Article

Compression of Text in Selected Languages—Efficiency, Volume, and Time Comparison

Beniamin Stecuła 1, Kinga Stecuła 2,* and Adrian Kapczyński 1

1 Faculty of Applied Mathematics, Silesian University of Technology, Akademicka 2A, 44-100 Gliwice, Poland
2 Faculty of Organization and Management, Silesian University of Technology, Akademicka 2A, 44-100 Gliwice, Poland
* Correspondence: kinga.stecula@polsl.pl

Abstract: The goal of the research was to study the possibility of using the planned language Esperanto for text compression, and to compare the results of the text compression in Esperanto with the compression in natural languages, represented by Polish and English. The authors performed text compression in the created program in Python using four compression algorithms: zlib, lzma, bz2, and zl4 in four versions of the text: in Polish, English, Esperanto, and Esperanto in x notation (without characters outside ASCII encoding). After creating the compression program, and compressing the proper texts, authors conducted an analysis on the comparison of compression time and the volume of the text before and after compression. The results of the study confirmed the hypothesis, based on which the planned language, Esperanto, gives better text compression results than the natural languages represented by Polish and English. The confirmation by scientific methods that Esperanto is more optimal for text compression is the scientific added value of the paper.

Keywords: processing; compression; coding; compression algorithms; languages

1. Introduction

Data compression is one of the achievements of the information revolution. This revolution has changed our lives, especially in the field of application and the use of digital content. Despite the fact that the phenomenon of data compression is often not visible, it has become more and more ubiquitous. Data compression is part of the information technology of our lives. It is one of the enabling technologies for each of the aspects of the multimedia revolution. Images, audio, and video files are available for placement on the website thanks to compression algorithms. For a long time, data compression was the domain of relatively small groups of engineers and scientists, but now it is widespread [1]. The daily activity of people on the Internet is possible thanks to compression. Nowadays, more and more information is in digital form. The number of bytes needed to represent multimedia data can be huge. Although hardware manufacturers and companies produce tons of hardware in an attempt to provide a better solution to work with large amounts of data, it is almost impossible to keep those data uncompressed [2].

The explosive growth in data generation led to development in data transmission and storage. The possibilities and efficiency of transmitting and storing content depend, on the one hand, on technology, and on the other hand on input data. Nowadays, more and more data are generated, and they must be stored. Ways to optimally store data should be looked for. This refers to different types of files, including text files. Text data can be compressed and stored in different languages, and a lot of them use English; for example [3–7]. However, it turns out that the issue of which language is the most optimal in terms of compression efficiency, time and volume remains insufficiently researched. There is no research in the literature or practice on the problem of which language is most optimal for compressing text files. There is also little research on artificial languages...
Sensors 2022, 22, 6393 and their application in computer science, especially in text compression. Studies on the compression and storage of text do not take into account the influence of the type of language. The authors of this article noticed that the length (volume) of the same text for different languages is different. It means that as input data for compression, the same text differs in the case of different languages. Therefore, the authors decided to conduct research in the field of comparing the compression of the same texts for selected languages. The authors studied the Esperanto language, which is an artificial (planned) language, in the context of its role in compression, and decided to study the problem of text compression depending on the language. The authors decided to investigate effects that would be obtained by compressing a text in Esperanto. The Esperanto language is characterized by simplicity, regularity, and often repetitive fragments [8]. The words in this language are composed of unchanging morphemes, which makes it naturally suitable for computer compression. In order to compare the effects of compression, the authors also chose Polish and English for the research. The Polish language was chosen due to the fact that it is the native language of the authors of the research. In turn, the English language is one of the most popular and widely used languages; moreover, it is the language in which this publication is written.

A new and thus-far unexplored direction of research is searching for an optimal language suitable for text compression. The authors undertook the study, choosing Esperanto due to the above-mentioned features of this language. The main goal of this research was to study the possibility of using the Esperanto language for text compression and to compare the results of text compression in Esperanto, Polish, and English. The authors formulated the research gap as the lack of direct text comparison of compression in the planned language Esperanto and natural languages. In the research, the following research question was asked: “Is the planned language Esperanto more suitable for the compression process than the natural languages represented by Polish and English?” The hypothesis was formulated as: “Planned language Esperanto gives better text compression results than natural languages represented by Polish and English”. The scientific added value of the research was determining the language that gives better text compression results. The scope of the research included the following:

- Study of Esperanto grammar and vocabulary.
- Development of the theoretical background of the solution.
- Selection of research tools.
- Finding the text in Esperanto, Polish, and English as input data for the compression process.
- Implementation of a program for text compression.
- Process of text compression.
- Analysis of the results.
- Development and update of the program in the future.

The paper is divided into five sections. Section 2 discusses related work on the topic of text compression. Section 3 describes the materials and methods used in the research. This section includes the selection of the planned language Esperanto, input data, a research tool, and a developed program for text compression. The compression program was developed by the authors. Section 4 presents the results of text compression and an analysis of the results of this compression. Section 5 includes conclusions.

2. Related Work

Data compression is a process that reduces the size of the data, removing excessive information [9]. A smaller data size is suitable because it reduces costs. Compression aims to reduce redundancy in stored or communicated data, thus increasing the effective density of the data. Compression is the representation of data in a reduced form so that data can be saved by using a small amount of storage and sent with limited bandwidth. In practice, data compression is based on reducing the size of the text by finding regularities such as long sequences of the same characters or repetitive sequences of letters and words [1]. Data
compression can be divided into two techniques, which are the following: lossless and lossy [1,10]. Lossless compression reproduces the data perfectly from its encoded bit stream. The original information content can be recreated in its original and unchanged form. This type of compression is only about changing the recording of the information. In lossy compression, in turn, less significant information is removed. It is a more efficient type of compression. It keeps the most important properties of the output information, but it may lose some details. It is often used to compress graphics and video files in such a way that only information noncrucial to the user is lost. An example is playing a movie with a lower resolution than the original. Computer compression usually needs to be uncompressed to be readable for humans [1]. There are many types of lossless text compression. They include the Burrows–Wheeler transform, Huffman coding, arithmetic coding, run-length coding, Deflate, Lempel–Ziv 77 (LZ77), Lempel–Ziv–Welch (LZW), GNU zip (Gzip), Bzip2, Brotli, and many more [1,11]. Some statistical methods assign a shorter binary code of variable length to the most frequently repeated characters, and examples of this method are Huffman and arithmetic coding. According to [12], the Huffman coding is one of the best algorithms in this category. As [13] claims, Deflate provides slightly poor compression, but its encoding and decoding speeds are fast. In addition, in the literature, a lot of research on the compression issue can be found; for example [14–17].

