HANCaps: A Two-Channel Deep Learning Framework for Fake News Detection in Thai

Krishanu Maity[1], Shaubhik Bhattacharya[1], Salisa Phosit[2], Sawarod Kongsamlit[2], Sriparna Saha[1], and Kitsuchart Pasupa[2,3(✉)][0000−0001−8359−9888]

[1] Department of Computer Science and Engineering,
Indian Institute of Technology Patna, Patna 801103, India
{krishanu 2021cs19, shaubhik 2111cs19, sriparna}@iitp.ac.in
[2] School of Information Technology,
King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand
{63070242, 63070245, kitsuchart}@it.kmitl.ac.th
[3] AI Governance Clinic, Electronic Transactions Development Agency (ETDA),
Bangkok 10310, Thailand

Abstract. The rapid advancement of internet technology, widespread smartphone usage, and the rise of social media platforms have drastically transformed the global communication landscape. These developments have resulted in both positive and negative consequences. On the one hand, they have facilitated the dissemination of information, connecting individuals across vast distances and fostering diverse perspectives. On the other hand, the ease of access to online platforms has led to the proliferation of misinformation, often in the form of fake news. Detecting and combatting fake news has become crucial to mitigate its adverse effects on society. This paper presents an investigation into fake news detection in the Thai language. It addresses current limitations in this domain by proposing a novel two-channel deep learning model named HANCaps, which integrates BERT and FastText embeddings with a hierarchical attention network and capsule network. The HANCaps model utilizes the BERT language model as one channel input, while the other channel incorporates pre-trained FastText embeddings. The proposed model undergoes evaluation using a benchmark Thai fake news dataset, and extensive experimentation demonstrates that HANCaps outperforms state-of-the-art methods by up to 3.28% in terms of F1 score, showcasing its superior performance.

Keywords: Fake News · Thai · Hierarchical Attention Network · Capsule Network

1 Introduction

With the advent of the internet, the world has witnessed a revolutionary shift in communication patterns. The proliferation of smartphones and the ubiquity of social media platforms have played pivotal roles in shaping the way information is disseminated and consumed [16]. This digital transformation has empowered individuals to participate actively in the news ecosystem, enabling them to share, comment, and contribute to online content. According to Statista, as of January 2023, there are approximately 5.18 billion active internet users worldwide, with 4.8 billion being active social media users[4]. This exponential growth in online connectivity has resulted in an unprecedented amount of information being shared and accessed globally. Consequently, the flow of information has become more decentralized, challenging traditional gatekeeping mechanisms and introducing a vast array of perspectives. As of August 2018, approximately two-thirds (68%) of Americans rely on social media platforms as their primary source of news[5].

The democratization of information has inadvertently led to the spread of misinformation and fake news. Fake news refers to deliberately fabricated or manipulated information presented as factual news [1]. It often aims to mislead, manipulate public opinion, or achieve various socio-political objectives. The consequences of fake news can be severe, ranging from undermining trust in legitimate sources of information to influencing elections, exacerbating social tensions, and even inciting violence.

Significant efforts have been made in the research community to address the challenge of fake news detection. Many studies [11,3,12,15] have focused on the English language, benefiting from large datasets and resources. Various approaches, including machine learning techniques, natural language processing (NLP) algorithms, and deep learning models, have been applied to analyze textual content and identify patterns indicative of fake news. Several existing works have achieved promising results in English fake news detection.

While extensive research has been conducted on fake news detection in English, the study of low-resource languages such as Thai remains limited, as described in the next section. This presents a significant challenge, as these languages often lack sufficient labeled data and specialized resources. Consequently, existing models may not perform optimally when applied to the Thai language. Addressing this limitation requires the development of robust and tailored approaches that consider the linguistic characteristics and contextual nuances specific to Thai.

