# Classification of Tweets Related to Illegal Activities in Thai Language

Sumeth Yuenyong, Narit Hnoohom, Konlakorn Wongpatikaseree and Teerapong Pheungbun na ayutthaya
*Department of Computer Engineering*
*Faculty of Engineering, Mahidol University*
Nakhon Pathom, Thailand
{sumeth.yue, narit.hno, konlakorn.won}@mahidol.ac.th, Thirapongpdg@gmail.com

***Abstract*—This paper presents classification of tweets related to illegal activities in Thai language. The unfiltered nature of Twitter allows it to be used as platform for communication about illegal activities. The sheer number of tweets makes an automatic tweet classification needed to detect these illegal tweets. Very little had been done about this issue, especially in the Thai language. Tweets classification is more difficult that standard text classification due to their short length colloquial nature. Furthermore, the training data is imbalanced because legal tweets are very easy to find while illegal tweets of specific types are quite hard to come by. We propose a tree-like hierarchical model where each node is a full deep neural network based on convolutional LSTM architecture. In order to deal with highly imbalanced training data, tweets were classified in two stages: legal/illegal first before being classified among the illegal classes. Furthermore, ensemble classifiers were used to detect difficult illegal classes that were misclassified as legal by the first stage. Experiment result shows that this approach has significantly better performance than the baseline of using only a single network to classify among all classes in a single stage.**

***Index Terms*—tweet classification, text classification, illegal tweets, natural language processing, deep learning**

## I. Introduction

The most popular micro blogging platform - Twitter, allow users to communicate in almost real time to thousands of other users about any topic. Due to the ubiquitous nature of Twitter users and mobile devices, the messages on the Twitter platform - tweets, contain a wealth of beneficial information that can be extracted. For example, to get a picture of what is happening on the ground in a disaster-struck area, in order to be able to better prioritize recuse effort [1]. However, the very features that make Twitter useful are also what is making it a platform for communication about illegal activity, such as selling drugs, due to the fact that tweets are largely unfiltered. While stated explicitly in Twitter's policy that it should not be used for illegal activities, users collectively generate hundreds of millions of tweets each day. The sheet number of tweets makes it impractical to filter them manually. Some algorithm which can automatically classify tweets is needed. For the Thai language, there is repository on Github[1] that deals with classifying Thai tweets into toxic/non-toxic, but their paper is not yet published. To best of our knowledge there is no existing study that deals with this particular issue in any language. In Thai it is particularly challenging due to word plays and other tactics such as nickname for drugs that are single syllable words or synonyms with common words.

In order to address the problem of Thai tweet classification, we propose a machine learning model that classifies tweets into normal (class 0) and 5 illegal categories: pornography (class 1), sex toys (class 2), prostitution (class 3), drugs (class 4) and gambling (class 5). Due to the highly imbalanced training data between legal and illegal tweets, the model is hierarchical consisting of many sub models working in stages. The ideas was to distinguish between legal/illegal tweet first before deciding which of the five illegal classes an illegal tweet belongs to. Furthermore, ensemble classifiers were used to detect difficult illegal classes that were misclassified as legal in the first stage. The final model achieved sensitivity above 80% for all classes with many classes above 90%. The specificity of the class 0 was 95.56%.

## II. Related Works

Tweets are basically short pieces of text, thus tweet classification is very similar to standard text classification. The main difference is that the length of tweets is limited to only 150 words. This makes it problematic to use standard term frequency and inverse document frequency features (TF-IDF) that are standard in text classification. The short length of tweets also makes algorithmic feature engineering [2] challenging, as the metrics to evaluate the quality of features, just like TF-IDF require long pieces of text to estimate accurately. As a result, tweet classification traditionally required hand-engineered features and meta/auxiliary data beside the actual tweets themselves. One example is [3] where keywords and their synonyms were chosen manually based on the topic of interest. Meta/supplemental data such as part of speech tagging, name entity tagging and Twitter username were also part of the features. The work in [4] also used a similar approach with the addition of context words which are words immediately before and after keywords in a tweet. They also use tweet date as a feature, which is an important piece of information for the authors' objective of flagging tweets related to an earthquake event. Retweets — tweets that get tweeted again by other users was used for clustering tweets into different clusters which corresponds to different topics by [5]. In [6] the authors extracted meta data from users'

[1] https://github.com/tmu-nlp/ThaiToxicityTweetCorpus

profiles and combined it with 8 additional binary feature that can be extracted from tweets, for example, whether or not it had been retweeted. It can be seen that a combination of manual feature engineering and tweet meta/auxiliary data was needed for tweet classification.

