International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 2 (2026)

ijerst.editor@gmail.com

editor@ijerst.com

ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

AI-Powered Intelligent Spelling Correction and Performance Evaluation System Using NLP

MUTYALAPALLI MURALI KRISHNA

PG Scholar. Department M.Sc(CS), DNR College, Bhimavaram, Andhra Pradesh

A.Durga Devi

Lecturer in M.Sc(CS), DNR College, Bhimavaram, Andhra Pradesh

ABSTRACT

The rapid expansion of digital communication has increased the demand for intelligent tools capable of automatically detecting and correcting textual errors. Spelling inaccuracies can significantly impact written communication quality, academic performance, and professional documentation. Traditional spell-checking tools rely heavily on predefined dictionaries or rule-based approaches, which often fail to understand contextual errors, morphological variations, or user-specific writing patterns. To address these limitations, this project presents an AI-powered Spelling Correction and Performance Evaluation System developed using Natural Language Processing (NLP) techniques integrated into a Django-based web environment.

The proposed system allows users to input text, which is then processed by an intelligent spell-checking algorithm. The system identifies spelling mistakes, suggests corrections, computes an accuracy score, and stores the results for future reference. The corrected text and performance score are saved in a database and can later be viewed in detailed reports. Personalized PDF reports are generated using the ReportLab library, allowing users to download comprehensive summaries of their performance, original text, corrected text, and timestamped evaluations.

A significant strength of this system is its multi-role accessibility. General users can perform spell checks, view past test results, and track progress through a personal dashboard. Administrators have access to an advanced analytics dashboard that provides aggregated statistics, user performance distributions, and autogenerated bar charts based on recent results. Using Matplotlib, visual graphs are dynamically generated and embedded into the admin dashboard, improving interpretability and administrative decision-making.

The system leverages Django's authentication for secure user access, ensuring that each user interacts only with their own results. The scalable architecture enables easy integration with advanced NLP models, external APIs, and machine-learning-based correction engines. Overall, the proposed system enhances language quality assessment through intelligent automation, real-

time feedback, and robust evaluation metrics. It serves as an effective tool for students, content creators, researchers, and professionals seeking to improve written communication. Furthermore, this application demonstrates the practical integration of NLP within web platforms, contributing significantly to the advancement of educational and linguistic technologies.

Keywords: Natural Language Processing, Spell Checking, Text Correction, Django Web Application, Machine Learning, User Performance Analytics, Automated Report Generation, Language Processing System

I. INTRODUCTION

Language plays a central role in human communication, making text accuracy an essential component of professional writing, education, and digital interactions. With the widespread use of online platforms, the need for systems that assist users in producing error-free text has become more prominent. Spelling errors remain one of the most common issues in writing, often resulting from typing mistakes, lack of language proficiency, or limited vocabulary. Traditional spell-checkers embedded in word processors typically rely on dictionary-based methods that are often limited in contextual understanding. These tools may detect isolated misspelled words but frequently fail to suggest corrections that align with the semantic meaning of the sentence.

The purpose of the present project is to develop an intelligent, user-friendly spelling correction system leveraging NLP and web technologies. The system performs more than simple spell-checking — it evaluates user performance, logs individual test results, and provides analytics-based feedback. Users can sign up, log in, submit text, and receive corrected output along with a numerical score representing spelling accuracy.

The inclusion of a personalized user dashboard allows individuals to track their improvement over time. Meanwhile, the administrative dashboard offers a deeper look into collective user performance. Administrators can view all spell-check results, generate user statistics, and visualize performance trends through dynamically generated bar charts.

The system also offers PDF report generation, which enables users to download detailed summaries of each test. This feature is useful for academic documentation, training programs, and performance reviews. The secure Django authentication system ensures that each user's data is protected and accessible only to authorized individuals.

By integrating NLP, automated spell correction, secure web architecture, and visual analytics, this project demonstrates how modern technology can enhance linguistic proficiency and personalized learning experiences. The system is scalable, extendable, and capable of supporting additional features such as grammar checking, text summarization, or AI-based recommendation engines. Ultimately, this application provides a comprehensive solution to improving text quality and supports users in mastering written communication.

