Optimizing Readability Using Genetic Algorithms

Jorge Martinez-Gil

Software Competence Center Hagenberg GmbH
Softwarepark 32a, 4232 Hagenberg, Austria
jorge.martinez-gil@scch.at

Abstract

This research presents ORUGA, a method that tries to automatically optimize the readability of any text in English. The core idea behind the method is that certain factors affect the readability of a text, some of which are quantifiable (number of words, syllables, presence or absence of adverbs, and so on). The nature of these factors allows us to implement a genetic learning strategy to replace some existing words with their most suitable synonyms to facilitate optimization. In addition, this research seeks to preserve both the original text's content and form through multi-objective optimization techniques. In this way, neither the text's syntactic structure nor the semantic content of the original message is significantly distorted. An exhaustive study on a substantial number and diversity of texts confirms that our method was able to optimize the degree of readability in all cases without significantly altering their form or meaning. The source code of this approach is available at `https://github.com/jorge-martinez-gil/oruga`

Keywords: Text readability, Text Optimization, Genetic Algorithms

1. Introduction

Readability is a measure that tells us how easy it is to read a text. It corresponds to the level of literacy that is expected from the readers in the target audience. In this way, readability is considered one of the most critical factors that facilitate the user experience when consuming information. It is crucial because it is key to establishing a trusting relationship between information producers and consumers. It must be considered that some factors, such as complexity, legibility, or typography, contribute to making a text readable. However, not all factors are quantifiable and cannot be optimized by automatic techniques. In this paper, we focus solely and exclusively on factors of a quantifiable nature, which always revolve around basic or advanced statistics associated with the text to be optimized.

Therefore, text readability refers to how simple it is to read and comprehend a given text, depending on its unique characteristics. These characteristics are usually measurable through metrics like the number of syllables in a sentence. The diversity of words used to create a readability score can be used to gauge this measure [5]. Therefore, ORUGA is intended to facilitate the reader's ability to understand a text by optimizing its readability.

In the context of this work, a clear distinction between readability and quality should be made. Quality emphasizes essential elements like grammar, spelling, and voice. However, the goal of producing textual information as plainly as possible and better matching it with its audience is what is meant when a text is said to be readable. In this way, text readability partially overlaps with the notion of text difficulty, which is also an essential aspect of human language, and has an impact on the daily lives of most people who consume written information.

Our research is based on the premise that readability can be measured by considering metrics related to the text and then using a specific mathematical formula to calculate it. Because different readability scores are calculated using different mathematical formulas, it is possible to design a strategy that replaces many of the terms in the text with synonyms using a global optimization scheme. Such optimization is about maximizing or minimizing the result yielded by such mathematical formulas at the user's convenience. In practice, such a strategy can be implemented through a genetic algorithm, as shown throughout this work.

Therefore, our research offers a viewpoint from the field of computer science, concentrating on the fundamental text representations and metrics utilized by readability assessment methods. In this way, the most significant contributions of this work to state-of-the-art are the following:

- We present, for the first time, a technique that can automatically optimize the readability of any text, i.e., we can minimize or maximize the degree of readability of a text automatically without substantive changes.
- We study which are the best sources of synonyms currently available for text readability optimization. Specifically, we have studied Wordnet, word2vec, and web scraping and established a classification around optimizing up to ten texts of different natures.
- We present an additional method based on multi-objective optimization, whose mission is to ensure that the minimum number of words needed in the original text is replaced in such a way that the structure of the text is not impacted.
- Last but not least, we studied several strategies that allow us to measure (and therefore optimize) the semantic distance between the original text and the generated text. In this way, the impact on the original text's meaning can be controlled.

This research work is structured as follows: Section 2 shows state-of-the-art methods and tools for improving text readability. Section 3 presents the technical details of our proposal; these technical details are based on the design of a genetic cutting strategy that allows us to explore a vast search space while consuming a reasonable amount of resources. Section 4 explains how to minimize the impact on the form of the original text using a multi-objective optimization technique. Section 5 shows how to preserve the essence of the original text by ensuring that the distance between the original text and the text obtained is kept as small as possible. It is necessary to remark that sections 3, 4, and 5 present information about the design of different experiments and the raw results obtained, and their subsequent analysis. Finally, we conclude with the main lessons that can be learned from this research.

2. State-of-the-art

Let us begin with a formal definition of readability; for instance,[3] defines readability as the "*total number of elements in a given text that affect a reader's success.*" This reader's success is a measure of how well a text that is read at an optimal speed can be understood. At the same time, [21] defines readability as "*the level at which certain people find reading material convincing and understandable.*" Beyond these definitions, we are particularly interested in the quantifiable aspects of readability, i.e., what can be objectively measured. Otherwise, it would not be easy to proceed to its optimization using a computer. Let us see what the literature says about these quantifiable aspects.

2.1. Text readability in the scientific literature

Readability metrics usually use simple features to calculate the degree of readability of a text. Some commonly used features are the number of sentences or words, the ratio of unique words, the total number of syllables, the proportion of unique words in the text, the number of digits, the number of words with many syllables, etc. Although, at first, it may seem that these metrics are simplistic, they are very commonly used for two important reasons: they are much cheaper and faster to use than the alternatives consisting of human surveys, and according to experts, they usually give exceptionally reliable results that are in line with reality.

The goal of improving readability is to increase the chances that readers can understand the thoughts and ideas reflected in the text. So that misunderstanding is minimized, information processing is facilitated without requiring much effort and energy consumption. With this goal in mind, many sources can be found that advise how to improve a text's readability. However, these are manually compiled protocols that a human operator must translate into reality by modifying the text manually. For example, for a given metric, it is better to shorten sentences; for another, it is better to replace complex words with simpler ones, etc.

This is precisely where our contribution to the state-of-the-art lies. Optimizing texts automatically using a metric as a target means we do not have to concern about taking any manual action leading to altering the text. The genetic algorithm will find a way to proceed automatically.

2.2. Why is text readability important?

There are several contexts and population groups for which readability is critical. Especially when it is necessary to convey a written message to an audience. For example,

- Teachers need to be sure of the readability of a text before deciding whether it is appropriate for their students. This is particularly important in language learning. With the method presented here, the text can be optimized for a certain niche of learners.
- In the world of advertising, readability allows for building a trust relationship between advertisers and potential consumers. Advertising goods or services using texts with high readability is usually not a good idea since the message might not reach an essential part of the population. This is even more important in the search engine optimization sector, as many search engines use readability metrics as a ranking factor when responding to user searches.
- Readability is also relevant for professionals who work on websites [27], news [29], or even educational materials [1]. In some countries, there is even a legal requirement that government agencies provide textual information with certain readability levels to reach the entire population.

2.3. Readability metrics

There are several metrics to quantify how readable a text is [22]. Most metrics have been designed for the English language [14], although works also explore readability in other languages [13]. Without being exhaustive, we can mention, in chronological order, some of the metrics that enjoy or have enjoyed more significant popularity when dealing with the English language.

Text readability depends not only on the characteristics of the text but also on the educational background of the individuals interested in understanding the text. We will see this reflected below when measuring readability using formulas. We will see how readability metrics take a text as input and calculate a numerical score that usually corresponds to the level of education required to understand the text.

2.3.1. Dale-Chall readability

The Dale-Chall readability formula [6] requires a list of 3,000 words that fourth-grade U.S. students could reliably understand, as shown in Equation 1.

$$DCRF = 0.1579\left(\frac{difficult\ words}{total\ words} \times 100\right) + 0.0496\left(\frac{total\ words}{total\ sentences}\right) \tag{1}$$

2.3.2. SMOG readability

The SMOG readability level [21] can be assessed through a formula originally used for checking health messages, as shown in Equation 2. It corresponds to the years of education necessary to understand the text.

$$SMOG = 1.0430\sqrt{number\ of\ polysyllables \times \frac{30}{number\ of\ sentences}} + 3.1291 \tag{2}$$

2.3.3. ARI readability

The ARI assesses the U.S. grade level required to read a text [31]. In some ways, it is similar to other formulas. Its difference is that rather than counting syllables, it counts characters: the more characters, the more complex the word. It also counts sentences as shown in Equation 3. This sets it apart from some other formulas.

$$ARI = 4.71\left(\frac{total\ Characters}{total\ Words}\right) + 0.5\left(\frac{total\ Words}{total\ Sentences}\right) - 21.43 \tag{3}$$

2.3.4. Flesch Kincaid readability

Flesch Kincaid's readability score, as shown in Equation 4, is a metric based on grade levels that is used commonly in the insurance industry. Grade levels made it much easier for people to understand [10]. A Flesch Kincaid Grade Level (FKGL) between 8 and 10 means that the text should be accessible to the public. FKGL remains the most widely-used formula today.

$$FKGL = 206.835 - 1.015\left(\frac{total\ words}{total\ sentences}\right) - 84.6\left(\frac{total\ syllables}{total\ words}\right) \tag{4}$$

Over the past decade, several natural language processing (NLP) techniques have been proposed to determine the readability of a text. Thus, as opposed to the classical approach of using formulas that measure a limited set of text features, these new variants have attempted to measure the difficulty of understanding sentences and words and even the complexity of the syntax [15]. Even some techniques based on predictors of readability, such as cohesion and coherence, have received considerable attention. However, so far, these approaches have yet to be able to predict the readability of a text better than the classical techniques discussed here [33].

