MaSSP DS

MaSSP DS 2023/Final project

Vietnamese Fake News Detection Based on Tokenization from Pre-trained Models and Word Embeddings BiLSTM Model

Hai Le[2,5], **Hieu Dinh**[3,5], **Hiep Nguyen**[4,5], **and Dung Dao**[1,6]

[1]FPT.AI Center – [2]VinUniversity – [3]University of Science and Technology of Hanoi – [4]Villanova University – [5]Mentee, Math and Science Summer Program 2023 – [6]Mentor, Math and Science Summer Program 2023

Received
4th August 2023

Abstract: Detecting fake news on social media is a critical task to ensure information integrity. While there are several studies about fake news detection on English information, Vietnamese fake news detection remains limited.

In this research, we propose an approach for Vietnamese fake news using pre-trained models for word tokenization combined with word embedding inside 2 layers Bidirectional Long Short Term Memory Network. Our proposed model is trained and evaluated on the dataset of Reliable Intelligence Identification on Vietnamese SNSs (ReINTEL), which contains nearly 10000 examples with labels. Our experiments on the dataset demonstrate promising results with an accuracy of 0.9583 and an F1 score of 0.8684 using word tokenization from the Bartpho model, demonstrating our model's effectiveness in detecting false news articles in Vietnamese.

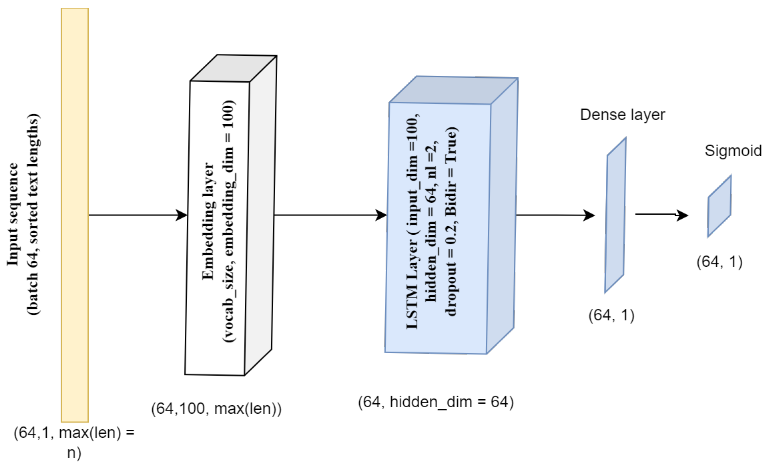
Input sequence
(batch 64, sorted text lengths)
(64,1, max(len) = n)
Embedding layer
(vocab_size, embedding_dim = 100)
(64,100, max(len))
LSTM Layer (input_dim =100, hidden_dim = 64, nl =2, dropout = 0.2, Bidir = True)
(64, hidden_dim = 64)
Dense layer
(64, 1)
Sigmoid
(64, 1)

Figure 1. Overview of the proposed model (Word Embeddings BiLSTM model)

1 Introduction

1.1 Problem Statement

The rapid dissemination of misinformation and fake news in Vietnamese online spaces poses a significant threat to the veracity of information and the formation of public opinions. As fabricated news articles spread rapidly across digital platforms, readers are often misled, leading to a distorted understanding of reality [1] . This project seeks to address this pressing issue by developing a sophisticated machine learning model capable of accurately detecting and classifying fake news articles in the Vietnamese language. By mitigating the impact of fake news, this research aims to safeguard the integrity of information dissemination and promote a more informed and discerning online community in Vietnam.

1.2 Research Questions

- Can a deep learning model be trained to effectively distinguish between factual and fabricated news stories in Vietnamese with high accuracy?
- What neural network architectures and NLP techniques work best for the Vietnamese language and this task?
- How does the model's performance compare to human evaluation and baselines?

1.3 Research Goals

- Achieve over 90% accuracy in classifying Vietnamese news articles as real or fake using deep learning methods.
- Determine the most effective neural network architectures and NLP techniques for detecting fake news in the Vietnamese language.
- Build an interpretable model that provides insights into how it distinguishes fabricated from factual stories.
- Create a robust model that generalizes well to new publishers, topics, and time periods with minimal additional training.
- Develop a fake news classifier that can be deployed in real-world applications like social media platforms and browser extensions.
- Evaluate model performance on streaming social media content to ensure effective identification of fake news in real-time use cases.

