Generating Representative Phrase Sets for Text Entry Experiments by GA-Based Text Corpora Sampling

Sandi Ljubic 1,2,∗ and Alen Salkanovic 1,2,©

1 University of Rijeka, Faculty of Engineering, Vukovarska 58, HR-51000 Rijeka, Croatia; alen.salkanovic@riteh.hr
2 Center for Artificial Intelligence and Cybersecurity, University of Rijeka, R. Matejcic 2, HR-51000 Rijeka, Croatia
∗ Correspondence: sandi.ljubic@riteh.hr

Abstract: In the field of human–computer interaction (HCI), text entry methods can be evaluated through controlled user experiments or predictive modeling techniques. While the modeling approach requires a language model, the empirical approach necessitates representative text phrases for the experimental stimuli. In this context, finding a phrase set with the best language representativeness belongs to the class of optimization problems in which a solution is sought in a large search space. We propose a genetic algorithm (GA)-based method for extracting a target phrase set from the available text corpus, optimizing its language representativeness. Kullback–Leibler divergence is utilized to evaluate candidates, considering the digram probability distributions of both the source corpus and the target sample. The proposed method is highly customizable, outperforms typical random sampling, and exhibits language independence. The representative phrase sets generated by the proposed solution facilitate a more valid comparison of the results from different text entry studies. In addition, we provide heuristic guidelines for preparing the text entry experiments, which consider the experiment’s intended design and the phrase set to be generated with the proposed solution.

Keywords: text entry; phrase sets; text corpus sampling; genetic algorithm; Kullback–Leibler divergence

1. Introduction

Text entry has been a significant area of research in human–computer interaction (HCI) since its inception, driven by the increasing need for office automation. This trend has continued in modern mobile computing, where text entry occurs on various devices using different interaction techniques. Touchscreen-based technology is now a platform of particular interest, as it enables the development of innovative software-based keyboard solutions. Various keyboard customizations, automatic adjustments, character layouts, and input assistance methods are being introduced to enhance convenience, reduce errors, improve efficiency, and ensure accessibility for all users.

New text entry methods are usually evaluated in controlled user experiments that focus on input speed and error rates as output metrics. In such an experimental approach, text-copy tasks requiring that users transcribe the provided phrases are favored over text-creation tasks, which assume the free input of arbitrary text [1].

While preselected phrases are commonly used as control stimuli in text entry experiments, there is currently no formally standardized phrase set that can serve as a reference. Kristensson and Vertanen [2] argue that choosing an appropriate phrase set may be important not only for the internal validity of the experimental research but also for its reproducibility, the heterogeneity of the study, and the external validity of the results.

Citation: Ljubic, S.; Salkanovic, A. Generating Representative Phrase Sets for Text Entry Experiments by GA-Based Text Corpora Sampling. Mathematics 2023, 11, 2550. https://doi.org/10.3390/math11112550

Academic Editors: Grigoreta-Sofia Cojocar, Adriana-Mihaela Guran and Laura-Silvia Diosan

Received: 20 March 2023
Revised: 23 May 2023
Accepted: 30 May 2023
Published: 1 June 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland.
This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).
Different ad hoc generated phrase sets make systematic review and comparison with the previously published results of text entry studies difficult. External validity requires conclusions that can be generalized to contexts outside the laboratory. This is difficult to achieve if the phrase sets do not adequately represent the domain or language in question.

There have been some notable attempts to create commonly accepted phrase sets. While we mention the major efforts in Section 1.1, this paper focuses on creating a representative phrase set from a single, presumably large, and pre-existing text corpus.

Predictive modeling is another way to evaluate text entry methods. It is an alternative to the empirical approach, which does not necessitate explicit testing with real users. Soukoreff and MacKenzie [3] presented a quantitative prediction technique that combines both a motion and a linguistic model:

$$WPM_{\text{max}} = \frac{1}{\sum_{i \in C} \sum_{j \in C} (P_{ij} \cdot MT_{ij})} \cdot \frac{60}{5}$$  \hspace{1cm} (1)$$

While the movement model aims to predict the time $MT_{ij}$ required to enter a character $j$ preceded by a previously entered character $i$, the linguistic model uses digram frequencies in a given language (with the character set $C$), resulting in computed occurrence probabilities $P_{ij}$ for each digram. These models are combined to develop a prediction for the average number of characters entered per second, and thus for the theoretical upper limit of text entry speed $WPM_{\text{max}}$ ($WPM$ here stands for words per minute). This approach generally uses a variation of the Fitts’s law [4] to predict the movement time between two consecutive keys, leading to the following formalization:

$$MT_{ij} = a + b \cdot \log_2 \left( \frac{A_{ij}}{W_j} + 1 \right)$$  \hspace{1cm} (2)$$

Equation (2) shows that the predicted time $MT_{ij}$ depends on $A_{ij}$—the distance between the center of key $i$ and the center of key $j$; and $W_j$—the width of the target key. The coefficients $a$ and $b$ are constants obtained by linear regression.

The predictive modeling of text entry efficiency based on Fitts’s law and the digram probability distribution has been successfully applied in several domains, including the development of various keyboards and input devices [3,5,6] and in target languages other than English [7,8]. This approach proved particularly important for modeling text entry methods that rely on a single pointing device.

The practical utility of Fitts’s law in modeling text input is why we focus on the digram-based statistical properties of input text. Namely, predictive modeling results are often compared to the empirically obtained outcomes in text entry research. In this context, it is helpful to use the digram distribution to develop representative phrase sets for experiments since, as shown above, the same digram statistics are also used in the predictive modeling process. Therefore, our main goal is to develop an efficient method for extracting phrase sets from existing large corpora with the following subtasks:

- When searching for the “most representative” phrase set by sampling from a text corpus of a given language, representativeness should be considered as a function of the digram probability distribution;
- The proposed sampling procedure should outperform existing methods;
- A proof-of-concept should be demonstrated for several different languages.

1.1. Related Work

This subsection describes publicly available phrase sets that have already been considered a resource in text entry research. The main features are described for each phrase set, and the development process in those cases where the design procedure is known.
MacKenzie and Soukoreff [9] published a phrase set of 500 English sentences described as “moderate in length, easy to remember, and representative of the target language”. As a metric for representativeness, the authors used the correlation coefficient between the distribution of individual letters in their phrase set and the letter frequencies previously derived by Mayzner and Tresselt [10]. Thus, the representativeness of this phrase set is derived from an outdated source of 20,000 words, which is an undersized sample compared to contemporary text corpora. Nonetheless, the corresponding phrase set has been used by many researchers, even in the mobile text entry domain. However, the phrases provided are unlikely to match well with text written in actual mobile messaging.

The AAC Research Group at the University of Nebraska [11] published several vocabulary lists and context-specific messages. The latter represents a collection of short conversational phrases suggested by AAC specialists that can be used as stimuli in text entry experiments. The phrases are categorized according to context, e.g., conversation control phrases, communication repairs/corrections, location markers, temporal markers, and social comments. It seems reasonable to use this set when a text entry experiment replicates a particular context of a written conversation. However, the representativeness of the language was not considered in the development of this phrase set.

Paek and Hsu [12] presented a method for generating phrase sets by randomly selecting n-grams, i.e., phrases with $n$ words, from a large corpus, choosing the set with the digram probability distribution closest to the digram probability distribution of the source corpus. Hence, they proposed “a more mathematically principled method” based on the notion of representativeness on information theory. To compare the digram probability distributions between the source corpus and the target phrase set, they used relative entropy—also known as Kullback–Leibler divergence (KLD):

$$D(p||q) = \sum_{x \in C \times C} \left( p(x) \cdot \log_{2} \frac{p(x)}{q(x)} \right)$$  \hspace{1cm} (3)

In Equation (3), $p(x)$ gives the probability distribution of character digrams for a sample phrase set, and $q(x)$ is the probability distribution of character digrams for the source corpus. At the same time, $C$ represents the set of characters in a given language. Paek and Hsu, in interpreting relative entropy, state that it is not a true “distance metric” since it is neither symmetric nor satisfies the triangle inequality [13]. However, since it is always non-negative and only equals zero when $p(x)$ and $q(x)$ are identical, one can argue that the phrase set is more representative of the source corpus when the relative entropy is closer to zero. The single sampling trial from the authors’ proposed procedure is shown in Figure 1.