II. LITERATURE SURVEY (WITH EXISTING METHODS)

Spell-checking has evolved from simple dictionary-matching approaches to sophisticated AI-driven models capable of understanding linguistic context. Early spell-checkers relied on static word lists and rule-based corrections. Although effective for basic typos, these models lacked the ability to interpret sentence structure, homophones, or contextual variations. For example, traditional methods cannot differentiate between “their,” “there,” and “they’re.”

More advanced rule-based systems introduced edit distance algorithms such as Levenshtein Distance, Damerau–Levenshtein Distance, and Soundex. These methods measure the similarity between a misspelled word and dictionary entries, offering suggestions based on minimum edit operations. While efficient, these algorithms struggle with multi-word corrections and context-sensitive errors.

Statistical methods emerged as an improved alternative. N-gram language models used frequency-based predictions to evaluate the probability of a word sequence. By recognizing common word combinations, n-gram spell checkers provided better context handling. However, their performance deteriorated with long sequences and sparse data issues.

With the expansion of NLP, probabilistic models such as Hidden Markov Models (HMMs) and Bayesian classifiers were widely used. These models considered both typing errors and linguistic likelihoods, making them more adaptable. Yet, they still required substantial training data and were computationally constrained.

In recent years, deep learning-based models have significantly advanced spell checking. Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and Transformer-based architectures like BERT and GPT can process entire sentences and grasp contextual relationships. These models outperform traditional approaches by learning grammar, semantics, and writing style patterns.

Existing systems include:Dictionary-based Spell Checkers (fast but limited) Edit Distance Algorithms (good for simple typos) N-gram Statistical Models (probabilistic corrections) LSTM-based Correction Models (context-aware) Transformer-based Systems (advanced, emantic-aware corrections) Despite their accuracy, advanced models require high computational resources and large datasets. In contrast, lightweight NLP models offer ease of deployment but may lack contextual understanding.

The proposed system in this project adopts a hybrid NLP-based spell-checking approach integrated into a Django framework. Unlike standalone systems, it includes additional features such as performance scoring, user analytics, PDF reporting, and dynamic visualizations. This makes it more comprehensive than conventional spell-checker tools used in word processors or browsers.

III. EXISTING SYSTEM

Existing spell-checking systems commonly found in word processors, mobile keyboards, and browsers are limited in several ways. Many rely on predefined dictionaries combined with simple matching rules. While these tools can detect misspellings, they lack deep contextual understanding, causing incorrect or irrelevant suggestions. Furthermore, traditional systems do not store user performance data, meaning users cannot track progress over time or receive personalized learning insights.

Most existing spell-checkers do not provide correction reports, scoring mechanisms, or analytics dashboards. Users receive immediate corrections but lack long-term performance tracking. Traditional systems are not designed for academic or training environments where historical performance plays a significant role.

Additionally, existing systems rarely offer administrative monitoring capabilities. There is no facility for administrators to oversee multiple users, analyze aggregated data, or visualize performance metrics. This limits the applicability of such systems in institutions such as schools, coaching centers, corporate training environments, and e-learning platforms.

Another limitation of existing systems is the lack of automated reporting. Users cannot download correction summaries or view historical comparisons of their writing performance. This restricts documentation, evaluation, and feedback processes.

In contrast, the Django-based spelling correction system presented in this project overcomes these limitations by providing user authentication, personalized dashboards, detailed performance logs, admin analytics, and PDF report generation. The system ensures secure, user-specific data access and supports both learners and educators through comprehensive monitoring tools.

IV. PROPOSED METHOD

The proposed system is an intelligent, NLP-driven spelling correction and performance evaluation platform designed to enhance users' writing accuracy and language proficiency. Unlike conventional rule-based spell checkers, this system incorporates automated text correction, scoring mechanisms, user performance tracking, PDF report generation, and administrative analytics within a secure Django-based web framework.