This paper introduces a novel two-channel deep learning framework called HANCaps, which effectively detects fake news in the Thai language. HANCaps harnesses the power of BERT, a pre-trained language model, along with hierarchical attention network (HAN) and capsule networks, to capture the hierarchical relationships inherent in textual features. The model's first channel utilizes the BERT language model, while the second channel incorporates pre-trained FastText embeddings. The proposed framework is evaluated using a benchmark Thai fake news dataset, and extensive experimentation demonstrates HANCaps' significant superiority over state-of-the-art (SOTA) methods. By leveraging the diverse tags available in the LimeSoda dataset [10] and contextual cues present

[4] https://www.statista.com/statistics/617136/digital-population-worldwide/
[5] https://www.pewresearch.org/journalism/2018/09/10/news-use-across-social-media-platforms-2018/

in the Thai language, our model strives to enhance the accuracy and effectiveness of fake news detection, contributing to advancing this crucial field.

2 Related Works

The first study related to fake news in the Thai language is [2]. They con- ducted a detection of misinformation from Twitter texts by extracting tweet features and testing them using conventional machine learning techniques, i.e., Support Vector Machines (SVM), Na¨ıve Bayes (NB), and Multilayer Percep- tron (MLP). However, the content of the texts was not considered in this study. Following that, there was an attempt to detect unreliable medical articles on Thai websites [14]. They extracted article features from websites in conjunction with content features, TF-IDF, and Bag-of-Words, extracted from some selecting keywords. These features were then tested using conventional machine Learning techniques, including XGBoost, Decision Trees, SVM, Logistic Regression, and k-Nearest Neighbors (kNN). Kaothanthong et al. [6] classified the headline types of articles as clickbait or non-clickbait, as clickbait articles tend to be associated with fake news. They introduced the use of Headline2Vec, a feature derived from the last layer of a Convolutional Neural Network (CNN), and compared it with basic features such as n-Grams and TF-IDF. The features were tested using SVM, NB, and MLP. Meesad [8] presented a framework for detecting fake news, classifying news into three categories: real, fake, and suspicious. They utilized NLP techniques to extract features from the content. Experimental results showed that Long Short-Term Memory (LSTM) achieved the best performance among the other algorithms.

Due to the COVID-19 pandemic, numerous instances of fake news have emerged. In response, Mookdarsanit and Mookdarsanit [9] attempted to develop a system for detecting COVID-19-related misinformation in the Thai language. Since there is a lack of fake Thai-language news datasets specifically related to COVID-19, they adapted a model from the COVID-19 news open datasets, which were translated into Thai. The model was tested using data crawled from Thai websites. Additionally, they employed the feature-shifting technique to increase the number of Thai-language samples for model training. Experimental results demonstrated that ULMFiT outperformed other deep learning models, e.g., BERT and GPT. Subsequently, Payoungkhamdee et al. [10] created a dataset called “LimeSoda”, focusing on fake news in the health domain. They evaluated the dataset using deep learning models, including Bidirectional LSTM (Bi-LSTM) with attention, BERT with a linear model, and WangChanBERTa with a linear model. Among them, BERT combined with a linear model yielded the best results. Furthermore, they attempted to understand how the models made decisions by analyzing token-level annotations and attention weights in Recurrent Neural Network-based models or using an embedding layer for transformer-based models. The findings suggest that while machine learning models provide explanations that differ somewhat from human judgments, there are common patterns in how humans and machines categorize words, indicating shared lexical interpretations.

3 Methodology

This section presents the methodology used for detecting fake news in Thai. We introduce a two-channel HAN-based deep neural network model, *HANCaps*, specifically designed for this purpose.

3.1 Proposed *HANCaps* Model

The proposed *HANCaps* model incorporates two distinct channels to capture different aspects of input sentences. The overall architecture of our proposed *HANCaps* model is illustrated in Figure 1. The first channel employs BERT [4] followed by HAN and capsule network, while the second channel uses pre-trained FastText [5] embedding followed by HAN and capsule network. Given K input sentences where a sentence $S = \{w_1, w_2, \ldots, w_n\}$ comprising n words, both channels process the input using a series of operations as follow.