![Figure 1. A phrase set sampling method by Paek and Hsu [12]. The sample is randomly generated from the source corpus, and its representativeness is calculated using the relative entropy based on the respective digram distributions.](image)

Paek and Hsu used a predetermined number of random samples to find the most representative phrase set. They targeted four-word phrases only, regardless of whether they formed a meaningful sentence. In addition, phrase duplicates were allowed to remain...
in the phrase set. The phrase set generated from the Enron Email corpus [14] was made publicly available.

Vertanen and Kristensson [15] focused on mobile text entry and the memorability of the provided phrases. Their primary text resource was the Enron Email corpus, but only messages written on mobile devices were used for further analysis. The authors enriched each phrase in this dataset with memorability, expected input speed, and error rate information. This information was obtained empirically by conducting crowdsourcing experiments in which actual users performed text-memorization and text-copy tasks. The authors published five sets of 40 phrases for evaluations where memorable text was required. In addition, they created four phrase sets that are recommended when representativeness is desired instead. In their search for representative phrase sets, they used the procedure described by Paek and Hsu [12]. However, they modified it slightly to discard n-grams in the middle of the sentence, thus only obtaining intelligible phrases. They restricted their phrase sets to sentences with 3–9 words.

In their continued effort to provide phrase sets for mobile text entry, Kristensson and Vertanen [2] analyzed the NUS SMS corpus [16]—a publicly available collection of actual SMS messages. Since the source corpus contained noise due to many meaningless messages with poor grammar, it was first appropriately filtered. The authors demonstrated that selecting a particular phrase set as a stimulus for text entry experiments makes a difference. Namely, the SMS-based phrase set proved to be significantly more error-prone due to the “strange language, abbreviations, and sentence fragments”. On the other hand, using the AAC phrase set resulted in significantly higher input rates, likely due to “simple, short, and familiar phrases that avoided proper names, unusual vocabulary, and difficult grammar”.

Leiva and Sanchis-Trilles [17] focused on generating sets of memorable phrases while trying to preserve the language representativeness. In doing so, they decided to model the character error rate (CER) as a function of several language-independent features to predict memorability. Statistical analysis was performed on the corpus from which the phrases were taken to ensure representativeness. The final score is assigned to each candidate phrase, favoring low CER (i.e., high memorability) and high representativeness. The proposed approach requires two different corpora: one large enough to describe the target language and one from which the phrases are selected. This method was successfully applied in [18], wherein a collection of 30 datasets for ten major languages was obtained.

Yi et al. [19] proposed a phrase set sampling procedure that emphasizes a word clarity metric based on probability theory, which indicates how likely the word is to be confused with other words. The proposed method additionally considers digram frequency and memorability. However, the proof-of-concept phrase sets were not developed from a large-text corpus but from a phrase set developed by MacKenzie and Soukoreff.

Gaines and Vertanen [20] developed a 5000-phrase set from comments on the Reddit web forum. Each phrase in this set is assigned a difficulty rating from 1 to 10. The difficulty rating is based on the character error rate, which was determined by simulations of text input. In their research, however, the authors did not consider the representativeness of a target phrase set.

Abbot et al. [21] utilized text-to-speech systems to synthesize the audio clips of all phrases from the set of MacKenzie and Soukoreff. They provided a set of 92 phrases that could be transcribed without comprehension errors. Although representativeness was not considered in this study, it is still interesting because it shows that a phrase set does not have to be strictly in the form of the text as an experimental stimulus in a text entry experiment.

In general, related work suggests that three different sampling approaches can be automated with the goal of producing phrase sets that are comparable across languages and domains:

- The random selection of phrases from available text corpora where representativeness is not explicitly targeted.
- Random selection of phrases, where phrase sets with better representativeness are selected [12].
Specific methods that consider the representativeness of a phrase set but focus on other metrics such as memorability [17] or word clarity [19] and analyze the aggregated metrics accordingly.

Table 1 summarizes the features of the phrase sets described in the related work.

<table>
<thead>
<tr>
<th>Phrase Set</th>
<th>Source Corpus</th>
<th>Development Procedure</th>
<th>Representativeness</th>
<th>Main Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAC</td>
<td>N/A</td>
<td>Phrases suggested by AAC specialists; context-specific</td>
<td>N/A</td>
<td>928 phrases 3771 words 17,221 characters</td>
</tr>
<tr>
<td>Paek and Hsu [12]</td>
<td>Enron Email</td>
<td>Random sampling of n-word phrases; searching for the representative sample</td>
<td>Relative entropy between the target phrase set and the source corpus (KLD)</td>
<td>500 phrases 2000 words 11,275 characters</td>
</tr>
<tr>
<td>Vertanen and Kristensson [15]</td>
<td>Enron Email (filtered)</td>
<td>Metadata tagging; filtering; random sampling for representative datasets</td>
<td>Relative entropy between the target phrase set and the source corpus (KLD)</td>
<td>600 phrases 3730 words 19,211 characters</td>
</tr>
<tr>
<td>Kristensson and Vertanen [2]</td>
<td>NUS SMS (filtered)</td>
<td>Filtering based on phrase length and existing dictionary lists</td>
<td>N/A</td>
<td>769 phrases 5442 words 25,261 characters</td>
</tr>
<tr>
<td>Sanchis-Trilles and Leiva [18]</td>
<td>OpenSubtitles 2011, Wikipedia</td>
<td>Modeling over language-independent features; statistical analysis of the corpus from which phrases are selected</td>
<td>Weighting phrases with a probability density function over several phrase features</td>
<td>Language-specific; see [18] for details</td>
</tr>
<tr>
<td>Yi et al. [19]</td>
<td>MacKenzie and Soukoreff</td>
<td>Combined analysis of word clarity, digram frequency, and memorability</td>
<td>Relative entropy between the target phrase set and the source corpus (KLD)</td>
<td>Four sets with 20, 40, 80, and 160 phrases; see [19] for details</td>
</tr>
<tr>
<td>Abbot et al. [21]</td>
<td>MacKenzie and Soukoreff</td>
<td>Synthesizing audio clips; targeting phrases with no comprehension errors</td>
<td>N/A</td>
<td>96 audio clips</td>
</tr>
</tbody>
</table>

### 1.2. Contributions and Structure

In the context of research on representative phrase sets, we want to develop a new method that: (1) targets phrase set representativeness; (2) outperforms typical random sampling; (3) can work with multiple languages; and (4) requires a single-large-text corpus. We consider KLD a metric for representativeness because digram probability is essential for curating phrase sets and the predictive modeling of text entry (see Equation (1)). Our method is directly motivated by the work described in [12]; however, we attempt to provide a more efficient method of corpus sampling and more representative phrase sets. Although we are not targeting the memorability feature, we want to design the proposed method to
use many different parameters, including the desired thresholds for the number of words in a phrase. Since the most memorable phrases are those with the fewest words, memorability could be targeted in this, albeit trivial, way.

To the best of our knowledge, no method has been proposed that uses (heuristic) optimization techniques (or the like) to exclusively target representativeness metrics, intending to find such a small phrase set from a single-large-text corpus that matches both the digram statistics of the original corpus and the intended design of the text-entry experiment. Therefore, we aim to provide this method not only as a proof-of-concept but, more importantly, as an out-of-the-box implementation that all text entry experimenters can use immediately and adapt as needed. In addition, we want to propose heuristic guidelines for preparing and running a text entry experiment, which would include generating a phrase set to be utilized in its entirety (which is often not the case).