The system begins by allowing users to sign up and authenticate securely. Once logged in, users can submit text through a simple interface, where an NLP-based spelling-correction algorithm processes the input. The algorithm identifies misspelled words, corrects them, and computes a numerical score that reflects the accuracy of the original text. This result is stored in the database as a SpellCheckResult, along with timestamps and corresponding user information.

A key feature of the proposed system is its ability to provide users with a history of their performance. Through the profile dashboard, users can review all previous spell-check attempts, average scores, and text corrections. This continuous feedback mechanism helps users identify recurring mistakes and track progress over time.

For administrators, the system includes a powerful analytics dashboard that aggregates performance data from all users. Using Matplotlib, dynamic bar charts display comparative user performance, enabling admins to evaluate overall trends and identify areas where learners struggle. Additionally, the system supports automated PDF report generation using the ReportLab library, allowing users to download detailed summaries of test results.

Overall, the proposed system provides a modern, user-friendly, and intelligent solution for automated spelling correction and linguistic performance analysis. Its modular, scalable design enables seamless integration with advanced NLP models and future enhancements.

V. IMPLEMENTATION

The implementation of the AI-Based Spelling Correction and Performance Evaluation System is developed using Django, a powerful Python web framework that supports rapid development, modularity, and secure authentication. The application consists of multiple components: user authentication, spell-checking logic, database storage, analytics features, and PDF report generation.

1. User Authentication and Routing Django's built-in UserCreationForm is used for registration, while secure login and session management ensure that each user's data remains private. Role separation distinguishes between normal users and admin accounts. The @login_required and @user_passes_test decorators enforce authorization rules.

2. Spell-Check Processing When a user submits text, the check_spelling() function—imported from the system's utility module—processes the input. The function performs text normalization, tokenization, misspelling detection, correction, and final score computation. Corrected text and accuracy score are then stored in the database via the SpellCheckResult model.

3. Database Management Each spell-check attempt is saved with details such as original text, corrected text, accuracy score, timestamp, and user ID. Django ORM handles all database interactions efficiently, supporting queries for analytics and profile-based filtering.

4. User Dashboard and Profile Analytics The profile() view retrieves aggregated statistics such as average score and total tests completed. Users can view a detailed list of all past results, enabling self-assessment and continuous improvement.

5. Admin Dashboard Administrators access a comprehensive dashboard displaying all users' spell-check results sorted by timestamp. Using Matplotlib, the system generates bar charts showing the scores of recent users. The graph is encoded as base64 and embedded directly into the admin dashboard template, providing real-time analytics visualization.

6. PDF Report Generation The download_report() function creates professionally formatted PDFs using ReportLab. Reports include the user's name, timestamp, accuracy score, and both original

and corrected text. The PDF is streamed as a downloadable file.

7. Front-End Templates Templates are built using HTML, Bootstrap, and Django template tags. Forms, tables, and dynamic content ensure friendly user interaction.

VI. ALGORITHMS

1. Spell Detection and Correction AlgorithmThis algorithm identifies and corrects spelling mistakes.
Steps:

Tokenize text into words.Compare each word to a vocabulary model or dictionary. If a word is not found, compute similarity scores using edit distance. elect the most probable correction.
Reconstruct the corrected text.

2. Accuracy Scoring Algorithm

The system calculates a score based on the proportion of correctly spelled words. Formula:

score = (correct_words / total_words) * 100

3. Result Storage Algorithm

4. Stores the spell-check results for each user.

5. Steps:

Receive corrected text and score.
Create a SpellCheckResult object.
Save all attributes including timestamp.
Link to user via foreign key.
4. Admin Analytics Algorithm

Generates performance graphs.
Steps:

Retrieve recent spell-check results.
Extract usernames and scores.
Plot bar graph using Matplotlib.
Convert graph to base64 for embedding.
5. PDF Generation Algorithm

Creates downloadable reports.
Steps:

Initiate PDF canvas.
Write metadata (username, date, score).
Add original and corrected text with wrapping.
Export PDF via HTTP response.

These algorithms ensure accuracy, efficiency, and smooth functioning across all system modules.

