and adjusted location weights accordingly. Table 2 displays some examples. To find candidates for keywords, keyword scores of all words in a discussion thread were calculated, as in (2). We used the scores to rank the top 10 potential keywords.

Table 1: Examples of HTML tag weights

Tag	Meaning	Weight
b	Bold	0.8
h1	Heading 1	1.0
strong	Strong emphasis	0.8
u	Underline	0.7
i	Italic	0.3
title	Page title	1.0
th	Table header	0.5

Table 2: Examples of location weights

Location	Weight
Thread page topic	1.75
Thread owner posts	1.25
Comments	0.75
Top comments	1.0

In addition, we used TF-IDF to help improve the ACE performance, since it penalized words that appear frequently in most documents. To find keywords, the system calculated TF-IDF scores for the top 10 candidates. Those words, with a TF-IDF score higher than the average of all 10 words, are considered as the keywords. More specifically, the Term Frequency (TF) part of TF-IDF was calculated as in (2), while the Inverse Document Frequency (IDF) was computed as in (3). For (2), let *d* be a term identified to be in the top 10 candidates of a post, $k(d)$ is the keyword score of term *d* and $w_i(d)$ represents the HTML weight value of occurrence, *i*, of term *d*.

$$k(d) = TF_a(d) + \frac{w_1(d)+w_2(d)+\cdots+w_n(d)}{n} \quad (2)$$

$$\mathrm{TF}(t) = \frac{\text{Number of times term } t \text{ appears in a discussion thread}}{\text{Total number of terms in the discussion thread}} \quad (3)$$

$$\mathrm{IDF}(t) = \log\left(\frac{\text{Total number of discussion threads}}{\text{Number of discussion threads with term } t \text{ in it}}\right) \quad (4)$$

To assess the performance of the our method, we first randomly selected 300 posts from the collected corpus. Then, we asked two independent readers to read the posts and their choice of keywords for each post (in most cases, more than two keywords were tagged). We compared the resulting keywords of each post and keep only those that were consistently identified. Finally, the human-tagged keywords were used as the benchmark for the evaluation of our keyword extraction approach. The results are described in section 4.

3.2 Sentiment Classification

We adopted a supervised learning approach to sentiment classification. In order to choose the best machine learning technique, from the corpus, we randomly selected 5,000 comments. Those comments were given to postgraduate students at our faculty. The students classified each comment into three categories: positive, neutral and negative. We then randomly selected 4,000 tagged comments to create five classification models using five different algorithms: SVM, decision tree, naïve Bayes, logistic regression and a neural network. Although the philosophies behind there five learning techniques are quite diverse, each has been explored and was found to be effective in prior sentiment classification studies. Furthermore, this study focused on features bases on unigrams, bigrams and TF-IDF: so 15 experiments were run. To evaluate performance, we applied the models to the other 1,000 unseen comments. The model classifications were then compared with the classifications provided by a human, to measure model accuracy. Both model construction and evaluation were executed on the Microsoft Azure Machine Learning platform [29]. From the results, the SVM model was found to perform best. Evaluation results, as well as extensive justification of the decision, are set out in section 4.

3.2.1 Opinion scorer

We started classifying user sentiment for an extracted keyword by searching through the corpus for all comments with the keyword. This set of comments was then input to the selected classification model. The results were then used to identify whether the keyword is an online crowd trend. We call them the "opinion scores". Let d be an extracted candidate term produced by the proposed extraction procedure, mentioned in section 3.1. Let c be a comment containing keyword k. $s(c)$ is the sentiment score of comment c from SVM classification model. The opinion scores ($o(d)$) of keyword d were calculated from:

$$o(d) = \left| \frac{\sum_1^n \left(\frac{s(c_i) - min(s(c_1,c_2,c_3,\ldots,c_n))}{\max(s(c_1),s(c_2),s(c_3),\ldots,s(c_n)) - \min(s(c_1),s(c_2),s(c_3),\ldots,s(c_n))} \right)}{n} \right| \quad (5)$$

