Simplify: Automatic Arabic Sentence Simplification using Word Embeddings

Yousef SalahEldin
German International University,
New Administrative Capital, Egypt
yousef.hamouda@student.giu-uni.de

Caroline Sabty
German International University,
New Administrative Capital, Egypt
caroline.sabty@giu-uni.de

Abstract

Automatic Text Simplification (TS) involves simplifying language complexity while preserving the original meaning. The main objective of TS is to enhance the readability of complex texts, making them more accessible to a broader range of readers. This work focuses on developing a lexical text simplification system specifically for Arabic. We utilized FastText and Arabert pre-trained embedding models to create various simplification models. Our lexical approach involves a series of steps: identifying complex words, generating potential replacements, and selecting one replacement for the complex word within a sentence. We presented two main identification models: binary and multi-complexity models. We assessed the efficacy of these models by employing BERTScore to measure the similarity between the sentences generated by these models and the intended simple sentences. This comparative analysis evaluated the effectiveness of these models in accurately identifying and selecting complex words.

1 Introduction

Automatic Text Simplification (TS) aims to make text less linguistically complex without changing its meaning or original information. This involves rewriting a complex text by performing various edit operations such as deletion, replacing words, splitting sentences, and changing the order of words. These actions are part of the TS natural language processing task (Al-Thanyyan and Azmi, 2021).

TS can benefit individuals who struggle with reading and writing, such as those with low literacy skills, dyslexia, or learning a new language. Different simplification techniques can be employed depending on the desired purpose and the end user. Additionally, TS can enhance written communication by ensuring that the target audience comprehends the intended message. (Rello et al., 2013). In addition, automated systems for simplifying text can help make the language more accessible to individuals who are not fluent in it or have limited proficiency.

Detecting text complexity is crucial in TS systems as it helps determine if the text needs to be simplified. It is also helpful in evaluating the results generated by the simplification system. TS systems primarily depend on syntax or lexical simplifications. (Shardlow, 2014).

Text simplification is related to techniques such as creating paraphrases, summarizing text, and machine translation in Natural Language Processing (NLP). Many strategies and evaluation methods used by Text Simplification are derived from these areas. In the past, rule-based syntactic simplification was used as a pre-processing step to improve various NLP tasks like parsing and formulating questions. (Sikka and Mago, 2020).

Arabic is a widely spoken language consistently listed as one of the top 10 most spoken languages. This emphasizes the importance of incorporating different natural language processing tasks for Arabic (Hatab et al., 2022). We utilized the latest technologies in the field of NLP to carry out a straightforward simplification task. We developed two models for identification purposes: one that categorizes text as either complex or non-complex and another that classifies text into various levels of complexity. As a result, we utilized BERT (Devlin et al., 2018) and FastText (Grave et al., 2018) to create the simplification model. We assessed the simplification phase using BERTScore (Zhang* et al., 2020), which involved the two identification models. Furthermore, we conducted a manual evaluation to ensure the quality of the simplified text.

2 Related Work

Unlike English and other languages, only a few researchers have explored Arabic Automatic Text Simplification. In (Al-Subaihin and Al-Khalifa,

Proceedings of the The First Arabic Natural Language Processing Conference (ArabicNLP 2023), pages 418–422

2011), they presented a text simplification tool named "AlBaseet". The tool's structure consisted of four main stages: complexity assessment, lexical simplification, syntax simplification, and diacritization. They followed the LS-pipeline approach to simplify the text and produced synonyms by creating a new vocabulary or utilizing ArabicWordNet (Rodríguez et al., 2008).

The second attempt to construct an Arabic ATS was made by (Al Khalil et al., 2017). Their semi-automatic simplification approach was meant to simplify modern Arabic fiction; a linguist applied ACTFL (American Council on the Teaching of Foreign Languages) language proficiency requirements for simplifying five Arabic books using a web-based tool. They intended to create a readability measurement identifier using various machine learning classifiers to develop a graded reader scale of four levels.

In (Hazim et al., 2022), a method for identifying and visualizing complex words is presented. The authors' method combines lexical and syntactic analysis techniques, such as part-of-speech tagging and dependency parsing, to extract relevant information and create visualizations highlighting individual words' complexity.

A system was proposed in (Khallaf, 2023) that utilizes linguistic resources and rule-based transformations to identify complex linguistic structures and simplify them accordingly.