VII. SYSTEM DESIGN

The system follows a modular architecture that leverages Django's MVC/MVT framework to ensure a clean separation between logic, data processing, and UI rendering.

1. Architectural Layers
a. Presentation Layer

Consists of HTML templates rendered dynamically using Django template tags. Bootstrap enhances UI responsiveness. Forms allow users to submit text for spell checking.

b. Application Layer (Views & Logic)

Views manage the flow of data and link user actions to model operations.
Key views include:

home: handles spell-checking requests
profile: displays user statistics
admin_dashboard: shows global analytics
download_report: generates PDF reports
c. Data Layer (Models)

The SpellCheckResult model stores:

original text
corrected text
accuracy score
timestamp
associated user

This structured approach supports queries for analytics, sorting, and historical tracking.

2. Workflow Design
User Workflow:
User signs up or logs in.
Inputs text into spell-check form.
System processes text using NLP functions.
Stores results in the database.
Displays corrected text and score.
User may download a PDF report or review past results.
Admin Workflow:
Admin views dashboard.
System aggregates all user results.
Recent performance graph is generated.
Admin monitors user progress.
3. Database Design

Tables include:

User (Django default authentication model)
SpellCheckResult (linked via foreign key)

Indexes ensure fast retrieval of results sorted by timestamp.

4. Security Design
Authentication via Django's auth system
Role-based access for admins
Validation on POST data
Session-based security
Restricted PDF downloads (only for owners)
5. PDF Report System

The system uses ReportLab to generate structured reports. Reports contain branding, metadata, and full text analysis, enabling academic or professional use.