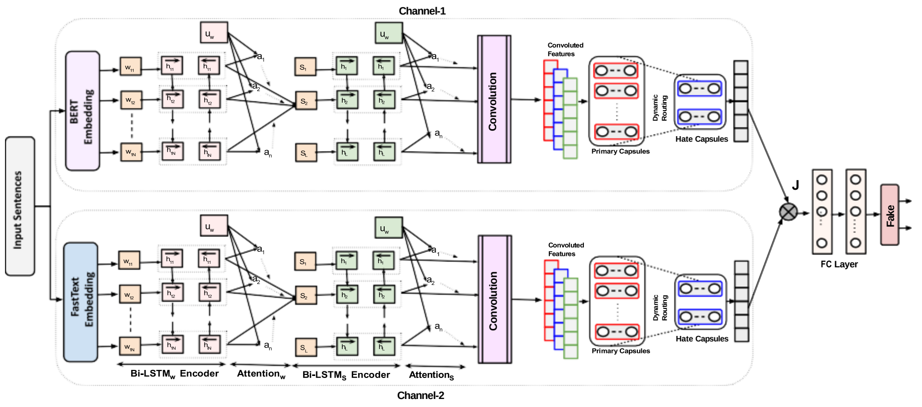

Fig. 1. *HANCaps* architecture

Channel-1

BERT Embedding The BERT model processes the input sentence S and generates a sequence output $E_B \in \mathbb{R}^{n \times d}$ of dimensions *max sequence_length* $\times$ 768. The output E_B is then fed through the HAN operates by incorporating attention mechanisms at different levels of the document hierarchy, allowing it to capture local and global dependencies.

Hierarchical Attention Network We enhanced our model configuration to a hierarchical form to represent context-rich data samples. First, we used Bi-LSTM as a word-level encoder to compute sentence representation.

(i) **Bi-LSTM:** To enhance the contextual representation of the input sequence, we have integrated a Bi-LSTM layer that takes the embedding vector E_B generated by the BERT model as its input. This layer can capture contextual information in both forward and backward directions by processing the input sequence in both directions. At each time step, the hidden state h_t of the Bi-LSTM is obtained by concatenating the hidden state of the forward LSTM $\overrightarrow{h_t}$ and the hidden state of the backward LSTM $\overleftarrow{h_t}$. Consequently, the output of the Bi-LSTM layer is a sequence of hidden states H_e that includes all the hidden states of the input sequence. This representation can be expressed as $H_e = [h_1, h_2, h_3, \ldots, h_n]$.

(ii) **Attention Layer:** We incorporate a word attention layer after the Bi-LSTM layer, which allows the model to focus selectively on important words in the sentence. Formally, given the hidden state h_i of the Bi-LSTM at time step i and the weight vector u_a, the attention score a_i for the i-th word is computed as

$$a_i = \frac{\exp(u_a^T h_i)}{\sum_{j=1}^{n} \exp(u_a^T h_j)}, \tag{1}$$

where n is the length of the input sentence. The word-label (wl) sentence representation S^{wl} is then obtained as the weighted sum of the Bi-LSTM hidden states multiplied by attention weight,

$$S^{wl} = \sum_{i=1}^{n} a_i h_i. \tag{2}$$

Thus, we obtained $E_X^{wl} = [S_1^{wl}, S_2^{wl}, S_3^{wl}, \ldots, S_L^{wl}]$ for an input post X.

(iii) **Sentence-label Encoder:** Next, we apply the Bi-LSTM in the same way as a sentence-label encoder where input is $[S_1^{wl}, S_2^{wl}, S_3^{wl}, \ldots, S_L^{wl}]$. The output generated by Bi-LSTM passes through the attention layer to get the attention score a_i in the sentence label. Here, we simply multiply this attention score with Bi-LSTM hidden states output to get the sentence-label representation E_X^{sl} without performing the weighted sum operation as in the next layer, we apply CNN, which requires 2D input.