Recently, neural network models with many layers, also known as deep neural network, had achieved state-of-the-art performance in many machine learning tasks, perhaps most famously for natural image recognition [7]. This approach is now known as deep learning [8]. The main advantage of deep learning over traditional machine learning is that no manual feature engineering is necessary to achieve good classification results. For the task of text recognition, an early paper that applied deep learning to the problem is [9]. The authors treated texts as images by extracting a 2 dimensional array from them. Each word is converted to its representation in a vector space by an embedding layer commonly known as word2vec [10]. The representation of each word corresponds to a row in the 2D array. Then image recognition neural network which consists of convolutional layers [11] that progressively builds higher level representations before some fully-connected layers (FC) that performs the actual classification from the features automatically extracted by the convolutional layers. A similar approach was used in [12] but at the character level instead of at the word level. The authors claimed higher accuracy than recognition at the word level for many common text classification datasets.

Another popular type of neural network architecture for text classification is based on recurrent structure, where the output — also know as state, of a layer at a time step, i.e., the current word or character, becomes part of the input at the next time step. The work in [13] consists of a recurrent layer whose output is the feature for the previous word, an embedding layer to encode the current word, an another recurrent layer similar to the first but for the next word instead. This last layer is fed with the text reversed at the word level. The outputs from all three layers are then concatenated together and then fed to fully connected layers for classification.

Standard recurrent layer does not perform very well when the input sequence is long, due to the effect of a word on the current state being diminished the further back the word is. Long Short Term Memory (LSTM) was invented to solve this problem [14]. The idea of LSTM is to introduce gates, which basically are modulation factors which control how much of the current input/state to remember or forget before advancing to the next time step. In [15], the authors combined convolutional layers with LSTM for text classification and achieved better results than using either convolutional layers or LSTM layers by themselves.

In this work, we draw on these results for general text classification using deep neural network architectures and applied them to the task of tweet classification, which have additional challenges. Tweets are written in colloquial manner which means that they contain slangs, typos, emoji, and special characters which must either be corrected and/or eliminated before actual text processing can take place. In addition, illegal tweets are hard to find as they do not get retweeted and the user accounts associated with them tend to get abandon often. Finally people employ many tricks to try to hide the illegal nature of their tweets, such as word plays or substitution. The training data was also imbalanced between normal tweet and illegal tweets of certain categories that we wanted to classify. The proposed method addresses these challenges.

## III. Proposed Method

The proposed model is hierarchical as shown in Fig. 1. The motivation for this design is the following. Due to the fact that normal legal tweets are practically limitless, the training data will always be biased toward class 0 (legal). One could just pick the same number of legal tweets as the illegal tweets available to make the legal/illegal sets balanced. However, this would not reflect the actual size of the classes where class 0 is much bigger than the rest, due to the fact that anything not related to the five illegal classes are considered legal. Thus one would want to have the training examples for class 0 cover a broad range of legal tweet topics, but this creates the problem of imbalanced training data. Our early attempt at tweet classification using only a single network to classify all six classes resulted in a classifier that is biased toward class 0, as shown by the result in Section IV-B.

In order to solve this problem, we decided to attack it in two stages using two networks. The first network (network 1) was trained as a binary classifier where the classes are legal/illegal. Pooling all illegal tweets into the same class helps alleviate the problem of legal/illegal imbalanced to a certain degree, as well as making the job of network 1 easier. Any tweet classified as illegal by network 1 would then enter the second network (network 2), which was trained to classify only among the illegal classes (class 1-5). Using this approach, the training data for network 2 is relatively balanced. The network architecture at this point can be seen in Fig. 2. The left branch of network 1 is considered legal tweets, while the right branch is considered illegal and then passed to network 2 to classify among the five illegal classes. The "with user_description" attached to network 1 in Fig. 1 means that it uses the Twitter user description as well as the actual tweet text. We observe from reading many of them that user descriptions can be very indicative that an account is likely to be engaged in illegal activities. However it can cause confusion when trying to decide between the illegal classes themselves as one account may be engaged in more than one kind of illegal activities. Thus we decided to incorporate user description only in network 1 and not for network 2.