2.4. Semantic Similarity

The field of semantic similarity measurement [16] is one of the most active in several different research communities (information retrieval, database integration, natural language processing, and so on) [30]. This is due to its significant implications on many available frameworks, methods, and tools [26] both in industry and academia. The literature around this topic has skyrocketed in the last few years [4]. However, most research works focus on determining the similarity between words [36], phrases, or documents [17, 18].

In this way, rarely the likelihood of effectively throwing out the most similar words to a given one has been discussed. To this effect, there are synonym libraries that have been manually compiled and some word embedding techniques that do the job well. The most prominent solutions in this direction are WordNet [28], word2vec [23], or BERT [8]. However, it must be taken into account that the resource requirements for techniques based on the computation of word embeddings are very high [20].

2.5. Contribution over the state-of-the-art

Determining text readability based on a formal analysis of the structures and words used has been a recurring theme in the literature over the last few years. As a result, many metrics have been proposed to measure text readability. A common denominator of all these metrics is that a high score usually means the text is difficult to understand. In other words, a higher degree of study is needed to understand it. For this reason, many communication professionals often use tools to help them discern whether the text fits a given audience. However, none of these tools can do the professional's job automatically, which is why our contribution is essential.

To the best of our knowledge, this is the first time anyone has proposed automatically improving the readability of text without significantly altering the content or the text's form. In addition, we put this innovative method through its paces using a wide range of texts that varied in subject matter and level of readability. Furthermore, we provide the source code of the first implementation for anyone interested in experimenting with or improving the method.

3. Part I: Design and Implementation of a Functional Solution

In the following, we explain our proposal for text readability optimization using a method based on genetic algorithms. In this section, we outline the technical preliminaries, discuss the implementation, show an illustrative example of how our approach works in practice, and conduct an experimental study using real data and use cases that can help us get an idea of the performance of this approach in practice.

3.1. Technical Preliminaries

Our hypothesis is that we can find a set of synonyms to replace some words in the original text so that the value returned by the readability formula can be optimized. For example, if the readability formula rewards or penalizes long words, we have to find synonyms of less length. In reality, we only have to concern about understanding which formula best represents readability in our specific scenario and indicate it as a fitness function. The genetic algorithm will *understand* how to proceed. Thus, we are faced with a classical optimization problem.

We intend to act only on the vocabulary since it is one of the most critical parts of that language. It is widely assumed that vocabulary is the essential part of a language because, without vocabulary, it is impossible to compose any message [34]. We do not act on proper nouns, prepositions, or other stop words to avoid distorting the original message.

While this problem can be solved by a brute-force search over the range of the words of a given text $w_0, w_1, ..., w_n$, the GA method scales very well when dealing with large texts. In this case, a brute-force search would be prohibitively expensive. We could act on other aspects, such as the structure, but then we would risk distorting the original text's essence again.

Our idea is to find the combination of words (which will be encoded in the form of an individual) that optimizes the desired objective. The choice of the fitness function is effortless and has the advantage that the solution automatically identifies what kind of words lead to the optimization.

In this work, we work with three main sources of synonyms:

- *WordNet* [24], which is a thesaurus that has been manually compiled. It is probably one of the most widely used dictionaries in information systems and will give us several alternatives to substitute each candidate word. No surprises are to be expected in this library, except perhaps the substitution of words with a synonym that has a sense far from the original. In any case, we have a solution to this problem.

- *wordvec* [23], which is an approach that calculates the vectors associated with each word according to a textual corpus of relevance. Once each vector has been computed, a computation process can be performed by which the N vectors most similar to a given one are identified. This way, synonyms of relevance are obtained for the word in question. Note that this process is computationally expensive.

- *Web scraping* [25], which consists of obtaining the synonyms from some websites, usually specialized, so that we can shuffle several alternatives per candidate word. This method obtains many high-quality word candidates, but it should be used responsibly because it can cause problems on the server side if many hundreds or thousands of requests are made. Therefore, the method is fine for experimentation, but it would be unreasonable to exploit it.

3.2. Implementation

The implementation of this novel approach is based on a classical optimization scheme using genetic algorithms. Algorithm 1 briefly explains in pseudo-code how the whole process is performed by adapting the classical structure of the genetic algorithm [9] where different operators capable of implementing selection, cross-over and mutation processes are considered in order to evolve a given population towards the desired objective.

Please note that before starting the evolution process, a pre-processing of the text must be done in order to identify the candidate words to be replaced by a synonym. We propose that all words are candidates except: proper names, stop words, and prepositions. The reason for this is that they are often words for which it is really difficult to find a synonym.

In relation to the genetic algorithm itself, the parametric details of the solution will be discussed below, but it is possible to see how we implement it in the form of a classical evolutionary process. This means that a population of individuals is selected randomly, and their readability score is calculated. We then proceed with an iterative process of selection, crossover, and mutation; the best individuals are passed from generation to generation until one of the stopping criteria is met:

Algorithm 1 Optimizing Readability Using Genetic Algorithm

```
1: procedure ORUGA
2:     population ← generationRandomIndividual (population)
3:     calculateReadabilityScore (population)
4:     while (stop condition not reached) do
5:         parents ← selectionOfIndividuals (population)
6:         offspring ← Crossover (parents)
7:         offspring ← Mutation (offspring)
8:         offspring ← calculateReadabilityScore (offspring)
9:         population ← updatePopulation (offspring)
10:    endwhile
11:    optimizedText ← optimizedIndividual (population)
12:    optimizedText ← correctErrorsIfNecessary (optimizedText)
13:    return optimizedText
```

the highest possible has been reached (unlikely), or the number of iterations has been exhausted (very likely).

Finally, at the end of the evolutionary process, we correct the text in case grammatical errors are produced by substituting a synonym that does not fit the tense and the form of the sentence in which it is framed. This way, we obtain a corrected readability score, which may vary slightly from the one derived automatically. In return, we ensure that the results are usable, or at least close to being usable.

3.3. Illustrative examples

Example 1 shows us how ORUGA works with written material about science extracted from Wikipedia. In fact, our aim is to observe how ORUGA behaves when trying to minimize the FKGL readability score using synonyms from the library WordNet. Let us remember that FKGL (or one of its variants) is probably the most widely used metric and its optimization brings advantages in several fields. Please note that the text to be treated can be of any length, but to facilitate the presentation to the reader, we have chosen (and will choose throughout this paper) one that contains only several sentences.

Example 1. Science

Original text **Source:** Wikipedia

"The sea moderates the climate and has important roles in the water cycle, carbon cycle, and nitrogen cycle. Humans harnessing and studying the sea have been recorded since ancient times, and evidenced well into prehistory, while its modern scientific study is called oceanography. The most abundant solid dissolved in seawater is sodium chloride. The water also contains salts of magnesium, calcium, potassium, and mercury, amongst many other elements, some in minute concentrations. Salinity varies widely, being lower near the surface and the mouths of large rivers and higher in the depths of the ocean; however, the relative proportions of dissolved salts vary little across the oceans." **FKGL score: 14.06**.

ORUGA - minimizing the FKGL score - library Wordnet

The sea moderates the climate and has important part in the H2O cycle, C cycle, and N cycle. Humans harnessing and studying the sea have been taped since ancient times, and attested well into prehistory, while its modern scientific study is called oceanography. The most abundant solid fade out in brine is Na chloride. The H2O too contains salts of magnesium, calcium, potassium, and mercury, amongst many other elements, some in min concentrations. Salinity changes widely, being got down near the surface and the mouths of large rivers and higher in the depth of the ocean; however, the relative proportions of fade out salts change little across the oceans. **FKGL score: 11.85**

As can be seen, ORUGA can optimize the textual input by first automatically identifying which words can be replaced by a synonym, and then undertaking a process of searching for synonyms that improve the results of the metric to be optimized. In this way, the impact on the initial message is minimal, although it is true that some synonyms can slightly distort the meaning of the text, and therefore final supervision by the user is required. However, there is no need to concern because this problem will be addressed later in the paper.

Let us look now at Example 2, which is a written text about the history of Austria that has been also extracted from Wikipedia, and we would like to to minimize the FKGL readability score using synonyms automatically obtained by web scraping.