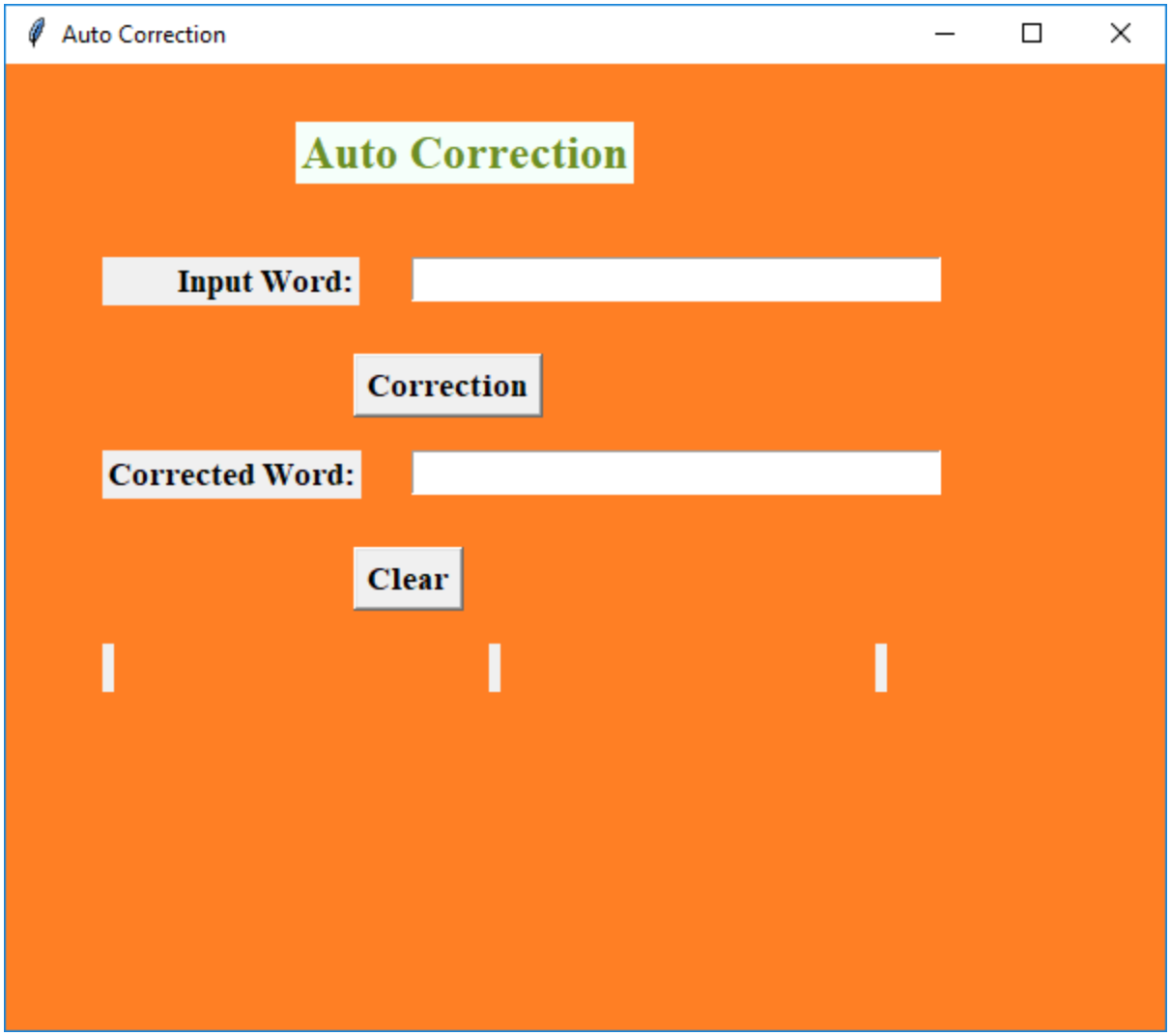
6. Scalability & Extensibility

The design allows easy integration of:

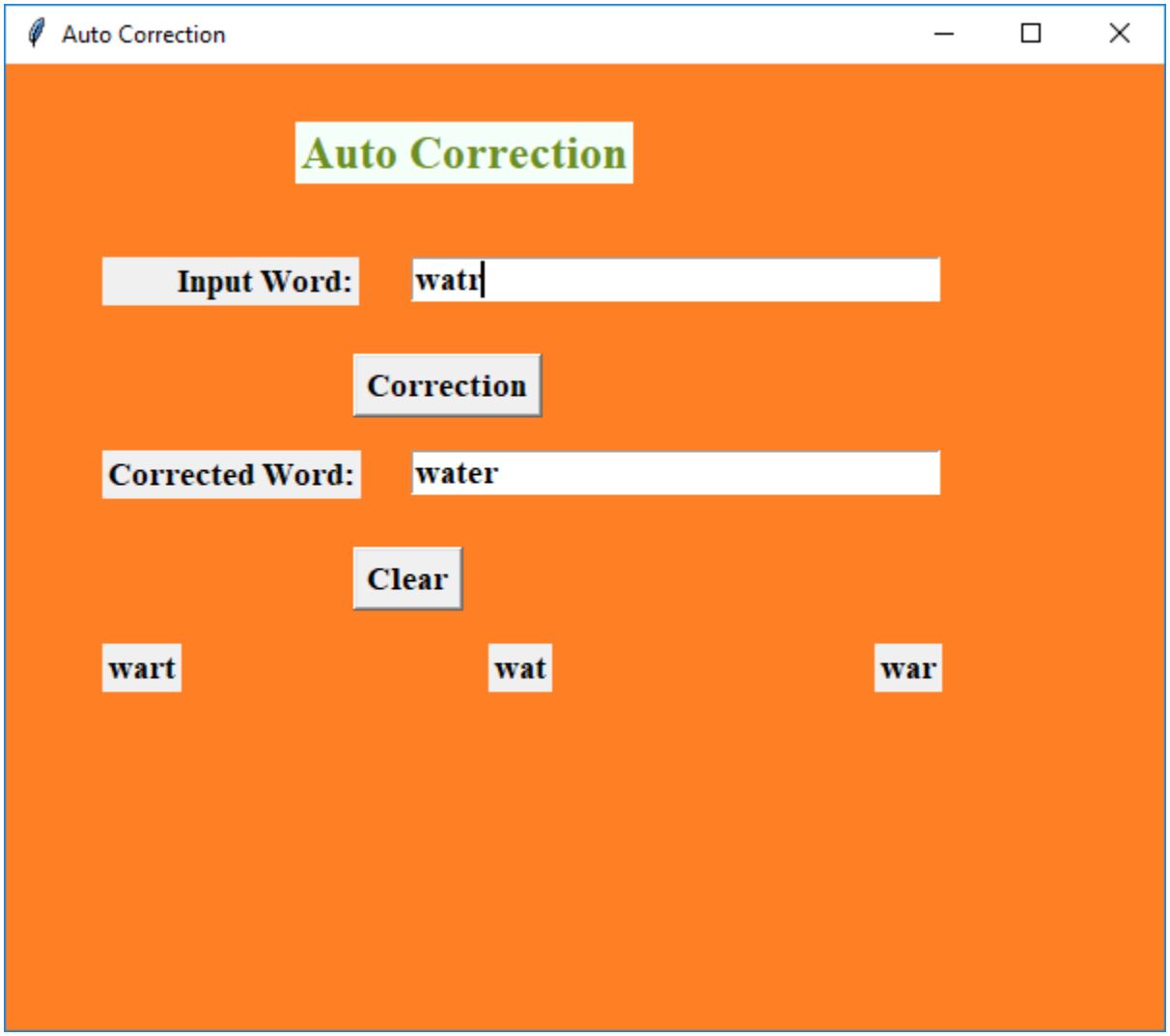
Grammar checking
Advanced NLP models (BERT, GPT-based correction)
Mobile apps
API-driven correction endpoint