3 Simplification Approach

There are three stages involved in simplifying complex sentences. Initially, we need to recognize the complex words used in the sentence. After identifying these complex words, we generate alternative options for them that are simpler and more comprehensible. These alternatives can include synonyms, definitions, or rephrasing of the original word. Ultimately, we choose the most appropriate replacement for every intricate term, considering the surrounding context and the overall message conveyed in the text.

3.1 Complex Word Identification

The initial phase, known as Complex Word Identification (CWI), is extremely important because if a complex word is not identified, it will hinder the generation of substitutions in the entire LS architecture. Therefore, the accuracy of the CWI step determines the simplification pipeline's success. Multiple steps are carried out on the given input sentence during this stage.

Initially, we assign a Part-of-Speech tag (POS tag) to every word. Next, we determine specific POS tags that may require simplification. We only focus on examining verbs, nouns, and adjectives for simplification. Additionally, we subject complicated words to a machine-learning algorithm aided by a frequency list. Then, we obtain the complexity of each word. Initially, when provided with a sentence as input, we employ POS tagging to determine the Part-of-Speech for each word. We utilized the Farasa modules (Abdelali et al., 2016) to identify POS Tags in an Arabic sentence.

3.1.1 Pre-processing of Identification Dataset

After identifying the POS Tags of a given word, we determine whether such a word is complex. We trained an ML model using an available Arabic frequency list (Kilgarriff et al., 2014) to train an ML model. The frequency list contained 8904 Arabic words and their level of complexity based on the Common European Framework (CEFR) and the corresponding frequency.

Due to the large percentage of null values in the frequency column, we added our frequency score using Wordfreq[1]. Also, we added a POS Tag for each word using Farasa (Abdelali et al., 2016). Moreover, we added the stem of each word as a new feature, assuming that we want to know the complexity of the origin, as different words will have the same stems, and we removed redundant rows. The final data contains 4258 unique words and their corresponding stem, POS tag, frequency, and label, whether complex or not.

3.1.2 ML Identification Model

We built an ML model that can classify the complexity of each word. We considered building a model using the C-Support Vector Classification (SVC). We did try different combinations of independent features for the ML model. The input of the model contains the stem, POS Tag, and frequency as independent features. A different approach was to give the model word itself rather than its stem, as a stem can vary in complexity in different instances. Accordingly, we did implement two different identification models. The first model, Multi-Comp, was implemented by converting CEFR levels from 1 being the most minor complex to 6 being the most complex, according

[1]https://doi.org/10.5281/zenodo.7199437

to levels ranging between B1 to C2, respectively. We implemented the second model by categorizing CEFR levels into two binary formats. We determined that levels A1 to B1 are classified as not complex, assigning them a value of 0. On the other hand, levels B2 to C2 are considered complex and are given a value of 1. This model is referred to as the Binary model.

3.2 Generation Substitutions

The second stage is to generate substitutions for the complex word. We implemented two approaches: the first was using FastText, and the second was using BERT.

In the first approach, where we used FastText, we calculated the cosine similarity between words using the nearest neighbor module. We implemented a method to determine five similar candidates for a given complex word. However, FastText just produced words in different forms by the nearest neighbor. For example, the word 'ذهب' can be spoken as "Thahaba" or "Dahab", yet both words have entirely different meanings.

In the second approach, we used AraBERT (Antoun et al., 2020).

The masking language model works simply by masking a specific word in the sentence, and the model tries to predict what word can fit that place, given its right and left words. Accordingly, we utilized such a module for substitution generation. Once we have a list of complex words in a sentence, we mask a complex word per time and feed it to AraBert. AraBert then tries to predict the word appropriately fitting into the masked area.

3.3 Selection of Substitutions

We have constructed a sentence where we have inferred difficult words and identified five potential options for each difficult word. AraBERT provides a list of five words and their respective confidence scores, which indicate the level of certainty the model has for each candidate. Therefore, our initial strategy was to replace complex words with those with the highest certainty level. Unfortunately, two obstacles arose. The main obstacle was that sometimes, the word associated with the highest certainty rating was the same complex word. The second point is that we need a way to confirm whether the substituted word is more straightforward. Therefore, we deemed it necessary to include something that ensures the replacement of a complex word with its simpler equivalent.