To explain this equation, we first normalized the obtained sentiment score of comment c (or $s(c)$) using the min-max normalization technique. Then, the average value of the scores was computed to obtain the opinion score of keyword k (or $o(d)$). Combining the keyword score and TF-IDF (being normalized with min-max normalization method), we suggest that the online crowd trend can be identified. Similarly, the keyword scores of all keywords were normalized, using mean normalization, prior to the addition to the opinion score which resulted in a "popularity score" :

$$\text{Popularity score of term } d_i = \text{normalized}(k(d_i)) + \text{normalized(TF-TDF score)} + o(d_i) \quad (6)$$

In the next section, the case study of online crowd trend extraction from Pantip.com is described. A web-application was also developed to visualize the extraction results and to show how businesses would benefit from adopting the approach to keep tabs on what people are talking about online.

4.0 CASE STUDY – EXTRACTING POPULAR TRENDS FROM PANTIP.COM

Pantip.com is a popular online forums in Thailand. The website has several discussion *rooms*, categorized by common crowd interests. For example, a *blue planet* room discusses leisure and travelling; *Lumpini park* discusses health related issues and *Bangkhunprom* discusses television's series, actors and stars. A user first chooses a room in which to post and initiate a discussion thread. Then there are usually many comments (namely, responses, feedback, etc.) from other users followed by more responses or more posts from the thread owner. Note that each discussion thread has its own set of tags. These tags will also be stored in the database and be used in our application. In order to extract popular crowd trends from this website, we used two main components. The first is a web-application that automatically extracts keywords from posts and assesses their popularity. The second involves the keywords extraction and sentiment classification methods as well as the results measuring performance.

We created a web application to collect data from the website, process the data and visualize the detected crowd trends which was briefly described previously [30]. The web application was written in Java with NoSQL database (MongoDB). The main components of the system are:

- The backend system crawls the website to collect data. We used jSoup [31] – a Java HTML parser – to extract data from the website and then store it in a local database. We collected all the discussion threads on Pantip.com during the Songkran holiday (13 – 14 April 2017) as we wanted to find what the 'hot' topics during the holiday were. A total of 4,142 threads were collected.

```
(objectId:1,
thread_id: 35625262
thread _title: {CR Review โรงแรม..,}
tag: {เที่ยวไทย..,}
room: blueplanet
thread_user: 3405203
thread_date: 2016-07-23 14:36
comment:{สวัสดีค่ะเพื่อนๆ…, …}
clean_comment:{สวัสดีค่ะเพื่อนๆ…, …}
keywords: {หัวหิน, …}
ace: {9.0, …}
tf_idf_score: {0.015, …}
opinion_score: {0.89, …})
```

Fig. 1: Example of a discussion thread document

- The raw data were passed to a pre-processing module. First, special characters and emoticons were removed from the collected text. Secondly, the text was segmented into series of words. We employed LexTo for Thai word segmentation [27]. The library uses a dictionary-based approach to word segmentation. Then, stop words were removed using a custom-made dictionary developed by us. Finally, we stored the collected data into our MongoDB database – i.e., a NoSQL database. Our main MongoDB collection collected rows of discussion thread documents. The document structure is shown in Fig. 1. To clarify, at this stage, keywords, ace, and tf_idf_score were yet to be calculated.
- The third key component extracted the keywords. TF scores, HTML scores and consequently keyword scores of all words were computed. Based on the keyword scores, we identified a list of top-ten candidate terms for each discussion thread. Ultimately, TF-IDF score for each candidate was obtained.
- Next, comments containing the candidate terms were forwarded to the sentiment classification component. Specifically, we used an SVM classifier on the Microsoft Azure platform [29]. A web service was created as an interface between our web application and the model on Azure. For each candidate keyword, sentiment scores of all relating comments were produced by the model. We then used these sentiment scores to determine an opinion score for the keyword.
- By adding keywords, TF-IDF and opinion scores, we obtained the popularity score of a candidate term. The results were visualized on our web application in the form of a word cloud. The web application will be discussed in detail in the next subsection.