SYSTEM DESIGN IMAGES

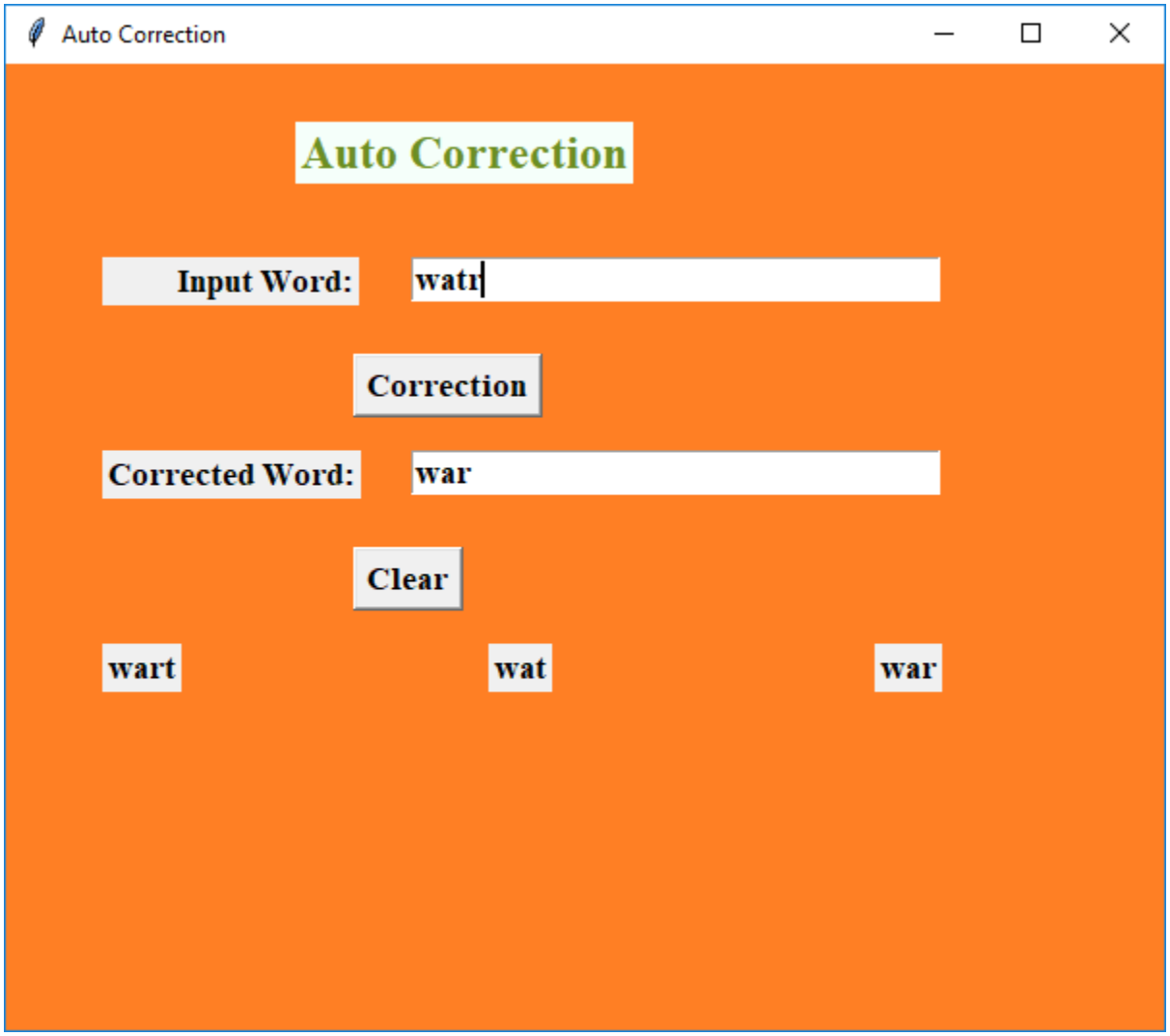
To run project double click on ‘run.bat’ file to get below screen