Compression is a subject of some research that is described in the literature. The authors of [18] proposed a novel frequent pattern-mining-based Huffman encoding algorithm for text data and used a hash table in the process of frequent pattern counting. Their algorithm operates on pruned set of frequent patterns, and is efficient in terms of database scanning and storage space, reducing the code table size. The main objective of the paper [19] was to identify which compression technique was better in text, image, and audio compression applications. According to authors, for text files, due to a shorter compression ratio and lower compression time, the Huffman Algorithm is recommended. For the image files, Lempel–Ziv was better compared to the Huffman Algorithm because of the huge difference between their compression ratios and compression times. For audio files, the comparison between the compression times of the two algorithms was inconclusive because there was no significant difference between the two. Another research study [20] focused on an engineering perspective of Data Mining, using it as a tool for efficient data compression. The research exploited the principle of assigning shorter codes to frequently occurring patterns in relation to the single-character-based code assignment approach of Huffman encoding. Research in [4] presented a new algorithm for the compression of very short text messages. Their algorithm converts the input text consisting of letters, numbers, spaces, and punctuation marks commonly used in English writings to a format which can be compressed in the second phase, which is a transformation reducing the size of the message by a fixed fraction of its original size.

The topic of compression is also related to the Internet of Things (IoT) issue. As [21] claims, a key challenge of the modern digital world is reducing the size of the transmitted data without sacrificing their quality. A natural solution is to compress data at the sensing devices. Therefore, this problem is the subject of many studies. The authors of [22] proposed a novel method to significantly enhance transformation-based compression standards, such as JPEG, by transmitting much fewer data from an image at the sender’s end. They proposed a two-step method by combining the state-of-the-art signal processing-based recovery method with a deep residual learning model to recover the original data. The authors of [21] introduced Sprintz, which is a compression algorithm for multivariate integer time series that achieves state-of-the-art compression ratios across a large number of publicly available datasets. The work of [23] describes a novel multivariate data compression scheme for smart metering IoT. The proposed algorithm exploits the cross-correlation between different variables sensed by smart meters to reduce the dimension of the data. The studies of [24] presents an optimal compression technique using CNNs for remote sensing images. The method uses CNN to learn the compact representation of the original image which held the structural data and was then coded by the Lempel–Ziv-Markov chain
algorithm. There is also more work on lossless compression algorithms suitable for IoT; for example, [25–28].

In the literature, there are more studies that focus on the compression of English texts; they include, for example: [29–34]. There are not many works that focus on different languages. The first example is [35], which shows transliteration-based Bengali text compression. Other studies focused on German [36,37] and Arabic [38,39], but there is also a study on compression comparison of English, German, French, Japanese, and Chinese [40], and English, German, French, Italian, Czech, Hungarian, Finnish, and Croatian [41]. When it comes to Polish, there are not many studies [42–45]. However, based on the literature review, it can be claimed that there are no studies on compression comparison of English, Polish, and Esperanto at all—and this justifies the authors’ need to conduct research in this field.

3. Materials and Methods

3.1. Selection of the Planned Language Esperanto

In 1887, Ludwik Zamenhof officially published the first version of the international language he had created, known as Esperanto. This language was created in order to reconcile nations in the territory he inhabited. These nations communicated in three or more languages, which resulted in frequent misunderstandings, divisions, and prejudices [46,47]. Due to his language, Ludwik was nominated eight times for the Nobel Peace Prize [48]. In 1908, the Universala Esperanto Asocio (UEA), i.e., the World Esperantists’ Union, respected by such organizations as the United Nations, UNICEF, UNESCO, the Council of Europe, the International Organization for Standardization (ISO), and the Organization of American States, was funded [49].

Esperanto is an artificial language. Its main characteristic is its regularity (due to the immutability of morphemes). Due to its unambiguous and analytical nature, Esperanto is very precise. It should be noted that parts of speech are easily distinguished due to specific grammatical endings. In the context of sentence order, the order of words is characterized by great freedom. The grammar of the language is described in Zamenhof’s Foundation of Esperanto (epo. Fundamento de Esperanto) [8].

The vocabulary of Esperanto has its roots in European languages, mainly Romance (around 75%) and Germanic (around 20%) [50]. There are several hundred words in the Esperanto Foundation, and nowadays, their number exceeds a thousand. All of these words form a complete language, as they also form the basis for more complex expressions that are formed by putting together existing words.

Thus, the vocabulary in Esperanto forms a certain system that allows us to obtain a range of meaning many times greater than in the case of languages without extensive prefixing and suffixing. Therefore, it should be stated that within the framework of the currently existing database, it is possible to create concepts that do not yet have equivalents in natural languages.

