CX 4240: Final Report
Optimal Spanish Keyboard

Members (GTID): Hamza Adel Yaafar - 903676655, Raquel Ovadia - 903685438, Augusto Chang - 903846106, Alexandra Guahnich - 903702101, Jose A Pages - 903593698.
Professor: Chao Zhang

Abstract

The goal of this research is to produce an optimized Spanish keyboard layout to enhance typing efficiency and comfort for users. This report presents two keyboard layouts - one based upon Genetic Algorithm (GA) and the other based on a TF-IDF approach - both of which were derived from a thorough analysis of a combination of Spanish words. Traditional QWERTY configurations, conceived in the era of typewriters, are not ideally suited for typing efficiency. Therefore, our study utilized a corpus containing a billion words to train machine learning models, enabling the analysis of common typing patterns and leading to the design of our optimized layouts.

Our methodologies included the TF-IDF approach, focusing on character importance and term frequencies to strategically place keys to minimize finger movement, and a GA approach, iterating through potential keyboard layouts to evolve an optimal arrangement based on typing effort. The VUCK layout and WBTM layout - names derrived from the firt top left letters - demonstrated an impressive improvements in efficiency over the traditional QWERTY design.

This abstract encapsulates the motivation, methodology, results, and implications of our research, providing a pathway for future investigations to further refine and personalize keyboard layouts, with a consideration for broader linguistic datasets, user adaptability, and the prevailing QWERTY standard. The study points to the potential of algorithmic optimization in keyboard design and sets the stage for subsequent real-world applications and user-centric evaluations.

1 Problem and Dataset

1.1 Background

The motivation behind optimizing the Spanish keyboard layout stems from the desire to enhance typing efficiency and reduce physical strain for typists. Traditional keyboard layouts, like QWERTY, were designed in the 19th century to prevent typewriter jams rather than to optimize typing speed. With the evolution of digital typing devices, the need for layouts that minimize finger movement, reduce errors, and increase typing speed has become more apparent. In the context of the Spanish language, which has unique linguistic characteristics and letter frequency patterns, there is a significant opportunity to design a keyboard layout that caters specifically to its typists. This project aims to leverage machine learning to analyze large datasets of Spanish text, identify the most common letter combinations, and design a keyboard layout that minimizes the distance between frequently used keys, thereby maximizing typing efficiency and comfort.

1.2 Problem Statement

The primary problem this project addresses is the inefficiency of existing keyboard layouts for Spanish-language typists. Traditional layouts may not reflect the frequency and patterns of letter use in Spanish, leading to increased typing time, higher error rates, and potential physical discomfort. The challenge lies in developing a data-driven, optimized keyboard layout that aligns with the natural typing habits and linguistic features of the Spanish language. The goal is to create a layout that reduces finger movement and improves typing speed and accuracy, which can have a significant impact on productivity and ergonomic health for individuals who type in Spanish extensively.

1.3 Data

For this project, the data source used is the "Spanish Billion Words Corpus" available on Hugging Face `https://huggingface.co/datasets/spanish_billion_words`. This dataset is a comprehensive collection of Spanish text compiled from various sources, providing a broad and diverse linguistic foundation suitable for analyzing letter frequency and common word patterns in the Spanish language. The corpus contains a billion words, ensuring that the analysis covers a wide range of vocabulary and linguistic structures. By leveraging this extensive dataset, we can train machine learning models to accurately identify and analyze the typing patterns and frequencies of letters and letter combinations in Spanish, forming the basis for designing an optimized keyboard layout.

2 Methodology

To solve our problem, we implemented two different methodologies in order to compare the results and decide which would be the best overall method to create an optimal keyboard for Spanish speakers.

2.1 Approach 1 - TF IDF:

To implement these methods, we first loaded the cleaned csv dataset to python. Then, we initialized the $tf - df$ DataFrame, with rows representing each character from the Spanish alphabet, including all special characters like 'ñ' and punctuation marks, and the columns represented each individual document. To fill in the term frequencies DataFrame, we employed two different functions. First, we used the function $split - accent$ to handle accented characters by separating the accent from the alphabetic character, allowing us to analyze their frequency separately. For each document, we counted the frequency of each character using the $count - chars$ function, while also updating the $tf - df$ DataFrame accordingly. Subsequently, to calculate the TF-IDF values, we then multiplied the term frequency of each character in each document by its respective IDF value to obtain the TF-IDF matrix, and

$$\text{IDF(character)} = log\left(\frac{Total\ number\ of\ documents + 1}{Number\ of\ documents\ containing\ the\ character + 1}\right) + 1$$

Figure 1: TF-IDF Character Formula

then computed character importance by adding the TF-IDF values across rows. In addition, we calculated bigram frequencies by extracting adjacent characters from each document and counting their occurrences across the entire dataset. This provided insights into the most common pairs of characters used in Spanish.