In line with the above, the main contributions of this paper are as follows:

• A novel method for sampling text corpora using the GA approach aiming to achieve near-optimal representativeness of the phrase set;
• An open source implementation of the proposed method that can be readily used regardless of the target language, available corpora, target character set, number of words in the phrases, size of the phrase set, and choice of GA parameters;
• A set of heuristic guidelines for preparing text entry experiments that consider the experiment’s intended design and the set of phrases to be generated by the proposed solution.

This paper is structured as follows. Section 2 describes our proposal for generating representative phrase sets from available text corpora, focusing on mapping the domain problem into a GA implementation. The results of the initial comparison with the random sampling method, GA tuning, and application of the solution for different languages are presented in Section 3. The discussion of the obtained results is presented in Section 4, and Section 5 brings the final conclusions.

2. Materials and Methods

Finding the most representative phrase set from the large source corpus is a search problem with a wide range of possible solutions. To better understand the size of such a typical search space, we can consider a hypothetical text corpus with 50,000 phrases, a moderate volume from today’s point of view. We can further assume that the desired number of phrases in the target phrase set is 200. If we want to evaluate the representativeness of all possible phrase sets that can be derived by corpus sampling (with no duplicates allowed), the number of available combinations in our case increases to \(C(50K, 200) = 5.297e + 564\).

Of course, an exhaustive search would be both time-consuming and resource intensive in terms of computational power. As mentioned earlier, Paek and Hsu [12] used a predetermined number of samples in their method, i.e., they searched for the best solution out of 100 samples. They argued that the number of sample phrase sets researchers want to analyze depends on how much time they can spend collecting and evaluating candidates. Nevertheless, such an approach leaves much of the search space unexplored.

Proposing an alternative sampling method requires a benchmark to evaluate its efficiency. Therefore, we implemented our version of Paek and Hsu’s method and named it “brute force” sampling (BF). The BF method is shown in Figure 2.

First, for a given source corpus (SC), the probability distribution \(q(ij)\) of character digrams is computed. Here, \(ij\) represents a general term of the character digram, where both \(i\) and \(j\) must be elements of a previously defined target character set. For example, if only a lowercase English text without punctuation and non-alphabetic symbols other than the space character is considered, the total number of digrams is \(27 \times 27 = 729\). After computing \(q(ij)\), the source corpus is filtered to extract only n-grams, and thus condensed to the reduced corpus (RC). Unlike the original procedure of Paek and Hsu, we selected candidates from the RC that contain formed sentences with exactly \(n\) words (where \(n\) is a user-defined value). The sampling procedure of RC itself is quite simple and consists
of a predefined number of trials (TRIALS) where the best of the observed solutions is retained. The single sampling trial involves: (i) generating the phrase set candidate (PSC) by randomly selecting it from RC; (ii) computing the digram probability distribution \( p(ij) \) for the generated PSC; (iii) computing the Kullback–Leibler divergence (KLD); and (iv) declaring the current PSC as the target phrase set (TPS) if the associated KLD has the lowest value so far. Since the described procedure leads to finding the best-of-TRIALS phrase set, whose digram distribution is closest to the digram distribution over the entire SC, we declare the final TPS as the “most representative” phrase set determined by BF sampling.

![Figure 2. Procedure for finding representative phrase sets using the source corpus BF sampling.](image)

Our case is obviously a typical mathematical optimization problem in which a suboptimal solution is sought in a large search space. To explore this search space of possible phrase sets more efficiently than BF, we propose a meta-heuristic approach using a genetic algorithm (GA). The basic idea of the corresponding method is shown in Figure 3.

As can be seen, the initial operations remain the same as in the BF sampling approach: based on the input target character set, the digram probability distribution \( q(ij) \) is computed for the SC. At the same time, the n-gram reduction is performed to obtain the RC dataset. The main difference is that the genetic algorithm handles the process of actual sampling and sample evaluation. As a heuristic routinely used to generate solutions for optimization and search problems by mimicking the process of natural selection, a genetic algorithm seems a logical choice for our case. For now, we can consider GA as a black box that can generate a target phrase set that is likely to be more representative compared to the BF approach. However, this needs to be supported by actual data, so the GA parameters such
as population, chromosome length, fitness function, termination criteria, and operators such as mutation, crossover, and elitism need to be appropriately applied to the problem domain.

![Genetic Algorithm Diagram](image)

**Figure 3.** Procedure for finding representative phrase sets using the source corpus GA sampling.

### 2.1. Mapping the GA to the Problem Domain

Although a huge amount of literature deals with the theoretical foundations of genetic algorithms, only the basic principles and notations are briefly explained here. The implementation of a GA starts with a population of candidate solutions, usually randomly selected and declared as chromosomes. The population then enters an evolutionary process, representing one generation in a single iteration. Within a given generation, each chromosome is evaluated against a defined fitness function, usually corresponding to the main criteria for solving the problem. Chromosomes that represent a better solution to the target problem are then given more chances to “reproduce”, i.e., they are subjected to crossover and random mutation after the intermediate selection process. As such, a new generation of candidate solutions is formed for the next iteration. Various termination criteria can be used to terminate the genetic algorithm, such as the specified maximum number of generations, the desired level of fitness, or the number of generations that do not increase fitness. More information on the computational behavior of GAs and how they support complex search problems can be found in [22].

Making an initial random population within our problem domain is illustrated in Figure 4.

Individual phrases from the reduced corpus represent genes in terms of the genetic algorithm. The phrases are randomly selected and grouped to form a single phrase set candidate (PSC) corresponding to a single chromosome. The number of phrases in the PSC (chromosome length) is a user-defined value. Our implementation does not allow for phrase duplicates to be present in a given chromosome. The number of PSCs we want to analyze iteratively represents a population size, which is also an arbitrary parameter.
Once the initial population of PSCs is created, their evolution towards better solutions can begin.

Each PSC from the population is evaluated using the fitness function, which in our case, is represented by the Kullback–Leibler divergence. The relative entropy between a given PSC and the SC can indicate how representative the phrase set in question is, i.e., how “good” the corresponding chromosome is. Concerning the KLD, chromosomes whose associated value is closer to zero are considered fitter.

Figure 4. Population, chromosomes, and genes representation in the text corpus sampling problem domain.

One of the basic principles of GA assumes that the best candidates should be retained and bred to produce even better solutions for the next generation. Choosing chromosomes for a potential crossover operation represents a selection process, and several methods provide this functionality in GA. Simple truncation selection, for example, eliminates a fixed percentage of the weakest candidates and is considered less sophisticated because weaker solutions may not survive at all. Weaker solutions can sometimes be an advantage in the crossover process since there is always some chance of obtaining genes that might prove useful. Therefore, we opted for fitness selection, also known as roulette wheel selection, in which the probability of an individual being selected for a crossover increases with the fitness of the individual in question. We used the two-point technique for the crossover method, in which two parent chromosomes are combined to generate two new child chromosomes (see Figure 5).

Figure 5. Two-point crossover swaps the text phrases between the parent chromosomes, based on the positions of two points, resulting in the generation of two new PSCs.

The resulting offspring replace the parent chromosomes in the current population, thus performing a generational change. The main advantage of such an approach is its ease of implementation; however, the obvious disadvantage is that highly fit individuals are not guaranteed to pass unchanged into the next generation. For this reason, our implementation includes an additional elitism operator. Elitism allows a certain percentage of the best candidates (elitism percentage) to be included in the next generation without being modified beforehand, ensuring that future generations do not lack quality.
Finally, we had to account for genetic diversity by including the mutation operator. Unlike the binary crossover operator, the unary mutation operator acts on a single chromosome by changing one or more gene values. While crossover directly supports convergence to a local optimum, the mutation operator is, in turn, used to cause divergence by adding new genetic information. Mutation occurs with a user-defined mutation probability, usually set relatively low. However, setting the best value for the mutation rate is highly problem-specific, as this value defines the desired trade-off between the exploration and exploitation capabilities of the GA.