To guarantee the replacement of the word, we depended on Gensim, an open-source library (Rehurek and Sojka, 2011). Gensim includes a module that measures the similarity between two words. We used this module by setting a condition that if the MLM model identified the complex word as the top candidate, we would calculate the similarity between the complex word and the other candidates. Currently, we possess two distinct identification models. The initial model evaluates complexity using a binary system, assigning either a 1 or 0. On the other hand, the second model assesses complexity using a scale of values ranging from 1 to 6, known as the Multi-Comp Model. We decided to add another condition for the second model to solve the second challenge we faced. The condition states that we will replace the complex word only if the replaced candidate has a lower complexity value. Even if it has the same value as a complex word, we will still keep the complex word to preserve the meaning better. Additionally, we ensured that the replaced candidate was not any ambiguous replacement, so we identified what variations the AraBERT model predicted and eliminated unnecessary replacements.

4 Evaluation & Results

In order to evaluate our models, we needed a parallel corpus. A parallel corpus is a collection of complicated texts and their simplified versions in the same language. To the best of our knowledge, there is only one available parallel corpus for the Arabic language (Al-Raisi et al., 2018). The corpora are in different sizes. The small size contains 8 sentence pairs, the medium-sized size contains 69 sentence pairs, and the large contains 765 sentence pairs.

4.1 Automatic Evaluation

We first evaluated the SVC identification models using different independent features. After, we evaluated our simplification approach using BERTScore. This was because BERTScore overcame the limitations of other metrics and supported the Arabic language.

As shown in Table 1, we tried four different combinations.

As demonstrated in Table 1, we found that using the stem of the word in combination with its frequency resulted in an F1-score of 0.88. From this,

Features	F1 Score
Word/PosTag/Frequency	0.79
Stem/PosTag/Frequency	0.77
Word/Frequency	0.86
Stem/Frequency	**0.88**

Table 1: Table showing results of different identification models

we determined that including a POS tag would only confuse the model, as its variations are quite different in various positions. By comparing features based on the stem or the words, we found that using the stem is more effective. It is more accurate to always provide the model with the stem of a word rather than providing various forms of the word, as this can lead to confusion in the model.

To assess the performance of both identification models in a sentence simplification system, we opted to examine their effectiveness using varying sizes of parallel corpora. Small, medium, and large sizes were evaluated by BERTScore using 'bert-base-multilingual-cased', which supports the Arabic language and many different languages. The results we obtained are shown in Table 2:

Lexical	P	R	F1
	Small		
Target/Binary-Model	0.836	0.843	0.830
Target/Multi-Comp-Model	0.848	0.858	**0.853**
	Medium		
Target/Binary-Model	0.864	0.872	0.868
Target/Multi-Comp-Model	0.876	0.885	**0.885**
	Large		
Target/Binary-Model	0.863	0.871	**0.867**
Target/Multi-Comp-Model	0.858	0.866	0.862

Table 2: Results showing both models on different sizes of parallel corpora

The findings suggest that the Multi-Comp model outperformed the Binary Model for both small and medium-sized datasets in the machine translation system. However, the Binary Model performed better than the Multi-Comp model when evaluating extensive corpora. This suggests that the Binary Model is more adaptable in dealing with diverse text types. This is likely because large corpora usually cover a range of topics and text formats, and the Binary Model is less likely to become confused when replacing words, unlike the Multi-Comp model, which may struggle with the complexity involved.

4.2 Manual Evaluation

We aimed to assess our model's performance by collaborating with human experts. To achieve this, we designed a survey comprising 20 randomly selected samples. Each sample included both input and output texts. The input text was a complex passage from the parallel corpora, while our models generated the output text. We evaluated the model by including 3 features, which are: 1) Meaning Preservation (MP), 2) Grammaticality (G), and 3) Simplicity (S) (Laban et al., 2021). We asked their experts to rate every sample on the three features on a scale of 1 to 5.

When addressing meaning preservation, we found that the Multi-Comp model outperforms the Binary model with a 69% rate of preserving meaning in the output texts. Moreover, it also outperformed the Binary model grammar-wise with a rate of 84% sustaining the grammar in the outputs. The only measurement that the Binary model leveraged was the most critical measurement, which is simplicity. Among the output texts, 79% were simpler than inputs.