Esperanto was not the first artificial (planned) language. It is worth mentioning the Volapük language, which directly preceded Esperanto [51]. The creator of this language, Johann Schleyer, did not agree with any reforms proposed by the academy of this language. The creator’s resistance led to a schism, and then to a decline in the language’s popularity. Volapük was a regular and schematic language, but the extensive and difficult-to-remember and pronounce vocabulary meant that even the creator himself was not able to use it fluently [52]. Nowadays, the language is still used on the Internet [53]. However, it appears only in written form—in oral form, it is used only by hobbyists. After Esperanto, many other artificial languages have emerged, such as Occidental (Edgar von Wahl, 1922), Novial (Jesperen, 1928), Interlingua (1951), and Romanid (Zoltan Magyar, 1956) [54]. Moreover, some languages were directly based on Esperanto, including Ido (1907), Reform Esperanto (1910), and Latin Esperanto (1911) [54]. However, it should be emphasized that all these popular languages had the disadvantage of being “naturalistic”. This means that they
imitated natural languages to such an extent that they consequently lost their original feature; that is, regularity.

Based on the idea accompanying the Esperanto language, a movement called “Esperantism” was founded. Its main assumptions were the following [8]:

- Striving to introduce a neutral human language around the world;
- Popularization of Esperanto through practice, including increasing the library of sources (both original and translations);
- Waiver of rights by the author of the Esperanto language—Esperanto belongs to everyone;
- No one can introduce new rules into the language. The only source is the Esperanto Fundament;
- Every user of Esperanto is an Esperantist.

It is worth noting that these goals are very similar to the currently popular idea behind many programming languages or software, which is creating and sharing open source code. Due to the simplicity, regularity, and specific grammar and vocabulary of the Esperanto language, the authors decided to undertake research on the use of this language in computer science. The authors studied the grammar, vocabulary, and principles of the language. Based on the analysis, the authors concluded that Esperanto is optimal for use as a human language, which will be easily analyzed and processed by computer software. It is also worth noting that, thanks to the popularity of this language, numerous source materials that can be used as valuable research sources were created.

Moreover, it is worth noting that due to its regularity, the Esperanto language was applied in many projects and became the subject of many studies in the field of computer science; for example, creating an algorithm for morphological segmentation of Esperanto words [55], a study on Zipf’s law [56], building a model of the world context [57], enhancing the development of human–machine communication [58]. Moreover, the authors [59] focused their paper on discussing the digital presence of Esperanto. They assessed its digital vitality on the basis of its language ideology and other sociolinguistic data. Other researchers [60] quantified the irregularity of different European languages belonging to four linguistic families and an artificial language (Esperanto). They worked on modifying a well-known method to calculate the approximate and sample entropy of written texts. They based their method on the search for regularities in a sequence of symbols and consistently distinguishing between natural and synthetic randomized texts. The mentioned reviewed research examples, followed by the authors’ experience, inspired the authors of this article to conduct their research on text compression in Esperanto.

3.2. Input Data

For a text compression program to produce meaningful results, it had to operate with varied and long text as input research material. The authors had to look for a text available in each of the three selected languages. Therefore, the study uses the entirety of the novel by the Polish writer Henryk Sienkiewicz—Quo vadis. The author of the novel was the first Polish writer to receive the Nobel Prize for Literature—this was in 1905 [61]. Quo vadis, written in Polish and published in 1896, was recognized as a worldwide bestseller at that time [62]. The novel has been translated into over 30 languages. It is a novel in the public domain with readily available translations, including English and Esperanto. The selection of this novel as the subject of the research carried out allowed us to obtain input text (input data) with very similar characteristics and length, minimizing the differences that could affect the result of the experiment in an unexpected way.

The authors searched the database of novels in the public domain and found the novel Quo vadis by Henryk Sienkiewicz. The sources of the downloaded novel and its translations were the following:

- In Polish—wolnelektury.pl (accessed on 5 February 2022) [63];
- In English—gutenberg.org (accessed on 5 February 2022) [64];
- In Esperanto—tekstaro.com (accessed on 15 January 2022) [65].
The data were saved in the .txt format in 8-bit Unicode Transformation Format (UTF-8) coding.

The most popular compression algorithms were used to compress the text. Four algorithms selected for the research include the following:

- Zeta Library (zlib);
- Lempel–Ziv–Markov chain algorithm (lzma);
- Bz2;
- Lz4.

The processing and presentation of the collected data included the following:

- Examination of the compressed text and measurement of the compression time;
- Data presentation in text form was provided through a specially created class, which stores the collected data and writes them to the console;
- Entering data into the Microsoft Excel spreadsheet table and presenting the results in the form of tables and graphs.

3.3. Research Tools

The programming language used in the compression program in this study was Python 3.9.5, a general-purpose high-level language. Its author is Guido van Rossum [66]. The greatest advantages of Python include the readability and transparency of the code, which facilitate code management, development, and debugging. Thanks to these advantages, working in this language is efficient, and the programmer can pay more attention to the logic of the implemented algorithm. Thanks to the interpretability, errors can be easily spotted—in the event of a stoppage, the program shows the place that requires attention. Python is a dynamically typed language, so there is no need to specify the type of variable to be used; it is possible to use the variable without having defined it previously [67]. The Python programming language is a valued language in the world, which is confirmed by the fact that it was chosen by Alphabet Inc., Mountain View, CA, USA (a conglomerate holding company created through a restructuring of Google) as one of its main programming languages [67].

PyCharm is an Integrated Development Environment (IDE) for professional Python developers. Its producer is the software producer Jet-Brains, known from other projects, such as IntelliJ IDEA, RubyMine, and the Kotlin language [68]. It was chosen as a research tool in this study due to its comprehensive approach to programming, which includes automated code refactoring, autocompletion, convenient keyboard shortcuts, and suggestions for improving readability based on the PEP8 standard.

The UTF-8 format was selected as a text form. It is a Unicode encoding system, created in accordance with the American Standard Code for Information Interchange (ASCII) [69]. UTF-8 is used by over 98% of the largest websites in the world and by over 97% of all types of websites [70]. It was the optimal choice for research due to its compatibility with the Polish, English, and Esperanto alphabets.