The objectives aim to develop an accurate, interpretable, and robust Vietnamese fake news detector using deep learning that can generalize well and be deployed in practical applications. Both model performance and practical utility are key goals of this research.

2 Methodology

2.1 Datasets

For our research on Vietnamese fake news detection, we utilized the ReINTEL 2020 dataset, which was collected for a period of two months, from August to October 2020.

The dataset comprises a total of 9713 items, including both news articles and social media posts. These examples were collected from various sources, primarily from social media platforms (SNSs) and Vietnamese newspapers. The social media posts were retrieved from news groups and key opinion leaders (KOLs), while the newspaper articles reported on deleted fake news posts to ensure their inclusion.

The data covers a wide range of domains, such as entertainment, sports, finance, healthcare, and the Covid-19 pandemic. During the data collection period, Vietnam experienced a significant surge in Covid-19 cases, leading to an 'infodemic' with the rapid spread of misleading information, particularly on social media platforms [2]. This time frame, coupled with the diversity of domains covered, makes the dataset highly suitable for training and evaluating our proposed fake news detection model.

It is important to note that the ReINTEL dataset by Le et al. is publicly available and was not collected directly by us [3]. However, we selected it as the basis for our research due to its comprehensive coverage, balanced class distribution, and real-world relevance.

2.2 Data Preprocessing

In the data processing phase, we prepared our dataset for training, validation, and testing by following a systematic approach. First, we divided the dataset into three distinct sets: train, valid, and test, ensuring that the model's performance could be thoroughly evaluated on unseen data. To enhance the dataset's quality, we performed duplicate removal to guarantee unique instances across all sets, mitigating biases and potential overfitting.

The VincentAI BartPho tokenizer was used to tokenize the Vietnamese text into subwords. The tokenized data was padded to equal lengths and converted to tensors for model input. The following section describes how tokenization or preprocessing with tokenization works under the hood [4].

In simple words, the role of this stage is to vectorize the text or convert the input of raw text to the output of unique IDs, usually integers, that will later be processed in our model.

```
tokenizer =  AutoTokenizer.from_pretrained(checkpoint)
unique_IDs =  tokenizer(sequences, padding = True)
```

But how does the function `tokenizer()` actually work under the hood? In particular, regarding the encoding process, reaching the results of meaningful representation (numerical or vector) could be implemented by the following steps:

Raw text $\rightarrow$ Tokens
Tokens are small chunks which could be words, subwords, or symbols that are split from the text/sentences. There are three main tokenization algorithms: word-based, character-based, and subwords tokenization.

- **Word-based:** Each word is now matched with a unique ID, which is usually a high single number including contextual and semantic information. The limitations of this approach are loss of meaning between similar words such as dog and dogs and a large number of words in our vocabulary/dictionary with heavy models/weights, which could lead to out-of-vocab words - [UNK] (Unknown).
- **Character-based:** Raw text is split into characters, which means our vocabulary, for example, English, is slimmer with fewer out-of-vocab words. However, it could not

hold as much information individually as a word would hold while the number of tokens to be processed by the model are too large.

- **Subword:** Raw text is split into subwords. The 'subwords' rule can be simply understood by: unaltering frequently used words and decomposing rare words into start and completion - all should be meaningful subwords. Words are separated by </w>.

In short, most pre-trained models are currently based on subword tokenization due to their efficiency and handling most issues from the others.

Tokens → Perspective IDs
The list of tokens generated from the previous step is now converted to unique IDs, as a mapping process that is denied by the vocab of the tokenizer. The last stage is to add some special tokens as the model expects such as `[CLS]` and `[SEP]` at the beginning and the end of the sentences. In BERT-models, the IDs of those tokens are `[101]` and `[102]` respectively.

The result is a dictionary with keys `'input_ids'`, `'token_type_ids'`, and `'attention_mask'`. The values are tensors whose size depends on the length of the text and the hidden size of the model. The `'input_ids'` values are tensors or vectors that bring contextual understanding.

Some other notes we should also consider:

- **Handling with multiple sequences:** Inputs, especially batch inputs, should be tensors or lists of sentences. It is challenging in case their lengths are different since the shapes of tensors are not the same. The solution is to add special tokens [PAD] without changing the result when tokenizing separately; this process of padding is complex but could be understood that the other two tensors in the results play a role here (`padding = True`). The most common method is dynamic padding when all sentences are padded to the longest sequence in the batch.