EXAMPLE 2. HISTORY

Original text **Source:** Wikipedia

"Austria emerged from the remnants of the Eastern and Hungarian March at the end of the first millennium. Originally a margraviate of Bavaria, it developed into a duchy of the Holy Roman Empire in 1156 and was later made an archduchy in 1453. In the 16th century, Vienna began serving as the empire administrative capital and Austria thus became the heartland of the Habsburg monarchy. After the dissolution of the Holy Roman Empire in 1806, Austria established its own empire, which became a great power and the dominant member of the German Confederation. The defeat in the Austro-Prussian War of 1866 led to the end of the Confederation and paved the way for the establishment of Austria-Hungary a year later." **FKGL score: 13.20**

ORUGA - minimizing the FKGL score - web scraping

Austria looms from the debris of the Eastern and Hungarian March at the end of the first millennium. Originally a margravate of Bavaria, it matured within a duchy of the Holy Roman Empire in 1156 and was next made an arch duchy in 1453. In the 16th century, Vienna lead plate as the command departmental central and Austria thus come the heartland of the Habsburg monarchy. After the divorce of the Holy Roman Empire in 1806, Austria settled its own empire, that come a great power and the dominant branch of the German Confederation. The defeat in the Austro-Prussian War of 1866 led to the end of the Confederation and brick the way for the founding of Austria-Hungary a year later. **FKGL score: 11.35**

Once again, we can see how the genetic algorithm has acted intelligently to decrease the text readability score and therefore make the text accessible to more people. It is clear that the suitability of some words may be subject to debate, but the first objective of this research, i.e., to optimize the readability score, has been achieved. As we mentioned before, we will be concerned to outline a final product later in this paper.

Finally, let us look at the opposite case, i.e., let us see if we can make a text more difficult to read without losing its essence. In Example 3, we have a text about sports also extracted from Wikipedia, and we do not want to make the text accessible to as many people as possible, but on the contrary, we want to increase the level of readability. We will now try a different metric, e.g., ARI.

Example 3. Sports

Original text **Source:** Wikipedia

"Real Madrid Club de Futbol, meaning Royal Madrid Football Club, commonly referred to as Real Madrid, is a Spanish professional football club based in Madrid. Founded in 1902 as Madrid Football Club, the club has traditionally worn a white home kit since its inception. The honorific title real is Spanish for Royal and was bestowed to the club by King Alfonso XIII in 1920 together with the royal crown in the emblem. Real Madrid have played their home matches in the Santiago Bernabeu Stadium in downtown Madrid since 1947. Unlike most European sporting entities, Real Madrid members (socios) have owned and operated the club throughout its history." **ARI score: 12.69**.

ORUGA - maximizing the ARI score - web scraping

Real Madrid Club de Futbol, connotation Royal Madrid Football Club, frequently referred to as Real Madrid, is a Spanish professional football business established in Madrid. Founded in 1902 as Madrid Football Club, the business has consistently timeworn an alabaster home kit since its inception. The appellation title real is Spanish for Royal and was entrusted to the business by King Alfonso XIII in 1920 together alongside the aristocratic culmination in the emblem. Real Madrid have played their familiar matches in the Santiago Bernabéu Stadium in downtown Madrid afterward 1947. Unlike most European sporting entities, Real Madrid assemblage (socios) have owned and negotiated the business throughout its history. **ARI score: 15.53**

As can be seen, after processing the fragment related to *Real Madrid* extracted from Wikipedia, the genetic algorithm selects synonyms that are much longer and more complicated to read to increase the ARI score. Although a use case consisting of making the text less accessible to people is hard to imagine, it may find application in some specific niches of learning, education, etc.

3.4. Experimental study

In this section, we explain the details of an empirical study that we have carried out to know the feasibility of our new method. To do so, we first established the conditions of the experiments by setting up an experimental setup. Secondly, we have run and obtained the raw results. Furthermore, thirdly, and lastly, we have proceeded with the analysis of the data obtained.

3.4.1. Experimental setup

Adjusting the parameters of the genetic algorithm differs from the focus of this work. Therefore, we have chosen a classical parameter setting, which has been shown to work quite well. Standard tuning of the parameters, e.g., through a grid search, is a possible line of future work. In the meantime, the parameters we have used for our experiments are the following:

- Population size {10, 15, 20}: **20**
- Number of parents mating {10, 15, 20}: **10**
- Number of genes: one per candidate word to be substituted by a synonym
- Fitness function: the user can choose among Equations 1, 2, 3, 4
- Stop condition: {100, 200, 300}: **300** generations

To test whether our approach can optimize text readability, we semi-randomly selected ten texts extracted from Wikipedia. These texts are classified into several categories engineering, geography, history, science, and sports. Some metrics, such as SMOG, require at least 30 sentences to apply their formula. For cases where we do not reach 30 sentences, we will duplicate the text until we reach that number of sentences. It should also be noted that depending on which readability category each use case represents. For example, magenta represents the highest difficulty (FKGL between 15 and 18), equivalent to an academic paper. Red represents medium-high difficulty (FKGL between 12 and 15). The black color represents a medium difficulty (FKGL between 9 and 12). Furthermore, the blue color represents a low-medium difficulty. It is estimated that up to 80% of the native English-speaking population could understand text with an FKGL between 6 and 9, which is what the blue value represents.

Furthermore, every experiment was run on a standard computer with 32 GB of primary memory and an Intel Core i7-8700 processor running at 3.20 GHz on Microsoft Windows 10 64-bit. Most of the functionality has been implemented using the library PyGAD[1], an open-source Python library for building genetic algorithms.

3.4.2. Experiments

The first experiment is the one that is the most useful in practice. It consists of trying to minimize the readability of the ten proposed texts so that the processed text can reach the largest possible audience. To do that, we want to use readability metrics which estimate the readability of a text based on simple aspects such as syllable and word counts. We use the FKGL score since this metric, or some of its variants, has been used for decades on traditional texts, and it is still one of the most common and widely used traditional readability measures.

[1]https://pypi.org/project/pygad/

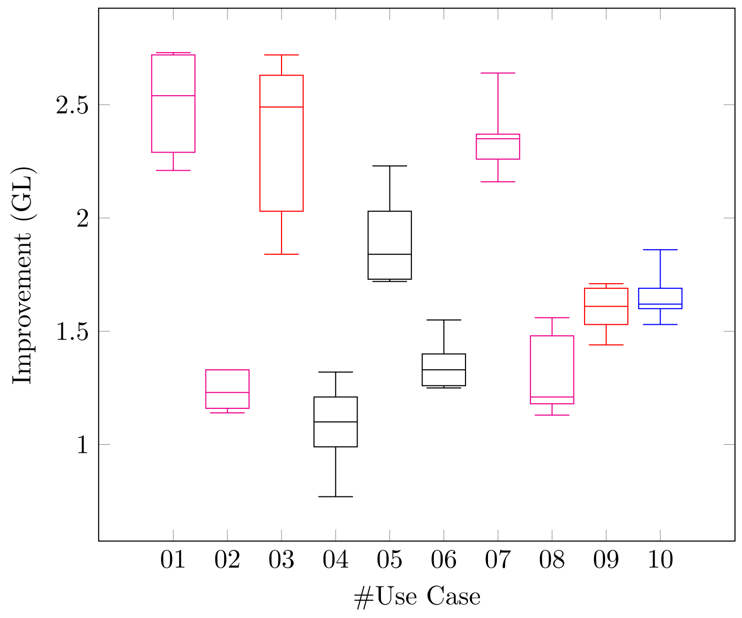


Figure 1: Results for the **minimization** of the **FKGL** score using **WordNet**

Fig 1 shows us the results obtained when reducing the FKGL for the texts under consideration using the library WordNet. The results shown are the summary of ten independent runs per use case. Since we are dealing with cold start methods with random values, and there is even a component of randomness in the mutations, we will almost always get a different result, so it is essential to show the results in this way.

Positive results have been obtained in all 100 experiments performed (ten runs for ten use cases). In addition, minimum improvements, average improvements, and even maximum improvements have been achieved in the range between 0.77 and 2.73 points on the FKGL scale.

Fig 2 shows us the results obtained using the synonym calculation method using word2vec. As it is possible to see, the variance of the results is enormous, which means that using this library will make the results not very predictable. At the same time, as in the previous case, all 100 experiments were able to obtain readability improvements that ranged between 0.42 and 3.90-grade levels.

Fig 3 shows us the results we have obtained by searching for synonyms with web scraping techniques. The results this time have a smaller variance than in the previous case. In addition, we have again obtained favorable results in all 100 experiments performed. The optimization achieved ranges between 1.02 and 4.20-grade levels.