CNN [7] Let the output feature map of this CNN layer be denoted as F. The element-wise dot product is performed between the E_X^{sl} and different filters c_i of size $h \times d$ in the CNN layer. This produces the feature map f_i corresponding to a particular n-gram's presence in the input sentence. The dimension of F is given by $(n - h + 1) \times k$, where h is the filter size and k is the number of filters used. Therefore, F is a collection of k feature maps obtained by sliding the filters over the entire input sequence,

$$F = [\mathbf{F}_1, \mathbf{F}_2, \mathbf{F}_3, \ldots, \mathbf{F}_k]. \tag{3}$$

In our proposed *HANCaps* model, instead of applying a pooling operation to the feature maps, we have used a capsule network [13] to retain the special features often lost during pooling.

Primary Capsule Layer The capsule network's first layer combines the convolutional features produced by the CNN and creates primary capsules that represent each element in the feature maps using a group of neurons, thereby preserving local word order and semantic representations of words as instantiation parameters rather than scalar values. To generate a set of capsules, denoted as $p_i \in \mathrm{R}^l$, a kernel, denoted as K_i, is applied over the feature maps F, where l is the number of neurons in a capsule. Within the main capsule layer, a channel C_i consisting of a collection of capsules p_i is defined as follows:

$$C_i = \iota(F * K_i + b_i), \tag{4}$$

where ι refers to a non-linear activation function known as the squashing function, and b_i is a bias term.

Dynamic Routing between Capsules In this layer, each capsule in the previous layer sends its output vector to all capsules in the next layer. The coupling coefficient between capsule i in the previous layer and capsule j in the next layer, denoted as $c_{i,j}$, is determined by a softmax function over all capsules in the next layer, and is calculated as follows:

$$p_{i,j} = \frac{\exp(b_{i,j})}{\sum_k \exp(b_{i,k})}, \tag{5}$$

where $b_{i,j}$ is the log prior probability that capsule i should be coupled with capsule j. The output of each capsule in the next layer is then calculated as a weighted sum of the predictions from all capsules in the previous layer, weighted by the coupling coefficients:

$$s_j = \sum_i p_{i,j}\hat{a}_{j|i} \text{ and } \hat{a}_{j|i} = W_{ij}a_i, \tag{6}$$

where $\hat{a}_{j|i}$ is the prediction vector of capsule i for the presence of an entity of class j and is defined as the dot product between the output vector of capsule i and a transformation matrix $W_{i,j}$, which learns to represent the instantiation parameters between capsule i and class j. Finally, the output vector of each capsule j is passed through a non-linear squashing function to ensure that its length is between 0 and 1.

Classification Capsule Layer The final layer of the proposed capsule network is the classification capsule layer, which consists of k capsules with 16-dimensional instantiated parameters. In this layer, each capsule is dedicated to identifying a specific type of hate speech. The hate capsules are generated by routing the previous layer's output to the final layer. The output of the hate capsule layer is a flattened 1D vector of dimension ($k \times 16$). This vector is concatenated with the features generated by Channel-2.

Channel-2 We keep the same architecture as mentioned in Channel-1. The only distinction lies in the choice of embedding generation strategy. While Channel-1 employed BERT for generating input token embeddings, in Channel-2, we utilized FastText for this purpose. This allowed us to examine the impact of different embedding techniques within the same framework.

Fully Connected (FC) Layers The concatenated outputs of Channel-1 and Channel-2 form a combined representation, denoted as J, for the input sentence X. Subsequently, this representation J is fed into FC layers, consisting of FC_1 with 200 neurons and FC_2 with 100 neurons. Finally, a softmax output layer is applied to predict the probabilities of the sample belonging to the target classes.