A custom mutation operator is proposed and implemented to support specific requirements within the problem domain. Along with the mutation probability, we introduced an additional parameter—a number of phrases to be altered in a mutating PSC. In our case, the mutation probability represents the chance that a given PSC will undergo the mutation process. When such a mutation is eventually triggered, a predefined number of phrase swaps are made at random positions within the PSC. The corresponding mutation process is shown in Figure 6.

**Figure 6.** The mutation operator used in our implementation.

Each single mutation session consists of three basic operations: (i) based on the predefined number of gene modifications, phrases to be randomly replaced within the respective chromosome are determined; (ii) the same number of new phrases are randomly extracted from the subset of RC; and (iii) the determined current phrases are swapped with the new ones. In Figure 6, the number of mutations is assumed to be 3. Phrases $P_2$, $P_7$, and $P_{n-3}$ are swept and become part of the RC subset in the next generation if they are not already contained in one of the remaining chromosomes.

The design of our mutation operator, in which several phrases (genes) in a set (chromosome) can be changed simultaneously, retains all the essential features of a typical mutation operator. This facilitates the exploration by including diversity even in cases where a representative set with a large number of phrases is required. When introducing multiple mutations, avoiding excessive population disruption is important. Care must be taken to ensure that the number of changed phrases in a set is reasonable and does not lead to chaotic or unpredictable behavior. In addition, our mutation operator ensures that diversity is maintained in the population, as it only replaces existing phrases with those not yet present in the entire population. Even though our operator allows a kind of “fast” mutation within a generation, combinations of phrases that have previously shown a good fit are protected from mutation by the elitism operator. Thus, the additional parameterization of the mutation operator allows further adaptation to our domain problem and the properties of the target phrase set. After all, the proposed GA method can be used to set the number of target genes for mutation in the described operator to 1, effectively turning our mutation operator into the default one.

Figure 7 shows the structure of the GA process pipeline used in our implementation.
2.2. Implementation of GA-Based Text Corpora Sampling

The proposed procedure for sampling a text corpus using a genetic algorithm is implemented in C# within the .NET platform. To support the functionality of GA, we used the freely available genetic algorithm framework (GAF)—the single multi-thread assembly that provides programming resources for many types of genetic algorithms [23]. While the most common genetic operators are built into GAF, external operators can also be developed and added to the standard GA process pipeline. No problems were encountered in mapping the properties of the problem domain to the appropriate programming structures, and implementing our custom mutation operator was quite straightforward.

As a final result of our project development phase, we provided several C# classes with a number of useful methods for dealing with corpora sampling. Some of the functionalities provided by our implementation, which includes both BF and GA approaches, are as follows:

- Loading the corpus from a text file;
- Reducing the available source corpus to the dataset containing only n-word phrases;
- Computing the letter/digram probability distribution for the provided text corpus based on the defined target character set;
- Computing the relative entropy (KLD) between two digram probability distributions;
- Parameterized BF sampling;
- Setting GA by defining the basic parameters, the processing pipeline, and the input datasets (RC and the character set for a target language);
- Specific GA event handling (managing the completion of a single generation or a GA run).

We provide simple C# code snippets that perform all the necessary analysis and computation to demonstrate how easily the provided implementation can be used to obtain the phrase sets. As can be seen in Figure 8, corpus sampling can be invoked with just a few lines of code, regardless of whether the BF or GA approach is preferred. Furthermore, the corresponding methods are parameterized to allow the easy customization of the sampling process. For example, defining completely new GA parameters at one point is possible. This forms the basis for a simple GA parameter tuning in the later phase of the analysis of the results.

Other useful features of our implementation can be highlighted:

- The n-gram reduction from the SC is performed on sentence-based premises so that only meaningful phrases can be included in the further analysis;
- The RC does not contain duplicate phrases; consequently, each phrase set candidate, as well as the target phrase set, is duplicate-free;
- The probability distributions do not have to be recalculated, but can be loaded from the respective files with the previously computed digram/letter frequencies (this is especially useful while working with large text corpora);
• The methods are designed in such a way that large text corpora can be analyzed independently of the available hardware resources: file processing is implemented using byte streams so that large text files do not have to be entirely loaded into memory;
• The whole sampling procedure depends on a defined target character set (which can be easily changed), while the algorithm itself is not language-specific since corpora can be sampled by analyzing the whole or restricted alphabet of a given language, with or without punctuation and additional characters;
• The sampling steps are recorded in the corresponding log file, including all relevant parameters and intermediate results, so that the performance of both BF and GA can be checked in detail.

BF sampling

```csharp
CorpusTextSampler.BruteForceCorpusSampling_FromScratch()
{
    "SC.txt", // source corpus file (SC)
    100, // number of random sampling trials
    500, // number of phrases in the target phrase set
    new[4][4], // character set for a given language
    "RC.txt", // source corpus limited to n-grams only (RC)
    "TPS.txt", // output: target phrase set (TPS)
    "SC-letters.csv", // output: letter probability distribution for SC
    "TPS-letters.csv", // output: letter probability distribution for TPS
    "SC-digrams.csv", // output: digram probability distribution for SC
    "TPS-digrams.csv", // output: digram probability distribution for TPS
    "log.csv" // output: log file for the analysis of the BF sampling
}
```

GA sampling

```csharp
Distribution corpusDist = new Distribution("SC.txt", "charset.txt", true, new[4][4], "RC.txt");
HashSet<string> reducedCorpus = CorpusTextSampler.loadReducedDatasetFromFile("RC.txt");

GA myGA = new GA(
    "charset.txt", // character set for a given language
    corpusDist, // digram probability distribution for SC
    500, // number of phrases in the target phrase set
    reducedCorpus, // source corpus limited to n-grams only (RC) - as Hashset
    100, // GA population size
    true, 5, // elitism operator (true/false), elitism percentage
    0.8, // crossover probability
    0.1, 10, // mutation probability, number of genes to be mutated
    10000, // maximum number of generations
    "PPS.txt", // output: target phrase set (TPS)
    "log.csv" // output: log file for the analysis of the GA sampling
);
myGA.Run();
```

Figure 8. Starting BF and GA sampling with the provided C# methods. In both cases, only two resources are required as input arguments: the source corpus (a file that presumably contains a large amount of text) and the target character set for a given language.

The implementation's source code is freely available as an open source project on GitHub [24]. Researchers can use this resource and experiment by “feeding” the provided methods with different text corpora and different sampling features to obtain representative phrase sets with the desired properties (language, character set, phrase set size, and phrase length). The result of the GA-based sampling procedure can be examined in more detail using different combinations of GA parameters and compared with the benchmark BF approach. In addition, the provided source code can be modified and/or adapted to support some specific domain problems and tackle performance issues. For example, a different way of handling unwanted characters in the source corpus can be considered, alternative BF sampling approaches can be used, a new version of the GA mutation operator can be developed, and so on. A completely different fitness function can even be introduced to analyze possible new metrics for phrase set representativeness.
3. Results

This section gives an overview of all the results obtained by applying the proposed solution. In doing so, we describe the initial tests and compare the effects of BF and GA on the available Croatian text corpus. The Croatian language was chosen for the reason that it is the native language of the authors of this work, and there are no representative phrase sets that could be used for the text entry experiments in this language. Then, we show the procedure and the results of tuning the GA, and finally, we present the results of applying the tuned algorithm to text corpora from several different languages. All results were obtained on the following computer PC configuration: CPU Intel i7-5600U @3.20 GHz, 12 GB RAM, 512 GB SSD.