Fig. 2 presents an overview of our crowd trend detection system.

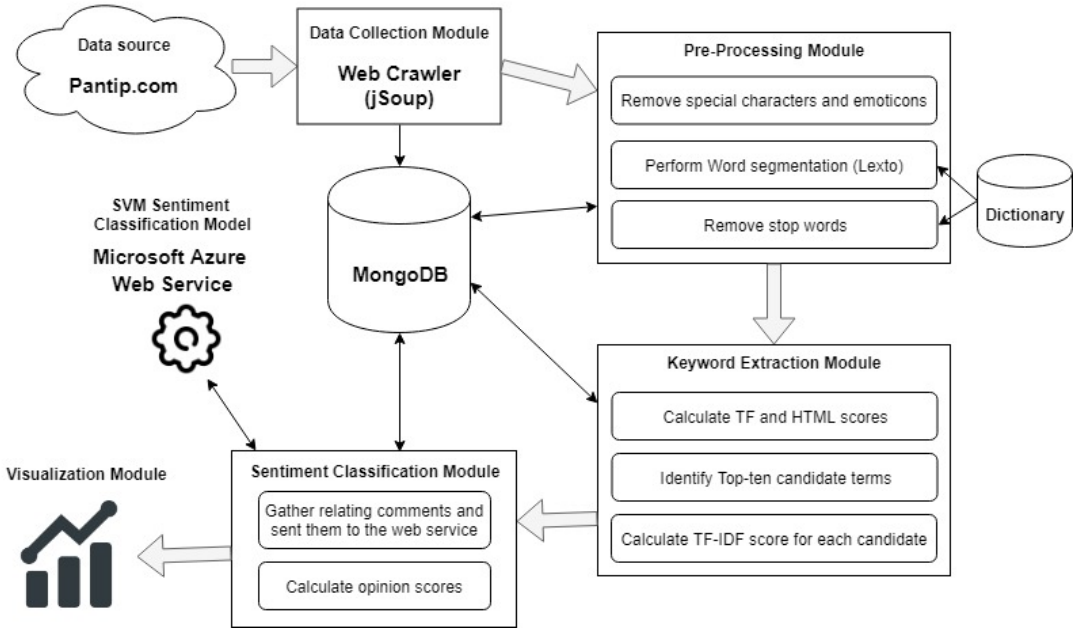

Fig. 2: Overview of the system

4.1 Web Application

In addition to the data processing, pre-processing, keyword extraction, sentiment classification modules mentioned above, we describe the visualization module in this section: it has three key features: (1) word clouds, (2) sentiment results dialog box, and (3) product or service recommendation.

For the first feature, the system displays keywords as a word cloud based on popularity score of each keyword as shown in Fig. 3. More popular words use a larger font size. The font color of a term reflects its opinion score with darker tones for lower scores, while lighter tones represent higher scores. This allows users to easily identify trendy keywords at a glance. Moreover, the word cloud can be filtered using two criteria, namely rooms and tag words (i.e., a discussion thread usually has many tag words assigned by the initiator). Therefore, a cloud of popular words can be created for (1) the whole Pantip.com, (2) a particular discussion room or (3) a particular tag. When a particular tag is chosen, a word cloud is created from those discussion threads containing the tag word.