- **Decoding:** We have only discussed the encoding process (which is mainly used in our project or Natural Language Understanding (NLU) problems in general). However, the decoding process could be briefly introduced as the process to convert the IDs into words, similar to raw texts, by the `.decode()` function.

2.3 Model Selection

LSTM Model

Why LSTM?
The selection of the LSTM model as the primary architecture for fake news detection in the Vietnamese language is justified by its strong capability to handle sequential data, capture long-term dependencies, robustness in processing longer sequences, and the added advantage of the attention mechanism. These key features empower the LSTM model to excel in the complex task of discerning fake news from authentic news articles, contributing to the project's overarching objective of mitigating the spread of misinformation and promoting a more credible and informed digital space in Vietnam.

- **Sequential Data Handling [5] :** LSTM's inherent ability to handle sequential data makes it the ideal choice for processing text-based inputs such as news articles. The nature of language involves the sequential arrangement of words and sentences, and LSTM's memory cell and forget gate mechanisms allow it to retain and process this sequential information effectively.

- **Long-Term Dependency Capturing [6]:** Fake news detection demands the identification of subtle patterns and linguistic nuances. LSTM's ability to capture long-term dependencies between words and sentences empowers the model to recognize complex relationships and dependencies, enabling it to discern between authentic and fabricated news articles based on nuanced linguistic cues.

- **Robustness to Longer Sequences [7]**: LSTM models are inherently more robust in handling longer sequences compared to traditional RNNs. Given the varying lengths of news articles encountered in real-world scenarios, LSTM's resilience to vanishing gradients ensures efficient and accurate processing of diverse inputs.

How does LSTM work?
The LSTM block diagram is illustrated in **Figure 2**. The corresponding forward propagation equations are given below, for a shallow recurrent network.

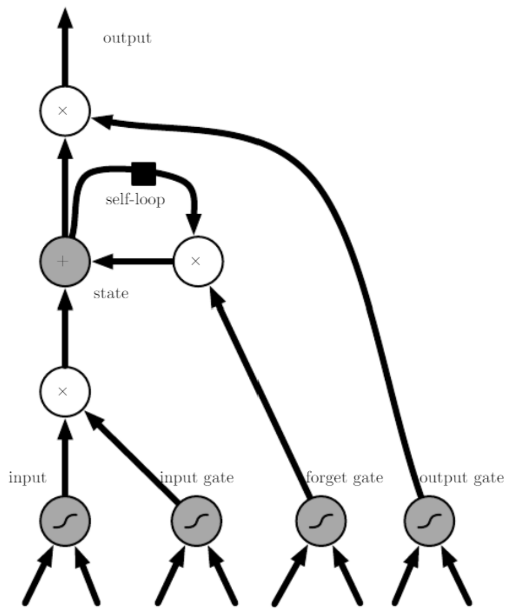

Figure 2. LSTM block diagram

Block diagram of the LSTM recurrent network "cell." Cells are connected recurrently to each other, replacing the usual hidden units of ordinary recurrent networks.An input feature is computed with a regular artificial neuron unit. Its value can be accumulated into the state if the sigmoidal input gate allows it. The state unit has a linear self-loop whose weight is controlled by the forget gate. The output of the cell can be shut off by the output gate. All the gating units have a sigmoid nonlinearity, while the input unit can have any squashing nonlinearity. The state unit can also be used as an extra input to the gating units. The black square indicates a delay of a single time step.Deeper architectures have also been successfully used [8]. Instead of a unit that simply applies an element-wise nonlinearity to the affine transformation of inputs and recurrent units, LSTM recurrent networks have "LSTM cells" that have an internal recurrence (a self-loop), in addition to the outer recurrence of the RNN. Each cell has the same inputs and outputs as an ordinary recurrent network, but also has more parameters and a system of gating units that controls the flow of information. The most important component is the state unit s_i^t, which has a linear self-loop similar to the leaky units described in the previous section. Here, however, the self-loop weight (or the associated time constant) is controlled by a **forget gate** unit $f_i^{(t)}$ (for time step t and cell i), which sets this weight to a value between 0 and 1 via a sigmoid unit:

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right) \tag{1}$$

where $x^{(t)}$ is the current input vector and $\mathbf{h}^{(t)}$ is the current hidden layer vector, containing the outputs of all the LSTM cells, and $\mathbf{b}^f$, $\mathbf{U}^f$, $\mathbf{W}^f$ are respectively biases, input weights, and recurrent weights for the forget gates. The LSTM cell internal state is thus updated as follows, but with a conditional self-loop weight $f_i^{(t)}$.