3.1. Initial Assessment of BF and GA Sampling Effects

The proposed procedure was first tested for sampling the Croatian text corpus. As the main source corpus, we chose fHrWaC [25], a filtered version of the Croatian web corpus hrWaC, initially compiled by Ljubešić and Erjavec [26]. hrWaC is a web corpus collected from the .hr top-level domain, containing 1.2 billion tokens in its first version and distributed under the CC-BY-SA license. Since this resource was found to contain segments of non-textual content, such as code snippets and formatting structures, as well as some encoding errors and foreign language content, we decided to use the filtered version instead. The final filtered corpus contains 50,940,598 sentences, is licensed under CC-BY-SA 3.0, and can be downloaded from a website maintained by the Text Analysis and Knowledge Engineering Lab (TakeLab) [27] at the University of Zagreb.

A sampling of the fHrWaC corpus was initially conducted using the BF approach. It was decided that an analysis would be performed based on the character set consisting of the lowercase alphabet typically used in Croatian QWERTY layouts. This includes all 26 English characters as well as five additional Croatian diacritical marks: ˇ c, ´ c, ¯d, š, and ž. Except for the space character, no other punctuation characters were considered, so our digram matrix includes exactly $26 \times 26 = 1024$ records. As the first step of the sampling procedure, we derived the digram probability distribution for the source corpus. Table 2 shows the statistics for the ten most frequent digrams in the fHrWaC corpus.

Table 2. Ten most frequent digrams in the fHrWaC corpus (based on the used character set).

<table>
<thead>
<tr>
<th>Character Digram</th>
<th>Number of Occurrences</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-space</td>
<td>215,614,936</td>
<td>0.032759789</td>
</tr>
<tr>
<td>e-space</td>
<td>190,822,444</td>
<td>0.028992904</td>
</tr>
<tr>
<td>i-space</td>
<td>155,658,921</td>
<td>0.023650279</td>
</tr>
<tr>
<td>je</td>
<td>152,481,601</td>
<td>0.023167528</td>
</tr>
<tr>
<td>space-s</td>
<td>115,031,757</td>
<td>0.017477528</td>
</tr>
<tr>
<td>o-space</td>
<td>110,503,789</td>
<td>0.016789564</td>
</tr>
<tr>
<td>u-space</td>
<td>107,987,401</td>
<td>0.016407233</td>
</tr>
<tr>
<td>space-p</td>
<td>105,068,195</td>
<td>0.015963699</td>
</tr>
<tr>
<td>na</td>
<td>87,110,973</td>
<td>0.013235341</td>
</tr>
<tr>
<td>space-n</td>
<td>80,275,325</td>
<td>0.012196756</td>
</tr>
</tbody>
</table>

Based on the calculated fHrWaC digram probability distribution $q(ij)$, it can be seen that digrams containing the blank are generally more frequently used, which is not surprising if we know that they represent the beginning and/or the end of words. As mentioned earlier, the sampling procedure should now search for a phrase set with its digram probability distribution $p(ij)$ being the most representative of the obtained $q(ij)$ with respect to the KLD.

We decided to retrieve the target phrase sets of exactly 200 phrases and focus on phrases of 3–15 words. Naturally, shorter phrases are the preferred stimuli in text entry experiments where the text to be typed must be memorized beforehand. In comparison, shorter and longer phrases can be used simultaneously in typical experimental text typing.
tasks. A total of 1000 trials were conducted, resulting in 13 different phrase sets whose representativeness values are shown in Figure 9. It can be seen that both the mean relative entropy and its minimum value decrease with increasing phrase length. Thus, we conclude that better representativeness is more difficult to achieve when the target set contains shorter phrases. Since most text entry experiments use short phrases, finding representative solutions becomes even more challenging.

Figure 9. Sampling 200 phrase sets from an fHrWaC reduced to \( n \) words using the BF approach with 1000 random trials. In addition to the KLD mean value, the corresponding standard deviation is also shown.

We set the benchmark phrase length to five words for further analysis. According to the initial results from BF, the minimum value of relative entropy obtained for a phrase set containing 200 phrases of five words is 0.079179. Next, the effect of the phrase set size was assessed by an additional round of sampling with six different sizes. Again, 1000 random trials were used in each BF sampling run. The corresponding results are shown in Figure 10.

Figure 10. Sampling phrase sets with a variable size from a 5-word reduced fHrWaC using the BF approach with 1000 random trials. In addition to the KLD mean value, the corresponding standard deviation is also shown.
Again, the results show that relative entropy (both average and minimum values) generally decreases as the amount of text in the target phrase set increases. This confirms analogous results from Paek and Hsu [12], where the corresponding trend was attributed to smaller quantization effects for larger phrase sets. Thus, it has already been argued that “it is much more important to make sure that a sample phrase set is representative of the source corpus when there is a small number of phrases”. Therefore, we decided to stick with 200-phrase sets for further analysis.

Finally, we wanted to analyze how the number of random trials in the BF procedure affects the best value of relative entropy. As mentioned earlier, this number significantly depends on how much time can be spent on BF sampling concerning the search space under investigation. We sampled the RC eight times, from a run with only ten random trials to the more time-consuming processes, the last of which involved half a million samples. The results obtained are shown in Figure 11.

Figure 11. Sampling 200-sentence phrase sets from the 5-word reduced fHrWaC using the BF approach with a different number of random trials. In addition to the KLD mean value, the corresponding standard deviation is also shown.

No major differences were observed between the mean values of relative entropy (ranging from 0.105109 to 0.107923) and their respective standard deviations. However, the KLD minimum values associated with the most representative phrase sets found in each BF sampling procedure generally showed a negative trend. This is an understandable effect since a more extended search means a higher probability of finding a more appropriate phrase set. We can now be interested in the absolute gain in terms of relative entropy values: a BF run with 500,000 random trials yielded the best sample with $\min(KLD) = 0.075669$. In contrast, the second best result was observed in a procedure with 1000 random trials with $\min(KLD) = 0.079320$. The best value obtained serves as the final BF benchmark to be compared with the proposed GA sampling procedure.

After analyzing the effects of BF sampling, the next step was to test the support of GA. In line with the previous analysis, the proposed GA sampling procedure was used to obtain a representative set of 200 phrases with exactly five words. The reduced corpus was already available through the BF sampling procedure. The initial parameters of GA were set as follows: a population of 50 phrase sets (chromosomes), elitism of 2%, crossover probability of 0.65, mutation probability of 0.1, five new genes involved in each mutation process, and a maximum number of generations (500) as the criterion for GA termination. This preliminary GA run produced the results shown in Figure 12.

Within the initial population, the most representative phrase set of 50 randomly selected candidates had the corresponding KLD value of 0.086755. Further crossovers and mutations in the GA process pipeline resulted in the KLD value being more than halved after only 500 generations. Thus, even the first run of GA yielded a target phrase set
which was more representative of the source corpus \((KLD = 0.032082)\) than any phrase set previously obtained using the BF approach. This was somewhat surprising since the BF procedure had already involved a large number of trial runs without being able to derive a phrase set with a value even close to the KLD value obtained in the first run of GA. Thus, the most representative phrase set yet was easily obtained, but the next task was to experiment with the parameters of GA to search for possibly better solutions.

Figure 12. Sampling 200-sentence phrase sets from the 5-word reduced fHrWaC using the GA approach with initially set parameters.

3.2. GA Parameter Tuning

Determining the “right” parameters for the GA to solve a particular search problem is anything but a trivial task. Generally, this task can be divided into parameter tuning and parameter control. As explained in Eiben and Smit [28], parameter control assumes that parameter values change during a GA run, so initial parameter values must be fixed and appropriate tuning strategies must be used. On the other hand, parameter tuning is less complicated because no changes occur during a GA run, and only a single value per parameter is required. However, since there are a large number of options that can be manipulated, parameter tuning remains a serious issue. Considering that there are several possible approaches for tuning the GA parameters, a possible analysis of the respective tuning algorithms is beyond the scope of this paper. The existing tuning algorithms and their impact on GA-based corpus sampling, both in terms of performance improvements and computational costs, are viable options for future studies. The parameterized features of the provided implementation (including GAF support and our sampling methods) allow such an analysis without modifying the genetic algorithm.