3.4. Developed Program for Text Compression

The developed text compression program initially loads the input data in each of the languages that are the subject of research. After loading the data, the developed program compresses and saves the work. Finally, the results obtained are presented. Figure 1 presents pseudo code for the developed program. The program reads the text of the book in binary form and stores it in the text variable. Each language has a separate text file named after the language/notation abbreviation adopted by the authors. Their names are the following:

- pl—stands for Polish language;
- en—stands for English language;
- eo—stands for Esperanto language;
- eox—stands for Esperanto language in notation x.
read input data
foreach language do:
    foreach algorithm do:
        timer start
        compress data
        timer stop
        save compression time
        save compression ratio
show results

Figure 1. Pseudo code for the developed program.

Each language has its own unique features presented in Figure 2 and described below, based on the study and the experience of the authors.

|     |     |     |
|-----|-----|-----|
| pl  | jest | 4 characters |
| en  | is   | 2 characters |
| eo  | estas | 5 characters |
| eoX | estas | 5 characters |
| pl  | przemówił | 9 characters |
| en  | spoke | 5 characters |
| eo  | ekparolis | 9 characters |
| eoX | ekparolis | 9 characters |
| pl  | pełen życia | 11 characters |
| en  | filled with life | 16 characters |
| eo  | vivoplena | 9 characters |
| eoX | vivoplena | 9 characters |
| pl  | posiłki | 7 characters |
| en  | reinforcements | 14 characters |
| eo  | tačmentoj | 9 characters |
| eoX | tačmentoj | 10 characters |
| pl  | czy czyni podobnym | 18 characters |
| en  | or something of that sort | 25 characters |
| eo  | aŭ io similа | 12 characters |
| eoX | aŭ io similа | 13 characters |
| pl  | A nie powoziaż? | 15 characters |
| en  | And dost thou drive a chariot? | 30 characters |
| eo  | Ču vi ĉarveturas? | 17 characters |
| eoX | Ču vi ĉarveturas? | 19 characters |

Figure 2. The comparison of the lengths of the characters in each language.

Text in Polish—the Polish language is characterized by having characters in its alphabet that are not in ASCII encoding, so each such letter takes up more space.

Text in English—the English language is characterized by the fact that the basic words are short, but the names of more complex concepts often require the use of a few words, so that the volume of the translated text is larger than the original.

Text in Esperanto—Esperanto is characterized by longer base words than the English language, but shorter words for more complicated concepts. Therefore, long and complicated texts are smaller than those in Polish and English.
Text in Esperanto in x notation—in order to exclude the influence of encoding on compression, text written with x notation was used.

Letters with signs above them (for example “ˆs” or “˘u”) have their ASCII equivalents followed by the letter x (e.g., instead of “šaūmo” it would be “sxauxmo”). These letters appear in a minority of words, so most of the text remained unchanged. Table 1 shows the data on uncompressed text. It can be noticed that the text length is the same for the eox and eo versions. Encoding letters with special signs takes two bytes, and converting them to two ASCII characters results in the encoding of two single-byte letters, which ultimately does not affect the size of the text.

**Table 1.** Data on uncompressed text.

| Parameter | pl          | en          | eo          | eox         |
|-----------|-------------|-------------|-------------|-------------|
| The volume of the uncompressed text (bytes). | 1,187,923  | 1,198,403   | 1,174,480   | 1,174,480   |

Table 2 shows the sample text from *Quo vadis* in different languages and the adequate number of characters.

**Table 2.** The sample text from *Quo vadis* in different languages and the adequate number of characters.

| Quotation                                                                 | Number of Characters |
|--------------------------------------------------------------------------|----------------------|
| “I tak minal Nero, jak mija wicher, burza, pozar, wojna lub móř, a bazylika Piotra panuje dotąd z wyżyn watykańskich miastu i świata”. | 131                  |
| “Therefore, Nero passed, as a whirlwind, as a storm, as a fire, as war or death passes; but the basilica of Peter rules till now, from the Vatican heights, the city, and the world”. | 174                  |
| “Tiel pasis Nero, kiel pasas uragano, fulmotondro, brulo, milito aŭ pesto, dum la baziliko de Petro regas ĝis nun de la Vatikana altajo la urbon kaj la mondo”. | 157                  |
| “Tiel pasis Nero, kiel pasas uragano, fulmotondro, brulo, milito aux pesto, dum la baziliko de Petro regas gxis nun de la Vatikana altajxo la urbon kaj la mondon”. | 160                  |

The data listed in the console included, inter alia, the following:
- The length of the uncompressed text;
- The name of the language for which the compression is performed;
- The name of the compression algorithm;
- The compression time;
- The percentage of space taken in relation to uncompressed text;
- Bytes space taken up by compressed text.

The timeit library was used to measure the compression time. It allows an easy measurement of the time of repeatedly providing a function. It gives a value that is the time in seconds of all the completed loops, and the value is finally divided by the number of loops to obtain the unit information. In the research, there were five hundred repetitions of each measurement.

Table 3 shows the data contained in the console.
During the research, the authors improved the text compression program. Examples of changes are the following:

- Adding a text version with x notation;
- Collecting the length of compressed texts in bytes;
- Collecting the length of uncompressed texts in bytes;
- Adding more algorithms in order to verify the set hypothesis and obtain more reliable results.

### 4. Results

#### 4.1. Text Volume before and after Compression

The subject of compression was Henryk Sienkiewicz’s novel *Quo vadis*. Figure 3 shows the uncompressed text volumes in each language. The unit is bytes. It is worth noting that the uncompressed Esperanto text occupies less space than the English and Polish texts. Compressed text, on the other hand, allows us to save even more space, as can be seen in the graphs in Figure 4.

