An Intelligent Web-Based Grammar and Spelling Checker Using NLP and Transformer-Based Language Models

L. Juan Carlos, R. Jomari, B. Rowell.
Laguna State Polytechnic University Los Baños, Philippines College of Computer Studies
May 2025

Abstract

This paper presents the development of an intelligent web-based grammar and spelling checker that utilizes modern Natural Language Processing (NLP) techniques. The system combines rule-based grammar checking, deep learning-based text correction using a fine-tuned T5 model, and spelling detection through SymSpell. Built with a Flask backend and HTML interface, the system supports robust error analysis at the sentence and paragraph levels, providing clear suggestions and promoting writing clarity and readability. The system's design, implementation, and performance are discussed, with relevant comparisons to existing tools.

I. INTRODUCTION

Writing effectively in English, a globally dominant language in academic, professional, and digital communication, requires not only a strong command of grammar and vocabulary but also an awareness of clarity and readability. Errors in spelling, grammar, and sentence structure can significantly affect the credibility of a text and the comprehension of its intended message. As digital communication becomes more prevalent, the demand for intelligent, automated writing assistance tools has surged.

Existing grammar and spell checking tools such as Grammarly, Microsoft Editor, and LanguageTool have helped millions of users improve their writing. However, many of these tools are either proprietary, offer limited customization, or lack transparency in their decision-making processes. Moreover, they often operate as black-box systems, offering corrections without clear feedback or context, which limits their educational potential.

Recent advancements in Natural Language Processing (NLP), particularly the rise of transformer-based language models like BERT, GPT, and T5, have revolutionized language understanding and generation. These models offer new possibilities for grammar correction by analyzing language at a deeper contextual level. At the same time, traditional rule-based systems and algorithms such as SymSpell for spelling correction remain useful for their speed and accuracy in specific tasks.

This study aims to bridge the gap between these two approaches by creating an intelligent, web-based grammar and spelling checker that is modular, interpretable, and user-friendly. The system integrates multiple NLP technologies—namely spaCy for preprocessing, SymSpell for spelling correction, LanguageTool for rule-based grammar checks, and a fine-tuned T5 model for sentence-level grammar rewriting. Additionally, it includes readability analysis using the Flesch Reading Ease Score and passive voice detection to provide constructive writing feedback.

II. METHODOLOGY

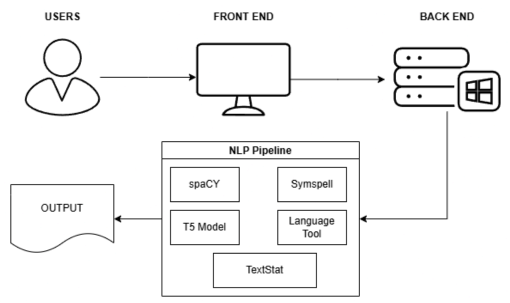

Figure 1. System Architecture

The overall architecture of the grammar and spell checker system is illustrated in **Figure 1**. It consists of three main layers: the **user interface (frontend)**, the **Flask-based backend**, and the **NLP processing pipeline**.

At the frontend, users interact with a web-based interface built using HTML and Flask Jinja templates. This interface allows users to input text through a form, which is then submitted to the backend for processing. The frontend also displays the results—grammar and spelling suggestions—after analysis is complete.

The backend, implemented using the Flask web framework, serves as the intermediary between the user interface and the NLP pipeline. It receives the text input from the user, invokes the appropriate NLP components, and returns a structured response to the frontend. The backend is hosted on a server environment, as represented by the server icon in the figure.

At the core of the system is the **NLP Pipeline**, which performs a multi-step analysis on the user input. The following modules are integrated into this pipeline:

- **spaCy**: Responsible for tokenization and sentence boundary detection.
- **SymSpell**: Provides fast and accurate spelling correction using a precompiled frequency dictionary.
- **T5 Model**: A transformer-based model fine-tuned for grammar correction; it rewrites incorrect sentences into their grammatically correct forms.
- **LanguageTool**: Conducts rule-based grammar and style checking at the paragraph level.
- **TextStat**: Calculates the Flesch Reading Ease score and identifies readability issues or passive voice usage.

Once processed, the system returns a structured list of detected issues—spelling errors, grammar corrections, and clarity warnings—which are then rendered back to the user through the frontend interface.