3.2 Loss Function

For the purpose of parameter optimization and back-propagation of loss, the categorical cross-entropy loss function $L_{CE}(\hat{Y}, Y)$ has been utilized in this study. It is defined as follows:

$$L_{CE}(\hat{Y}, Y) = -\frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{N} Y_i^j log(\hat{Y}_i^j), \tag{7}$$

where $\hat{Y}_i^j$ represents the predicted label and Y_i^j represents the true label. The term N denotes the number of tweets in the dataset, while M represents the number of classes.

4 Experimental Setup

4.1 LimeSoda Dataset

In this research, the LimeSoda dataset [10] was utilized for the purpose of fake news detection in the Thai language. This dataset comprises a total of 7,191 documents sourced from various platforms such as official healthcare departments, official news sources, article websites, web boards, e-commerce sites, and social media platforms, all within the healthcare domain. Each document in the dataset is assigned one of the following classifications: fact, fake, or undefined. Notably, the dataset also includes token-level annotations that facilitate the validation of classifier decisions. These annotations encompass five high-level tags: misleading headline, imposter, fabrication, false connection, and misleading content.

To better utilize the tags mentioned in [10], we have created a zipped sentence where each word is succeeded by its corresponding tag as mentioned in Figure 2. To investigate the impact of incorporating tags in the fake news detection task, we conducted experiments using two input variations: input sentence only and input sentence with tags (referred to as '+Tags'). By comparing these two settings, we aimed to demonstrate the influence of tag inclusion on the effectiveness of fake news detection.

Input Sentence	‘โค’, ‘วิด’, ‘19’, ‘แพ▪’, ‘กระเทียม’, ‘ตม▪ ’, ‘แล▪ว’, ‘จิบ’, ‘บ▪อยๆ’
Translation	COVID-19 is susceptible to garlic. Boil it and drink it frequently.
Tags	‘Fb-Refer’, ‘Fb-Refer’, ‘Fb-Refer’, ‘Fb-Refer’, ‘Fb-Refer’, ‘’, ‘’, ‘’, ‘’
Input+Tags	โค Fb-Refer วิด Fb-Refer 19 Fb-Refer แพ▪ Fb-Refer กระเทียม Fb-Refer ตม▪ ‘’ แล▪ว ‘’ จิบ ‘’ บอยๆ’

Fig. 2. A sentence from a fake news sample in the LimeSoda dataset is provided. Here, the “Fb-Refer” tag, representing fabrication, signifies the presence of a regular prelude leading to a fabricated reference.

4.2 Experimental Settings

All our experiments are performed on a machine with an AMD EPYC 7552 48-Core Processor, 512 GB DDR4 RAM, and 5x Nvidia Ampere A100 GPUs totaling 200 GB of graphics memory. To prepare for the experiments, we partitioned the dataset into testing, validation, and training sets, with ratios of 10%, 10%, and 80%, respectively. The models were trained ten times with different random splits to ensure robustness, and the average performance was reported. Various network configurations were tested, and we achieved the best results with a batch size of 16, a learning rate of $1e{-}4$, and 30 epochs. All models were implemented using Scikit-Learn and PyTorch.

In the baseline setup, we included four commonly used machine learning models: Na¨ıve Bayes, SVM, and Random Forest, using various embedding techniques. For machine learning baselines, we utilized the 768-dimensional pooled output of multilingual BERT (mBERT), which was pre-trained in 104 different languages, including Thai. For FastText embedding, we employed a pre-trained Thai FastText model to extract the embedding of each token and computed the average to obtain a 300-dimensional vector representing the entire sentence.

We established various variants of single-channel and double-channel deep learning baselines by varying the input embedding models followed by different deep learning models such as CNN, Bi-LSTM, HAN, and Capsule network. In the case of single-channel baselines, we first passed the input tweet through BERT or FastText to generate a 2D embedding matrix (E_m) of dimension ($max\ sequence\ length \times d$), where $d = 300$ for FastText and $d = 768$ for BERT. We then passed this E_m through different deep learning models as follows:

(i) HAN: The input embedding E_m is passed through word-label encoder (Bi-LSTM$_w$) followed by a sentence-label encoder (Bi-LSTM$_s$). The weighted sum of the Bi-$LSTM_s$ hidden states multiplied by attention weight is fed into an FC layer (100 neurons) followed by a softmax layer for prediction.

