Natural Language Processing

Text Error Correction

Sharif University of Technology
Ali Nikkhah, Sarina Zahedi, Ramtin Khoshnevis*

May 30, 2024

*Equal Contribution

Contents

1 Project Explanation

The aim of this project is to standardize and correct Persian text extracted from tweets. This involves addressing various errors in the text, which can be categorized into three main types:

1.1 Typographical Errors

Typographical errors include missing, extra, or swapped characters within words. Examples include:

- **Missing Characters**
- **Extra Characters**
- **Swapped Characters**

1.2 Grammatical Errors

Grammatical errors involve incorrect sentence structures and improper use of punctuation. Examples include:

- Misuse of punctuation
- Incorrect use of possessive markers

1.3 Colloquial Writing

Colloquial writing includes non-standard abbreviations and informal language. Examples include:

- Informal writing
- Abbreviations

The project involves editing and correcting these errors to bring the text to a standard form.

2 Approach Overview

This project uses a hybrid approach combining distance algorithms and transformer models to correct errors in Persian text. The pipeline includes the following stages:

2.1 Data Gathering

Collect a large dataset of Persian tweets with annotated errors and their corrections.

2.2 Preprocessing

Clean and preprocess the text data to standardize the format and remove noise.

2.3 Dataset Augmentation

Augment the dataset with synthetic errors to improve the robustness of the model, focusing on typographical errors such as missing, extra, or swapped characters.

2.4 Distance Algorithms

Apply distance algorithms like Levenshtein distance to correct simple typographical errors.

2.5 Transformer Models

Use transformer-based models to handle complex corrections requiring contextual understanding, such as grammatical errors and colloquial writing.

2.5.1 Comparison of Approaches

- **Seq2Seq Approaches**: Sequence-to-sequence models are effective for capturing dependencies in the text but might struggle with very long sentences and require large amounts of data for training.
- **GPT Approach**: Generative Pre-trained Transformers (GPT) models excel in understanding context and generating coherent text, making them suitable for correcting complex grammatical errors and colloquial writing.

3 Block Diagram

The following diagram illustrates the hybrid approach pipeline:

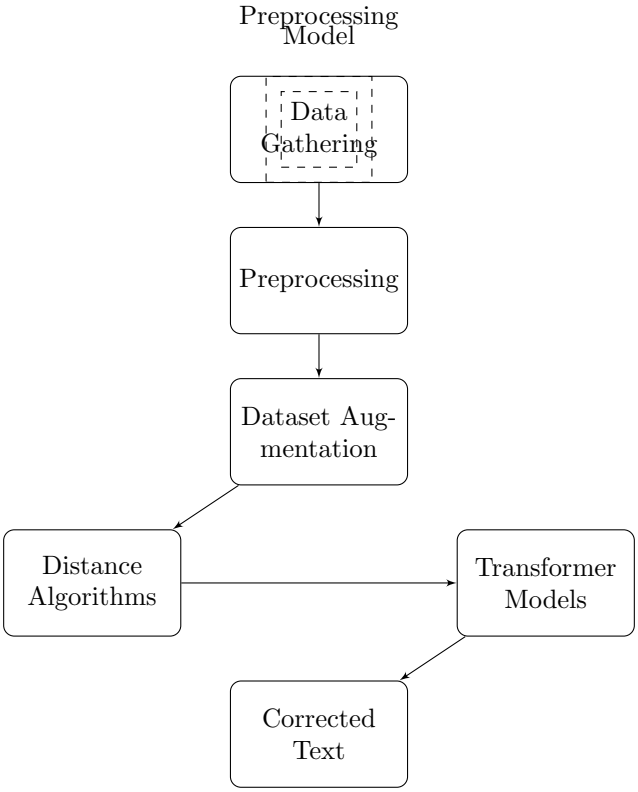

Figure 1: Hybrid Approach Pipeline

4 Installation Instructions

To set up the NLP bot, follow these steps:

4.1 Create Configuration File

Create a `yaml` configuration file and set up a connector to connect to the bot account as shown below:

```
connectors:
  - name: matrix
    homeserver: "https://matrix.org"
    room: "!your-room-id:matrix.org"
    mxid: "@your-bot:matrix.org"
    password: "your-password"
```

4.2 Login and Invite Bot

Login using the group's account through Element and send an invitation to your bot's account. Start the `opsdroid` server and specify a public address for the group. Register this group address in the rooms section of the configuration file.