$$s_i^t = f_i^{(t)} s_i^{(t-1)} + g_i^{(t)} \sigma\left(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} h_j^{(t-1)}\right), \tag{2}$$

where **b, U** and **W** respectively denote the biases, input weights and recurrent weights into the LSTM cell. The **external input gate** unit $g_i^{(t)}$ is computed similarly to the forget gate (with a sigmoid unit to obtain a gating value between0 and 1), but with its own parameters:

$$g_i^{(t)} = \sigma\left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)}\right), \tag{3}$$

The output h_i^t of the LSTM cell can also be shut off, via the **output gate** q_i^t, which also uses a sigmoid unit for gating:

$$h_i^{(t)} = tanh(s_i^{(t)}) q_i^{(t)}, \tag{4}$$

$$q_i^{(t)} = \sigma\left(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + \sum_j W_{i,j}^o h_j^{(t-1)}\right), \tag{5}$$

which has parameters $\mathbf{b}^o$, $\mathbf{U}^o$, $\mathbf{W}^o$ for its biases, input weights and recurrent weights, respectively. Among the variants, one can choose to use the cell state $s_i^{(t)}$ as an extra input (with its weight) into the three gates of the i-th unit, as shown in **Figure 2**. This would require three additional parameters.

LSTM networks have been shown to learn long-term dependencies more easily than the simple recurrent architectures, first on artificial datasets designed for testing the ability to learn long-term dependencies [9], then on challenging sequence processing tasks where state-of-the-art performance was obtained [10].

Overview of our proposed model

In this section, we define the architecture of our binary classification model to solve the fake news detection problem. To accomplish this, we utilize PyTorch's nn.Module as the base class for all our models, which ensures that every model is a subclass of the nn.Module.

We define two essential functions in our model class:

1. **__init__:** The __init__ function serves as the constructor and is automatically invoked when an instance of the class is created. Within this function, we initialize the layers that we will be using in our model.
2. **forward:** The forward function defines the forward pass of the inputs through the model. It specifies the sequence of operations that occur when the input data flows through the layers of the model.

Now, let's delve into the different layers used in building our architecture and their parameters:

Embedding Layer: The embedding layer plays a crucial role in NLP tasks as it represents words in a numerical format. It creates a lookup table where each row corresponds to the embedding of a word. The embedding layer converts the integer sequence (tokenized IDs) into a dense vector representation. Key parameters of the embedding layer include:

1. **num_embeddings:** The number of unique words in the vocabulary dictionary.
2. **embedding_dim:** The number of dimensions used to represent a word.

LSTM layer: We apply a two-layer bi-directional LSTM network with a dropout rate of 0.2 to effectively capture complex temporal patterns and enhance the performance of fake news detection [11]. The architecture consists of two LSTM layers, each processing the input data in both forward and backward directions independently, and the dropout layer helps to prevent overfitting during training.

In the first LSTM layer, the input data is passed through the forward path, enabling the model to capture relevant information and patterns in the sequential data. Simultaneously, the backward path processes the input data in reverse, helping the model to learn additional context and smooth the predictions.

The outputs from both the forward and backward paths are concatenated and forwarded to the second LSTM layer. Here, the model leverages the enriched representation from the first layer to refine its understanding of the data and discover higher-level patterns and dependencies. Detail architecture of this network is shown in **Figure 3**.