Consequently, a more conservative approach to parameter tuning was adopted in our work. Values are selected by convention (general principles or rules of thumb such as “lower mutation rate”), ad hoc decisions (e.g., two-point crossover), and, most importantly, experimental comparisons with a limited number of parameter combinations. When experimenting with different parameter values, particular emphasis was placed on algorithm performance, convergence to the best solution, and final KLD values after a certain number of GA generations. Some parameters are directly related to the observed results, e.g., a larger population size and/or a higher number of generations increases the time required to complete GA. On the other hand, it is not immediately obvious how a particular combination of parameters, here including the probabilities of crossover and mutation, affects the outcomes of GA. Approximately 60 combinations were tested, resulting in a similar decreasing trend of the KLD value.

All tuning sessions showed a faster decline in KLD values in the first generations and a slower convergence to the best solution in the later stages of the GA run. Regarding the execution speed, a single GA run with 1000 generations and a population size in
the range of 500–1500 was completed in less than an hour with the aforementioned PC configuration. Since we considered this time a reasonable cost for the investment in parameter tuning, and due to the similarity of the obtained KLD trends, it was decided that the final GA parameters should be selected according to the best obtained KLD value across all tuning runs with 1000 generations. As a result, the parameter values were set as follows: population size—500; elitism—5%; crossover probability—0.8; mutation probability—0.2; and number of genes involved in the mutation process—20.

When testing different values for the custom mutation operator, we observed a typical effect: for better results, the combination of a very high mutation rate and a large number of variable genes should be avoided, as well as a combination of a very low mutation rate and a small number of variable genes. Several aspects can explain the positive effects of a somewhat higher mutation rate in our case. First, the search space for representative phrase sets is very large, corresponding to the size of modern text corpora. The risk of premature convergence, where GA gets stuck in a suboptimal solution, is more pronounced in such a large domain. A higher mutation rate introduces additional randomness into the population, helping to explore different regions of the search space and avoid premature convergence by leaving local optima. In addition, a higher mutation rate promotes exploration by introducing more disruptive changes into candidate phrase sets. Such exploration can help discover new combinations of phrases that exhibit higher representativeness, even if they may initially be far from the optimum. Moreover, a higher mutation rate helps maintain diversity by introducing new phrases that are not yet present in the population, which is ensured by the design of our mutation operator. This diversity promotes the exploration of different linguistic patterns and increases the chances of finding more representative phrase sets.

Once the final parameters for GA were established, it was possible to formally compare the BF and GA sampling methods. Assuming that 500 solutions (GA population size) are evaluated in every single GA generation, a complete GA run with 1000 generations would include a total of $500 \times 1000 = 500,000$ evaluations. In this respect, comparing such a GA run with the BF procedure involving the same number of random sampling trials is fair. The result of this analysis is presented in Figure 13.

![Figure 13. Comparison of sampling results between the BF procedure with 500,000 random trials and a GA with 1000 generations and a population size of 500 (200-sentence phrase sets from the 5-word reduced fHrWaC were sampled).](image-url)
The ordinal number of BF trials can be tracked on the upper horizontal axis and is relevant for the highly variable KLD values obtained with the BF method (upper part of the graph). In contrast, the ordinal number of GA generation can be tracked on the lower horizontal axis and is relevant to the decreasing KLD values obtained with the GA approach (lower part of the graph).

The advantage of the GA approach is apparent in terms of the final solution and performance. The most representative phrase set found from the fHrWaC corpus reduced to five words using the BF procedure has the associated KLD value of 0.075669493. The phrase set with better representativeness was found within the first generations of the corresponding GA run. Moreover, the GA eventually resulted in a target phrase set with a considerably lower KLD, namely 0.023117662. In terms of execution speed, GA also outperforms BF sampling. However, this can be attributed to the multi-threading support in GA implementation, in contrast to the BF procedure, which relies on serial iterations with a sample-and-evaluate rate of almost 200 phrase set candidates per minute. Therefore, the results of BF shown in Figure 14 were obtained in no less than 40 h, which is significantly more compared to the GA approach.

![Graph showing KLD values over GA generations](image)

Figure 14. Final results of fHrWaC corpus GA-based sampling (only the first 100 generations are shown).

### 3.3. The Final Outcomes and a Multi-Language Context

In order to obtain representative phrase sets for the Croatian language, in a final step, we decided to target sets of exactly 200 phrases with seven different phrase lengths (3–9 words). We used the proposed GA sampling procedure, setting the parameters of GA as previously described. In addition, we configured GA to terminate after 10,000 generations to further improve the final solutions. Since all n-word reduced corpora \( n \in \{3, \ldots, 9\} \) were previously generated by BF sampling runs, additional n-word filtering was not required. Each GA run was then applied to the corresponding reduced corpus, and thus, the most representative phrase sets after 10,000 generations were found.

The results confirm that the relative entropy of the best solutions decreases when the target set contains phrases with more words. To see the differences between the seven KLD trends more clearly, we show the GA sampling results, but only within the first 100 GA generations, where the convergence speed is higher (see Figure 14).

The characteristics of the phrase sets obtained, the corresponding values of the KLD metric, and an example sentence from each phrase set (with an English translation) are given in Table 3.
Table 3. Characteristics of the most representative phrase sets sampled from the fHrWaC source corpus found using the proposed GA-based approach. Example phrases in Croatian are additionally translated into English (EN) for clarity.

<table>
<thead>
<tr>
<th>Phrase Length</th>
<th>Phrase Set Characteristics</th>
<th>Example Phrase</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 words</td>
<td>200 phrases, 600 words</td>
<td>jako ih volim</td>
</tr>
<tr>
<td></td>
<td>4.57 characters/word</td>
<td>EN: I like them very much</td>
</tr>
<tr>
<td></td>
<td>KLD(fHrWaC): 0.035726331</td>
<td></td>
</tr>
<tr>
<td>4 words</td>
<td>200 phrases, 800 words</td>
<td>tajna je u preljevu</td>
</tr>
<tr>
<td></td>
<td>4.87 characters/word</td>
<td>EN: the secret is in the dressing</td>
</tr>
<tr>
<td></td>
<td>KLD(fHrWaC): 0.025814862</td>
<td></td>
</tr>
<tr>
<td>5 words</td>
<td>200 phrases, 1000 words</td>
<td>želim otići iz toga kaosa</td>
</tr>
<tr>
<td></td>
<td>5.06 characters/word</td>
<td>EN: I want to leave this chaos</td>
</tr>
<tr>
<td></td>
<td>KLD(fHrWaC): 0.020257031</td>
<td></td>
</tr>
<tr>
<td>6 words</td>
<td>200 phrases, 1200 words</td>
<td>igranje bila za visoku ocjenu</td>
</tr>
<tr>
<td></td>
<td>5.11 characters/word</td>
<td>EN: the game was not for high rating</td>
</tr>
<tr>
<td></td>
<td>KLD(fHrWaC): 0.019554355</td>
<td></td>
</tr>
<tr>
<td>7 words</td>
<td>200 phrases, 1400 words</td>
<td>ostala sam jedan dan više u bolnici</td>
</tr>
<tr>
<td></td>
<td>5.22 characters/word</td>
<td>EN: I stayed one more day in the hospital</td>
</tr>
<tr>
<td></td>
<td>KLD(fHrWaC): 0.014567244</td>
<td></td>
</tr>
<tr>
<td>8 words</td>
<td>200 phrases, 1600 words</td>
<td>za večeru pojedite pitu ili čips od jabuke</td>
</tr>
<tr>
<td></td>
<td>5.25 characters/word</td>
<td>EN: have apple pie or chips for dinner</td>
</tr>
<tr>
<td></td>
<td>KLD(fHrWaC): 0.012153204</td>
<td></td>
</tr>
<tr>
<td>9 words</td>
<td>200 phrases, 1800 words</td>
<td>na njegovom posljednjem albumu ona je napisala osam pjesama</td>
</tr>
<tr>
<td></td>
<td>5.29 characters/word</td>
<td>EN: on their last album she wrote eight songs</td>
</tr>
<tr>
<td></td>
<td>KLD(fHrWaC): 0.011310758</td>
<td></td>
</tr>
</tbody>
</table>

