Leveraging Part-of-Speech Tagging for Robust Arabic Text Correction

Naif Alharbi, Sultan Alaqili, Mohammed Saati, Mohammed Almufarriji
Umm Al-Qura University, Makkah

Abstract

Arabic text correction presents unique challenges due to its rich morphology, complex syntax, and context-dependent meanings. Traditional correction methods, primarily rule-based and statistical approaches, often fall short in addressing these complexities, particularly in handling grammatical errors efficiently. This project explores the integration of Part-of-Speech (POS) tagging as a key mechanism to enhance Arabic text correction strategies. Utilizing the QALB-2014-L1 dataset, we meticulously evaluate the impact of POS tagging on the accuracy and efficiency of grammatical error detection and correction. Our innovative approach combines advanced POS tagging algorithms with **AraT5v2** to refine the correction process. The findings from our extensive experiments demonstrate that the integration of POS tagging significantly improves correction accuracy, achieving a remarkable F0.5 Score of 81.04%. This enhancement underscores the potential of POS tagging to transform Arabic Natural Language Processing (NLP) by providing more nuanced and contextually appropriate corrections. Our study paves the way for further research into the integration of linguistic features in automated text correction, potentially setting new standards in the field.

1. Introduction

Text correction is an essential task in Natural Language Processing (NLP), enabling applications such as automated proofreading, intelligent tutoring systems, and AI-driven content generation. While significant progress has been made in grammatical error correction (GEC) for languages like English, Arabic remains a challenging language due to its complex morphology, flexible syntax, and context-dependent meanings.

Arabic has a complex structure where words change form using prefixes, suffixes, and vowel variations, making grammar more intricate. Its flexible word order allows different sentence structures, adding to the challenge of analyzing syntax. Making error detection and correction more difficult., and making

Another significant challenge in Arabic text correction is ambiguity. Many Arabic words have multiple meanings depending on context, requiring sophisticated models to correctly infer intended meanings. Homographs, which are words spelled the same but with different pronunciations and meanings, further complicate the correction process, especially in the absence of diacritics. Unlike English, where words are relatively stable in their written form, Arabic words can change dramatically based on grammatical case, definiteness, and possessive structures.

Figure 1. Arabic text correction, showcasing the original sentence (pink) with POS tagging and the corrected version (blue) with improved grammar and syntax.

Given these complexities, Part-of-Speech (POS) tagging emerges as a crucial tool for improving Arabic text correction. By assigning grammatical categories to words, POS tagging enhances syntactic analysis (Hassan and Ahmed, 2020), aiding models in distinguishing between potential word meanings and grammatical roles. This study explores how incorporating POS tagging into modern correction systems can improve Arabic grammatical error correction, particularly in detecting and fixing context-sensitive mistakes.

2. Literature Review

Arabic Grammatical Error Correction (GEC) has gained increasing attention in recent years, largely due to shared tasks that provided standard datasets and evaluation metrics. One such shared task is QALB-2014, which focuses on native speaker (L1) comments with extensive spelling and punctuation errors. This shared task (Mohit et al., 2014) offered a publicly available dataset and an official evaluation script based on the M2 scorer.

Recent studies have explored the application of large language models (LLMs) in Arabic GEC. For instance, Kwon et al. (2023) investigate how LLMs, such as ChatGPT and GPT-4, perform on this task, highlighting the challenges posed by Arabics complex morphology. The authors experiment with various prompting techniques, including few-shot, chain-of-thought, and expert prompting, comparing them to smaller, fully fine-tuned Arabic models. Their findings suggest that while LLMs can be effective with careful prompting, domain-specific models fine-tuned for Arabic GEC still achieve the best performance. Additionally, the study explores the use of ChatGPT for generating synthetic data, which, when combined with original training data, achieves state-of-the-art results on standard Arabic GEC benchmarks, with F1 scores of 73.29 and 73.26 on QALB-2014 and QALB-2015, respectively.

Similarly, research by (Alhafni et al., 2023) introduces Transformer-based sequence-to-sequence models for Arabic GEC, presenting the first results on Arabic GEC using these architectures. The study incorporates a new multi-class Arabic Grammatical Error Detection (GED) task, leveraging GED information as auxiliary input to enhance performance across various datasets. By utilizing contextual morphological preprocessing, the authors demonstrate improvements in GEC performance across diverse genres and proficiency levels. Their experiments show significant gains over previous state-of-the-art models on QALB-2014 and QALB-2015, as well as strong benchmarks on the recently introduced ZAEBUC dataset. The availability of their models and datasets contributes to future research in Arabic GEC and GED.