Testing tweet classification using the model in Fig. 2, the result (Section IV-C) shows that class 1 (pornography) and 4 (drugs) are especially problematic as they are often classified as class 0. These two classes are particularly hard to distinguish from legal tweets due to the way people avoid using explicit words and instead use words which are perfectly normal in other contexts. For example using nicknames that are synonyms with common words for different drugs or women names for types of pornography. These users can be

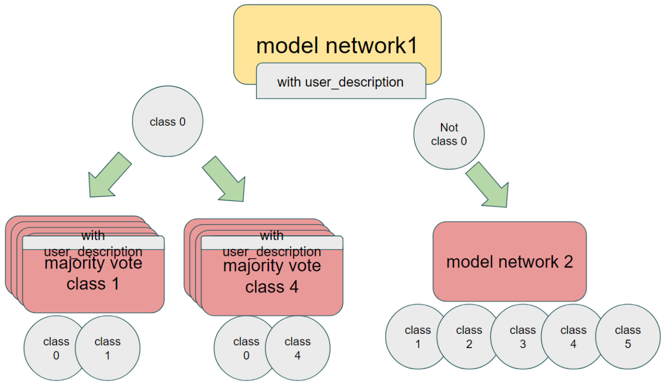

Fig. 1. The full hierarchical model. Network 1 decides between just legal and illegal tweets, while network 2 classify among the five illegal classes. The left branch of network 1 leads down to two separate sets of boosting ensemble classifiers whose jobs are to catch classes 1 and 4 respective that were misclassified as legal by network 1.

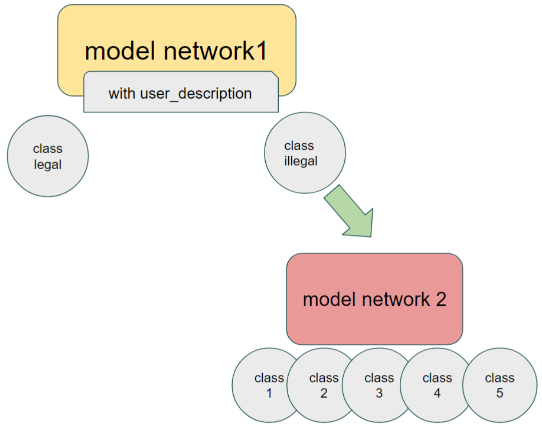

Fig. 2. The hierarchical model where tweets are classified in two stages. Network 1 decides between just legal and illegal tweets, while network 2 classify among the five illegal classes.

very creative with wording in order to get the message across to their target audience while hiding the true intension from others. Unfortunately due to the explicit nature of the content we cannot show examples here. We decided to deal with this problem as follows. Two sets of boosting ensemble classifiers [16] were trained where each individual models was trained with all of the available training set for class 1 or class 4 and a random subset of class 0 such that the classes are balanced. That is, each model in the ensembles was trained with a different subset of class 0. This is such that the ensembles as a whole had seen a significant portion of class 0 without each individual classifier being biased toward the class. The size $N$ of the ensembles was 7. The ensembles decide the classes by majority vote.

Each individual node (network 1, network 2, and each of the network in the two ensembles) in the model in Fig. 1 are actually quite sophisticated neural network models on their own. The internal structure of each node (referred to from here on as "sub model") is shown in Fig. 3 and Fig. 4. The only

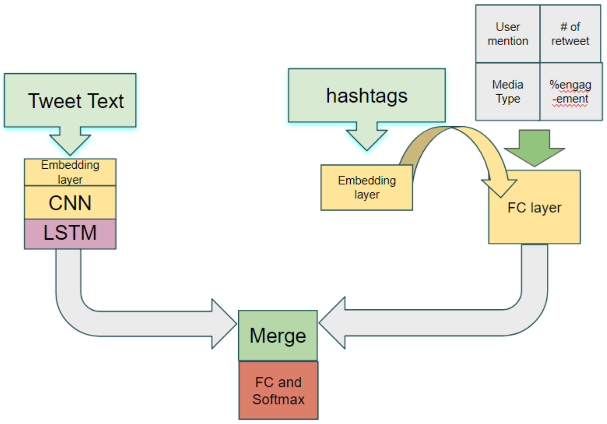

Fig. 3. Architecture of the sub model with no user description.