Subsequently, the authors compared the texts after compression in terms of volume. Table 4 shows the collected data on volumetric compression efficiency. Data in rows correspond to individual algorithms; columns—languages/notations; and the values indicate the percentage of volume of the compressed text in relation to the uncompressed text—Equation (1), that is, written in an open, human-readable form, in a .txt file in UTF-8 encoding. The applied algorithms are deterministic, which means that their multiple provisions gave the same results each time.

\[
E_c = \frac{c_a}{c_b} \times 100\%
\]  

where: \(E_c\) —efficiency of compression; \(c_a\)—text volume after compression; \(c_b\)—text volume before compression.

| Algorithm | pl | en | eo | eox |
|-----------|----|----|----|-----|
| zlib      | 38.38 | 37.2 | 35.8 | 35.71 |
| lzma      | 30.66 | 29.45 | 28.58 | 28.56 |
| bz2       | 27.93 | 26.86 | 26.03 | 25.98 |
| lz4       | 43.47 | 41.87 | 40.52 | 40.48 |
Figure 5 shows the compression efficiency in the form of graphs. It can be concluded that, in terms of efficiency, compression is the least affected by Polish, then English, and Esperanto is the best. It is worth noting that the very application of the x notation, despite the fact that it does not change the text volume, improves compression. It is minimal: below one tenth of a percentage point; however, it still is an improvement. The obtained results confirm the hypothesis presented in the research, which was formulated as: “Planned language Esperanto gives better text compression results than natural languages represented by Polish and English”.

Figure 3. Volume of uncompressed text in the given languages.

Figure 4. Volume of compressed text in the given languages.
The efficiency of text compression in relation to text volume in Polish can be calculated using Equation (2):

$$E_{c_{pl}} = \frac{c_a}{c_{b_{pl}}} \times 100\%$$

where: $E_{c_{pl}}$—efficiency of compression in relation to text volume in Polish; $c_a$—text volume after compression; $c_{b_{pl}}$—text volume in Polish before compression.

**Figure 5.** The efficiency of text compression.

The authors decided to make comparisons of text compression in each of the analyzed languages, adopting a common denominator in these comparisons. In other words, the authors decided to refer each of the comparisons to the original text of *Quo vadis* in Polish. They referred compressed texts in individual languages to the uncompressed original text, i.e., in Polish. The equation to determine the effectiveness in relation to the volume of the text in Polish is presented as Equation (2). The results of the analysis are presented in Table 5 and in the graph in Figure 6. Based on this comparison, it was possible to visualize the practical percentage savings, which in the case of the LZ4 algorithm was not 1.35 pp, but 2.18 pp.

**Figure 6.** The efficiency of text compression in relation to text volume in Polish.

### Table 5. Data on the efficiency of compression in relation to the volume of text in Polish [%].

| Algorithm | pl | en | eo | eox |
|-----------|----|----|----|-----|
| zlib      | 38.38 | 37.2 | 35.8 | 35.71 |
| lzma      | 30.66 | 29.45 | 28.58 | 28.56 |
| bz2       | 27.93 | 26.86 | 26.03 | 25.98 |
| lz4       | 43.47 | 41.87 | 40.52 | 40.48 |

### Table 6. Data on the compression time [ms].

| Algorithm | pl | en | eo | eox |
|-----------|----|----|----|-----|
| zlib      | 68 | 68 | 72 | 72 |
| lzma      | 444 | 455 | 447 | 446 |
| bz2       | 82 | 81 | 81 | 82 |
| lz4       | 81 | 93 | 93 | 93 |
Table 5. Data on the efficiency of compression in relation to the volume of text in Polish [%].

| Algorithm | pl    | en    | eo    | eox   |
|-----------|-------|-------|-------|-------|
| zlib      | 38.38 | 37.53 | 35.40 | 35.31 |
| lzma      | 30.66 | 29.71 | 28.26 | 28.23 |
| bz2       | 27.93 | 27.10 | 25.73 | 25.68 |
| lz4       | 43.47 | 42.24 | 40.06 | 40.02 |

4.2. Time of Compression

Table 6 presents the collected data on the time of a single compression for the selected algorithm and language/notation. These data are presented graphically in Figure 7. Based on these charts, it can be observed that there are no major differences in the execution time of individual algorithms in different languages. The data shown were performed with five hundred repetitions of each measurement. Then, these measurements were summed and divided by their number to give the average. The total compression time, including data in all languages, all algorithms, and five hundred repetitions for each of them, was over 23 min. When analyzing the compression data, one can only conclude that there are no clear differences in compression times between languages. It is worth noting that lzma algorithms take the longest time to compress. This is in accordance with the results of another study [13], which confirm that this algorithm has the lowest compression speed compared to other algorithms.

Table 6. Data on the compression time [ms].

| Algorithm | pl | en | eo | eox |
|-----------|----|----|----|-----|
| Zlib      | 68 | 68 | 72 | 72  |
| Lzma      | 444| 455| 447| 446 |
| bz2       | 82 | 81 | 81 | 82  |
| lz4       | 81 | 93 | 93 | 93  |

Figure 7. The compression time.
4.3. Additional Comparison of Compression of Text Translated in Google Translate

To support the received results, the authors decided to carry out an experiment. Using the document translation option in Google Translate (GT), the entire novel of *Quo vadis* was also translated from English (as it is the default language for GT) into eight other languages. Then, the compression was performed with the developed program and the results were presented in Figures 8 and 9. As can be observed, among the tested languages, Esperanto achieved the best results in terms of saving disk space. It is also worth noting that the compression efficiency depends only on the language of the text and is almost independent of the algorithm used for compression (Esperanto in each algorithm got the best results, and Russian the worst).