4.3 Load Language Model

After training your language model, load it in the bot's skill file and generate the appropriate responses. The bot will match inputs to the conditions defined in the skill file and provide the correct outputs as shown below:

```
# Example Python code to load model in bot skill
import your_model

def your_skill_function(input_text):
    model = your_model.load('path/to/your/model')
    corrected_text = model.predict(input_text)
    return corrected_text
```

5 Model Selection and Error Handling

5.1 Why Transformers?

Transformers have revolutionized the field of natural language processing due to their ability to handle long-range dependencies and their scalability with large datasets. Unlike traditional sequence-to-sequence models, transformers use self-attention mechanisms to capture the relationships between all words in a sentence simultaneously, which makes them highly effective for tasks that require understanding context, such as text correction.

5.2 Why T5 Model?

The T5 (Text-To-Text Transfer Transformer) model, developed by Google, is particularly suitable for our task because it frames all NLP tasks as text-to-text problems. This unification allows us to leverage a single model architecture for various types of text correction tasks. Additionally, the T5 model has shown excellent performance in a wide range of NLP tasks, making it a robust choice for correcting typographical, grammatical, and colloquial errors in Persian text.

5.3 Handling Typographical Errors

Typographical errors, such as missing, extra, or swapped characters, are addressed using distance algorithms like the Levenshtein distance. These algorithms measure the minimum number of single-character edits required to change one word into another, allowing us to efficiently identify and correct simple typographical errors.

5.4 Handling Grammatical Errors and Colloquial Writing

Grammatical errors and colloquial writing present more complex challenges that require a deeper understanding of context and syntax. The T5 model, pretrained on Persian language data, is employed to handle these errors. By fine-tuning the T5 model on our dataset of erroneous and corrected sentences, we enable it to learn the contextual patterns and nuances of Persian grammar and colloquial usage. This allows the model to generate corrected sentences that are grammatically accurate and adhere to standard language conventions.

5.5 Combining Approaches

Our hybrid approach leverages the strengths of both distance algorithms and transformer models. Distance algorithms are used for quick and efficient correction of simple typographical errors, while the T5 model handles more complex grammatical and contextual corrections. This combination ensures comprehensive error correction, improving the overall quality and standardization of Persian text.

6 Training Process

6.1 Using Pretrained T5 Model

For this project, we started with a pre-trained T5 model that had been trained on a large Persian text corpus. This gave us a robust starting point, as the model already had a good understanding of Persian language structure, grammar, and common usage patterns.

6.2 Integrating Word-Level and Sentence-Level Correction

To achieve high-quality text correction, we integrated both word-level and sentence-level correction into our training process. This involved a multi-stage fine-tuning approach:

6.2.1 Word-Level Fine-Tuning

First, we fine-tuned the pre-trained T5 model on a dataset specifically designed for word-level corrections. This dataset consisted of individual words with common typographical errors and their correct forms. Fine-tuning on this dataset helped the model learn basic error patterns, such as missing, extra, or swapped characters.

6.2.2 Sentence-Level Fine-Tuning

After the initial fine-tuning on word-level data, we further fine-tuned the same T5 model on a sentence-level dataset. This dataset contained sentences with various types of errors, including typographical, grammatical, and colloquial errors, along with their corrected versions. Fine-tuning on this dataset enabled the model to learn context-specific corrections, improving its ability to correct errors within the broader context of a sentence.

6.3 Multi-Stage Fine-Tuning Approach

The multi-stage fine-tuning approach involves the following steps:

1. **Initial Fine-Tuning on Word-Level Data**: This stage helps the model learn basic error patterns and corrections at the word level.

2. **Secondary Fine-Tuning on Sentence-Level Data**: This stage allows the model to learn how to apply corrections in context, ensuring that the corrections make sense within the entire sentence.

By fine-tuning the model in stages, we help it generalize better, combining the benefits of both word-level and sentence-level correction. This approach results in a model that is more effective at correcting errors within sentences, as it leverages both basic correction patterns and contextual understanding.

6.4 Benefits of the Multi-Stage Fine-Tuning Approach

- **Improved Generalization**: The model can handle a wider range of errors by first learning basic correction patterns and then contextualizing them within sentences.
- **Enhanced Performance**: Integrating word-level and sentence-level correction leads to better overall performance in text correction tasks.
- **Context-Aware Corrections**: The model makes more accurate corrections by considering the entire sentence context, reducing the likelihood of introducing new errors.

This multi-stage fine-tuning strategy ensures that the T5 model is well-equipped to handle a variety of text correction tasks, making it a powerful tool for standardizing and correcting Persian text extracted from tweets.

7 Challenges

The main challenges encountered during this project include:

7.1 Lack of Dataset for Persian Sentences

There is a significant scarcity of comprehensive datasets for Persian sentences. This limitation hinders the development of robust models, as they rely heavily on large and diverse datasets for training.