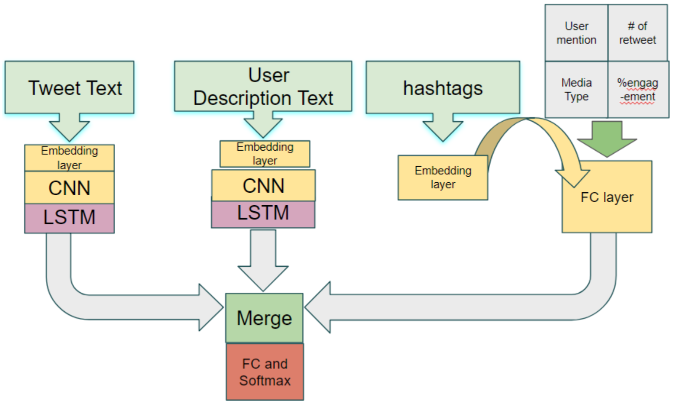

Fig. 4. Architecture of the sub model that incorporates user description text.

difference between these two figures is the presence/absence of the "user description text" branch. The structure of the sub model is inspired by [15] that used both convolutional layers and LSTM layers for text classification. The idea of the this sub model architecture is to have multiple paths for feature extraction from different sources: tweet text, user description text, hashtags and meta data. For the sub model that uses the user description (Fig. 4), the left two paths are respectively for the tweet text itself and the user description of the account that generated the tweet. They have the same sequence of layers: word embedding → 1D convolution → 1D max pooling → LSTM → dropout. The details of each layer is as follows:

- word embedding: embedding length = 32, top words = 15607, max text length = 150 words. They are trained with other layers during training.
- 1D convolution: kernel length = 3, same padding, relu activation function
- 1D max pooling: kernel length = 2
- LSTM: number of units = 120 for the tweet text branch and 100 for the user description branch
- Dropout: dropout probability = 40% for the tweet text branch and 20% for the description branch.

The next branch to the right is for the hashtags in the tweet.

Because hashtags are individual words with no semantical relationship between them, they simply have to be converted into vector representation by a word embedding layer. The embedding layer for the hashtag branch has the same hyperparameters as the ones for the tweet text and description text branches. Finally, the branch on the far right is for the meta data of tweets: user mention — the number times a tweet was mentioned to by other users, number of retweets, media type (no media, image, video), and percent engagement. The percent engagement of a tweet is defined by

$$\frac{\text{retweets}}{(\text{user follower count} + 1)(\text{user status count} + 1)} \times 100,$$

where *user status count* is the total number of tweets (including retweets) generated by the user. Since the number of retweets is included in the denominator, engagement is always less than 1. The other three meta data are also single numbers which can be easily extracted using Twitter's API. These 4 auxiliary features are normalized, before getting concatenated with the vector representation from the hashtag branch, then fed into a fully-connected (FC) layer on the far right of the sub model.

The features from all the four branches (or three if the user description is not used) are flatten where necessary and then concatenated together and fed into the output layers of the sub model. The output layers consist of one FC layer followed by a softmax layer. This concludes the architecture of the sub model.

### A. Making Prediction Using the Full Model

One can make prediction on new unseen tweets using the full model in Fig. 8 as follows: a new tweet passes through feature extraction and first fed into the network 1. If the prediction of network 1 is not class 0, the tweet is fed into network 2 whose prediction is the final classification results. If network 1 predicts class 0, the tweet must be processed by both ensembles. Since class 0 is much more likely than class 1 or 4, if either ensemble predicts class 0, that is taken to be the final classification result. If both ensembles predict non-normal class, the one with the bigger majority is taken to be the final classification result.

## IV. Experiment Result

### A. Preprocessing

The training set contains the following number of tweets: class 0 = 76,500, class 1 = 1,156, class 2 = 808, class 3 = 1,922, class 4 = 464 and class 5 = 1,407. We cleaned the tweets using regular expressions for text find and replace. Examples of preprocessing rules include: eliminate words that begins with #, replace common slangs with proper words, eliminate common spelling errors, eliminate emoji's such as T_T, convert dates and phone numbers into the same formats. Next, we processed the tweets with Deepcut[2] to extract individual words. The list of words in the training set was constructed and then sorted from the most common word to the least common. Tweets that are shorter than 150 words were padded.