(ii) HAN+Capsule: The input embedding E_m was fed into HAN. The Bi-LSTM$_s$ hidden states output multiplied by attention weight is fed into a 1D CNN with 64 window size two filters. The convoluted feature was then transferred via the capsule network, and the hatred capsule layer’s output was flattened and routed through an FC layer. Finally, for the final prediction, a softmax layer was used.

(iii) HAN+CNN: Here, E_m was passed through HAN followed by a 1D CNN with 64 filters of window size 2. We then performed Average Pooling on convoluted features followed by a softmax output layer.
(iv) Bi-LSTM: Input embedding E_m went through a Bi-LSTM layer with 128 hidden states, followed by an FC layer with 100 neurons, and concluded with a softmax layer for the final prediction.

5 Results and Discussion

Table 1 showcases evaluation outcomes for our model, *HANCaps*, and baselines regarding accuracy, precision, recall, and macro F1 score, yielding the subsequent insights: (i) SVM consistently outperforms the other machine learning baselines in terms of F1 score, achieving the best F1 score of 73.89% when combined with both BERT+Fasttext embeddings. (ii) Our proposed model *HANCaps* significantly outperforms the best machine learning baseline (BERT+Fasttext+SVM) with an improved F1 score of 20.57%. (iii) In terms of single-channel deep learning baselines, HAN+Caps network outperforms Bi-LSTM and HAN+CNN with both BERT and Fasttext embedding. BERT+HAN+Capsule with Tags achieved the best F1 score of 89.50% among the single-channel-based deep learning baselines, surpassing BERT+Fasttext+SVM by 15.61% in F1 score. This finding supports the efficacy of deep learning models over machine learning models for hate speech detection in noisy social media data. (iv) The singular results of Channel-1 (BERT+HAN+Caps) and Channel-2 (FastText+HAN+Caps) are 89.50% and 86.46% in terms of F1 score, respectively. However, combining both channels achieves an F1 score of 94.46%, indicating the efficiency of combining BERT and FastText embeddings for handling noisy text. (v) An additional noteworthy observation is that concatenating the associated tag with each word in the input post (represented by +Tags) consistently improves the F1 score. This contrasts the finding from [10], which states that machine learning models provide explanations that differ from human judgments in this dataset, suggesting that the tag is not helpful. However, in our case, utilizing the tag can guide the model and consistently improve the overall performance. (vi) We evaluated other variants of the proposed model and concluded that *HANCaps* (BERT+HAN+Caps, Fasttext+HAN+Caps) achieved the best performance with an F1 score of 94.46, significantly outperforming all the baselines. (vii) When comparing BERT with FastText, we observe that BERT embedded with any deep learning models always performs better than FastText. A similar trend is also observed in the case of machine learning baselines, except for Random Forest. This observation indicates the advantage of the transformer-based pre-trained language model XLNet over FastText in terms of efficient embedding generation of noisy social media text data.

We analyzed prediction errors for fake news by randomly selecting the utilized tag and the non-utilized tag model results from one out of ten trials. The utilized tag model misclassified 6.0% (61/1015), while the non-utilized tag model misclassified 7.0% (71/1015).