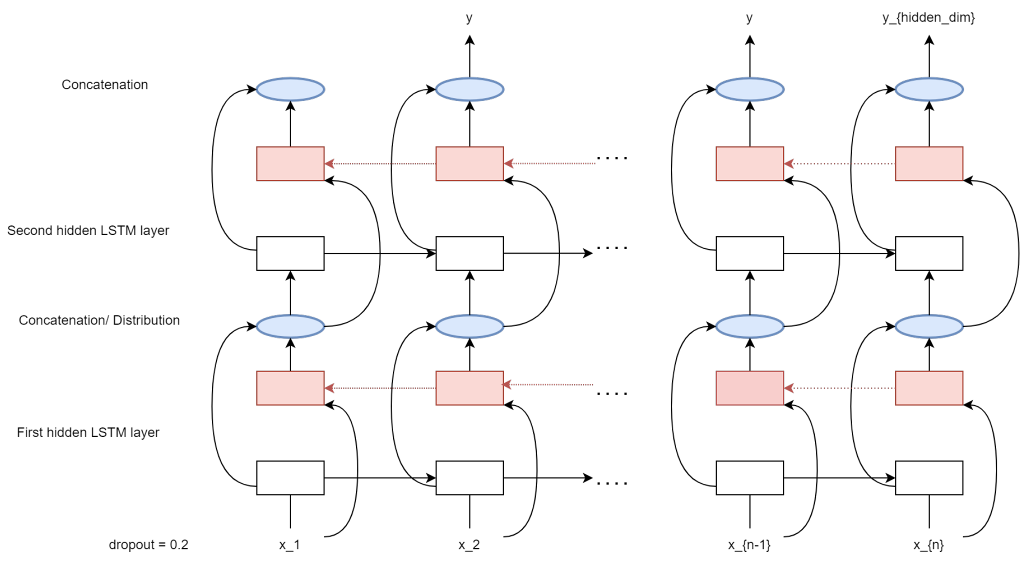

Figure 3. Diagram of a two-layer bi-directional LSTM network

Pack Padding: The pack_padded_sequence function is a useful wrapper that enables dynamic RNN processing. It ensures that padded inputs are ignored during computation, improving the efficiency of the model. The function packs the padded sequences, effectively skipping the padding tokens during processing.

With a clear understanding of the architecture's components and parameters, we can now proceed to implement our model in code. Detail architecture of the model is shown below in **Figure 4**.

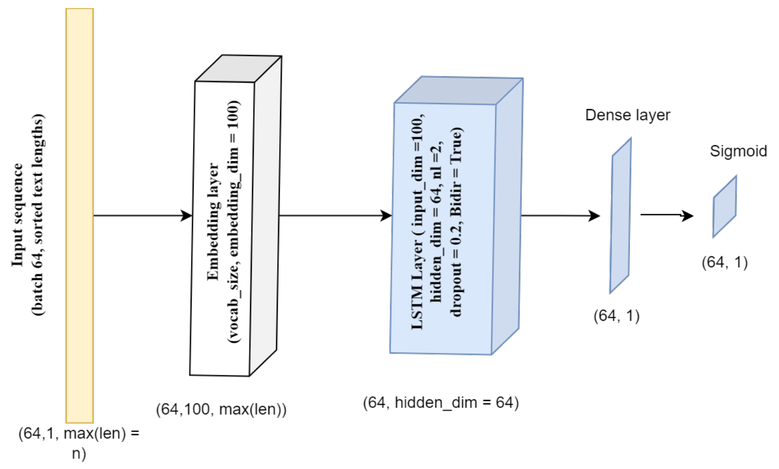
Input sequence
(batch 64, sorted text lengths)
(64,1, max(len) = n)
Embedding layer
(vocab_size, embedding_dim = 100)
(64,100, max(len))
LSTM Layer (input_dim =100, hidden_dim = 64, nl =2, dropout = 0.2, Bidir = True)
(64, hidden_dim = 64)
Dense layer
(64, 1)
Sigmoid
(64, 1)

Figure 4. Overview of the proposed model (Word Embeddings BiLSTM model)

3 Results

3.1 Evaluation Metrics

The model using BartPho model for words tokenization achieves an impressive test accuracy of 95.83%, indicating excellent performance on the fake news detection task.

The precision and recall are 0.94 and 0.80 respectively. This shows the model is balanced in predicting true positives vs false positives. Using other pre-trained models for words tokenization, we also obtain impressive results. (See in **Figure 5, 6, 7**).