Table 1 shows the summary results of the 300 experiments performed (100 for each synonym library). For WordNet, a median optimization of 1.63-grade levels is expected, very similar to

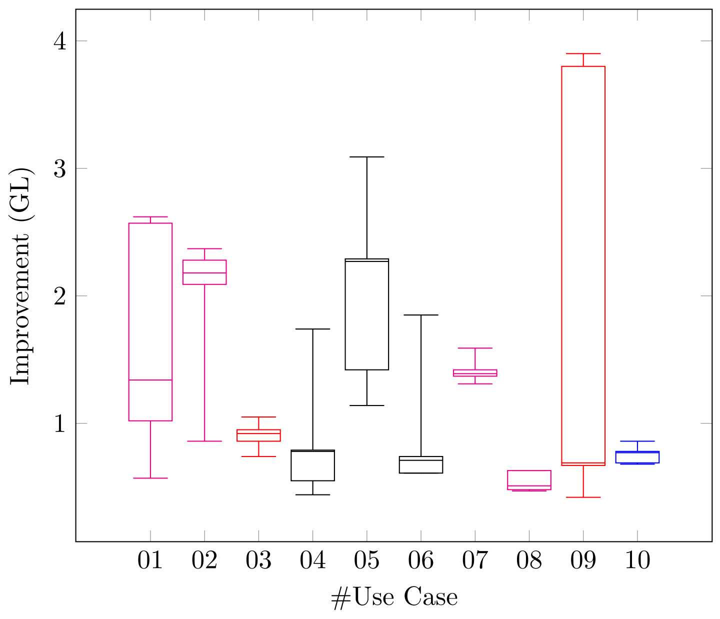


Figure 2: Results for the **minimization** of the **FKGL** score using **word2vec**

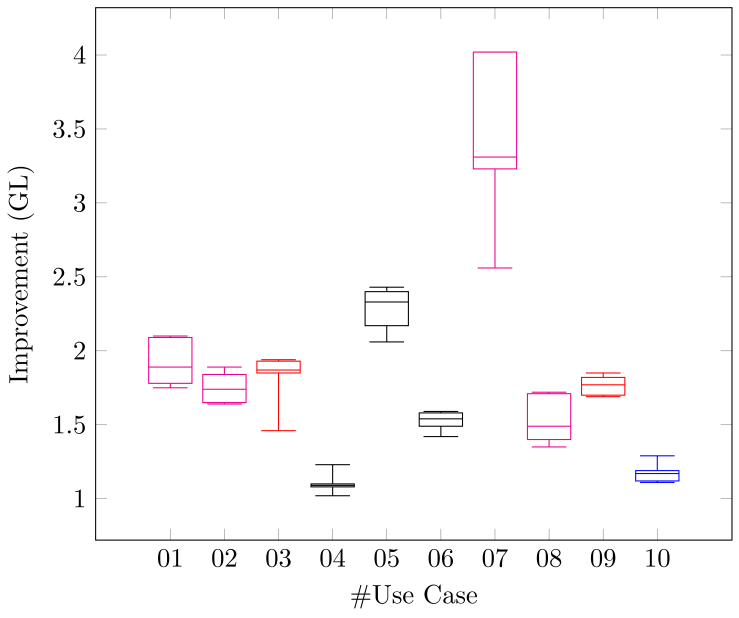


Figure 3: Results for the **minimization** of the **FKGL** score using **Web Scraping**

Library	Minimum	Median	Maximum
WordNet	0.77	1.63	2.73
word2vec	0.42	0.96	3.90
Web scraping	1.02	1.61	4.20

Table 1: Summary of the results obtained using the different libraries of synonyms

Use case	**DCRF**	**SMOG**	**ARI**	**FKGL**
01/science	1.11 ± 0.17	2.03 ± 0.13	2.90 ± 0.27	2.53 ± 0.15
02/history	0.29 ± 0.12	0.71 ± 0.08	0.84 ± 0.05	0.90 ± 0.04
03/geography	0.65 ± 0.07	0.91 ± 0.04	1.17 ± 0.12	1.33 ± 0.04
04/sports	0.78 ± 0.14	0.42 ± 0.12	0.83 ± 0.05	0.93 ± 0.10
05/engineering	0.59 ± 0.06	0.60 ± 0.10	0.66 ± 0.10	0.75 ± 0.09
06/geography	0.52 ± 0.15	0.70 ± 0.10	0.77 ± 0.19	0.77 ± 0.18
07/engineering	1.03 ± 0.15	0.51 ± 0.03	1.12 ± 0.21	1.32 ± 0.01
08/history	0.77 ± 0.09	0.27 ± 0.06	0.44 ± 0.00	0.73 ± 0.17
09/sports	0.46 ± 0.06	1.06 ± 0.28	0.99 ± 0.12	0.81 ± 0.10
10/history	0.74 ± 0.04	0.65 ± 0.03	0.77 ± 0.13	0.93 ± 0.04

Table 2: Summary of the results obtained for the experiments performed w.r.t. **minimization**. The numerical values represent absolute improvements after using ORUGA

that achieved by web scraping. The median optimization using word2vec is the worst of the three libraries considered. Please note that we are not judging here the quality of the replacements, but the values to optimize the input text independently of the quality of the final result.

Table 2 shows a summary of all the results we have obtained. The texts have been randomly obtained from Wikipedia. The range of values shown indicates the minimum value that could be reached and the maximum value (for minimization and maximization problems respectively).

We analyze the four formulas that we consider most representative, but the analysis of other formulas would not be a problem. When working with SMOG, we must be careful because the text to be analyzed requires at least 30 sentences. We have duplicated the text for these experiments so that these 30 sentences can be used.

Table 3 shows a summary of all the results we have obtained when maximizing the readability scores. As can be seen by comparing with the table above, it is much easier to increase the level of readability than to decrease it, i.e., it is easier to make a text difficult to read than the other way around.

3.4.3. Discussion

We have seen how it is possible to build a solution that optimizes the readability of texts of different natures. Moreover, such optimization is done respecting the content and the form of such texts, trying to minimize the impact of word replacements by synonyms that better fit the readability criteria of the different formulas we have studied. We have seen how optimization can occur in two ways. First, it can be done in such a way as to reduce the readability score, which

Use case	DCRF	SMOG	ARI	FKGL
01/science	1.98 ± 0.16	1.30 ± 0.06	1.80 ± 0.08	2.09 ± 0.14
02/history	1.85 ± 0.25	1.99 ± 0.09	3.88 ± 0.28	3.02 ± 0.08
03/geography	2.38 ± 0.12	1.23 ± 0.08	4.44 ± 0.13	4.00 ± 0.44
04/sports	2.09 ± 0.40	2.79 ± 0.07	4.63 ± 0.40	4.26 ± 0.04
05/engineering	2.10 ± 0.43	2.16 ± 0.11	4.55 ± 0.10	4.23 ± 0.35
06/geography	1.53 ± 0.21	1.81 ± 0.22	2.96 ± 0.04	2.90 ± 0.41
07/engineering	1.49 ± 0.38	1.71 ± 0.04	3.87 ± 0.46	3.34 ± 0.16
08/history	1.43 ± 0.18	3.09 ± 0.13	4.11 ± 0.63	3.70 ± 0.21
09/sports	1.60 ± 0.21	2.29 ± 0.16	3.54 ± 0.28	2.56 ± 0.04
10/history	2.06 ± 0.08	2.64 ± 0.04	3.99 ± 0.25	3.27 ± 0.52

Table 3: Summary of the results obtained for the experiments performed w.r.t. **maximization**. The numerical values represent absolute improvements after using ORUGA

will allow more people to understand the text perfectly. This is undoubtedly the most practical option in practice. Furthermore, secondly, it can be done to increase the readability score, allowing a smaller number of people to understand the processed text unambiguously. This option has less practical utility than can be discerned at first glance.

Based on the research we have carried out, possible improvements can be made. For example, a multi-objective optimization algorithm could simultaneously optimize the readability score by affecting as few words as possible. In this way, the impact of our method on the original text would be even more negligible. A solution front could allow the human operator to decide the trade-off between the score modification and the replaced words.

Finally, and as a limitation of our method, it can be observed that sometimes the processed text contains minor grammatical errors. This is because we have yet to use techniques that allow, for example, to choose the appropriate verb tense for the context. Here there are two alternatives. On the one hand, we can implement better methods for correcting grammatical errors to obtain a corrected readability score; on the other hand, we can give the user the option to edit or choose the word that best fits each moment.

4. Part II: Minimizing the impact on the form of the original text

While in Section 3, we have seen that it is possible to design a functional solution to optimize the text readability, we have also seen that the approach can be intrusive at times. That is, the replacement of many words by synonyms can lead to a distortion of the original message. For this reason, in this section, we focus on minimizing the impact of ORUGA on the original text by replacing as few words as possible. To do so, we will build a solution based on multi-objective optimization (MOO) that allows us to optimize readability and simultaneously minimize the number of replacements. This section comprises the technical preliminaries, the implementation, some illustrative examples, and an empirical study to test several texts of different natures.

4.1. Technical Preliminaries

We have already seen how in the field of text readability, there has been a great effort to build straightforward formulas that can be understood by people and have a good correlation to how easily a text can be read from a human perspective. Now we go one step further to obtain a higher quality result. MOO is a strategy in which two or more objectives are simultaneously optimized. This is the situation we find ourselves in, given that we want: on the one hand, to improve the readability of a text, and on the other hand, we want to reduce as much as possible the number of words that need to be replaced to improve readability, and thus to minimize the impact that our approach has on the original text form.