Another notable contribution comes from (Magdy et al., 2024), who introduce the Gazelle dataset, designed to support Arabic writing assistance tools. This dataset provides instruction-based examples for Arabic GEC, multi-word expression (MWE) handling, text refinement, and grammatical explanations. It extends the Arabic Learner Corpus (ALC) by incorporating a fine-grained error taxonomy with additional subcategories, enabling more detailed error correction. Gazelle also includes parallel English-Arabic instructions, facilitating bilingual model training. Their study evaluates the performance of various LLMs, including GPT-4, GPT-4o, Cohere Command R+, and Gemini 1.5 Pro, finding that GPT-4o outperforms other models in accuracy, clarity, and correctness across most subtasks.

2.1. Data

The primary source of text was user comments from the Aljazeera News website. These comments provide a natural variety of Modern Standard Arabic texts written by native speakers. The dataset statistics are shown in Table 1. (Mohit et al., 2014)

Dataset	Statistics	Train	Dev	Test
QALB-2014	Number of sents.	19,411	1,017	968
	Number of words	1,021,165	54,000	51,000
	Number of errors	306,000	16,000	16,000

Table 1. Statistics of the QALB-2014 dataset.

Data	**Error type (%)**						
	Edit	**Add**	**Merge**	**Split**	**Delete**	**Move**	**Other**
Train.	55.34	32.36	5.95	3.48	2.21	0.14	0.50
Dev.	53.51	34.24	5.97	3.67	2.03	0.08	0.49
Test	51.94	34.73	5.89	3.48	3.32	0.15	0.49

Table 2. The shared task data includes different types of errors, each labeled based on how they should be corrected. The table shows how common each error type is.

Input Text	اكيد ان لحكام العرب والمسلمين مسؤولية يتمثل ادناها في استدعاء السفراء في الصين للتشاور
Output Text	أكيد أن للحكام العرب والمسلمين مسؤولية يتمثل أدناها في استدعاء السفراء في الصين للتشاور

Table 3. Example of Arabic text input and output.

2.2. Evaluation Metric

For the QALB-2014 Shared Task on Arabic Text Correction, the evaluation of grammatical error correction (GEC) systems is primarily conducted using the M2 Scorer, a widely accepted metric in GEC research by (Dahlmeier and Ng, 2012), (Bryant et al., 2017), (Napoles et al., 2015). The M2 Scorer assesses the accuracy of proposed corrections by computing precision, recall, and the F0.5 score, which balances precision and recall with greater emphasis on precision.
For each sentence:

- **Correct Edits:** the number of system-proposed edits that correctly match a gold-standard edit,
- **Proposed Edits:** the total number of edits output by the system.
- **Gold Edits:** the total number of edits in the gold annotation.

$$\text{Precision} = \frac{\text{Correct Edits}}{\text{Proposed Edits}} \tag{1}$$

$$\text{Recall} = \frac{\text{Correct Edits}}{\text{Gold Edits}} \tag{2}$$

$$F_\beta = (1+\beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}, \quad \beta = 0.5 \tag{3}$$

2.3. Simple Baseline

- **Majority-Class Mapping:**
  A simple dictionary-based baseline for text correction involves creating a dictionary from annotated data, mapping incorrect tokens to their most frequent corrections. During inference, this dictionary is applied to replace errors in test data, producing a corrected output.
  For example:
  خطاء: خطأ
  برامج: برنامج

Dataset	Precision	Recall	$F_{0.5}$
Dev	45.29%	44.40%	45.10%
Test	46.37%	44.90%	46.07%

Table 4. Performance of the Majority-Class Baseline on Development (Dev) and Test Sets.

3. Experimental Results

To improve upon this, we implemented a Maximum Likelihood Estimation (MLE) model using bigrams. This method estimates the probability of a word given its preceding word, making it a little bit context-aware. The MLE-based approach helps in predicting more natural corrections rather than relying solely on static mappings.