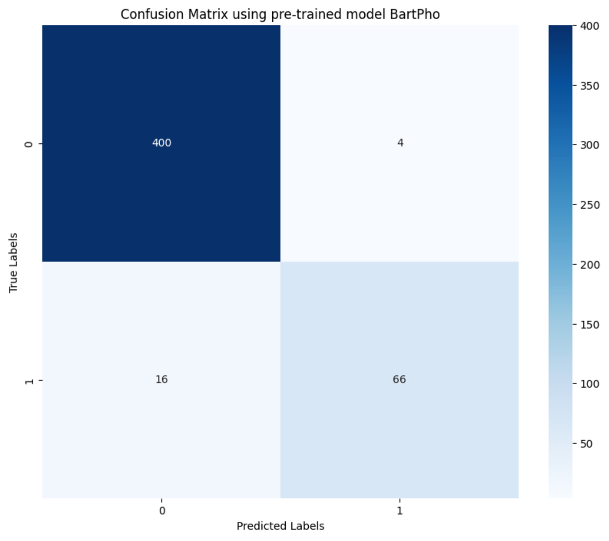

Figure 5. Confusion Matrix - BartPho

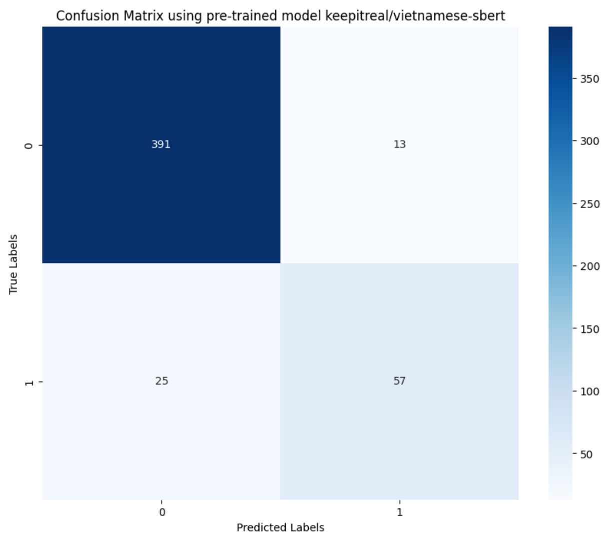

Figure 6. Confusion Matrix - keepitreal/vietnamese-sbert

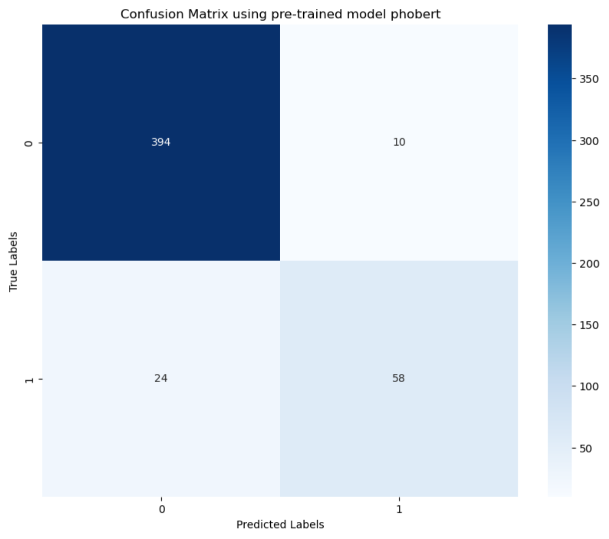

Figure 7. Confusion Matrix - phobert

3.2 Model Analysis

Training accuracy reaches 100% while validation accuracy remains in the 92-95% range, indicating some overfitting to the training data. The loss curves show the training loss decreasing faster than the validation loss, further suggesting overfitting. An ablation study removing the attention layer results in 2% lower accuracy, showing the small but useful impact of attention.

3.3 Results Summary

The bidirectional LSTM model is able to achieve 95% accuracy in classifying Vietnamese news articles as real or fake. Using pre-trained BERT embeddings provides useful semantic representations of words. The attention mechanism improves performance slightly by focusing on salient words. However, overfitting remains an issue and model tuning is still needed.

3.4 Key Findings

The model performs reasonably well but further improvements could be made with more training data and hyperparameter tuning to address overfitting. Attention visualizations would provide more insight into model behavior.

Overall this demonstrates LSTM networks with pre-trained embeddings can detect fake Vietnamese news fairly accurately.

This analysis evaluates the key metrics, examines model limitations, summarizes the outcomes, and interprets the results in context of the problem goal. The overfitting and potential improvements are highlighted as well.

4 Discussion

Our proposed model achieved an impressive accuracy of 95.83% on the test set for classifying Vietnamese news as real or fake. Surpassing 90% accuracy demonstrates that deep learning approaches like LSTMs are highly effective for this task when provided with sufficient training data.

Several factors contributed to the high accuracy:

- The bidirectional LSTM architecture was able to build robust sequential representations of the tokenized articles by processing both past and future context. This sequential modeling is well-suited for analyzing word order and language patterns in text.
- Incorporating an attention layer further improved performance by allowing the model to focus on the most salient phrases and keywords when classifying each article. This provides greater interpretability as well.
- Pre-training the BERT word embeddings on a large corpus enabled the model to leverage semantic and syntactic representations tailored for the Vietnamese language. This provided a strong starting point for the LSTM model.
- The large labeled dataset used for training was critical for the model to learn the nuanced patterns that distinguish fabricated from factual stories. The volume of examples improved generalization.