![Figure 8. The result of additional experiment—comparison of compression of text translated in Google Translate.](image)

![Figure 9. The summarization of the additional experiment (bytes).](image)

5. Conclusions

The authors developed a program for text comparison that can effectively compress text with the use of various algorithms, perform an examination of the work time and rate of text compression, as well as present the collected information. The created program allows us to compare any texts based on various compression algorithms in terms of the time needed to perform the work and the final compression effect. It is able to operate on texts in any language and various recording formats, as it operates on binary notation. The program uses four compression algorithms, and all of them indicated that the planned language Esperanto in any form of writing allows for a higher degree of compression than the studied natural languages. Therefore, Esperanto has been concluded to provide a more space-saving way to write information for storage on a computer. This conclusion is
significant, because nowadays more and more data are produced, and they must be stored. As the literature review revealed, methods for optimal data storage are being searched for. Esperanto gives better compression results because this language is more precise, unequivocal, and regular. It gives a chance for space savings, which in terms of the Internet of Things is very important. Based on the results, it can be claimed that Esperanto gives us a chance to substantially save resources. This conclusion creates a value-added and opens a broader perspective for further research on the use of Esperanto in data compression.

The developed text compression program was used successfully to conduct an analysis of the efficiency of text compression. It correctly compresses and collects data on compression rate and time. In the study, the main goal, which was to study the possibility of using the Esperanto language for text compression and to compare the results of text compression in Esperanto, Polish, and English, was realized. The authors found the answer to the set research question, which was as follows: “Is the planned language Esperanto more suitable for the compression process than the natural languages represented by Polish and English?”. The results enabled a positive answer to this question. In the criterion of the efficiency of text compression, the results confirmed that Esperanto receives better results. The obtained results confirmed the hypothesis—“Planned language Esperanto gives better text compression results than natural languages represented by Polish and English”. The experiment showed a higher compression ratio for Esperanto language compared to the natural languages—Polish and English. It took up more than two percentage points less space than in the case of the compression of the English text. The satisfactory result of the research was obtained due to the fact that the Esperanto language is based on unchanging morphemes, as well as on regularity combined with the absence of exceptions.

The program for text compression has potential to be developed in future in the several ways. First of all, the research can be improved by adding more natural languages to the comparison. Another way is to increase the base of various compression algorithms used in the study. After receiving the best results on comparison text in Esperanto, it can be claimed that it is worth adding and studying more text sources and comparing the results. Similarly, it is worth analyzing longer and shorter texts to see if the compression in Esperanto would give better results than in this study. Moreover, a big potential for study gives replacing the compression of the entire text with the use of algorithms using Machine Learning (ML) and Neural Networks (NN). It is also important to conduct research on saving space, especially in the context of the Internet of Things and increasingly used smart sensors. Therefore, an interesting solution would be to use Esperanto in applications such as speech-to-text and text-to-speech language rather than what is usually used: English. Broader research on the use of Esperanto to store larger texts is absolutely justified, and therefore, the authors plan to deal with them in the future.

Author Contributions: Conceptualization, B.S. and A.K.; methodology, B.S.; software, B.S.; validation, B.S.; formal analysis, B.S.; investigation, B.S. and K.S.; resources, B.S.; data curation, B.S.; writing—original draft preparation, K.S.; writing—review and editing, K.S.; visualization, K.S.; supervision, A.K.; project administration, A.K.; funding acquisition, B.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Silesian University of Technology, grant number BKM-662/RMS2/2022 (09/020/BKM22/0019).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.
References