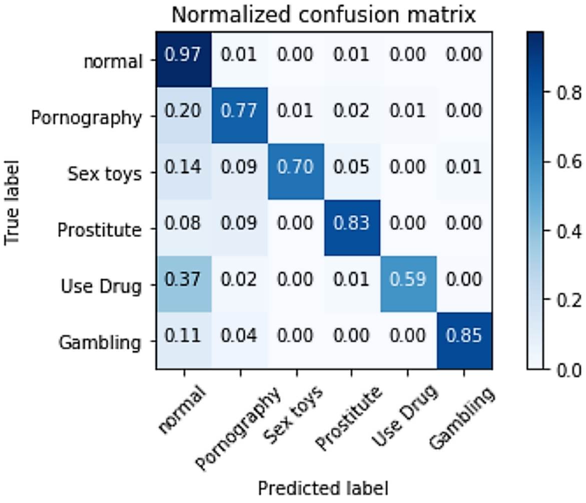

Fig. 5. The normalized confusion matrix of the baseline experiment using only a single network to classify all six classes.

TABLE I
ACCURACY, SENSITIVITY AND SPECIFICITY OF EACH CLASS IN THE BASELINE EXPERIMENT.

| | class 0 | class 1 | class 2 | class 3 | class 4 | class 5 |
|---|---|---|---|---|---|---|
| accuracy | 0.9711 | 0.9834 | 0.9970 | 0.9858 | 0.9976 | 0.9973 |
| sensitivity | **0.9798** | **0.7623** | **0.7069** | **0.8298** | **0.5941** | **0.8495** |
| specificity | **0.8567** | 0.9866 | 0.9999 | 0.9895 | 0.9999 | 0.9999 |

### B. Baseline Result

We first established a baseline result by using only a single network to classify all six classes. The test set contains the following number of tweets: class 0 = 13,500, class 1 = 204, class 2 = 143, class 3 = 339, class 4 = 82 and class 5 = 248. The normalized confusion matrix and performance measures are shown in Fig. 5 and Table I respectively. It can be seen that network is biased to class 0 due to the large amount of legal tweets in the training data. Most classification error was caused by incorrectly classifying class 1-5 as class 0, as expected for this kind of training data imbalance. Additionally, due to the test data also being imbalanced, the accuracy is not a good indicator of actual performance because of the large number of true positives for class 0 and true negatives for classes 1-5. The values that better reflect the performance and can be used to compare against later models in this case are the sensitivities of classes 1-5 and the sensitivity and specificity of class 0, which are bold in the table.

### C. Result of Using Hierarchical Model with Two Sub Models.

The first improvement was to use the model with two stages as shown in Fig. 2. As described above, the first model was trained as a binary classifier between class 0 and all the other classes, while the second model was trained to classify only among the classes 1-5. We present the result of this experiment

[2] https://github.com/rkcosmos/deepcut

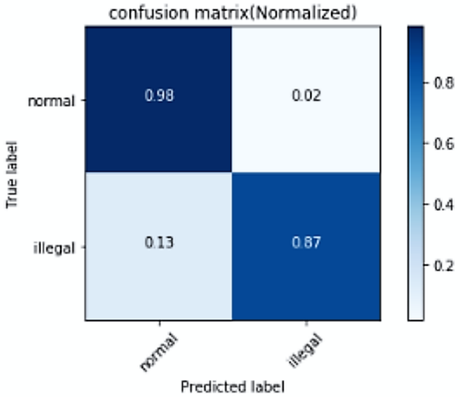

Fig. 6. The normalized confusion matrix of the experiment where the network in Fig. 2 was used as the classifier. Network 1 only (binary classification).

TABLE II
ACCURACY, SENSITIVITY AND SPECIFICITY OF EACH CLASS. WHERE THE NETWORK IN FIG. 2 WAS USED AS THE CLASSIFIER. NETWORK 1 ONLY (BINARY CLASSIFICATION).

| | class 0 | not class 0 |
|---|---|---|
| accuracy | 0.9723 | 0.9723 |
| sensitivity | **0.9495** | **0.8701** |
| specificity | **0.8701** | 0.9800 |

in two parts: the first is the binary classification result of network 1 only in Fig. 6 and Table II. It can be seen that the sensitivity to the illegal classes was improved compared to the baseline, and so was the specificity of class 0. This was expected since the training data is more balanced compared to the baseline, which made network 1 less biased toward class 0. The sensitivity of class 0 was reduced, which was also expected since the network was less likely to just predict class 0 compared to the baseline.

Next, we present the result when both network 1 and 2 are considered together as a six-class classifier. The confusion matrix is shown in Fig. 7 and the performance measures are shown in Table III. It can be seen that sensitivities of classes 1-5 are generally increased compared to the baseline, with the exception of class 1 and 4, which were often misclassified as class 0.