Furthermore, MOO is useful when decisions must be made despite potential trade-offs between more than one orthogonal objective. Again, this is our situation because our goals of maximizing readability while simultaneously replacing the fewest possible words require us to pursue two completely different goals. In situations like this, different solutions can simultaneously fulfill all objectives. As a result, all optimal solutions ought to be regarded as equivalent merit without any external evaluation from a human operator [19]. More formally, we can model a MOO problem as expressed in Equation 5.

$$\begin{gathered} \min\left(s_1(\vec{x}), s_2(\vec{x}), \ldots, s_n(\vec{x})\right) \\ \textit{subject to } \vec{x} \in X \end{gathered} \tag{5}$$

In MOO, no solution addresses all objective functions simultaneously. As a result, the priority should be placed on finding solutions that cannot make any goals better without making at least one of the other goals worse. Therefore, a solution $\vec{x}_1 \in X$ is said to dominate another one $\vec{x}_2 \in X$ if the conditions expressed in Equation 6 are met.

$$\begin{gathered} s_i(\vec{x}_1) \leq s_i(\vec{x}_2) \; \forall i \in \{1, 2, 3, \ldots, n\} \\ s_j(\vec{x}_1) < s_j(\vec{x}_2) \; \exists j \in \{1, 2, 3, \ldots, n\} \end{gathered} \tag{6}$$

In this way, a solution $\vec{x} \in X$ is optimal if no solution might dominate it. An element x is said to dominate another element y if x is not worse than y concerning all the goals and is strictly better than y for at least one. The elements of the search space that are not dominated give rise to a Pareto front, which represents the best possible solutions to the orthogonal objectives.

4.2. Implementation

There are several implementations for MOO strategies [11, 35]. It is beyond the scope of this paper to consider them all. However, we will look at one of the best ones, NSGA-II [7]. This strategy is summarized in Algorithm 2. Its mode of operation is based on the concepts of fronts and crowding distance.

NSGA-II adheres to the basic structure of a genetic algorithm but employs a different approach to mating and selection for survival. In the NSGA-II, the first step is to select individuals in a front-wise fashion. In this way, a situation will arise where it will be necessary to divide a front because it will not be possible for all individuals to survive.

The solutions are chosen according to the crowding distance for this particular splitting front. Within the parameters of the objective space, the Manhattan Distance corresponds to the crowding distance. On the other hand, it is desired that the extreme points be maintained with each new generation, and as a result, an infinite crowding distance is assigned to them. Algorithm 2 shows us a possible implementation of this approach.

Algorithm 2 MOO Technique for Optimizing Text Readability

```
1: procedure ORUGA2-MOO
2:     population ← initializePopulation ()
3:     population ← generationRandomIndividual (population)
4:     calculateReadabilityScore (population)
5:     assignRankBasedOnPareto (population)
6:     auxiliarPopulation ← generationChildPopulation (population)
7:     while (stop condition not reached) do
8:         for (each individual in population and auxiliarPopulation) do
9:             solution ← calculateReadabilityScore (population)
10:            solution ← assignRankBasedOnPareto (population)
11:            solution ← generateNonDominateSolutions (population)
12:            solution ← determiningCrowdingDistance (population)
13:            for (each solution) do
14:                population ← addingSolutionsNextGeneration (population)
15:            end for
16:        end for
17:        population ← selectPointsLowFrontHighCrowdingDistance (population)
18:        population ← generationNextPopulation (population)
19:    end while
20:    optimizedText ← optimizedIndividual (solution)
21:    optimizedText ← correctErrorsIfNecessary (optimizedText)
22:    return optimizedText
```

NSGA-II is an example of a genetic algorithm, and it possesses the three characteristics listed below: It operates based on an elitist principle, which states that only the most privileged members of a population are permitted to be passed down to subsequent generations. In addition, it employs a mechanism specifically designed to preserve diversity (crowding distance). As a direct consequence of this, it can identify non-dominated solutions.

4.3. Illustrative examples

Since we are trying to optimize two orthogonal objectives simultaneously, it is impossible to offer a single solution. Nevertheless, we can put in the hands of the human operator a front of solutions ranging from a total optimization by modifying the most significant number of words to a minor optimization by touching a minimum number of words. It is up to the human operator to decide which solution to choose.

Example 4 shows a real trace that controls the number of words to be replaced using the MOO technique known as NSGA-II. The text on which it operates is about geography and has been extracted from Wikipedia. Once again, we must insist that although, in theory, it would be possible to edit the words manually to satisfy the criteria of a given metric, this approach is transparent and works automatically for any desired metric. Please note that the words in blue are candidates to be replaced by a synonym.

EXAMPLE 4. GEOGRAPHY **Source:** WIKIPEDIA

Goal: *Minimize the FKGL score by using Wordnet synonyms and minimize the words to be replaced using NSGA-II.*

"Niagara Falls is a group of three waterfalls at the southern end of Niagara Gorge, spanning the border between the province of Ontario in Canada and the state of New York in the United States. The largest of the three is Horseshoe Falls, which straddles the international border of the two countries. It is also known as the Canadian Falls. The smaller American Falls and Bridal Veil Falls lie within the United States. Bridal Veil Falls is separated from Horseshoe Falls by Goat Island and from American Falls by Luna Island, with both islands situated in New York. Formed by the Niagara River, which drains Lake Erie into Lake Ontario, the combined falls have the highest flow rate of any waterfall in North America that has a vertical drop of more than 50 m (160 ft). During peak daytime tourist hours, more than 168,000 m3 (5.9 million cu ft) of water goes over the crest of the falls every minute." **FKGL score: 10.72**.

Words to be replaced	FKGL expected
5	10.43 (▽ 2.71%)
6	10.28 (▽ 4.10%)
7	10.13 (▽ 5.50%)
8	9.98 (▽ 6.90%)
9	9.91 (▽ 7.56%)
10	9.84 (▽ 8.21%)
11	9.76 (▽ 8.96%)
12	9.68 (▽ 9.70%)
13	9.61 (▽ 10.35%)

As it is possible to observe, minor changes can be made to the original text and still optimize readability. It is still an open question whether minor changes in form are not so minor in meaning. But that open question will be addressed later in this paper.

4.4. Experimental study

This section will explain the specifics of an empirical study we conducted to determine whether this novel approach is feasible. To accomplish this, we initially prepared an experimental setup so that we could determine the parameters of the experiments. Second, we completed the test and collected the unprocessed data. We have moved forward with the analysis of the data that we have gathered, which brings us to our third and last point.

4.4.1. Experimental setup

As was the case in the preceding part, the meticulous fine-tuning of the MOO strategy's parameters will not be the primary focus of this work. As a result, following a brief preliminary study based on a scheme of traditional parameter settings, one configuration works quite well. As a direct consequence of this, the following is a list of the parameters that we used for our experiments:

- Population size {10, 15, 20}: **20**
- Number of parents mating {10, 15, 20}: **20**
- Number of genes: one per candidate word to be substituted by a synonym
- Fitness function: the user can choose among Equations 1, 2, 3, 4, words to be replaced
- Stop condition: {300, 600, 900}: **900** generations

Furthermore, every experiment was run on a standard computer with 32 GB of primary memory and an Intel Core i7-8700 processor running at 3.20 GHz on Microsoft Windows 10 64-bit. Most of the functionality has been implemented using the library jMetalPy[2], an open-source Python library for designing and implementing MOO strategies [2].

4.4.2. Experiments

Now we will proceed with the experiments concerning minimizing the impact on the original message's form. While we focused previously on pure optimization, we focus here on minimizing the number of replaced words to affect how the message looks as little as possible.

In Figure 4, we can see a summary of the results obtained after our experiments. What we have done is try to minimize the readability score (FKGL) at the same time as the number of words to be replaced in the ten use cases we are using throughout this work. As can be seen, we always obtain a Pareto front of solutions which indicates that the fewer words replaced, the less optimization is carried out. However, it should also be noted that the fewer words replaced also means a more negligible impact on the form of the initial message. It should be noted that different colors have been used for the Pareto fronts as in the previous experiments. Each color represents the degree of difficulty of each case study.