1. Sayood, K. *Introduction to Data Compression*; Morgan Kaufmann: Burlington, MA, USA, 2017.
2. Rahman, M.; Hamada, M. Burrows-wheeler transform based lossless text compression using keys and Huffman coding. *Symmetry* 2020, 12, 1654. [CrossRef]
3. Linhares Pontes, E.; Huet, S.; Torres-Moreno, J.-M.; Linhares, A.C. Cross-language text summarization using sentence and multi-sentence compression. In Proceedings of the International Conference on Applications of Natural Language to Information Systems, Paris, France, 13–15 June 2018; Springer: Cham, Switzerland, 2018; pp. 467–479. [CrossRef]
4. Kalajdzic, K.; Ali, S.H.; Patel, A. Rapid lossless compression of short text messages. *Comput. Stand. Interfaces* 2015, 37, 53–59. [CrossRef]
5. Chubaryan, A.; Sargsyan, L. The Text-Organizing Function of Compression in English Scientific Discourse. *Armen. Folia Angl.* 2016, 12, 15–26. [CrossRef]
6. Inoue, K.; Miyazaki, T.; Sugaya, Y.; Omachi, S. Study on Compression of Images Including Text by Sparse Coding. *IEICE Tech. Rep.* 2016, 116, 5–10.
7. Teahan, W.J. A compression-based toolkit for modelling and processing natural language text. *Information* 2018, 9, 294. [CrossRef]
8. Zamenhof, L. Fundamento de Esperanto. Available online: https://www.akademio-de-esperanto.org/fundamento/ (accessed on 8 April 2022).
9. Rani, M.; Singh, V. A Survey on Lossless Text Data Compression Techniques. *Int. J. Adv. Res. Comput. Eng. Technol.* 2016, 5, 1741–1744.
10. Mentzer, F.; Goel, L.V.; Tschannen, M. Learning better lossless compression using lossy compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 6638–6647.
11. Oswald, C.; Ghosh, A.I.; Sivaselvan, B. An efficient text compression algorithm-data mining perspective. In Proceedings of the International Conference on Mining Intelligence and Knowledge Exploration, Hyderabad, India, 9–11 December 2015; pp. 563–575.
12. Rahman, M.A.; Hamada, M. Lossless Image Compression Techniques: A State-of-the-Art Survey. *Symmetry* 2019, 11, 1274. [CrossRef]
13. Gupta, A.; Bansal, A.; Khaduja, V. Modern lossless compression techniques: Review, comparison and analysis. In Proceedings of the 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India, 22–24 February 2017; pp. 1–8.
14. Popescu, C.; Grama, L.; Rusu, C. A Highly Scalable Method for Extractive Text Summarization Using Convex Optimization. *Symmetry* 2021, 13, 1824. [CrossRef]
15. Jalilian, E.; Hofbauer, H.; Uhl, A. Iris Image Compression Using Deep Convolutional Neural Networks. *Sensors* 2022, 22, 2689. [CrossRef]
16. Hu, W.; Zhu, M.; Zhang, H. Application of Block Sparse Bayesian Learning in Power Quality Steady-State Data Compression. *Energies* 2022, 15, 2479. [CrossRef]
17. Nonaka, K.; Yamanouchi, K.; Tomohiro, I.; Okita, T.; Shimada, K.; Sakamoto, H. A Compression-Based Multiple Subword Segmentation for Neural Machine Translation. *Electronics* 2022, 11, 1014. [CrossRef]
18. Oswald, C.; Sivaselvan, B. An optimal text compression algorithm based on frequent pattern mining. *J. Ambient Intell. Humaniz. Comput.* 2018, 9, 803–822. [CrossRef]
19. Bedruz, R.A.; Quiros, A.R.F. Comparison of Huffman Algorithm and Lempel-Ziv Algorithm for audio, image and text compression. In Proceedings of the 2015 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), Cebu, Philippines, 9–12 December 2015; pp. 1–6.
20. Oswald, C.; Ghosh, A.I.; Sivaselvan, B. Knowledge engineering perspective of text compression. In Proceedings of the 2015 Annual IEEE India Conference (INDICON), New Delhi, India, 17–20 December 2015; pp. 1–6.
21. Blaock, D.; Madden, S.; Guttag, J. Sprintz: Time series compression for the internet of things. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2018, 2, 1–23. [CrossRef]
22. Qiu, H.; Zheng, Q.; Memmi, G.; Lu, J.; Qiu, M.; Thuraisingham, B. Deep residual learning-based enhanced JPEG compression in the Internet of Things. *IEEE Trans. Ind. Inform.* 2020, 17, 2124–2133. [CrossRef]
23. Chowdhury, M.R.; Tripathi, S.; De, S. Adaptive multivariate data compression in smart metering Internet of Things. *IEEE Trans. Ind. Inform.* 2020, 17, 1287–1297. [CrossRef]
24. Sujitha, B.; Parvathay, V.S.; Lydia, E.L.; Rani, P.; Polkowski, Z.; Shankar, K. Optimal deep learning based image compression technique for data transmission on industrial Internet of things applications. *Trans. Emerg. Telecommun. Technol.* 2021, 32, e3976. [CrossRef]
25. Kagita, M.K.; Thilakaratne, N.; Boija, G.R.; Kaosar, M. A lossless compression technique for Huffman-based differential encoding in IoT for smart agriculture. *Int. J. Uncertain. Fuzziness Knowl. Based Syst.* 2021, 29, 317–332. [CrossRef]
26. Campobello, G.; Segreto, A.; Zanafi, S.; Serrano, S. RAKE: A simple and efficient lossless compression algorithm for the internet of things. In Proceedings of the 2017 25th European Signal Processing Conference (EUSIPCO), Kos, Greece, 28 August–2 September 2017; pp. 2581–2585.
27. Hadiatna, F.; Hindersah, H.; Yolanda, D.; Triawan, M.A. Design and implementation of data logger using lossless data compression method for Internet of Things. In Proceedings of the 2016 6th International Conference on System Engineering and Technology (ICSET), Bandung, Indonesia, 3–4 October 2016; pp. 105–108.

28. Pérez, R.; Leithardt, V.R.Q.; Correia, S.D. Lossless compression scheme for efficient gns data transmission on iot devices. In Proceedings of the 2021 International Conference on Electrical, Computer and Energy Technologies (ICECET), Cape Town, South Africa, 9–10 December 2021; pp. 1–6.

29. Gu, J.; Choi, H.; Jeong, J. English Text Compression using Huffman Coding. In Proceedings of the Korean Society of Broadcast Engineers Conference, The Korean Institute of Broadcast and Media Engineers, Seoul, Korea, 4 November 2016; pp. 69–71.

30. Cherkunova, M. V Means of Semantic Compression in Modern English Scientific Discourse (Based on Abstracts to the Articles From International Scientific Citation Databases). Prof. Discourse Commun. 2021, 3, 28–38. [CrossRef]

31. Bekali, O. Semantic-Styletic Tools in English. Kresna Soc. Sci. Humanit. Res. 2022, 8, 34–37.

32. Vijayalakshmi, B.; Sasirekha, N. Lossless text compression for unicode tamil documents. ICTACT J. Soft Comput. 2018, 8, 1635–1640.

33. Gilliver, P. The making of the Oxford English dictionary. Lexikos 2016, 26, 436–445.

34. Indurani, M.P.; Deepika, M.P.; Padma, M.P. A survey on big data compression. In Proceedings of the National Conference on “Future Research Perspectives in Computer Science and Information Technology”, Madurai, India, 21–22 February 2017.

35. Sarker, P.; Rahman, M. Introduction to Adjacent Distance Array with Huffman Principle: A New Encoding and Decoding Technique for Transliteration Based Bengali Text Compression. In Progress in Advanced Computing and Intelligent Engineering; Springer: Berlin/Heidelberg, Germany, 2021; pp. 543–555.

36. Gonzales, A.R.; Spring, N.; Kew, T.; Kostrzewa, M.; Säuberli, A.; Müller, M.; Ebling, S. A New Dataset and Efficient Baselines for Document-level Text Simplification in German. In Proceedings of the Third Workshop on New Frontiers in Summarization, Online, Dominican Republic, 7–11 November 2021; pp. 152–161.