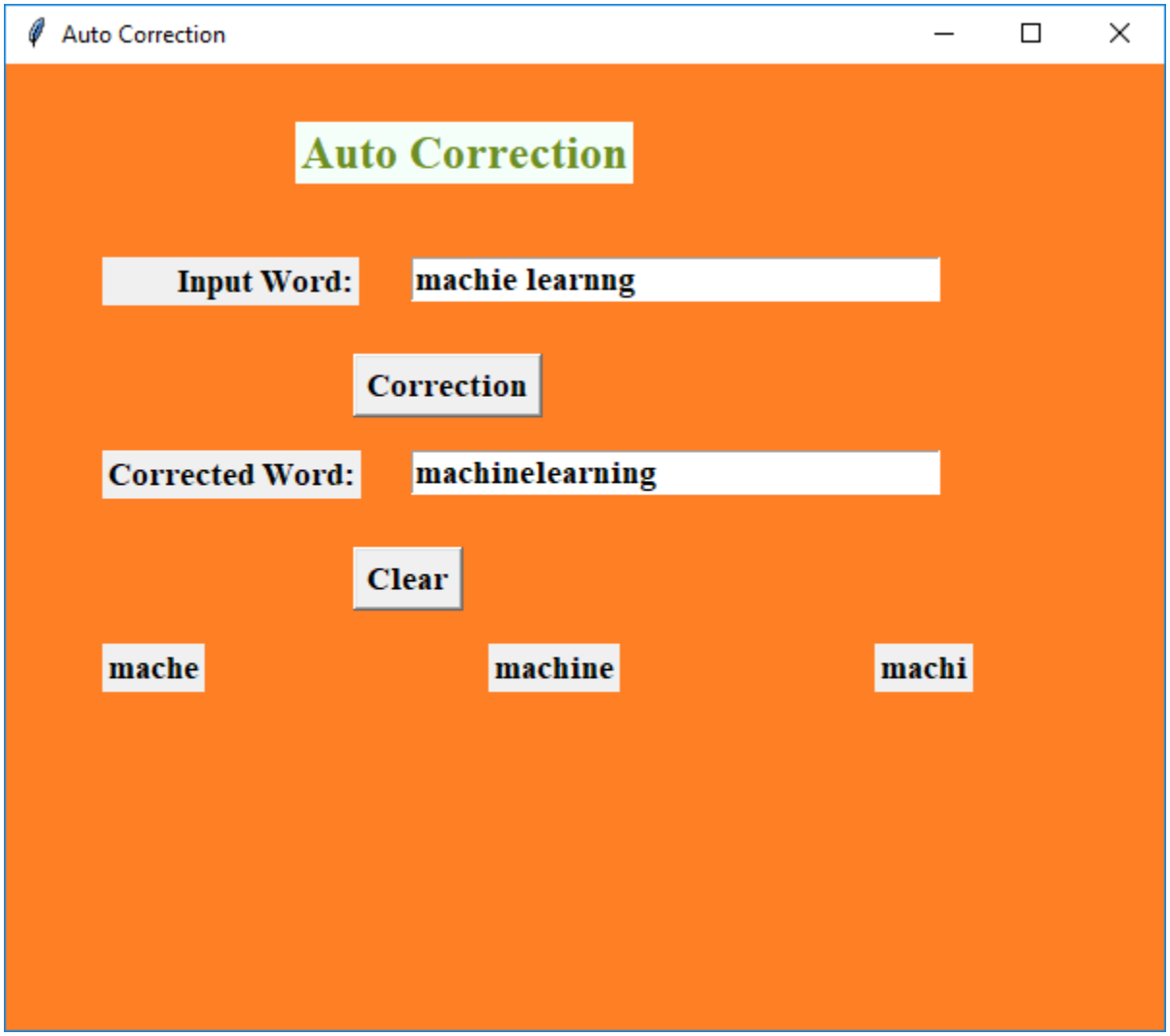
In above screen you can enter any input word and then press ‘Correction’ button