[2]https://jmetal.github.io/jMetalPy/

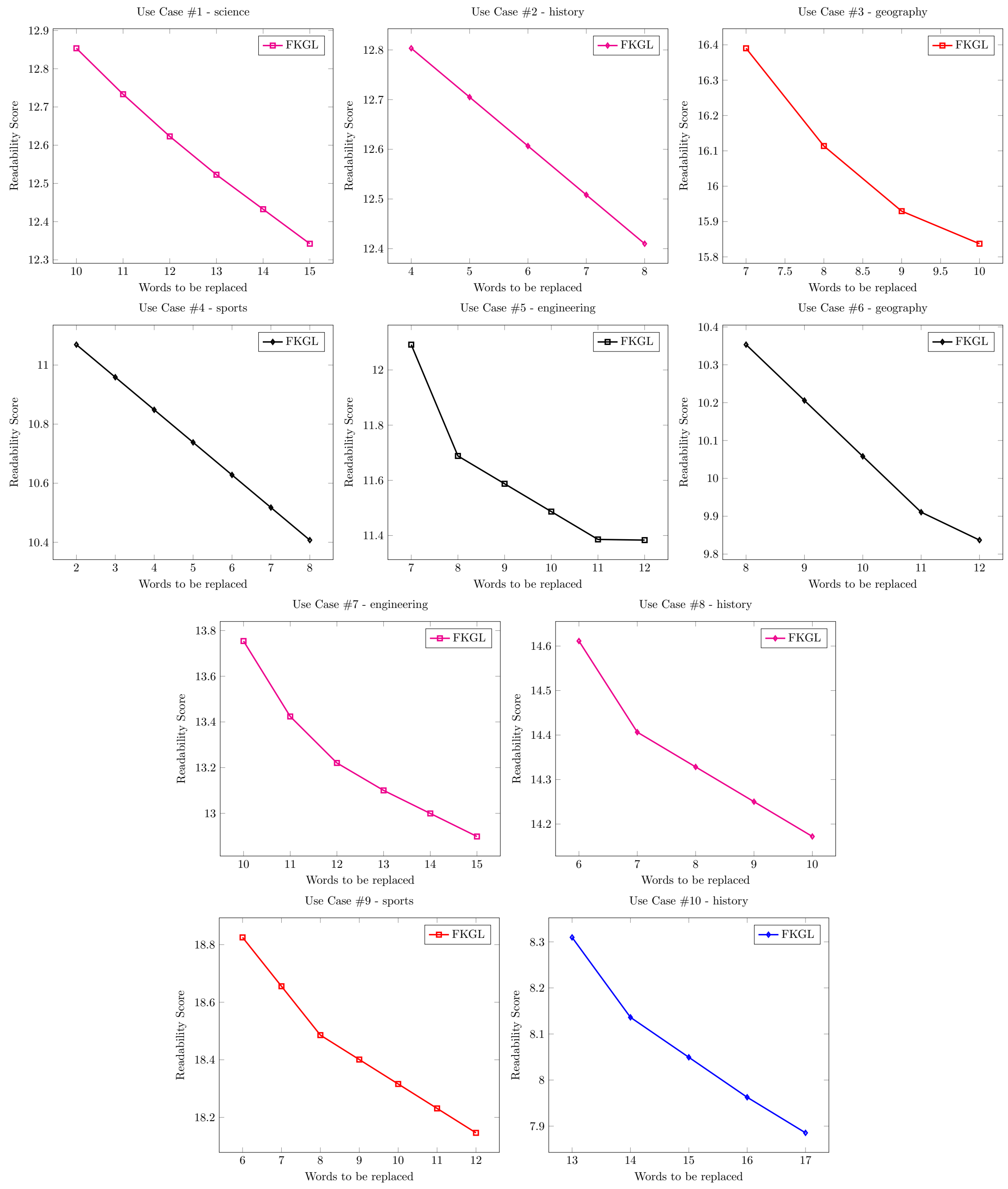


Figure 4: Non-dominated solutions for ten use cases obtained using NSGA-II

4.4.3. Discussion

We have shown how it is possible to design a solution that enhances the readability of texts, and we implemented that solution through MOO techniques. In addition, this sort of optimization is carried out while paying attention to the form of the texts in question to minimize the impact of replacing words with synonyms that better fit the readability criteria of the various formulas we have researched.

As we explained earlier the paper, this kind of optimization can also occur in two distinct ways: it can be carried out to lower the readability score, making it possible for more people to comprehend the text altogether. Also, as a second possibility, it is possible to do so in order to improve the readability score, which will enable a lesser number of individuals to comprehend the processed text.

Although the solutions will always be presented as a Pareto front for the human operator to choose his solution, it is also possible to define profiles: conservative, medium, and aggressive, which will opt for the conservative, medium, or more daring versions respectively without any question, without prejudice to the fact that it should be possible to manually edit a word that does not fit the context.

5. Part III: Preserving the essence of the original text

In Section 3, we have seen how it is possible to design and implement a functional solution to optimize the readability of any text. Although such a solution seems to work quite well, it cannot be considered fully automatic since a human user still has to evaluate whether any word used to replace original terms is not out of place in the context of the original message. In Section 4, we have used a MOO strategy to ensure that we only replace a minimum number of words in the original text in order to preserve its form, and we even go a step further and allow a human operator to decide the degree of impact of ORUGA on the form of the original text. In Section 5, we will address a remaining problem, which is to keep under control the semantic distance between the original text and the generated text so that a short distance guarantees that the original text has not been distorted, while a long-distance means that we have been able to optimize readability by a large amount, but at a high cost in terms of altering the meaning of the original text. The importance of this third and final part is that success in our strategy is what can guarantee that the solution is entirely unsupervised.

5.1. Technical Preliminaries

The additional element we will add in this part is to measure the semantic distance between the original text and the text to be delivered. In this way, we can control any significant change in the meaning of the message. This semantic distance will be the third objective in our MOO strategy.

In the literature, there are methods that allow us to determine the semantic distance between two pieces of text in a meaningful way, even when those pieces do not share any words. To do that, words are embedded as vectors using this method. It has been demonstrated to perform better than many of the methods considered to be state-of-the-art in the k-nearest neighbor's classification.

With the help of Word Mover's Distance (WMD) [12], and given pre-trained word embeddings, it is possible to automatically assess the semantic distance between two texts by computing the minimum distance that the embedded words of one text need to travel to reach the ones of another text. So, for example, we can use the WordNet library to calculate synonyms that allow us to optimize text readability. At the same time, we can use word2vec to supervise that the semantic between the generated text and the original text is minimized. We could use as a metaphor that it is an adversarial process.

The beauty of this approach is that the different synonym libraries now do not compete with each other but collaborate to try to measure (and therefore facilitate control) the semantic distance between the original text and the final text. Values close to zero will indicate that the meaning of the texts under consideration is practically equivalent, while distances approaching infinity indicate that the texts are incredibly different.

Moreover, we do not have to concern about whether the genetic algorithm replaces a word with one or more words (e.g., 'considering' by 'taking into account') since the WMD is prepared for this contiguity, as it assumes by design that the texts will not have the same length. Since when using WMD, each word is matched against all other words, but weighted by a flow matrix T that ensures the semantic distance will be symmetric, even when an unequal number of words must be matched.

5.2. Implementation

In this work, we have decided to use WMD to facilitate the measurement of the semantic distance between the initial text and the text that will be delivered at the end of the process. This ensures that the synonyms used to replace candidate words are coherent. This choice is because WMD can measure the amount of semantic distance that separates two pieces of text by comparing the words that are important to each other. This is true even if the two pieces of text do not share any words.

In addition to that, the method makes use of a representation known as the bag-of-words representation. The idea behind this method is that it should be possible to figure out how far apart two different texts are by figuring out the best way to move the distribution of the source text and the text being targeted.

We can formally define our strategy, so that let d and d' be the embedding representation of two texts, and $T \in \mathcal{R}^{n \times n}$ where $T_{ij} \geq 0$ means how much of word i in d travels to word j in d'. Furthermore, the distance between i and j might be $c(i, j) = \|x_i - x_j\|$. By $c(i, j)$, we denote the cost of moving from one word to another. In order to transform d into d', it is necessary to be sure that the flow from i is equivalent to d_i so that $\sum_j T_{ij} = d_i$. In this way, the minimum cumulative cost of moving d to d', given all these constraints, is provided by the solution shown in Equation 7.

$$
\begin{aligned}
arg\ min\ \sum_{i,j=1}^{n} T_{ij} c(i,j) \\
subject\ to\ \sum_{j=1}^{n} T_{ij} = d_i\ \forall i \in \{1,2,3\cdots n\}\ \wedge \\
\sum_{i=1}^{n} T_{ij} = d'_j\ \forall j \in \{1,2,3\cdots n\}
\end{aligned}
\tag{7}
$$

Therefore, we use a function that determines the distance between two texts as the cumulative sum of the minimum distance each word in one text must move in vector space to the closest word in the other text. Word embeddings derived from word2vec will be utilized for this work because of their capability to maintain critical aspects of the context in which a word is used.

WMD is used quite frequently these days to calculate semantic distances, and this is one of the reasons why. The only problem is that the complexity of computing the constrained minimum cumulative cost in the worst case is $O(p^3 \log u)$, where u is the number of unique words in the text [32]. Therefore, when working with texts that contain a large number of unique words, WMD may perform poorly. However, there are some techniques that improve performance.

In the context of this work, we have used the library *gensim*[3] implementation of the WMD fed by the word embeddings from word2vec [23]. In this way, what was previously a rival synonym library now becomes an adversarial library that helps keep semantic distance under control.