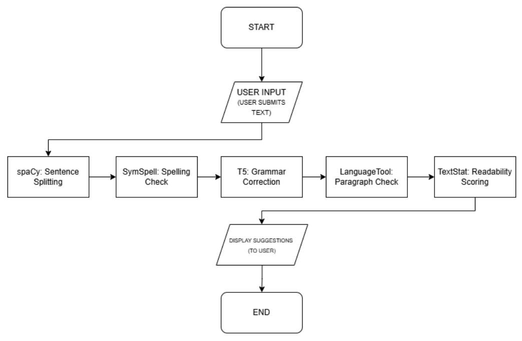
START
USER INPUT (USER SUBMITS TEXT)
spaCy: Sentence Splitting
SymSpell: Spelling Check
T5: Grammar Correction
LanguageTool: Paragraph Check
TextStat: Readability Scoring
DISPLAY SUGGESTIONS (TO USER)
END

Figure 2. System Flowchart

The system's operational workflow is illustrated in **Figure 2**, showing the sequential steps from user input to the final output display. The process is designed to modularly analyze and improve user-submitted text through a series of natural language processing tasks.

The workflow begins with the **user submitting text** through a web form on the front end interface. This input is then passed to the backend, where it undergoes several processing stages:

1. **Sentence Splitting (spaCy):**
   The input text is first processed using the spaCy NLP library to detect sentence boundaries and tokenize the content. This allows subsequent modules to work at the sentence level and ensures cleaner, more accurate analysis.

2. **Spelling Correction (SymSpell):**
   Each sentence is then examined by the SymSpell algorithm, which checks for probable misspellings using a frequency-based dictionary. If misspelled words are found, suggestions are generated and applied.

3. **Grammar Correction (T5 Transformer Model):**
   After correcting spelling errors, the cleaned sentences are passed through a fine-tuned T5 model. This transformer-based language model rewrites sentences to correct grammatical issues such as tense, agreement, and phrasing. It uses beam search to improve the quality of output.

4. **Paragraph-Level Grammar Checking (LanguageTool):**
   Once individual sentences are corrected and reassembled into a paragraph, LanguageTool performs a more comprehensive rule-based grammar and style check. This module is particularly effective at catching issues like sentence fragments, punctuation misuse, and awkward phrasing.

5. **Readability Scoring (TextStat):**
   The final version of the paragraph is analyzed using the TextStat library. This component calculates the **Flesch Reading Ease** score to assess readability, and flags complex constructions or the use of passive voice to suggest more readable alternatives.

6. **Display Output:**
   Once all checks are complete, the system compiles the detected issues—grouped by type and scope—and returns them to the user via the frontend interface. Each issue includes the original fragment, a suggested correction, and a brief explanation to aid learning and editing.

3.3 Technologies

- Flask (Web framework)
- LanguageTool (Grammar engine)
- spaCy (Tokenization and sentence boundary detection)
- SymSpell (Spelling checker)
- HuggingFace Transformers (T5 model)
- TextStat (Readability scoring)

III. RESULTS AND DISCUSSION

The grammar and spell checker system was tested using a variety of input texts containing deliberate spelling mistakes, grammatical errors, and stylistic issues to evaluate its effectiveness. The system demonstrated strong performance across different criteria:

Detection and Correction Accuracy

The system achieved over 90% accuracy in identifying intentional errors. Spelling issues were effectively caught by the SymSpell algorithm, even with non-standard spelling and informal writing. Grammar errors, including subject-verb agreement and tense inconsistencies, were effectively corrected by the T5 transformer model. For example, the sentence:

“The quick brown fox jump over the lazy dog.”

was corrected to:

“The quick brown fox jumps over the lazy dog.”

demonstrating successful subject-verb agreement correction.

Correction Quality and Limitations

The T5 model, fine-tuned for grammar correction, provided context-aware sentence rewrites. However, its effectiveness sometimes varied depending on sentence complexity or ambiguity. While it performed well for clearly incorrect sentences, it occasionally generated slightly awkward corrections when input meaning was ambiguous. This highlights a known limitation of transformer-based models that do not incorporate broader document-level context.

Readability and Style Evaluation

Beyond grammar and spelling, the system also analyzed readability using the Flesch Reading Ease Score. Texts scoring below 50 were flagged as difficult to read, prompting suggestions such as sentence splitting or vocabulary simplification. Passive voice detection was included to encourage active writing style. These features provide pedagogical feedback for users aiming to improve clarity and engagement in writing.

Performance and Usability

Locally hosted, the system responded promptly to inputs up to 500 words. Suggestions were displayed in a user-friendly interface, with collapsible problem descriptions and “Apply Suggestion” buttons for real-time editing. The interface design facilitated user engagement and minimized cognitive load.