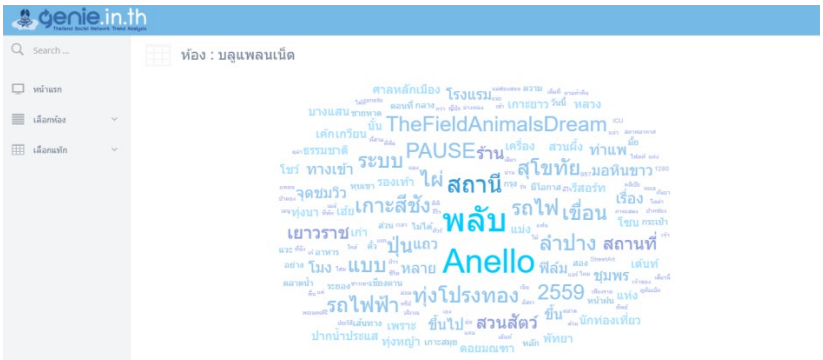

Fig. 3: Word cloud generated by the system

There is another key feature of our word cloud. Users can easily obtain sentiment analysis results by clicking on a word of interest in the cloud. To be more specific, when users click on a word, a pop-up dialog box containing all posts that contain the selected keyword and their sentiment scores. The dialog box also summarizes the number of relevant posts that are positive, negative or neutral (Fig. 4). We use the sentiment scores obtained from Azure web service to categorize the posts. Those that have scores of more than 0.75 are classified as positive, while those scored less than 0.65 are negative. The rests are classed as neutral.

Fig. 4: Pop-up dialog for sentiment results

Finally, the system searches three websites (www.aliexpress.com, www.agoda.com and www.thaifranchisecenter.com) for any information relevant to the top keywords that were identified. The objective is to demonstrate a use case showing how end-users can benefit from crowd trends. For example, if the system were to suggest that "fried squid" is a very popular food at the moment, users who are looking to start a business may want information about fried squid franchises from www.thaifranchisecenter.com. However, note that the system does not categorize a keyword, whether it is a destination, product, food, etc., before searching through the three websites. It rather searches through those websites for all identified keywords.

4.2 Experiment Results

Here, we discuss the results of two experiments. The first measured performance of the keyword extraction approach described in section 3.1. The second experiment measured sentiment classification accuracy.

In the key word extraction experiment, we randomly selected 300 discussion threads from the corpus. The selected threads were read by a reader, who extracted a list of five words representing the key concepts. For each thread, we compared the human list with a machine-extracted and top-five keywords produced from our algorithm. We achieved a precision of 0.69, suggesting that out of 100 keywords, extracted by the machine around 69 keywords overlap with manually selected concepts.

The sentiment classification experiment used the Microsoft Azure Machine Learning platform [22]. As mentioned in section 3.2, we randomly selected 5,000 comments and gave them to third year undergraduate students to classify them as positive, negative or neutral. We then split them into chunks of 4,000 for training and 1,000 for testing. We trained five different models, i.e., SVM, decision tree, naïve Bayes, logistic regression and neural network, focusing on three features - unigrams, bigrams and TF-IDF. Moreover, we assessed the model accuracy using unseen data of 1,000 records. The accuracies are displayed in Table 3. The best performers for each feature category were decision tree for unigram, SVM for bigram and naïve Bayes for TF-IDF. From Table 3 we see that SVM produced the most consistent results across all three features. Furthermore, the range of SVM accuracies is the lowest, which reflected consistency. We therefore decided to implement the SVM model as a part of the sentiment classification module of our trend detection system.

Table 3: Sentiment classification accuracy results

Feature / Technique	SVM	Decision Tree	Naïve Bayes	Logistic Regression	Neural Network
Unigrams	77.7%	77.8%	75.2%	77.2%	74.1%
Bigrams	77.7%	71.9%	75.0%	72.9%	71.0%
TF-IDF	75.4%	71.2%	77.79%	71.8%	72.2%
Range (max - min)	2.3%	6.6%	2.8%	5.4%	3.1%
Average	76.9%	73.6%	76.0%	74.0%	72.4%

After the decision to employ the SVM model was made, we assessed which choice of features provide the best results, using precision and recall values. Table 4 suggests that adopting TF-IDF as features produces the highest

precision, while bigrams provide the best recall. We chose the most appropriate features based on the results of Table 3 and 4. Unigrams and bigrams perform similarly. However, we chose bigrams. In sum, we adopted the SVM classification model with bigram features for our web application.