5.3. Illustrative examples

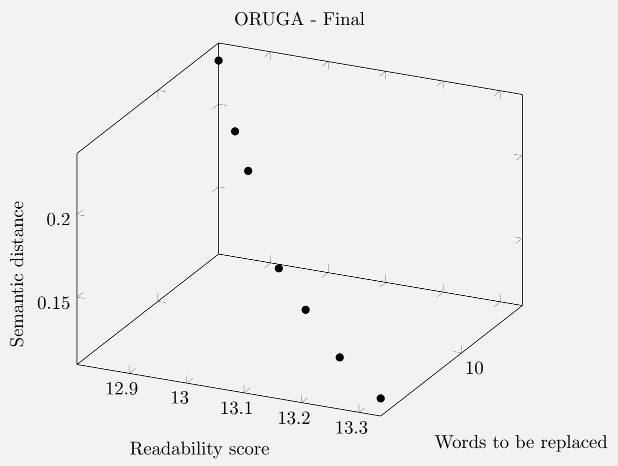
Example 5 provides us with information about engineering that was taken from Wikipedia. We are interested in observing how ORUGA operates while attempting to minimize the FKGL readability score, the number of words that need to be replaced, and the semantic distance between the original text and the text that has been processed. We are replacing the synonyms with the help of WordNet, and we are moving forward with the MOO with the help of NSGA-II. The ultimate goal is to ensure a very low risk of distortion of the original message that wanted to be communicated.

[3]https://pypi.org/project/gensim/

Example 5. Engineering

Original text **Source:** Wikipedia

"Big data refers to data sets that are too large or complex to be dealt with by traditional data-processing application software. Data with many fields (rows) offer greater statistical power, while data with higher complexity (more attributes or columns) may lead to a higher false discovery rate. Big data analysis challenges include capturing data, data storage, data analysis, search, sharing, transfer, visualization, querying, updating, information privacy, and data source. Big data was originally associated with three key concepts volume, variety, and velocity. The analysis of big data presents challenges in sampling, and thus previously allowing for only observations and sampling. Thus a fourth concept, veracity, refers to the quality or insightfulness of the data." **ARI score: 14.43**.



Minimize ARI score with the least risk of distorting the original message

Big data bring up to data sets that are too big or complex to be dealt with by traditional data-processing application software. Data with many fields (rows) offer greater statistical power, while data with higher complexity (more attributes or columns) may lead to a higher false finding rate. Big data analysis challenges include capturing data, data storage, data analysis, search, sharing, transfer, visualization, querying, updating, information privacy, and data source. Big data was originally tied in with three key concepts volume, variety, and velocity. The analysis of big data presents challenges in sampling, and thus previously let for only observations and sampling. Thus a fourth concept, veracity, refers to the quality or acumen of the data. **ARI score: 13.33 - ▽ 7.62%**

The example shows that we no longer operate as blindly as before. Now, we also minimize the words to be replaced and the semantic distance between the initial and the generated text. Therefore, the results are much more reasonable and can be relied upon to work in exploitation environments. The readability score optimization is less spectacular than before, but the risk of distorting the original message, both in form and content, is much more substantially reduced.

5.4. Experimental study

In this section, we have performed the empirical study to test this version of ORUGA. To do so, we designed the experiments through an experimental setup. We performed the experiments and collected the raw data. And finally, we proceeded with the analysis of the collected data.

5.4.1. Experimental setup

The accurate adjustment of the MOO strategy's parameters is not the primary focus here, as was the case with the previous parts of this research. One configuration works quite well after a brief preliminary study based on a conventional parameter-setting strategy. This came about as a result of what was mentioned above. The following is a list of the parameters that we have used:

- Population size {10, 15, 20}: **20**
- Number of parents mating {10, 15, 20}: **20**
- Number of genes: one per candidate word to be substituted by a synonym
- Fitness function: Equations 1, 2, 3, 4, words to be replaced, and WMD [12].
- Stop condition: {300, 600, 900}: **900** generations

5.4.2. Experiments

We are going to proceed with the experiments concerning minimizing the impact on the meaning of the original message. We focus now on being able to produce a result that can be put (or be close to being put into exploitation). To do this, we will conduct experiments to see if we can control the difference in meaning between the original message's content and the generated text.

Figure 5 shows us the summary of all the results obtained for the ten use cases that we have been studying throughout this research work. As can be seen, each use case, no matter how topical or challenging, corresponds to a good number of solutions ranging from the most conservative (the one that has the least risk of distorting the original message) to the most aggressive (the one that reduces the readability score more conclusively at the risk of distorting the original message). According to previous experiments, each color represents a degree of difficulty.

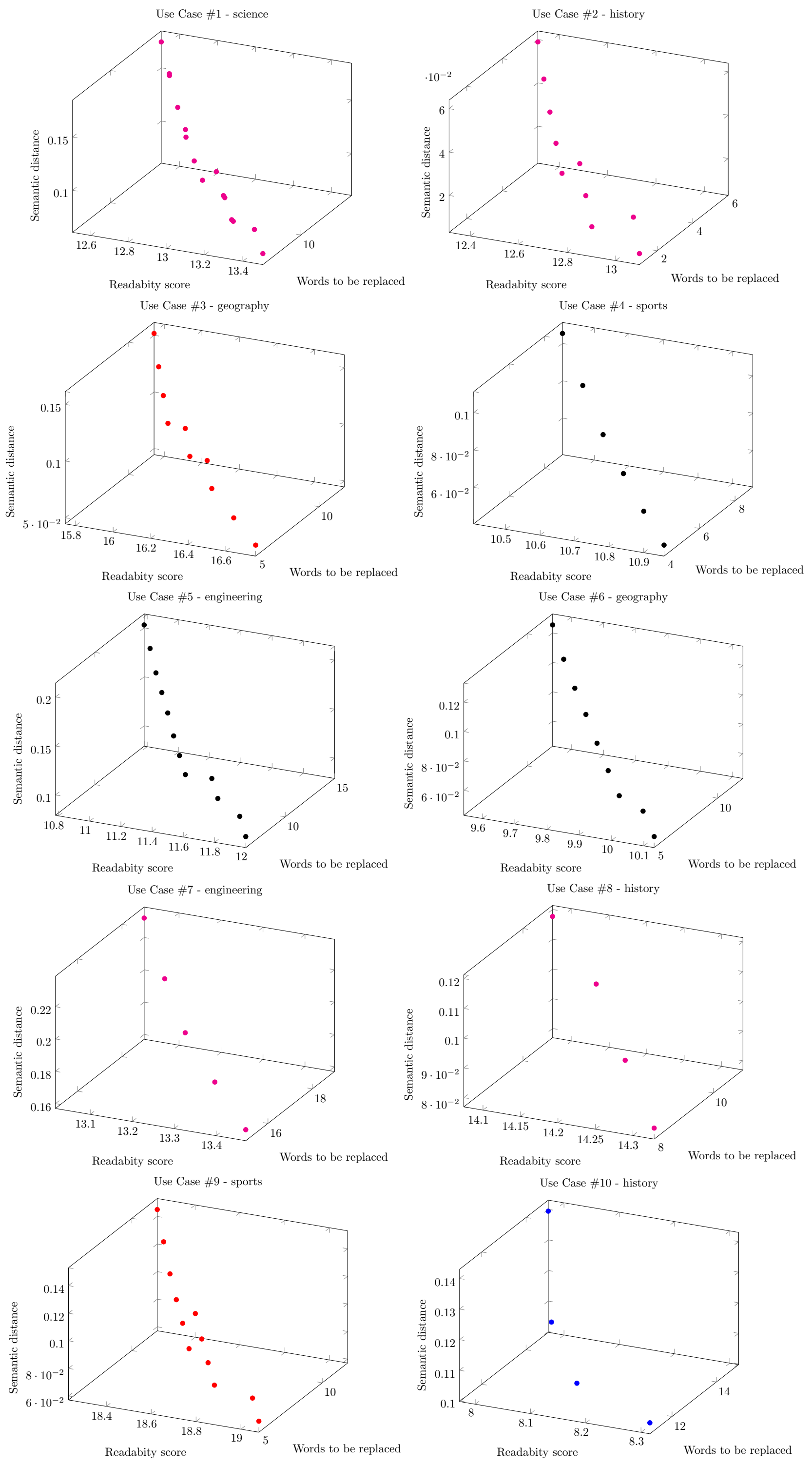


Figure 5: Summary of the results obtained for the third (and final) version of ORUGA

5.4.3. Discussion

We have seen how it is possible to build a solution that enhances the readability of texts of very different natures and readability levels, and we implemented that solution. In addition, this sort of optimization is conducted while paying attention to both the content and the structure of the texts in question to minimize the impact of replacing words with synonyms that are a better fit for the readability criteria of the various formulas we have researched.

Throughout this research, we have seen how it is possible to use genetic algorithms to improve the readability of any text formulated in English. As we have explained earlier, optimization is bi-directional. That is, it can automatically increase or decrease the readability of the text. Once we established the basis for the optimization, we could control the number of words that could be replaced; we could also control that the semantic distance between the original text and the final text does not skyrocket. Therefore, our initial goals of building a solution that improves the readability of any text without significantly altering its form or content have been achieved.

6. Conclusion

In this research work, we have seen how ORUGA can automatically optimize the readability of a text by using genetic algorithms. We have shown that by automatically replacing some words of the text to be optimized by their synonyms, we can optimize the readability levels in the direction (minimize or maximize) we wish. Neither the content nor the form of the text is altered because a minimal impact on the transformation of the original text is sought through various additional MOO techniques. Although, in theory, analogous solutions could be built using neural language models, this approach has a significant advantage: it is unsupervised and requires no training.