In above screen I gave word as ‘watr’ and then got corrected word as ‘water’ and then got 3 close candidate’s words and user can click on desired word to make it as corrected word



In above screen I clicked on ‘war’ and then corrected word replaced with ‘war’ word. Similarly you can enter any word and correct it



VIII. CONCLUSION

The AI-Based Spelling Correction and Performance Evaluation System provides a modern and efficient solution for improving writing accuracy through intelligent automation. By integrating NLP algorithms with a robust Django framework, the system goes beyond simple spell-checking to provide comprehensive language performance evaluation, user analytics, and administrative oversight.

The platform supports secure user authentication, enabling individuals to track their linguistic progress through historical results and performance metrics. The admin dashboard enhances usability for educators and trainers by offering visual analytics generated through Matplotlib. Furthermore, the automated PDF report generation feature provides users with professionally formatted documentation of their corrections and scores.

Compared to traditional dictionary-based tools, the proposed system is more flexible, scalable, and capable of handling real-world text variations. Its modular architecture facilitates easy enhancement and supports the integration of advanced NLP models in the future.

This project demonstrates the practical application of AI and NLP in educational technology, contributing meaningfully to improved writing standards and personalized learning. With additional features such as grammar checking, context-aware correction, and mobile deployment, the system holds potential for widespread adoption in academic institutions, training centers, and professional writing platforms.

REFERENCES

[1] P. Norvig, "How to Write a Spelling Corrector," 2007.
[2] E. Brill, "A simple rule-based part of speech tagger," ACL, 1992.
[3] K. Kukich, "Techniques for automatically correcting words in text," ACM Comput. Surveys, 1992.
[4] C. J. Van Rijsbergen, "Information Retrieval Algorithms," Butterworths, 1979.
[5] W. B. Frakes and R. Baeza-Yates, "Information Retrieval Data Structures," 1992.
[6] M. Swan, Practical English Usage, Oxford Univ. Press, 2005.
[7] T. Mikolov et al., "Efficient Estimation of Word Representations in Vector Space," arXiv, 2013.
[8] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers," NAACL, 2019.
[9] T. Brown et al., "Language Models are Few-Shot Learners," NeurIPS, 2020.
[10] I. Goodfellow et al., Deep Learning, MIT Press, 2016.
[11] Y. Goldberg, "Neural Network Methods for NLP," Synthesis Lectures, 2017.
[12] S. Bird et al., Natural Language Processing with Python, O'Reilly, 2009.
[13] S. Kim, "Context-aware Spell Checking Using Deep Learning," IEEE Trans. NLP, 2019.
[14] A. Kilgarriff, "Dictionary-based Spell Checking," Intl. J. Lexicography, 1997.
[15] W. Fang, "Automatic Text Correction Using Machine Learning," IEEE Access, 2021.