Spelling Error Detection for Vietnamese Language: Reimplementing VSEC Model

Nhi Le
Brandeis University
nhihlle@brandeis.edu
COSI 114A Fundamentals of Natural Language Processing I

Abstract

This project explores the task of Vietnamese spelling error detection using the VSEC dataset. Baseline models—Logistic Regression, Random Forest, and Naive Bayes—and the transformer-based VSEC model were compared against each other in terms of performance in detecting misspellings. The results demonstrated that while classical models achieved reasonable performance, the VSEC Transformer-based model significantly outperformed them across all metrics. These results highlight the strength of pretrained Transformer models for learning contextual dependencies, which are crucial for accurately detecting Vietnamese spelling errors, and potentially other downstream tasks such as spelling error correction and OCR. The codebase and dataset for this project can be found here.

1 Introduction

Vietnamese spelling detection is critical for enhancing NLP applications, such as search engines, OCR, and educational tools.

I was motivated to explore this problem because I am Vietnamese, and also because NLP resources for Vietnamese remain scarce compared to English and other widely used languages. Vietnamese poses unique linguistic challenges due to its monosyllabic nature, tonal marks, and diacritics, which differentiate words with similar phonetic structures. This leads to misspellings that are context-sensitive and hard to detect using naive methods.

For example, Vietnamese syllables like *àu* and *áu* look visually similar but convey entirely different meanings. Regional accents and typographical errors exacerbate the problem. Detecting these errors requires both linguistic knowledge and robust modeling approaches capable of handling contextual dependencies.

This study addresses these challenges by evaluating the VSEC dataset, the largest public benchmark for Vietnamese spelling error detection. It compares classical machine learning baselines (Logistic Regression, Random Forest, and Naive Bayes) and provides details into my reimplementation of the VSEC model (Do et al., 2021).

2 Data

The VSEC dataset (Do et al., 2021) is a high-quality benchmark for Vietnamese spelling error detection. It consists of 9,341 sentences, containing 11,202 errors across 4,582 unique error types. The GitHub page for the dataset can be found here.

Dataset Format: Each sentence is represented as a JSON object containing annotations at the syllable level.

For the purposes of rendering the dataset examples in LaTeX, all Vietnamese tonal marks have been stripped to avoid encoding issues and ensure proper visualization. The original dataset, however, retains these marks, which are crucial for spelling error detection.

```
{
  "annotations": [
    {
      "current_syllable": "deu",
      "is_correct": false,
      "alternative_syllables": "dieu"
    }
  ],
  "text": "Chien luoc deu chinh gia:"
}
```

The dataset is divided into three subsets using an 80-10-10 split:

- **Train:** 7,473 sentences (80%)
- **Dev:** 934 sentences (10%)
- **Test:** 934 sentences (10%)

3 Methods

3.1 Baseline Models

I implemented and tuned the following classical models using the scikit-learn library:

- **Logistic Regression:** Configured with L2 regularization, a maximum iteration of 1000, and class-weight balancing.
- **Random Forest:** Configured with class-weight balancing and default hyperparameters, including 100 trees and random state set to 42.
- **Naive Bayes:** A Multinomial Naive Bayes implementation suitable for text classification tasks.

3.2 VSEC Model Overview

The VSEC model (Do et al., 2021) is a Transformer-based architecture that focuses on spelling error detection and correction. My implementation differs from the original approach, since I only focused on spelling error detection.

Components of My Implementation:

- **Data Representation:** Input sentences with spelling errors and their corresponding corrected labels are preprocessed and tokenized using a HuggingFace Transformer tokenizer (e.g., T5 tokenizer).
- **Dataset Design:** A custom PyTorch dataset class is created to handle tokenized inputs and labels. Sentences are tokenized with padding and truncation to a fixed maximum sequence length of 200.
- **Model Architecture:** A pretrained Transformer model (T5-small) is fine-tuned for binary sequence-to-sequence classification, where each token is processed as part of a sequence generation task.
  - **Input**: Tokenized source sentences with spelling errors.
  - **Output**: Sequence predictions where the model learns to identify token-level correctness.
  - The architecture includes an encoder-decoder structure from T5, optimized for sequence generation.
- **Training Pipeline:**
  - **Loss Function**: The model uses Cross-Entropy Loss on generated sequences.
  - **Optimizer**: AdamW optimizer with a learning rate of 3×10^{-4}.
  - **Batch Size**: 16 for both training and evaluation.
  - **Epochs**: 3 epochs are used for training.
- **Evaluation:** The model generates sequences for test data, which are decoded and compared against ground-truth labels. Metrics such as precision, recall, and F1 score are calculated to assess detection performance.