Table 4: Precision and recall of SVM

Feature	Precision	Recall
Unigrams	78.0%	99.2%
Bigrams	77.9%	99.5%
TF-IDF	78.8%	93.5%

5.0 DISCUSSION AND FUTURE WORK

In summary, we described an unsupervised approach to capture social media crowd trends from Thai text posts and comments. A case study was based on a popular online discussion website in Thailand – Pantip.com. After collecting posts and comments from the website using the jSoup web crawler, the data was passed through three main modules. The language pre-processing module segmented Thai words, removed special characters, emoticons and stop words. The keyword extraction module initially extracted the a top ten keywords from each discussion thread by calculating a *keyword score* using a term frequency and HTML weighing scheme, emphasizing the page tags, owner posts and top comments. The sentiment classification module calculated a normalized *opinion score* for each keyword. These two scores were combined with TF-IDF score into a term *popularity score*. This information was stored in JSON format in a MongoDB database for a web application.

The crowd trends were shown on the web application as a *word cloud*, where the word size is based on the popularity score and the shade was based on its sentiment. Three ways of presenting crowd trends can be selected by users: (1) a general trend of the whole Pantip.com, (2) a particular trend within a discussion room and (3) a trend related to a particular tag word. In addition, users could click each word in the word cloud to explore its content in all posts together with the post sentiment - classified as positive, negative or neutral. One use of crowd trends was demonstrated by searching for trend related merchandise from three commercial websites and suggesting related products to end-users.

This approach has several advantages. First, this is one of the first studies to mine *general crowd trends* from a popular social media Thai discussion website containing Thai text posts and comments, such as pantip.com, which required some special techniques to extract both the main post and follow up comments to it. Haruechaiyasak et al [26] also mined general crowd trends, mostly from twitter. Other known works on Thai text mostly ran experiments on Thai text documents with a specific purpose, for example, to detect name entities or to extract domain specific keywords. A special language preprocessing module was necessary to prepare Thai text for the next step.

Second, our method used an *unsupervised approach*, based on term frequency and an HTML weighting scheme, so crowd trends can be detected without prior knowledge of domain specific terminology or a training corpus. Therefore, an emerging trend, that has not been seen before, can be detected. Our method achieved a precision of 0.69, compared to values reported by Zhang et al [15] of 0.2496 for ACE, 0.7396 for ICE and 0.6930 for KEA. Essentially, we extended ACE [14] and ICE [15] HTML scores with location scores and tf-idf. Our method performed better than ACE and achieved an acceptable level of precision when compared to the unsupervised method ICE, which is more complex, and the state-of-the-art method, KEA.

Third, to accurately classifying sentiment for each post or comment, 15 experiments selected the most appropriate sentiment classification model to be used. In addition, the aggregated opinion score was used to calculate overall sentiment for each candidate term that appeared in several posts and comments. Note that since the prior experiments to *choose the best classification model* has been conducted on the same source of data, i.e., from Pantip.com posts and comments, the resulting recall rate on the test data is very high – 99.5% using SVM and bigrams. Furthermore, the average accuracy rate of 76.9% in our study was satisfactory when compared to previous studies on Thai text sentiment analysis. For example, Lertsuksakda et al. [32] used a dataset of 40 Thai children's stories (1,964 sentences) and classified the sentences into three classes - positive, neutral, and negative. They used SVM with three different kernels (Linear, RBF, and Polynomial) and achieved the best accuracy of 72.1% with the RBF kernel. More recently, Vateekul and Koomsubha [33] applied two deep learning techniques (Long Short Term

Memory (LSTM) and Dynamic Convolutional Neural Network (DCNN)) to sentiment classification of Thai Twitter data. They used word vectors as features and tested 22,000 tweets. Both deep learning techniques performed at ~75% accuracy.

Fourth, most previous research either focused on extracting keywords or classified opinions of a given text. Our method combined keyword scores, TF_IDF scores and opinion scores into one *unified term popularity score*. This enabled terms to be *visualized as a word cloud* and to be explored for their actual contents in posts corresponding with sentiment classes. Thus this method achieved the goal of an easy apprehended visual presentation. Table 5 displays examples of keywords with the highest popularity scores from 100 randomly selected forum threads. Please note that all values in the table were normalized with min-max method.