The results of the manual evaluation show that there is a significant trade-off between the three measurements. The Binary model excels in simplicity but has a downside regarding meaning preservation and grammar. The model prioritizes simplifying complex words over preserving the meaning of the sentence, which leads to a loss of meaning preservation and grammar in the output. In other words, the model sacrifices meaning preservation and grammar to generate more straightforward text. This trade-off highlights the challenge of balancing multiple metrics in natural language processing tasks.

5 Conclusion and Futute Work

To conclude, we endeavored to develop a lexical text simplification system for Arabic. We introduced two models for identification: the Binary Model and the Multi-Comp Model. Furthermore, we suggested several simplification approaches utilizing FastText and AraBERT embeddings. Our perception of the lexical system restrictions is based on the fact that certain of the generated sen-

tence structures need to be better-formed, and the system can incorrectly recognize complex words from simple ones in the CWI phase. In the future, it would be beneficial to utilize more recent models for evaluation.

6 Limitations

Overall, we presented the advantages and disadvantages of our proposed approach. We specifically emphasized the drawbacks of the CWI step. One drawback of CWI is its limited ability to accurately identify complex words, primarily because it needs a dependable frequency list. Another crucial consideration in our proposed approach is finding a balance between simplifying a sentence without compromising its intended meaning and maintaining proper grammar. Furthermore, the availability of a parallel corpus is crucial for undertaking such a task, and we need more resources in Arabic.

References

Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A fast and furious segmenter for Arabic. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 11–16, San Diego, California. Association for Computational Linguistics.

Muhamed Al Khalil, Nizar Habash, and Hind Saddiki. 2017. Simplification of arabic masterpieces for extensive reading: A project overview. *Procedia Computer Science*, 117:192–198.

Fatima Al-Raisi, Weijian Lin, and Abdelwahab Bourai. 2018. A monolingual parallel corpus of arabic. *Procedia computer science*, 142:334–338.

Afnan A Al-Subaihin and Hend S Al-Khalifa. 2011. Albaseet: A proposed simplification authoring tool for the arabic language. In *2011 International Conference on Communications and Information Technology (ICCIT)*, pages 121–125. IEEE.

Suha S Al-Thanyyan and Aqil M Azmi. 2021. Automated text simplification: a survey. *ACM Computing Surveys (CSUR)*, 54(2):1–36.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. In *LREC 2020 Workshop Language Resources and Evaluation Conference 11–16*, page 9.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Ali L Hatab, Caroline Sabty, and Slim Abdennadher. 2022. Enhancing deep learning with embedded features for arabic named entity recognition. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4904–4912.

Reem Hazim, Hind Saddiki, Bashar Alhafni, Muhamed Al Khalil, and Nizar Habash. 2022. Arabic word-level readability visualization for assisted text simplification. *arXiv preprint arXiv:2210.10672*.

Nouran Abdelrahman Ahmed Khallaf. 2023. *An Automatic Modern Standard Arabic Text Simplification System: A Corpus-Based Approach*. Ph.D. thesis, University of Leeds.

Adam Kilgarriff, Frieda Charalabopoulou, Maria Gavrilidou, Janne Bondi Johannessen, Saussan Khalil, Sofie Johansson Kokkinakis, Robert Lew, Serge Sharoff, Ravikiran Vadlapudi, and Elena Volodina. 2014. Corpus-based vocabulary lists for language learners for nine languages. *Language resources and evaluation*, 48(1):121–163.

Philippe Laban, Tobias Schnabel, Paul Bennett, and Marti A Hearst. 2021. Keep it simple: Unsupervised simplification of multi-paragraph text. *arXiv preprint arXiv:2107.03444*.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2).

Luz Rello, Ricardo Baeza-Yates, Laura Dempere-Marco, and Horacio Saggion. 2013. Frequent words improve readability and short words improve understandability for people with dyslexia. In *IFIP Conference on Human-Computer Interaction*, pages 203–219. Springer.

Horacio Rodríguez, David Farwell, Javi Farreres, Manuel Bertran, Musa Alkhalifa, M Antonia Martí, William Black, Sabri Elkateb, James Kirk, Adam Pease, et al. 2008. Arabic wordnet: Current state and future extensions. In *Proceedings of The Fourth Global WordNet Conference, Szeged, Hungary*, 387-405.

Matthew Shardlow. 2014. A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications*, 4(1):58–70.

Punardeep Sikka and Vijay Mago. 2020. A survey on text simplification. *arXiv preprint arXiv:2008.08612*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.