### *D. Result of the Final Full Model*

From the previous section it can be seen that using hierarchical model improved performance, but classes 1 and 4 were problematic. They have the lowest sensitivity in Table III. As stated earlier, we observed that people tend to be particularly creative in talking about these things without using explicit words, which makes it hard to distinguish them from normal legal tweets. Furthermore, class 4 has the lowest number of training examples out of all six classes. In order to improve the performance for classes 1 and 4, we observed that when misclassification happen it is almost always class 0. Thus we extended the model in Fig. 2 by adding two ensembles on the left branch of network 1 that was classified as legal to try to “catch” classes 1 and 4 that escaped network 1. All of the sub-model in the ensembles incorporate user description. The confusion matrix is shown in Fig. 8 and the performance measures are shown in Table IV. It can be seen that the sensitivity of all illegal classes generally improved compared to the previous experiment, except for class 2 where it’s just slightly lower, and significantly better compared to the baseline. The accuracy and sensitivity of class 0 were lower than using just two networks with no ensemble, which suggest that more legal tweets were misclassified as illegal by the full model than the model without the ensemble. Overall however, the sensitivity to the illegal classes are high, which is good for this use case where the system is expected to act as filter, flagging potentially illegal tweets for humans to confirm. One is willing to sacrifice some accuracy for the legal class in order to be able to detect more illegal tweets.

TABLE III
ACCURACY, SENSITIVITY AND SPECIFICITY OF EACH CLASS. WHERE THE NETWORK IN FIG. 2 WAS USED AS THE CLASSIFIER. BOTH NETWORKS TOGETHER.

| | class 0 | class 1 | class 2 | class 3 | class 4 | class 5 |
|---|---|---|---|---|---|---|
| accuracy | 0.9868 | 0.9914 | 0.9969 | 0.9965 | 0.9968 | 0.9978 |
| sensitivity | **0.9955** | **0.6569** | **0.8182** | **0.9115** | **0.5976** | **0.9032** |
| specificity | **0.8780** | 0.9962 | 0.9987 | 0.9985 | 0.9990 | 0.9994 |

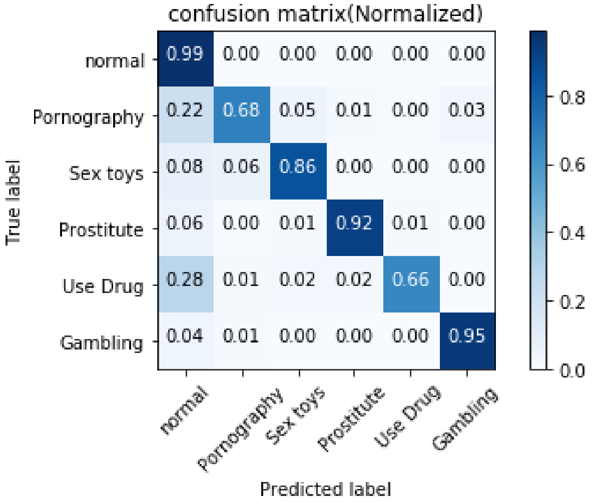

Fig. 7. The normalized confusion matrix of the experiment where the network in Fig. 2 was used as the classifier. Both networks together.

TABLE IV
ACCURACY, SENSITIVITY AND SPECIFICITY OF EACH CLASS. WHERE THE NETWORK IN FIG. 1 WAS USED AS THE CLASSIFIER. ALL NETWORKS TOGETHER.

| | class 0 | class 1 | class 2 | class 3 | class 4 | class 5 |
|---|---|---|---|---|---|---|
| accuracy | 0.9645 | 0.9841 | 0.9948 | 0.9962 | 0.9836 | 0.9972 |
| sensitivity | **0.9651** | **0.8088** | **0.8601** | **0.9233** | **0.8902** | **0.9476** |
| specificity | **0.9557** | 0.9866 | 0.9962 | 0.9980 | 0.9841 | 0.9980 |