With the character importance and bigram frequencies at hand, we proceeded to design the optimal keyboard layout. The layout was built based on the typing effort required for each key, based on the Colemak Mod-DH effort grid. Moreover, we assigned each character to an effort weight based on character importance, where the most important characters received the lowest effort score. For instance, characters 'e', 'a', 'o', and 's', which had the highest character importance scores, were placed in the keys with an effort score of 1.0.

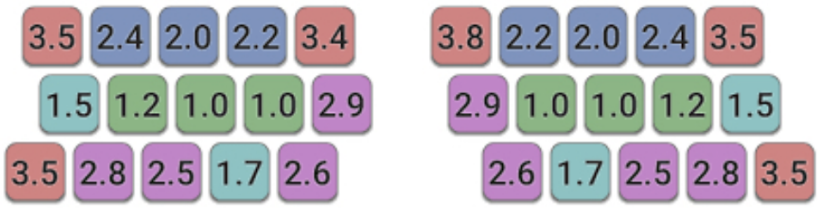

Figure 2: Keyboard Effort Calculator by Colemak Mod-DH project

To fine-tune the arrangement of letters within each section and key of the keyboard, we utilized the bigram frequency data. This approach ensured that pairs of characters frequently appearing together, such as 'e' and 's' from the bigram 'es', were positioned adjacently or close to one another. This strategic placement aimed to minimize the distance that fingers need to travel when typing common sequences, thereby enhancing typing speed and decreasing finger movement. By systematically applying these methods, we developed a keyboard layout that optimizes typing efficiency for Spanish speakers.

2.2 Approach 2 - Genetic algorithm:

The second method that we implemented was the genetic algorithm. The genetic algorithm is based on the theory of evolution and can be used to optimize an object through repeated crossover and mutation of the best elements from a randomly chosen population. The reason why we chose this method is because we felt that it would be a good way to optimize a complex problem like this one, since there are so many different permutations of keyboards.

The way the genetic algorithm works is that it selects a random population of keyboards with different layouts and measures the fitness (effort required) for each of those keyboards based on the dataset. Afterwards, it chooses the best elements from the population and has them compete against each other. Ultimately, the best layouts are probabilistically chosen to become the parents of new keyboard layouts that are a combination of both. There is also a random chance for keys to be mutated (randomly switched) since this will allow for diversity and prevent us from stopping at a local minimum. This new generation of children is then taken through the process again and everything is repeated.

In our code, fitness was determined by cumulative effort required by each keystroke given the effort scale we found online. For our selection process, we used tournament selection, since it is effective and allowed for less fitness evaluations to be conducted, while also preventing noise. As for crossover, we created a code that would shuffle the key-effort pairs of both parents and would then be randomly assigned to the child, with any surplus or deficit keys being placed in their respective spots. Finally, mutation was conducted through the random swapping of keys.

It was important to choose the right hyperparameters in order to optimize the keyboard as much as possible, but the genetic algorithm is already very computationally intensive and we wanted to prevent further growth in this intensity. We decreased the amount of data that we used to 1000 lines in favor of proper hyperparameters, since this would lead to a more generalized code that could be replicated with other datasets in the future. In order to tune these parameters, we created a parameter grid with different values for the mutation rates, population number, generation number, and crossover rate. It is important to find the proper population and generation numbers since we do not want to run the model for too long and risk overfitting the keyboard to our data. We also didn't want the mutation rate or crossover rate to be too low since this would prevent diversity which could lead to a local minimum, but if it were too high, the model would become too unstable.

We used the GridSearchCV function from the scikit-learn library in order to run a search of all permutations of our parameter grid. This function also uses cross validation in order to test the model's ability to generalize new data. We searched over 300 combinations of parameters. Ultimately, we found that the best hyperparameters given our data were a crossover rate of 0.8, 50 generations, a mutation rate of 0.05, and a population size of 150.