Table 5: Examples of keywords with highest popularity scores

Keywords	Keyword scores	TF-IDF scores	Opinion scores	Popularity score
สวนผึ้ง	0.383	0.730	0.857	1.97
PAUSE	0.670	0.179	0.938	1.787
นราธิวาส	0.296	0.783	0.654	1.732
เชียงคาน	0.339	0.530	0.820	1.689
น่าน	0.626	0.057	0.990	1.673
เกาะสีชัง	0.687	0.279	0.662	1.628
มหานคร	0.104	0.551	0.949	1.604
เกาะล้าน	0.6	0.117	0.832	1.549
เจดีย์	0.739	0.069	0.733	1.541
เขื่อนศรีนครินทร์	0.104	0.343	1	1.448

Fifth, the centralized MongoDB database stored the detailed scoring scheme and enabled fast and flexible processing. This in turn enabled the trends to be *explored via the web application in three levels*: from the whole website, from a specific room, and from a specific tag.

Lastly, as an example, we employed the detected crowd trends in automatic search for and *suggest related merchandise* to end users.

Nevertheless, there are also research areas that can be improved. First, we could group similar or related terms into one topic and to group related topics into a theme. This would enhance the homogeneity of posts and comments from different threads. However, this is an open research topic for Thai language processing. Second, the system can be improved to separate and parallelize the collection module from the other modules. The parallel components connected only to a common central database would be more suitable to handle incoming post streams, continuously processing them, and visualizing changing crowd trends in real-time. Another area might predict crowd trends in advance, instead of mining them from past posts and comments. For businesses, knowledge from the identified trends pervading in the market the could be used to predict customer needs and wants, then innovate products that suitable to their needs.

REFERENCES

[1] S. Pappas, "How Big Is the Internet, Really?," 2016. [Online]. Available: https://www.livescience.com/54094-how-big-is-the-internet.html.

[2] C. Newberry, "What is social listening?," 2017. [Online]. Available: https://blog.hootsuite.com/social-listening-business/#whatis.

[3] P. Netisopakul and G. Wohlgenannt, "A Survey of Thai Knowledge Extraction for the Semantic Web Research and Tools," *IEICE TRANS. INF. & SYST.,* vol. E101–D, no. 4, 2018.

[4] Alexa.com, "Top Sites in Thailand," 2017. [Online]. Available: http://www.alexa.com/topsites/countries/TH.

[5] Pantip.com, "Pantip Overview in 2015-2016," 2017. [Online]. Available: https://pantip.com/advertising.

[6] A. Cuesta, D. F. Barrero and M. D. R-Moreno, "A Framework for massive Twitter data extraction and analysis," *Malaysian Journal of Computer Science,* vol. 27, no. 1, pp. 50-67, 2014.

[7] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, Sapporo, Japan, 2003.

[8] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," *International Journal on Artificial Intelligence Tools,* vol. 13, pp. 157-169, 2004.

[9] K.-Y. Chen, L. Luesukprasert and T. C. Seng-cho, "Hot topic extraction based on timeline analysis and multidimensional sentence modeling," *IEEE transactions on knowledge and data engineering,* vol. 19, 2007.

[10] S. Rose, D. Engel, N. Cramer and W. Cowley, "Automatic keyword extraction from individual documents," *Text Mining: Applications and Theory,* pp. 1-20, 2010.

[11] J. Li and K. Zhang, "Keyword extraction based on tf/idf for Chinese news document," *Wuhan University Journal of Natural Sciences,* vol. 12, pp. 917-921, 2007.

[12] K. K. Bun and M. Ishizuka, "Topic extraction from news archive using TF* PDF algorithm," in *Proceedings of the Third International Conference on Web Information Systems Engineering*, Singapore, 2002.