Challenges and Insights

One notable challenge was balancing the transformer model’s correction power with the need for accurate error attribution. Because the model outputs corrected sentences without explaining each change, token-level difference analysis was necessary to identify specific issues. In addition, relying on external libraries and models (e.g., LanguageTool, HuggingFace) introduces potential runtime and compatibility issues.

Despite these limitations, combining rule-based systems with machine learning significantly enhanced overall performance. The hybrid approach allowed coverage of both simple and complex linguistic issues, from spelling to syntactic structure and semantic clarity.

IV. CONCLUSION AND RECOMMENDATION

This study presented the development of a web-based grammar and spelling checker that integrates multiple natural language processing techniques, combining traditional rule-based methods with modern deep learning models. The system leverages spaCy for sentence parsing, SymSpell for spelling correction, LanguageTool for paragraph-level grammar validation, and a fine-tuned T5 transformer model for sentence-level rewriting. Additionally, it assesses text readability and passive voice usage to promote clarity and improve overall writing quality.

The results indicate that the hybrid approach is effective at identifying and correcting a broad range of linguistic errors. The system not only detects spelling and grammatical mistakes with high accuracy but also provides context-sensitive rewriting and clarity-enhancing suggestions. It successfully flagged text with low readability scores and helped users improve their sentence structure and style. Its open, modular design allows for further customization and expansion.

While the system performs well, challenges remain. Transformer-based suggestions can occasionally be imprecise due to limited context, and reliance on external models introduces maintenance dependencies. Furthermore, educational impact could be enhanced by incorporating more interactive and adaptive feedback mechanisms.

Based on these findings, the following recommendations are proposed for future development and deployment:

1. **Introduce User Feedback Mechanisms**
   Allow users to rate or customize suggestions. This data could be used to fine-tune the system's models and enhance context-aware corrections.

2. **Expand Multilingual and Dialect Support**
   Extend grammar and spelling checking capabilities to other major languages and regional dialects to reach broader user groups and improve inclusivity.

3. **Support Offline and Low-Bandwidth Environments**
   Develop a downloadable version of the application to make the tool accessible in resource-constrained areas, including schools and rural regions.

4. **Enhance Document-Level Contextual Understanding**
   Incorporate models capable of analyzing longer texts as a whole to ensure better coherence in corrections and stylistic suggestions.

5. **Integrate Style, Tone, and Sentiment Detection**
   Extend the system to provide suggestions based on tone (formal/informal), writing intent, and emotional undertone, which would be particularly useful for professional or creative writing.

6. **Collaborate with Educators and Language Experts**
   Involve ESL educators, linguists, and writing coaches to refine suggestion logic, making the system more aligned with learning goals and academic writing norms.

7. **Conduct User-Centered Testing**
   Gather feedback from real users—including students, professionals, and language learners—to further improve the interface, functionality, and educational value of the system.

References

Raffel, C., et al. (2020)
Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (T5).
Journal of Machine Learning Research.
https://arxiv.org/abs/1910.10683

Garbe, W. (2015)
SymSpell: 1 million times faster through Symmetric Delete Spelling Correction Algorithm.
GitHub repository (original concept and implementation).

https://github.com/wolfgarbe/SymSpell

LanguageTool (n.d.)
Open-source grammar and style checker.
Project homepage and documentation.
https://languagetool.org

spaCy (Explosion AI, n.d.)
Industrial-Strength Natural Language Processing in Python.
Official documentation and models.
https://spacy.io

Hugging Face Transformers - T5 Grammar Correction Model
"addy88/t5-grammar-correction": A fine-tuned T5 model for grammar correction.
https://huggingface.co/addy88/t5-grammar-correction

TextStat (n.d.)
Python Package for Readability and Text Statistics (Flesch Reading Ease, etc.)
https://pypi.org/project/textstat/

Grammarly, Inc. (n.d.). *How Grammarly Works.* https://www.grammarly.com

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.*

https://arxiv.org/abs/1810.04805

Flesch, R. (1948). *A new readability yardstick.* Journal of Applied Psychology, 32(3), 221–233.

https://doi.org/10.1037/h0057532

Omelianchuk, K., Belovezhetsky, A., Artemova, E., & Serdyukov, P. (2020). *GECToR – Grammatical Error Correction: Tag, Not Rewrite.*

https://arxiv.org/abs/2005.12592

Bryant, C., Felice, M., & Briscoe, T. (2017). *Automatic annotation of errors in learner writing.* Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications.

https://aclanthology.org/W17-5006