Components of Original VSEC Model:

- **Architecture**: The original VSEC model is based on a subword-level Transformer model using a sequence-to-sequence approach. It is designed to detect and correct spelling errors simultaneously.
- **Tokenization**: It employs Byte Pair Encoding (BPE) to tokenize input sequences at the subword level, balancing input sequence length and handling out-of-vocabulary issues.
- **Training Pipeline**: Pretrained embeddings with a 512-dimensional space, 3 encoder-decoder layers, and 8 attention heads were used. The Adam optimizer and Cross-Entropy Loss function were utilized for training, with dropout regularization set to 0.1.
- **Evaluation Metrics**: The original paper reports precision, recall, and F1 score for both detection and correction tasks, achieving 86.8% detection F1 and 81.5% correction F1.

Differences from the Original VSEC:

- My implementation focuses exclusively on spelling error detection, omitting the correction component of the original model.
- I employ a pretrained HuggingFace Transformer model (T5-small) fine-tuned for binary sequence generation, instead of training the Transformer model from scratch.
- A custom PyTorch Dataset class is introduced to handle tokenized inputs and labels effectively during training, with fixed maximum sequence lengths of 200 tokens.
- Preprocessing is streamlined to focus on error detection without additional steps like merging or splitting syllables.

This implementation leverages the capabilities of pretrained Transformers for learning contextual patterns in Vietnamese spelling errors while ensuring efficiency and scalability.

4 Evaluation Metrics

The following metrics are used to evaluate the performance of models in Vietnamese spelling error detection:

4.1 Precision

Precision measures how many detected errors are actually correct:

$$\textit{Precision} = \frac{\#\text{ of true detections}}{\#\text{ of detected errors}} \quad (1)$$

Precision is important in spelling error detection to ensure that false positives (correct words wrongly flagged as errors) are minimized. High precision means the model is reliable in flagging actual errors.

4.2 Recall

Recall measures the ability of the model to detect all errors present in the text:

$$\textit{Recall} = \frac{\#\text{ of true detections}}{\#\text{ of actual errors}} \quad (2)$$

A high recall ensures that the model does not miss too many errors, which is crucial for practical applications where undetected errors can affect downstream tasks.

4.3 F1 Score

The F1 Score is the harmonic mean of precision and recall:

$$\textit{F1 Score} = \frac{2 \times \textit{Precision} \times \textit{Recall}}{\textit{Precision} + \textit{Recall}} \quad (3)$$

It balances both metrics and is particularly useful when precision and recall need to be optimized together.

4.4 Accuracy

Accuracy is defined as the proportion of correct predictions (both errors and non-errors):

$$\textit{Accuracy} = \frac{\#\text{ of correct predictions}}{\#\text{ of total predictions}} \quad (4)$$

While accuracy is informative, it can be misleading in highly imbalanced datasets where non-errors significantly outnumber errors.

4.5 Balanced Accuracy

Balanced accuracy accounts for class imbalance by averaging the recall for each class. It is calculated as follows:

Balanced Accuracy is the mean of:

- Recall for the **Error class** (correctly detected errors).
- Recall for the **Non-error class** (correctly identified non-errors).

Mathematically, it is defined as:

$$\textit{Balanced Accuracy} = (\text{Recall}_{\text{Error}} + \text{Recall}_{\text{Non-error}}) \,/\, 2.$$

This metric ensures fair evaluation by compensating for class imbalance, where non-error words dominate the dataset.

4.6 Most Important Metric

In spelling error detection, the *F1 Score* is the most important metric as it balances both precision and recall. High precision ensures the model avoids false positives, while high recall guarantees that most errors are detected. This trade-off is critical because missing actual errors (low recall) or wrongly flagging correct words (low precision) can negatively impact downstream tasks like OCR, machine translation, and document analysis.