7.2 Accuracy of Persian Words Dataset

The available datasets for Persian words often lack accuracy and consistency. This inconsistency affects the model's ability to learn correct word forms and reduces the overall effectiveness of the error correction process.

7.3 Lack of Dataset for Grammatical Errors in Persian

Another major challenge is the absence of datasets specifically focused on grammatical errors in Persian. Such datasets are crucial for training models to recognize and correct grammatical mistakes effectively.

7.4 High Resource Demand for Model Training

Training advanced models like transformers requires significant computational resources. The high resource demand poses a challenge, especially for projects with limited access to powerful hardware and computational capabilities.

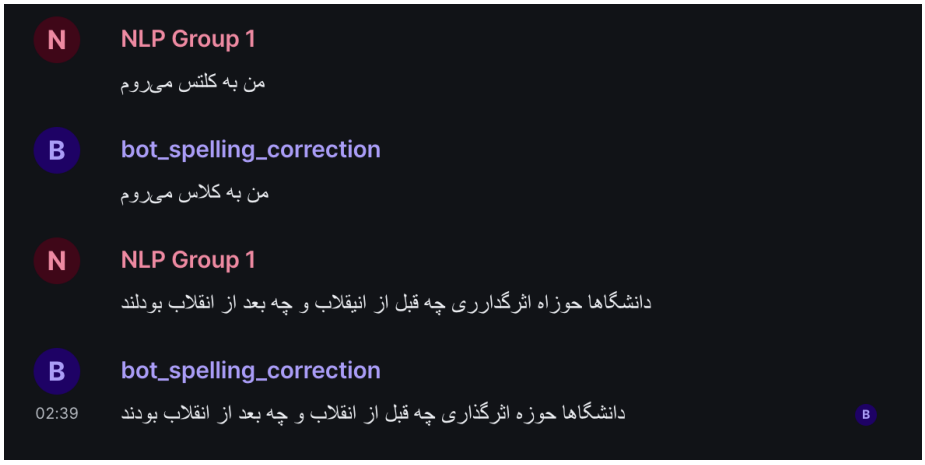

Figure 2: Sample Output Image 1

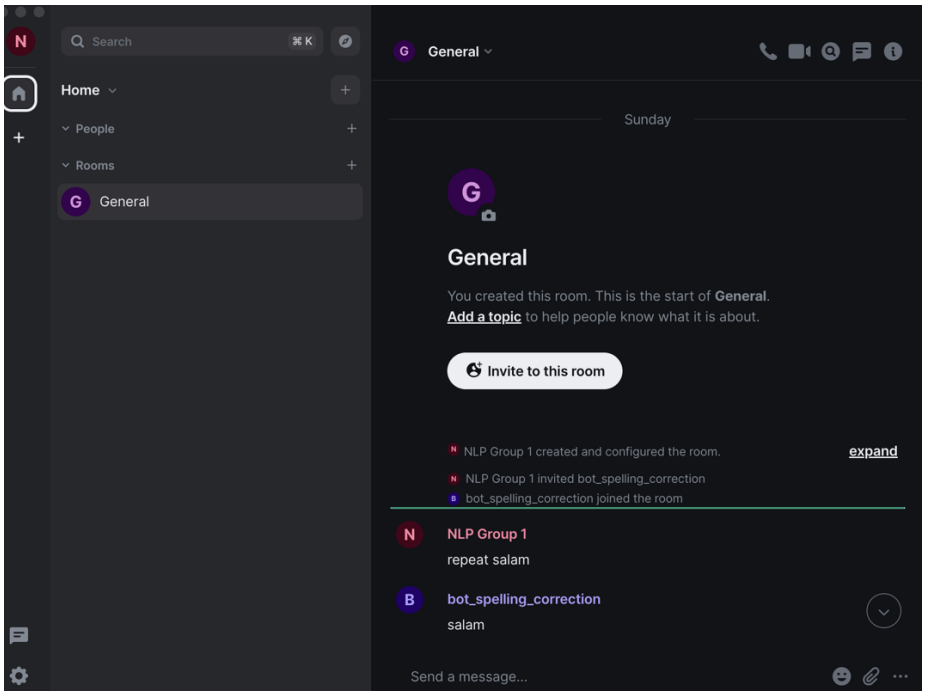

Figure 3: Sample Output Image 2

8 Datasets

The datasets for this project were sourced from Persian tweets. Each dataset was categorized and annotated based on the types of errors they contained. The annotations helped in training models to automatically detect and correct these errors.

9 References

- https://github.com/language-ml/parsi.io/tree/%20main/parsi_io/modules/space_punctuation_editor
- https://github.com/shahind/Persian-Words-Database
- https://github.com/rominaoji/PerSpellData/tree/main
- https://github.com/AUT-Data-Group/HeKasre/tree/main