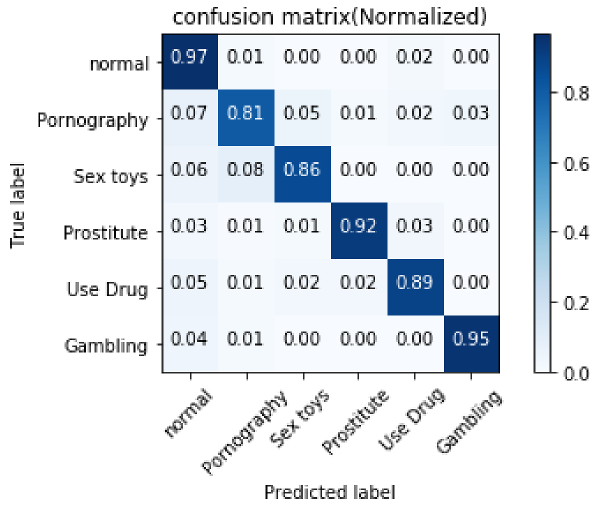

Fig. 8. The normalized confusion matrix of the experiment where the network in Fig. 1 was used as the classifier. All networks together.

## V. Conclusion

In this paper, we presented classification of tweets related to illegal activities in Thai language. The Twitter platform offer great freedom that allows anyone to communicate with thousand of followers. However the sheer number of tweets and their unfiltered nature also enable people to use Twitter to communicate about illegal activities. Tweet classification shares a lot with standard text classification, with added difficulty due to their short length and colloquial nature. Classification of tweets in Thai language had not received much attention, especially regarding communication about illegal activities. We presented a tree-like hierarchical model where tweets are first classified as legal or illegal by the first network, then those that were classified as illegal enter the second network that classifies among the illegal classes. The motivation of this approach was to address the problem of highly imbalanced training data, because illegal tweets of specific classes are hard to come by while legal tweets are very easy to find. This approach was enhanced further by using ensemble classifiers to catch illegal tweets that were misclassified by the first network as legal. It was highly beneficial for classes that are particularly hard to differentiate from legal due to word play and other tactics by users. The final result as indicated by the sensitivity of each class as well as the specificity of class 0 were significantly better than the baseline using only a single model to classify all six classes in one stage.

### A. Future Work

Future work includes applying the ensemble classifiers to all of the illegal classes and not just class 1 and class 4. Ablation study of the model parameters is also needed to determine the best configuration of the layers and their parameters. Deployment of the trained model for continuous monitoring of the interested subset of Thai tweet is also under way. Finally, it may be worth investigating Bayesian networks, such as the model also output how confident it is in a prediction as well as the prediction result.

## Acknowledgment

This work was financially supported by a research grant from Mahidol University through the National Research Council of Thailand (NRCT) under the project named “A system for forensic data analysis of user interaction with social media”, and the Department of Computer Engineering, Faculty of Engineering, Mahidol University, Thailand.

## References

[1] S. Kumar, G. Barbier, M. A. Abbasi, and H. Liu, “Tweettracker: An analysis tool for humanitarian and disaster relief.” in *ICWSM*, 2011.

[2] G. Forman, “An extensive empirical study of feature selection metrics for text classification,” *Journal of machine learning research*, vol. 3, no. Mar, pp. 1289–1305, 2003.

[3] R. Batool, A. M. Khattak, J. Maqbool, and S. Lee, “Precise tweet classification and sentiment analysis,” in *Computer and Information Science (ICIS), 2013 IEEE/ACIS 12th International Conference on*. IEEE, 2013, pp. 461–466.

[4] T. Sakaki, M. Okazaki, and Y. Matsuo, “Tweet analysis for real-time event detection and earthquake reporting system development,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 919–931, 2013.

[5] F. Toriumi and S. Baba, “Real-time tweet classification in disaster situation,” in *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 117–118.

[6] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas, “Short text classification in twitter to improve information filtering,” in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2010, pp. 841–842.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[8] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, p. 436, 2015.

[9] Y. Kim, “Convolutional neural networks for sentence classification,” *arXiv preprint arXiv:1408.5882*, 2014.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[12] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” in *Advances in neural information processing systems*, 2015, pp. 649–657.

[13] S. Lai, L. Xu, K. Liu, and J. Zhao, “Recurrent convolutional neural networks for text classification.” in *AAAI*, vol. 333, 2015, pp. 2267–2273.

[14] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] C. Zhou, C. Sun, Z. Liu, and F. Lau, “A c-lstm neural network for text classification,” *arXiv preprint arXiv:1511.08630*, 2015.

[16] R. E. Schapire, “The boosting approach to machine learning: An overview,” in *Nonlinear estimation and classification*. Springer, 2003, pp. 149–171.