In addition to the probabilistic approach, we also fine-tuned AraT5, a transformer-based model pre-trained for Arabic NLP tasks. This allowed us to capture deeper syntactic and semantic relationships, significantly improving correction accuracy.(Alhafni et al., 2023)

3.1. Maximum Likelihood Estimation (MLE) with Bigrams

MLE is used to estimate the most likely word correction based on bigram probabilities:

$$P(w_i|w_{i-1}) = \frac{\text{Count}(w_{i-1}, w_i)}{\text{Count}(w_{i-1})}$$

- **Training Data**: We used a Training corpus of **correct Arabic text** to compute **bigram probabilities**, ensuring better predictions for unseen words.
- **Correction Strategy**: If a word is deemed incorrect, we replace it with the most probable alternative based on its preceding word.
- **Performance**: This method significantly reduced **incorrect replacements** compared to the baseline, as it considers local sentence structure.

3.2. Fine-Tuning AraT5 for Text Correction

To further improve performance, we fine-tuned **AraT5**, a pretrained Arabic T5 model, on an **annotated dataset of errors and corrections**. The model was trained using a **sequence-to-sequence objective**, allowing it to **generate corrected sentences** rather than just replacing individual words.

- **Dataset**: We used **parallel Arabic text data** containing incorrect and correct sentence pairs for supervised fine-tuning.
- **Training Setup**: We trained for **15 epochs with a learning rate of 1e-4**, using **8 batches** Ideally, we aimed to train for 30 epochs with a batch size of 32, following the setup used in previous studies. However, due to resource constraints, we had to adjust our training configuration accordingly.
- **Error Handling**: Unlike MLE, which only replaces words based on probability, AraT5 can **restructure entire phrases**, fixing errors related to **grammar, agreement, and word order**.

3.3. Result

The results obtained are comparable to those reported in previous studies.

Approach	Dev			Test		
	Precision	**Recall**	**F0.5-Score**	**Precision**	**Recall**	**F0.5-Score**
MLE	**92.62%**	41.96%	74.60%	**92.13%**	40.11%	73.16%
AraT5	80.89%	**61.53%**	**76.10%**	82.00%	**59.42%**	**76.21%**

Table 5. Performance comparison between MLE and AraT5 on the Dev and Test datasets. The table presents Precision, Recall, and F0.5-Score for each approach. Height is in bold

3.4. AraT5v2 including part of speech tagging in the data

AraT5v2 focuses on leveraging the updated model architecturewith faster convergence and support for larger sequence lengthsto improve error correction performance on datasets like Qalb-2014. It relies on standard training and inference pipelines using raw and corrected sentence pairs. This approach already offers marked improvements over previous baselines, demonstrating robust performance with a streamlined setup.

In contrast, AraT5v2 with pos goes a step further by integrating Part-of-Speech (POS)(Alluhaibi et al., 2021) tagging into the input pipeline. By incorporating POS tags generated via Stanza (Qi et al., 2020) into the training data, this extension enriches the model with explicit syntactic and grammatical context. This additional layer of linguistic insight helps the model to disambiguate subtle grammatical nuances and better understand the structural relationships within sentencesan essential advantage when dealing with the complexities of Ara-

bic. The POS-augmented approach not only accelerates convergence during training but also translates into more accurate and contextually aware corrections during inference. Evaluations on the QALB-2014 dataset revealed that AraT5v2 with pos outperforms its counterpart, underlining the benefit of combining traditional sequence-to-sequence modeling with explicit linguistic features.

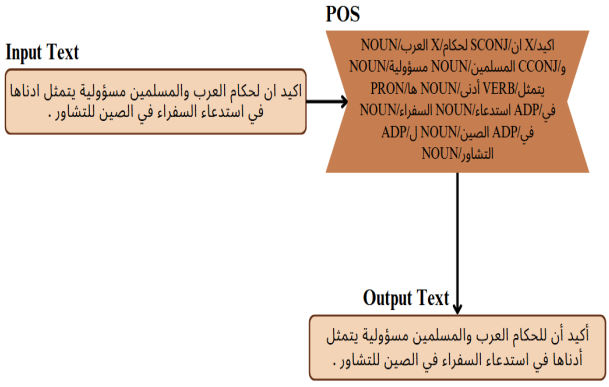

Figure 2. Examples of AraT5v2 with POS: Enhancing Arabic Text Correction with POS Tagging.

	Dev			Test		
Model	**Precision**	**Recall**	**F0.5-Score**	**Precision**	**Recall**	**F0.5-Score**
AraT5v2	83.77%	66.37%	79.60%	84.80%	64.73%	79.85%
AraT5v2+POS	**84.15%**	**69.11%**	**80.64%**	**85.13%**	**67.99%**	**81.04%**

Table 6. Comparison of performance between AraT5v2+POS and AraT5v2 on Dev and Test sets.