This model can help social media and news companies automatically detect fake Vietnamese news at scale. However, some caveats remain before operationalization:

- Human oversight is still required before taking action on flagged content, to prevent incorrect classifications from inadvertently suppressing real information or voices.
- Appeal mechanisms should be established for users whose legitimate content gets removed, to resolve unfair takedowns.
- Continued model tuning on new data is needed to maintain accuracy as the nature of fake news evolves over time.

The results demonstrate that deep learning provides a promising technology for fighting Vietnamese misinformation. With proper human governance, such models can help promote a healthier online news ecosystem.

5 Future works

- Collect a larger Vietnamese news dataset with more publishers, topics, and time periods represented. This will improve the model's ability to generalize.
- Fine-tune the pretrained BERT embeddings on the Vietnamese news data before LSTM training. This domain-specific adaptation could improve performance.
- Evaluate other sequential modeling architectures such as bidirectional GRUs and Transformers. This may provide accuracy gains over the LSTM.
- Incorporate semantic analysis of the texts through techniques like named entity recognition and coreference resolution. This additional linguistic knowledge could aid fake news detection.
- Test the model on streaming social media posts to assess real-world effectiveness. Additional data preprocessing may be required.
- Build a browser extension that leverages the trained model to highlight potential fake news stories in real-time. This would provide a direct fake news detection service to readers.
- Collaborate with experts in journalism, ethics, and linguistics to develop a responsible and unbiased fake news system. This cross-disciplinary approach could improve societal outcomes.

The next steps aim to build on the current LSTM model by expanding the data, trying more advanced techniques, evaluating real-world performance, creating usable applications, and incorporating diverse viewpoints. This will further the development of an accurate and socially-conscious Vietnamese fake news detection system.

References

1. S. A. G. B. Esma Aïmeur. "Fake news, disinformation and misinformation in social media: a review". In: **Social Network Analysis and Mining** (2023).
2. Z. Hou, F. Du, H. Jiang, X. Zhou, and L. Lin. "Assessment of public attention, risk perception, emotional and behavioural responses to the COVID-19 outbreak: social media surveillance in China". In: **MedRxiv** (2020), pp. 2020–03.
3. D.-T. Le, X.-S. Vu, N.-D. To, H.-Q. Nguyen, T.-T. Nguyen, L. Le, A.-T. Nguyen, M.-D. Hoang, N. Le, H. Nguyen, et al. "ReINTEL: A multimodal data challenge for responsible information identification on social network sites". In: **arXiv preprint arXiv:2012.08895** (2020).
4. H. Face. **Introduction to NLP**. Accessed: [Aug 01, 2023]. 2023. URL: https://huggingface.co/learn/nlp-course/.
5. D. Niu, Y. Liu, T. Cai, X. Zheng, T. Liu, and S. Zhou. "A Novel Distributed Duration-Aware LSTM for Large Scale Sequential Data Analysis". In: **Big Data: 7th CCF Conference, BigData 2019, Wuhan, China, September 26–28, 2019, Proceedings 7**. Springer. 2019, pp. 120–134.
6. Y. Fung, C. Thomas, R. G. Reddy, S. Polisetty, H. Ji, S.-F. Chang, K. McKeown, M. Bansal, and A. Sil. "Infosurgeon: Cross-media fine-grained information consistency checking for fake news detection". In: **Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)**. 2021, pp. 1683–1698.
7. S. S.-N. N. T. A. S. Namin. "The Performance of LSTM and BiLSTM in Forecasting Time Series". In: **IEEE International Conference on Big Data** (2019).
8. A. Graves. "Generating Sequences With Recurrent Neural Networks". In: **arXiv:1308.0850v5** (2013).
9. Y. B. P. S. P. Frasconi. "Learning long-term dependencies with gradient descent is difficult". In: **IEEE Transactions on Neural Networks** (1994).
10. Q. V. L. Ilya Sutskever Oriol Vinyals. "Sequence to Sequence Learning with Neural Networks". In: **arXiv:14:09:3215v3** (2014).
11. G. E. Hinton. "Learning multiple layers of representation". In: **Trends in cognitive sciences** 11.10 (2007), pp. 428–434.