After completing the search for representative phrase sets using the GA procedure, a problem remains that needs to be addressed. Large web-scraped corpora usually contain, in addition to the inevitable spelling and grammatical errors, some parts with offensive language that are difficult to filter out automatically. The fHrWaC corpus is no exception in this respect, so there is a possibility that the obtained phrase sets contain some undesirable texts. Therefore, it is recommended that the phrase sets are manually checked by correcting possible errors and removing unwanted content. Such a modification of the obtained phrase sets inevitably leads to a change in their digram statistics and, consequently, their relative entropy. A minor change in the KLD value can be expected if only a few grammatical issues are involved. However, finding the correct replacements may become a larger problem if many phrases need to be completely removed. Therefore, it would be ideal to have large text corpora that have been adequately “cleaned” beforehand at disposal.

In order to prove the usefulness and effectiveness of the proposed method on a general level, we decided to apply it to different languages. It is important to generate representative sets of phrases for different languages because it has already been proven that language plays a significant role in text entry experiments and directly affects the efficiency of text input [29]. We additionally extended the method’s capabilities to generate sets containing phrases with different word counts. In this way, it is left to the end user (the operator of the text entry experiment) to generate a representative set for the target language that best fits the intended design of the experiment. For example, if the goal is to test the input of memorable phrases,
generating a set of phrases containing between 3 and 5 words is possible. On the other hand, if one wants to study the effects of transcribing a larger amount of text, the method can be used to generate a phrase set with longer sentences.

The proof of concept was carried out, in parallel with the native Croatian language (HR), for English (EN), Italian (IT), Spanish (ES), Russian (RU), Polish (PL), French (FR), and German (DE). For each of the above languages, a corresponding character set had to be defined—a set of all letters/graphemes that can form valid digrams in that language. Following common practice in text entry research, we limited phrases to lowercase letters without punctuation, such as commas and periods. We chose phrase sets with different numbers of phrases (150, 200, 250, 300) containing different numbers of words (3–5, 4–6, 5–6, 6–7). Sampling was performed for each language using the BF and GA methods, thus allowing the comparison with the baseline method. In doing so, we decided to make two GA runs for each language—with two thousand and three thousand generations.

We used the OpenSubtitles 2013 collection [30,31] as a resource for text corpora in different languages. It should be noted that the mentioned corpora are not filtered with respect to the possible presence of undesirable parts of the text. The results of the sampling methods applied to this resource, targeting eight different languages, are shown in Figure 15.

![Figure 15](image-url)  

Figure 15. Results of applying the proposed method to several different languages from the OpenSubtitles 2013 corpora.

The upper part of the diagram shows box-and-whiskers plots illustrating the variability of the results of the separate experiments in which the BF approach was used (with 500,000 sample trials). In contrast, the lower part of the diagram shows the results of the GA approach—performed with a total of 1000 and 3000 generations. Since the best result of BF sampling is the one with the lowest KLD value, we connected all BF minima with a line. With this kind of visualization, we wanted to point out the following:
• The shape of the three lines shows that the final results of the different experiments (referring to different languages and target phrase sets) have a similar relationship, regardless of whether the BF or GA is used.
• The mutual distance between the three lines shows how effective GA is in finding better solutions, given the difference in the KLD values obtained.

As can be seen, it is once again shown that the proposed GA method outperforms the BF, this time in the context of different languages. Regardless of the target language, of how many phrases the target set contains, and of how many words phrases in the target set contain, the GA method is proven to be able to find a phrase set of higher linguistic representativeness. As expected, GA runs with 3000 generations yielded more appropriate solutions.

The details of the phrase sets obtained from sampling the corpora of different languages are given in Table 4.

### Table 4. Characteristics of the most representative phrase sets sampled from the OpenSubtitles 2013 (OS) source corpora found using the proposed GA-based approach with 3000 generations. Eight languages are selected for the analysis.

<table>
<thead>
<tr>
<th>Language</th>
<th>Phrase Length</th>
<th>Phrase Set Characteristics</th>
<th>Example Phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Italian (IT)</td>
<td>3–5 words</td>
<td>150 phrases, 656 words 4.72 characters/word KLD (OS – IT): 0.02193727</td>
<td><em>siete tutti sani</em> <em>prendila come un gioco</em> <em>hai chiamato a casa sua</em></td>
</tr>
<tr>
<td>Spanish (ES)</td>
<td>3–5 words</td>
<td>150 phrases, 679 words 4.45 characters/word KLD (OS – ES): 0.034925941</td>
<td><em>prepare el carro</em> <em>me bête de llorar</em> <em>ofender no era mi intención</em></td>
</tr>
<tr>
<td>Croatian (HR)</td>
<td>4–6 words</td>
<td>300 phrases, 1490 words 4.50 characters/word KLD (OS – HR): 0.012102423</td>
<td><em>to je samo sok</em> <em>bila sam jutros u banci</em> <em>ja ča ti paziti na opremu</em></td>
</tr>
<tr>
<td>English (EN)</td>
<td>4–6 words</td>
<td>300 phrases, 1567 words 4.08 characters/word KLD (OS – EN): 0.010343915</td>
<td><em>go with the plan</em> <em>that goes for every man</em> <em>love is like taking a train</em></td>
</tr>
<tr>
<td>Russian (RU)</td>
<td>5–6 words</td>
<td>200 phrases, 1103 words 4.94 characters/word KLD (OS – RU): 0.025255478</td>
<td><em>я вижу ты что знаешь</em> <em>я вернулся так скоро как смог</em></td>
</tr>
<tr>
<td>Polish (PL)</td>
<td>5–6 words</td>
<td>200 phrases, 1103 words 5.24 characters/word KLD (OS – PL): 0.021751797</td>
<td><em>nie ma leków na amnezję</em> <em>przysięgłem tak szybko jak tylko usłyszałem</em></td>
</tr>
<tr>
<td>French (FR)</td>
<td>6–7 words</td>
<td>250 phrases, 1618 words 4.49 characters/word KLD (OS – FR): 0.014245106</td>
<td><em>vous aurez votre chance cette nuit</em> <em>combiem de fois vous tu regender ça</em></td>
</tr>
<tr>
<td>German (DE)</td>
<td>6–7 words</td>
<td>250 phrases, 1622 words 4.87 characters/word KLD (OS – DE): 0.013459111</td>
<td><em>niemals mehr will ich dich verlieren</em> <em>man muss den kunden bei laune halten</em></td>
</tr>
</tbody>
</table>

### 4. Discussion

Text entry is still a hot topic among the HCI community. Many novel text entry methods and solutions for various application domains have recently been introduced. These include text entry in immersive virtual environments [32,33] and augmented reality systems [34], speech-based text entry [35], and typing on passive surfaces [36,37]. All these work uses a common phrase set, namely that developed by MacKenzie and Soukoreff, whilst related experiments sometimes only use part of the set. This suggests that, while there are ways to generate phrase sets with better language representativeness, many researchers have not yet utilized them. The above work points to another problem: text entry is mostly tested in English, even though the experiments are not necessarily conducted with native English speakers. Evaluations of text entry methods in other languages are limited, and when English is not the focus, either ad hoc phrase sets are invented [38] or MacKenzie and Soukoreff’s phrase set is translated [39]. These facts further contextualize the potential and the need for representative phrase sets.