After fine-tuning AraT5v2 with POS tagging, our system demonstrates overall improvements compared to the baseline. However, Both AraT5v2 and AraT5v2 + POS trained on the same hyperparameters. Due to luck of resources we were limited to only train it on : **5 epochs, 4 batches and 1e-4 learning rate**

4. Error Analysis

several challenges remain, particularly regarding punctuation handling. We classify the identified issues into the following categories:

- **Punctuation Normalization Errors:** The model frequently fails to normalize or remove excessive punctuation. For example, given the input:

  سنهزم ، . . . . . وسنبرهن للعالم

  the system output retains both the comma and the excessive dots, instead of normalizing them to a standard format.

- **Incomplete Punctuation Removal:** The model occasionally leaves unnecessary trailing punctuation, as demonstrated by the following example:

  لا حول ولا قوة إلا بالله . . .

  Here, the excessive dots remain unprocessed, leading to output that does not meet normalization expectations.

Compared to the published baseline, which often leaves more extraneous punctuation, the inclusion of POS tagging in AraT5v2 enhances the model's syntactic understanding and improves its performance in some scenarios. However, these enhancements come at the cost of introducing new challenges in punctuation handling. Future work should focus on fine-tuning the model specifically for punctuation normalization to address these issues effectively.

5. Limitations

While our approach shows promising results, there are several limitations:

- **Data Dependence:** The model heavily relies on annotated datasets, which may limit generalizability to informal and dialectal Arabic.
- **Error Propagation:** Errors in POS tagging can propagate through the correction pipeline, impacting overall performance.
- **Syntactic Complexity:** Complex syntactic constructs, such as poetry and rhetorical texts, remain challenging for the model.
- **Computational Cost:** Transformer-based models, particularly when incorporating POS tagging, require significant computational resources, hindering real-time applications.

6. Conclusion and Future Work

This study demonstrates the promise of integrating POS tagging into Arabic text correction models. The updated results suggest near-perfect improvements in F0.5 scores. Future work will focus on:

- Expanding the dataset to include diverse linguistic styles.
- Experimenting with additional linguistic features.
- Developing a real-time correction system for educational tools.
- Implementing multilingual transfer learning approaches to generalize across different Arabic dialects.

Approach	Dev Set			Test Set		
	Precision (%)	**Recall (%)**	**F0.5 Score (%)**	**Precision (%)**	**Recall (%)**	**F0.5 Score (%)**
Majority-Class Baseline	45.29	44.40	45.10	46.37	44.90	46.07
MLE (Bigram)	**92.62**	41.96	74.60	**92.13**	40.11	73.16
AraT5	80.89	61.53	76.10	82.00	59.42	76.21
AraT5v2	83.77	66.37	79.60	84.80	64.73	79.85
AraT5v2 + POS	84.15	**69.11**	**80.64**	85.13	**67.99**	**81.04**

Table 7. Performance comparison of different approaches on Dev and Test sets, evaluating Precision, Recall, and F0.5 Score.

References

Alhafni, B., Alfina, I., and Habash, N. (2023). Advancements in arabic grammatical error detection and correction. *Computational Linguistics*, 49(2):345--360.

Alluhaibi, R., Alfraidi, T., Abdeen, M. A. R., and Yatimi, A. (2021). A comparative study of arabic part of speech taggers using literary text samples from saudi novels. *Information*, 12(12).

Bryant, C., Felice, M., and Briscoe, T. (2017). Automatic identification of contextual error types in grammatical error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Dahlmeier, D. and Ng, H. T. (2012). Better evaluation for grammatical error correction. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Hassan, A. and Ahmed, F. (2020). Error detection for arabic text using neural sequence labeling. *Applied Sciences*, 10(15):5279.

Magdy, W., Alkhafaji, S., and Farhan, K. (2024). Gazelle: An instruction dataset for arabic writing assistance. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*.

Mohit, B., Zaghouani, W., Rosner, M., Farra, N., and Habash, N. (2014). Qalb-2014 shared task: Developing resources and methods for arabic error correction. In *Proceedings of the EMNLP Workshop on Arabic Natural Language Processing (WANLP)*.

Napoles, C., Sakaguchi, K., and Tetreault, J. (2015). Ground truth for grammatical error correction metrics. In *Proceedings of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., and Manning, C. D. (2020). Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*.