[13] B. Krulwich and C. Burkey, "Learning user information interests through extraction of semantically significant phrases," in *Proceedings of the AAAI spring symposium on machine learning in information access*, Palo Alto, California, 1996.

[14] P. M. Ramirez and C. A. Mattmann, "ACE: improving search engines via Automatic Concept Extraction," in *Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration*, Las Vegas, Nevada, 2004.

[15] Y. Zhang, R. Mukherjee and B. Soetarman, "Concept extraction and e-commerce applications," *Electronic Commerce Research and Applications,* vol. 12, pp. 289-296, 2013.

[16] B. O'Connor, M. Krieger and D. Ahn, "TweetMotif: Exploratory Search and Topic Summarization for Twitter," in *Int'l AAAI Conference on Weblogs and Social Media*, Washington, DC, 2010.

[17] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin and C. G. Nevill-Manning, "KEA: Practical automatic keyphrase extraction," in *Proceedings of the fourth ACM conference on Digital libraries*, Berkeley, CA, 1999.

[18] A. Imsombut and A. Kawtrakul, "Automatic building of an ontology on the basis of text corpora in Thai," *Language Resources and Evaluation,* vol. 42, no. 2, p. 137–149, 2008.

[19] B. Liu and L. Zhang, "A Survey of Opinion Mining and Sentiment Analysis," in *Mining Text Data*, Springer, 2012, pp. 415-463.

[20] B. Pang, L. Lee and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, Philadelphia, PA, 2002.

[21] E. Boiy and M.-F. Moens, "A machine learning approach to sentiment analysis in multilingual Web texts," *Information Retrieval,* vol. 12, no. 5, p. 526–558, 2009.

[22] M. Taboada, J. Brooke, M. Tofiloski, K. Voll and M. Stede, "Lexicon-Based Methods for Sentiment Analysis," *Computational Linguistics,* vol. 37, no. 2, pp. 267-307, 2010.

[23] K. Kraichit, J. Amarittakul, P. Netisopakul and B. Thanasopon, "Social commerce credibility analysis," in *8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, Yogyakarta, Indonesia, 2016.

[24] NECTEC, "S-Sense Social Sensing," 2016. [Online]. Available: http://www.ssense.in.th/.

[25] A. Andreevskaia and S. Bergler, "When Specialists and Generalists Work Together: Overcoming Domain Dependence in Sentiment Tagging," *ACL,* pp. 290-298, 2008.

[26] C. Haruechaiyasak, A. Kongthon, P. Palingoon and K. Trakultaweekoon, "S-sense: A sentiment analysis framework for social media sensing," in *Workshop on Natural Language Processing for Social Media (SocialNLP)*, Nagoya, Japan, 2013.

[27] NECTEC, "Thai Lexeme Tokenizer," 2016. [Online]. Available: http://www.sansarn.com/lexto/.

[28] NECTEC, "LEXiTRON: Thai - English Electronic Dictionary," 2009. [Online]. Available: http://lexitron.nectec.or.th/2009_1/.

[29] Microsoft, "Microsoft Azure Machine Learning," 2016. [Online]. Available: https://azure.microsoft.com/en-us/services/machine-learning/.

[30] B. Thanasopon, S. Nattawut, J. Buranapanitkij and P. Netisopakul, "Extraction and Evaluation of Popular Online Trends: A Case of Pantip.com," in *The 9th International Conference on Information Technology and Electrical Engineering*, Phuket, Thailand, 2017.

[31] J. Hedley, "jsoup: Java HTML Parser," 2017. [Online]. Available: https://jsoup.org/.

[32] R. Lertsuksakda, K. Pasupa and P. Netisopakul, "Sentiment analysis of Thai children stories on support vector machine," in *The Twentieth International Symposium on Artificial Life and Robotics 2015 (AROB 20th 2015)*, Beppu, Japan, 2015.

[33] P. Vateekul and T. Koomsubha, "A Study of Sentiment Analysis Using Deep Learning Techniques on Thai Twitter Data," in *13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Khon Kaen, Thailand, 2016.