Table 1. Results of different baselines and proposed frameworks for Fake news detection in Thai

Embedding	Model	Precision	Recall	F1	Accuracy
Machine learning baselines					
BERT	Naïve Bayes	64.35	52.58	54.85	51.86
	SVM	61.36	70.62	64.28	69.84
	Random forest	59.63	67.28	61.47	67.18
Fasttext	Naïve Bayes	48.35	42.58	46.65	47.23
	SVM	61.36	70.62	64.28	66.78
	Random forest	63.55	57.28	61.75	63.32
BERT+Fasttext	Naïve Bayes	63.75	55.36	57.23	55.36
	SVM	**76.36**	**72.85**	**73.89**	**72.86**
	Random forest	75.92	71.04	72.42	71.04
SOTA					
-	Bert & Linear	90.76	89.63	91.18	91.14
Deep Learning baselines					
Single channel					
BERT	HAN+Caps	88.39	87.38	88.38	88.38
	HAN	86.51	85.48	86.47	86.48
	HAN+CNN	85.25	84.36	86.15	85.23
	Bi-LSTM	82.02	82.35	82.07	82.09
	HAN+Caps (+Tags)	**89.57**	**88.48**	**89.50**	**89.51**
	HAN (+Tags)	87.21	87.20	86.52	87.18
	HAN+CNN (+Tags)	86.99	85.41	87.13	86.94
	Bi-LSTM (+Tags)	83.17	84.78	84.90	84.90
Fasttext	HAN+Caps	85.19	84.74	85.72	85.87
	HAN+CNN	83.74	82.25	84.68	83.72
	HAN	82.34	82.11	82.15	82.15
	Bi-LSTM	84.12	85.64	84.56	84.87
	HAN+Caps (+Tags)	**86.40**	**85.35**	**86.46**	**86.58**
	HAN+CNN (+Tags)	85.01	83.99	84.72	84.87
	HAN (+Tags)	85.82	85.76	85.76	85.77
	Bi-LSTM (+Tags)	85.94	86.83	86.11	86.33
Two channel					
BERT+HAN+Caps, Fasttext+HAN		**93.25**	**93.24**	**93.23**	**93.24**
BERT+HAN, Fasttext+HAN+Caps		91.16	91.14	91.14	91.14
BERT+HAN, Fasttext+HAN		89.41	89.27	89.37	89.37
BERT+HAN+Caps, Fasttext+HAN (+Tags)		94.30	94.29	94.29	94.29
BERT+HAN, Fasttext+HAN+Caps (+Tags)		92.16	92.12	92.13	92.12
BERT+HAN, Fasttext+HAN (+Tags)		90.56	90.48	90.50	90.48
Proposed Model (HANCaps)					
BERT+HAN+Caps, Fasttext+HAN+Caps		**93.27**	**93.19**	**93.24**	**93.37**
BERT+HAN+Caps, Fasttext+HAN+Caps (+Tags)		**93.17**	**94.58**	**94.46**	**94.48**

The errors observed in the non-utilized tag model can be summarized as follows: (i) Challenges in accurately predicting fact news in 25.4% (18/71), with 94.4% (17/18) referred to external organizations (Imposter tag) and 66.7% (12/18) contain clickbait words (Title Clickbait tag). (ii) Lack of ability to distinguish fake news nature in 22.5% (16/71), with 87.5% (14/16) consisting of fabricated content using common fact news words (Fabrication tag).

The utilized tag model achieved 98.6% (70/71) misclassified by the non-utilized tag model. However, false predictions still occurred in 6.0% (61/1015) of the messages. The errors observed in the utilized tag model can be summarized as follows: (i) Misclassifying fact news as fake news in 14.75% (9/61), with 88.9% (8/9) referred to external organizations and medical personnel (Imposter tag), 44.4% (4/9) using persuasive phrases (Misleading tag), and 33.3% (3/9) using attention-grabbing words (Title Clickbait tag) (ii) Misclassifying fake news as fact news in 19.67% (12/61), with 75% (9/12) involve exaggeration and fabricated sources (Fabrication tag), 41.7% (5/12) referred to external organizations, medical personnel, and external unreliable sources (Imposter tag), and 50% (6/12) misclassified the news contain with only one tag.

The most common misclassified tags for fake news were Fabrication and Imposter, while fact news most commonly misclassified were Misleading, Clickbait, and Imposter tags.