An exhaustive empirical study has shown that we have successfully performed all experiments. In the first instance, these experiments consisted of processing texts of different natures (history, geography, sports, nature, and science) using three synonym libraries and using different readability metrics. In the second instance: designing a MOO solution to control the number of words to be replaced. We have tested different texts to assess and compare the feasibility of this approach. Furthermore, in the third instance: measuring and controlling the semantic distance between the original text and the one that will finally be outputted. For this, we have used a novel technique that uses a library of synonyms to control the results obtained with another library of different synonyms in an adversarial process.

The results demonstrate that our hypothesis about text readability optimization at the beginning of this paper is valid. Despite the success of this research, it is necessary to bear in mind that simple formulas are typically simpler to put into practice, which is a limitation of this body of work. These formulas have a fundamental inability to model the semantics of word usage in context, which is needed to capture richer ideas of text difficulty.

As future work, a possible approach (yet computationally expensive) would be using a model of contextual embeddings such as BERT on a large dataset. Then, it should be necessary to store pairs of words and corresponding contextual representations and then use the nearest neighbors approach to identify synonyms that include the context. In this way, the impact of text processing will be even more limited.

Source code

The source code of this approach is published under MIT license in the following Github repository: `https://github.com/jorge-martinez-gil/oruga`.

Acknowledgments

The research reported in this paper has been funded by the Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation, and Technology (BMK), the Federal Ministry for Digital and Economic Affairs (BMDW), and the State of Upper Austria in the frame of SCCH, a center in the COMET - Competence Centers for Excellent Technologies Programme managed by Austrian Research Promotion Agency FFG.

References

[1] Ante, L. (2022). The relationship between readability and scientific impact: Evidence from emerging technology discourses. *J. Informetrics*, *16*, 101252. URL: `https://doi.org/10.1016/j.joi.2022.101252`. doi:`10.1016/j.joi.2022.101252`.

[2] Benítez-Hidalgo, A., Nebro, A. J., García-Nieto, J., Oregi, I., & Ser, J. D. (2019). jmetalpy: A python framework for multi-objective optimization with metaheuristics. *Swarm and Evolutionary Computation*, (p. 100598). URL: `http://www.sciencedirect.com/science/article/pii/S2210650219301397`. doi:`https://doi.org/10.1016/j.swevo.2019.100598`.

[3] Chall, J. S., & Dale, E. (1995). *Readability revisited: The new Dale-Chall readability formula*. Brookline Books.

[4] Chandrasekaran, D., & Mago, V. (2021). Evolution of semantic similarity - A survey. *ACM Comput. Surv.*, *54*, 41:1–41:37. doi:`10.1145/3440755`.

[5] Collins-Thompson, K. (2014). Computational assessment of text readability: A survey of current and future research. *ITL-International Journal of Applied Linguistics*, *165*, 97–135.

[6] Dale, E., & Chall, J. S. (1948). A formula for predicting readability: Instructions. *Educational research bulletin*, (pp. 37–54).

[7] Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.*, *6*, 182–197. URL: `https://doi.org/10.1109/4235.996017`. doi:`10.1109/4235.996017`.

[8] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (pp. 4171–4186). Association for Computational Linguistics. doi:`10.18653/v1/n19-1423`.

[9] Forrest, S. (1996). Genetic algorithms. *ACM Computing Surveys (CSUR)*, *28*, 77–80.

[10] Kincaid, J. P., Fishburne Jr, R. P., Rogers, R. L., & Chissom, B. S. (1975). *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel*. Technical Report Naval Technical Training Command Millington TN Research Branch.

[11] Kukkonen, S., & Lampinen, J. (2005). Gde3: The third evolution step of generalized differential evolution. In *2005 IEEE congress on evolutionary computation* (pp. 443–450). IEEE volume 1.

[12] Kusner, M., Sun, Y., Kolkin, N., & Weinberger, K. (2015). From word embeddings to document distances. In *International conference on machine learning* (pp. 957–966). PMLR.

[13] Madrazo Azpiazu, I., & Pera, M. S. (2020). Is cross-lingual readability assessment possible? *Journal of the Association for Information Science and Technology*, *71*, 644–656.

[14] Maqsood, S., Shahid, A., Afzal, M. T., Roman, M., Khan, Z., Nawaz, Z., & Aziz, M. H. (2022). Assessing english language sentences readability using machine learning models. *PeerJ Comput. Sci.*, *8*, e818. URL: `https://doi.org/10.7717/peerj-cs.818`. doi:`10.7717/peerj-cs.818`.

[15] Martinc, M., Pollak, S., & Robnik-Šikonja, M. (2021). Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*, *47*, 141–179.

[16] Martinez-Gil, J. (2022). A comprehensive review of stacking methods for semantic similarity measurement. *Machine Learning with Applications*, *10*, 100423.

[17] Martinez-Gil, J., & Chaves-Gonzalez, J. M. (2019). Automatic design of semantic similarity controllers based on fuzzy logics. *Expert Syst. Appl.*, *131*, 45–59. doi:`10.1016/j.eswa.2019.04.046`.

[18] Martinez-Gil, J., & Chaves-Gonzalez, J. M. (2020). A novel method based on symbolic regression for interpretable semantic similarity measurement. *Expert Syst. Appl.*, *160*, 113663. doi:10.1016/j.eswa.2020.113663.

[19] Martinez-Gil, J., & Chaves-Gonzalez, J. M. (2021). Semantic similarity controllers: On the trade-off between accuracy and interpretability. *Knowl. Based Syst.*, *234*, 107609. doi:10.1016/j.knosys.2021.107609.

[20] Martinez-Gil, J., & Chaves-Gonzalez, J. M. (2022). Sustainable semantic similarity assessment. *Journal of Intelligent & Fuzzy Systems*, *43*, 6163–6174. doi:10.3233/JIFS-220137.

[21] Mc Laughlin, G. H. (1969). Smog grading-a new readability formula. *Journal of reading*, *12*, 639–646.

[22] Meade, C. D., & Smith, C. F. (1991). Readability formulas: cautions and criteria. *Patient education and counseling*, *17*, 153–158.

[23] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.* (pp. 3111–3119).

[24] Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, *38*, 39–41.

[25] Mitchell, R. (2018). *Web scraping with Python: Collecting more data from the modern web*. " O'Reilly Media, Inc.".

[26] Navigli, R., & Martelli, F. (2019). An overview of word and sense similarity. *Nat. Lang. Eng.*, *25*, 693–714. doi:10.1017/S1351324919000305.

[27] Pantula, M., & Kuppusamy, K. S. (2022). A machine learning-based model to evaluate readability and assess grade level for the web pages. *Comput. J.*, *65*, 831–842. URL: https://doi.org/10.1093/comjnl/bxaa113. doi:10.1093/comjnl/bxaa113.

[28] Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). Wordnet: : Similarity - measuring the relatedness of concepts. In D. L. McGuinness, & G. Ferguson (Eds.), *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA* (pp. 1024–1025). AAAI Press / The MIT Press.

[29] Qin, Y. (2021). Comparable study on readability of machine generated news and human news. In *6th International Conference on Control, Robotics and Cybernetics, CRC 2021, Shanghai, China, October 9-11, 2021* (pp. 339–343). IEEE. URL: `https://doi.org/10.1109/CRC52766.2021.9620136`. doi:`10.1109/CRC52766.2021.9620136`.

[30] Rus, V., Lintean, M. C., Banjade, R., Niraula, N. B., & Stefanescu, D. (2013). SEMILAR: the semantic similarity toolkit. In *51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, Proceedings of the Conference System Demonstrations, 4-9 August 2013, Sofia, Bulgaria* (pp. 163–168).

[31] Senter, R., & Smith, E. A. (1967). *Automated readability index*. Technical Report Cincinnati Univ OH.

[32] Skianis, K., Malliaros, F. D., Tziortziotis, N., & Vazirgiannis, M. (2020). Boosting tricks for word mover's distance. In *International Conference on Artificial Neural Networks* (pp. 761–772). Springer.

[33] Todirascu, A., François, T., Bernhard, D., Gala, N., & Ligozat, A.-L. (2016). Are cohesive features relevant for text readability evaluation? In *26th International Conference on Computational Linguistics (COLING 2016)* (pp. 987–997).

[34] Wilkins, D. A. (1972). *Linguistics in language teaching* volume 111. Edward Arnold London.

[35] Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.*, *11*, 712–731. URL: `https://doi.org/10.1109/TEVC.2007.892759`. doi:`10.1109/TEVC.2007.892759`.

[36] Zhu, G., & Iglesias, C. A. (2017). Computing semantic similarity of concepts in knowledge graphs. *IEEE Trans. Knowl. Data Eng.*, *29*, 72–85. URL: `https://doi.org/10.1109/TKDE.2016.2610428`. doi:`10.1109/TKDE.2016.2610428`.