37. Dissemont, J.; Assenheimer, B.; Büttelmann, A.; Gerber, V.; Gretener, S.; Köhler-von Siebenthal, E.; Koller, S.; Kröger, K.; Kurz, P.; Läuchli, S. Compression therapy in patients with venous leg ulcers. JDDG J. Dtsch. Dermatol. Ges. 2016, 14, 1072–1087. [CrossRef] [PubMed]

38. Hilal, T.A.; Hilal, H.A. Arabic text lossless compression by characters encoding. Procedia Comput. Sci. 2019, 155, 618–623. [CrossRef]

39. Awajan, A.; Jrai, E.A. Hybrid Technique for Arabic Text Compression. Glob. J. Comput. Sci. Technol. 2015, 15, 1–7.

40. Xu, R.; Yang, Y. Cross-lingual distillation for text classification. arXiv 2017, arXiv:1705.02073.

41. Ignatowski, M.; Lerga, J.; Stanković, L.; Daković, M. Comparison of entropy and dictionary based text compression in English, German, French, Italian, Czech, Hungarian, Finnish, and Croatian. Mathematics 2020, 8, 1059. [CrossRef]

42. Marasek, K.; Brocki, L.; Kozinek, D.; Wolk, K.; Gubryniewicz, R. Spoken language translation for polish. arXiv 2015, arXiv:1511.07788.

43. Wolk, K.; Marasek, K. Polish-English statistical machine translation of medical texts. In New Research in Multimedia and Internet Systems; Springer: Berlin/Heidelberg, Germany, 2015; pp. 169–179.

44. Grzybowski, P.; Juralewicz, E.; Piasecki, M. Sparse coding in authorship attribution for Polish tweets. In Proceedings of the 2015 International Conference on Recent Advances in Natural Language Processing (RANLP 2015), Varna, Bulgaria, 2–4 September 2015; pp. 409–417.

45. Labužek, M.; Piasecki, M. English Translator ± A Bi directional Polish English Translation System. Available online: https://www.fi.muni.cz/isd2002/papers/108_Marek_Labuzek.ps (accessed on 12 April 2022).

46. Byram, M.; Golubeva, I. Conceptualising intercultural (communicative) competence and intercultural citizenship. In The Routledge Handbook of Language and Intercultural Communication; Routledge: England, UK, 2020; pp. 70–85.

47. Wagner, M.; Byram, M. Intercultural citizenship. Int. Encycl. Intercult. Commun. 2017, 13, 1–6.

48. The Nobel Foundation The Nobel Prize. Available online: https://www.nobelprize.org/ (accessed on 9 April 2022).

49. Universala Esperanto-Asocio Universala Esperanto-Asocio. Available online: https://uea.org/info (accessed on 11 April 2022).

50. PEJ—Polish Esperanto-Asocio. Available online: http://pej.pl/pl/o-esperanto/podstawy-jezyka/ (accessed on 8 April 2022).

51. Israel, N. Esperantic Modernism: Joyce, Universal Language, and Political Gesture. Modernism/Modernity 2017, 24, 1–21. [CrossRef]

52. Martin Camacho, J.C. La morfología de las lenguas artificiales. El caso del “volapuk” y de la “langue bleue”. Anu. Estud. Filol. 2019, 42, 189–213. [CrossRef]

53. Lafortune, P. The Village Voice. Available online: http://www.villagevoice.com (accessed on 12 April 2022).

54. Garv...

55. Cherkunova, M. V Means of Semantic Compression in Modern English Scientific Discourse (Based on Abstracts to the Articles From International Scientific Citation Databases). Prof. Discourse Commun. 2021, 3, 28–38. [CrossRef]

56. Omarov, D.; Tran, K.; Zhexembay, L.; Santana, M.; Hildebrand, A.J. Zipf’s Law: A Universal Law for Empirical Data from Word Frequencies to Olympic Records. Available online: https://faculty.math.illinois.edu/~hildebr/ugresearch/posters/urs2017-zipf-law.pdf (accessed on 12 April 2022).

57. Stecula, B. Budowanie Modelu Kontekstu Świata na Podstawie Tekstu w Języku Esperanto. Master’s Thesis, Silesian University of Technology, Gliwice, Poland, 2020.
58. Gobbo, F. Machine translation as a complex system: The role of Esperanto. *Interdiscip. Descr. Complex Syst. INDECS* 2015, 13, 264–274. [CrossRef]

59. Gobbo, F. Coolification and Language Vitality: The Case of Esperanto. *Languages* 2021, 6, 93. [CrossRef]

60. Hernández-Gómez, C.; Basurto-Flores, R.; Obregón-Quintana, B.; Guzmán-Vargas, L. Evaluating the Irregularity of Natural Languages. *Entropy* 2017, 19, 521. [CrossRef]

61. Nobliści.pl Laureaci Nagrody Nobla. Available online: http://www.noblisci.pl/1905-henryk-sienkiewicz/ (accessed on 8 April 2022).

62. Instytut Książki 115 Lat Temu Henryk Sienkiewicz Odebrał Nagrodę Nobla W Dziedzinie Literatury. Available online: https://instytutksiazki.pl/ (accessed on 8 April 2022).

63. Wólnelektury.pl Wolne Lektury. Available online: www.wolnelektury.pl (accessed on 11 February 2022).

64. Project Gutenberg Gutenberg. Available online: www.gutenberg.org (accessed on 11 February 2022).

65. Tekstaro de Esperanto. Available online: www.tekstaro.com (accessed on 11 February 2022).

66. Severance, C. Guido van rossum: The early years of python. *Computer* 2015, 48, 7–9. [CrossRef]

67. Kumar, C. Python Advantages and Disadvantages—Step in the Right Direction. Available online: https://techvidvan.com/tutorials/%0Apython-advantages-and-disadvantages/ (accessed on 11 February 2022).

68. JetBrains, s.r.o. PyCharm—The Python IDE for Professional Developers. Available online: https://www.jetbrains.com/company/ (accessed on 15 February 2022).

69. Yergeau, F. UTF-8, A Transformation Format of ISO 10646. Available online: https://tools.ietf.org/html/rfc3629 (accessed on 13 February 2022).

70. W3Techs Usage of Character Encodings Broken Down by Ranking. Available online: https://w3techs.com/technologies/cross/character_encoding/ranking (accessed on 13 February 2022).