Although GA can be considered an old technique, it continues to attract attention. In recent research efforts, it has been successfully applied in text analysis, particularly for text feature selection [40], text clustering [41], and text summarization [42].
Besides GA, other heuristic algorithms such as simulated annealing, particle swarm optimization, and ant colony optimization could also be considered to solve the observed problem. However, one of the main reasons for using the genetic algorithm was the interpretability of this method and its relation to the problem domain. Namely, the process of encoding phrases and their evolution by genetic operators allows the direct mapping of solutions to the problem domain. The set of phrases can be directly represented as a candidate chromosome, with each phrase representing a single gene. This representation is consistent with the nature of the problem and allows GA, with its crossover and mutation operators, to work directly with the units of interest. In addition, GAs are well suited for nonlinear optimization problems where the relationship between the input variables and the objective function is complex and may exhibit nonlinearity. In our phrase set selection, the fitness function is based on Kullback–Leibler divergence, which involves nonlinear relationships between character digrams, making GA a favorable choice for optimizing such an objective. Finally, since GAs have been extensively studied and implemented in various domains, numerous programming resources are available for implementing and tweaking GAs. Therefore, the proposed GA-based sampling method can be offered as a readily available resource for various platforms.

Although GA offers the above advantages, we recognize that the effectiveness of any optimization method depends on the specific problem and its characteristics and that it is important to also evaluate other algorithms. Implementing alternative optimization methods and comparisons with GA represents future research opportunities.

The current versions of all obtained phrase sets only include only lowercase phrases, as using only such phrases in HCI text entry experiments is common. If the inclusion of uppercase letters is preferred instead, corpora sampling without explicit lowercasing would be required, and the target character set would need to be defined differently. The same applies to including numbers, additional symbols, and punctuation marks. For example, suppose we want to add only simple punctuation (comma and period) to the current lowercase alphabet. In that case, the input charset file should be updated accordingly, e.g., for English, this would mean the following string: “abcdefghijklmnopqrstuvwxyz”. In this case, the digram matrix would have the size $29 \times 29 = 841$, with the corresponding digram frequency values calculated in the first step of the sampling procedure. Once computed, this digram probability distribution can be readily used as a language model for predictive text input evaluation. Namely, it can be combined with previously derived motion models (e.g., the well-known Fitts’s law in Equation (2)) to obtain predictions about text entry speed for a given input technique. At the same time, the digram probability distribution of the source corpus serves as a baseline for assessing the representativeness of the selected phrase set.

The proposed sampling procedure can be easily used to develop phrase sets with specific features that match the desired experimental design. The preferred source corpus, target language, associated character set, number of phrases, and phrase length can all be configured using input arguments from readily available programming methods. In this way, user-defined experimental stimuli can be generated without interfering with the proposed procedure’s internal implementation logic. The parameters of GA can also be configured in this way, allowing the further analysis of the efficiency of the procedure under different combinations of crossover probability, mutation rate, and elitism percentage.

Besides using the GA sampling procedure by simply applying the already provided methods, it is possible to modify the implementation’s source code to adapt GA sampling to more specific requirements. Here are some customization possibilities that can be performed, provided that some experience in .NET/C# programming has been acquired beforehand:

- A target phrase set could be generated according to a specific phrase length distribution (or average length);
- The use of a different language unit to evaluate the representativeness of a phrase set, e.g., a word-based KLD would include the relative entropy between the word probability distributions of the target phrase set and the SC;
• The initial preprocessing of the SC by removing phrases containing words from a list of known offensive terms;
• Introducing an adaptive mutation operator (e.g., with a phrase length-dependent number of genes involved in the mutation process).

Better source corpora will undoubtedly produce better target phrase sets. Therefore, it is desirable to obtain such text resources in which grammatical errors, rare dialect expressions, unpleasant content, and semantically meaningless phrases are rare. However, we must be aware that the language used in text entry is highly context-specific. For example, punctuation, grammar, and capitalization are largely ignored in today’s mobile messaging. At the same time, abbreviations and contractions (“1drfl” for “wonderful”), as well as specific shortenings (“lol” for “laughing out loud”) and pictograms (“i < 3 u” for “I love you”), are commonly used. Thus, if mobile text input is the subject of an experimental evaluation, it seems reasonable to use NUS SMS or public Facebook/Twitter feeds as the initial corpus for GA-based sampling, regardless of the potentially higher efforts required for initial filtering.

In empirical research on text entry, phrase sets should be used in such a way that the phrases they contain are evenly distributed among different tasks and participants so that the high linguistic representativeness of the sets supports the external validity of the results. In this context, the proposed method can either be used to generate a unique phrase set to standardize it for the target language or to create a customized phrase set corresponding to a specific experiment. For the latter case, we introduce the following heuristic guidelines to prepare the text entry experiment:
1. Select a high-quality and as large as possible text corpus for the target language and target domain of the text entry experiment;
2. Determine the number of test subjects to participate in the text entry experiment (N);
3. Determine how many unit tasks (typing different text phrases) the user must perform using the observed method (M);
4. If M is large (e.g., 100+ sentences), the proposed method can generate a representative set of M phrases, requiring participants to go through the entire set. On the other hand, if M is smaller (at the level of 10–20 sentences, which is common in text entry experiments), it is proposed to generate a set of $M \times N$ phrases, where each participant would type different M phrases from this set;
5. Report the statistics of the phrase set used, indicating the total number of phrases, the total number of words in the set, and the average number of characters in a word.

Conducting a text entry experiment with such a manipulation of phrase sets would help in standardization efforts, ensure the representativeness of the target language, and allow the valid comparison of results from different text entry studies.

Although it is known that target language and representativeness affect the efficiency of text input, we plan to study these effects in more detail in future work using phrase sets obtained with our method. In doing so, we plan to collaborate with researchers worldwide to organize experiments with participants who would write the text in both English and their native language.

5. Conclusions

Representative phrase sets play an essential role in empirical research on text entry, as they allow for a more accurate comparison of different text entry techniques. While Mackenzie and Soukoreff’s phrase set has emerged as a de facto standard, there have been notable efforts to improve the quality and contextual appropriateness of phrase sets. Instead of indiscriminately collecting random phrases, a more rigorous approach involves sampling from large, trustworthy text corpora to select linguistically more representative sets.

To tackle the optimization challenge of finding a relatively small, highly representative phrase set within the vast search space of a single-text corpus, we proposed a GA-based approach that utilizes Kullback–Leibler divergence to evaluate candidate sets. As demon-
strated in the paper, this method is highly customizable, outperforms typical random sampling, and exhibits language independence.

The implementation of the proposed method is readily available in the public GitHub repository [24], which promotes its immediate utilization and the reproducibility of this study. Researchers can quickly obtain representative phrase sets tailored to their designs of text entry experiments, considering factors such as the target language, source corpus, character set, phrase lengths, set size, and GA parameter values.

Finally, along with the proposed method and its source code, we provide heuristic guidelines for preparing and conducting text entry experiments. These guidelines include the generation of the target phrase set using the proposed method and its utilization in its entirety.

**Author Contributions:** Conceptualization, S.L.; methodology, S.L.; software, S.L.; validation, S.L. and A.S.; formal analysis, S.L. and A.S.; investigation, S.L. and A.S.; resources, S.L. and A.S.; data curation, S.L. and A.S.; writing—original draft preparation, S.L.; writing—review and editing, S.L. and A.S.; visualization, S.L.; supervision, S.L.; project administration, S.L.; funding acquisition, S.L.

All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

**Abbreviations**

The following abbreviations are used in this manuscript:

- **HCI** Human–Computer Interaction
- **GA** Genetic algorithm
- **KLD** Kullback–Leibler divergence
- **AAC** Augmentative and Alternative Communication
- **CER** Character Error Rate
- **BF** Brute Force (sampling)
- **SC** Source corpus
- **RC** Reduced corpus
- **PSC** Phrase set candidate
- **TPS** Target phrase set
- **GAF** Genetic Algorithm Framework

**References**


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.