6 Conclusion and Future Work

This paper presents HANCaps, a novel two-channel deep learning framework designed for detecting fake news in the Thai language. HANCaps leverages the integration of BERT and FastText embeddings with HAN and capsule networks to capture the hierarchical relationships embedded within textual features. Through extensive experimentation on a benchmark Thai fake news dataset, HANCaps demonstrates remarkable performance, surpassing existing SOTA methods by up to 3.28% in F1 score. By harnessing diverse tags and employing different embedding strategies, our model effectively enhances the accuracy of fake news detection. One limitation of this study is the utilization of token tags as input. The accurate detection of these tags plays a crucial role in achieving explainability, which is one aspect that we plan to address in our future work.

One limitation of this study is the utilization of token tags as input. The accurate detection of these tags plays a crucial role in achieving explainability, which is one aspect that we plan to address in our future work.

Acknowledgements This work was supported by the Ministry of External Affairs (MEA) and the Department of Science & Technology (DST), India, under the ASEAN-India Collaborative R&D Scheme.

References

1. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. Journal of economic perspectives **31**(2), 211–236 (2017)
2. Aphiwongsophon, S., Chongstitvatana, P.: Detecting fake news with machine learning method. In: Proceedings of the 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON). pp. 528–531 (2018)

3. Castillo, C., Mendoza, M., Poblete, B.: Fake news detection: A deep learning approach. ACM Transactions on the Web **13**(3), 1–28 (2019)
4. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR **abs/1810.04805** (2018)
5. Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T.: Learning word vectors for 157 languages. CoRR **abs/1802.06893** (2018)
6. Kaothanthong, N., Kongyoung, S., Theeramunkong, T.: Headline2Vec: a CNN-based feature for Thai clickbait headlines classification. International Scientific Journal of Engineering and Technology **5**(1), 20–31 (2021)
7. Kim, Y.: Convolutional neural networks for sentence classification. In: Moschitti, A., Pang, B., Daelemans, W. (eds.) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. pp. 1746–1751. ACL (2014)
8. Meesad, P.: Thai fake news detection based on information retrieval, natural language processing and machine learning. SN Computer Science **2**(6), 425 (2021)
9. Mookdarsanit, P., Mookdarsanit, L.: The COVID-19 fake news detection in Thai social texts. Bulletin of Electrical Engineering and Informatics **10**(2), 988–998 (2021)
10. Payoungkhamdee, P., Porkaew, P., Sinthunyathum, A., Songphum, P., Kawidam, W., Loha-Udom, W., Boonkwan, P., Sutantayawalee, V.: LimeSoda: Dataset for fake news detection in healthcare domain. In: Proceedings of the 16th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP). pp. 1–6 (2021). https://doi.org/10.1109/iSAI-NLP54397.2021.9678187
11. Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., Stein, B., Hagen, M.: A stylometric inquiry into hyperpartisan and fake news. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1567–1576. ACM (2017)
12. Ruchansky, N., Seo, S., Liu, Y.: CsiNet: Towards a more robust fake news detection framework. In: Proceedings of the 26th International Conference on World Wide Web. pp. 797–806. International World Wide Web Conferences Steering Committee (2017)
13. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. arXiv preprint arXiv:1710.09829 (2017)
14. Saengkhunthod, C., Kerdnoonwong, P., Atchariyachanvanich, K.: Detection of unreliable medical articles on Thai websites. In: Proceedings of the 13th International Conference on Knowledge and Smart Technology (KST). pp. 102–107 (2021)
15. Shahi, G.K., Vaibhav, G., Tiwari, A., Mishra, V., Bansal, S.: Deep learning models for fake news detection: A comparative study. In: Proceedings of the 9th International Conference on Software and Computer Applications. pp. 56–60 (2020)
16. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: A data mining perspective. ACM SIGKDD explorations newsletter **19**(1), 22–36 (2017)