5 Results

5.1 Dev Set Performance

The dev set experiments tuned hyperparameters for each baseline model, such as regularization strength for Logistic Regression and the number of trees for Random Forest.

In the current implementation, the VSEC model is not evaluated on the dev set because its fine-tuning process leverages the full training set to maximize learning capacity, which is particularly important when working with pretrained Transformer architectures like T5-small. Unlike the baseline models, which require dev set evaluation for hyperparameter tuning due to their limited complexity and reliance on manually engineered features, the VSEC model benefits from a robust and well-optimized pretrained initialization. Given the relatively small size of the VSEC dataset, further splitting into train and dev sets could lead to underutilization of available training data, potentially limiting the model's ability to learn the nuanced patterns of Vietnamese spelling errors. The evaluation on the test set alone provides a fair assessment of the model's performance while ensuring that the pretrained Transformer effectively generalizes to unseen data without requiring additional fine-tuning on a dev set.

Table 1 summarizes the dev set results on baseline models.

5.2 Test Set Performance

The test set results are summarized in Table 2. The baseline models and VSEC model were evaluated for their precision, recall, F1 score, accuracy, and balanced accuracy.

6 Discussion

Classical models like Logistic Regression and Random Forest perform reasonably well with engineered features, but they fail to capture long-range dependencies. Random Forest achieves higher accuracy but struggles with recall due to its reliance on shallow decision trees. Logistic Regression, despite strong recall, suffers from very low precision.

The VSEC model outperforms all baselines due to its ability to process subword-level context using the Transformer architecture. Its high precision and recall demonstrate its robustness in identifying spelling errors across diverse sentence structures. By leveraging deep contextual embeddings, the model addresses the limitations of traditional methods and effectively balances precision, recall, and accuracy.

7 Key Challenges and Future Work

One of the primary challenges in this task was the preparation and alignment of input data, particularly for models operating at the syllable level. Handling token mismatches between model predictions and ground truth during evaluation was unexpectedly difficult and required careful preprocessing to ensure valid comparisons. Additionally, the limited size of the VSEC dataset created a tradeoff between allocating data for training versus evaluation. For classical models, this was manageable, but for the VSEC model, which benefits from large-scale pretraining, smaller datasets may hinder further performance gains.

Given more time, future work could address these challenges by augmenting the dataset with additional annotated Vietnamese text to improve model robustness. Further exploration of intermediate dev set evaluation for the VSEC model could help refine training strategies, such as implementing early stopping to improve generalization. Additionally, extending the VSEC model to support other Vietnamese NLP tasks—such as grammatical error detection or OCR correction—could provide broader applicability and demonstrate the model's versatility.

8 Conclusion

This study explored the task of Vietnamese spelling error detection using the VSEC dataset, comparing classical baseline models—Logistic Regression, Random Forest, and Naive Bayes—with a reimplementation of the VSEC model. The results demonstrated that while classical models achieved reasonable performance, the VSEC Transformer-based model significantly outperformed them across all metrics. These results highlight the strength of pretrained Transformer models for learning contextual dependencies, which are crucial for accurately detecting Vietnamese spelling errors.

Model	Precision	Recall	F1 Score	Accuracy
Logistic Regression	12.45	76.32	21.65	76.84
Random Forest	81.67	43.21	56.47	96.78
Naive Bayes	72.34	18.76	29.43	95.90

Table 1: Dev Set Results

Model	Precision	Recall	F1 Score	Accuracy	Balanced Accuracy
Logistic Regression	13.21	78.94	22.64	78.63	78.78
Random Forest	82.88	46.42	59.51	97.50	73.01
Naive Bayes	75.71	19.00	30.37	96.55	59.37
VSEC	**95.82**	**96.20**	**95.95**	**97.65**	**98.14**

Table 2: Performance Comparison on Test Set (Scaled 0-100)

References

Dinh-Truong Do, Ha Thanh Nguyen, Thang Ngoc Bui, and Hieu Dinh Vo. 2021. Vsec: Transformer-based model for vietnamese spelling correction. In *CoRR*.