3 Results and Evaluation

The Genetic Algorithm (VUCK Layout) and the TF-IDF (WBTM Layout) displayed below are the result of the two methodologies used:

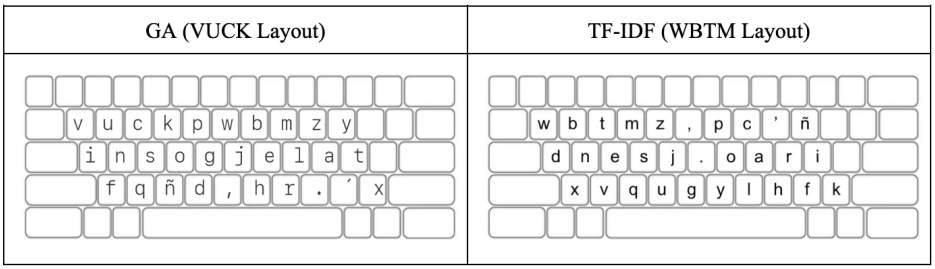

Figure 3: Generated keyboard layouts

From the outcomes we can see some similarities with the character position with equal weights assigned. These shared keys include: $O(1.0), E(1.0), I(1.5), C(2.0), J(2.9), X(3.5)$. The differences from each design rely on the nature of each algorithm.

For our genetic algorithm model, we used overall fitness as a metric for evaluation. As mentioned previously, this fitness is calculated through the effort scale that we found online. Our VUCK layout resulted in 186.42 effort per line on average, which, when compared to QWERTY's effort of 247.54 per line on average, we can see that there is an approximate improvement of around 32% in efficiency using this new layout. The loss diagram below shows the fitness of the best model across multiple generations. As we can see, we noticed that the model stopped decreasing at around 11 generations and then skyrocketed, leading into a plateau. This is the result of over-fitting that can occur from running the model for too many generations, which means that in the future, if we had more time and RAM we would extend the hyperparameter grid to include more alternatives for these hyperparameters. We chose the minimum of this evolution.

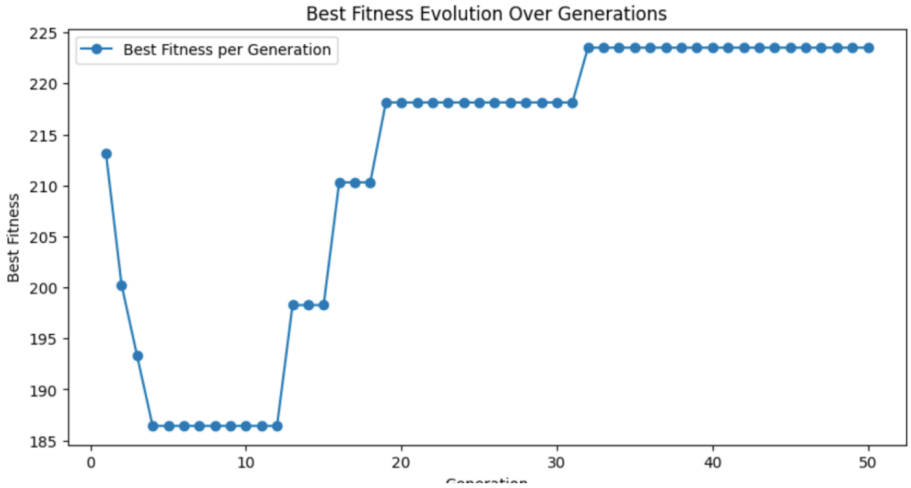

Figure 4: Fitness Evolution Graph

To evaluate the efficiency of the two generated keyboard layouts, we used TF-IDF as the metric that allowed us to determine through typing optimization scores the results of our project.

To do so, we focused on evaluating a dataset of words extracted from the Spanish corpus selected. We calculated the Term Frequency and the Inverse Document Frequency, using the smoothing formulation presented before, and we determined the TF-IDF Matrix. We assessed the significance of each word by summing the TF-IDF scores within the matrix rows. To examine the efficiency of the keyboard designs accurately, we selected a representative sample of words. This sample considered both word significance and word length to reduce noise from the short high-frequency words and avoid bias to prevent skewed results. Utilizing these two parameters yielded a filtered dataset with words ranging from 1 to 15 letters with importance levels above the mean.

By stratifying our sample by length and importance weight, we provided a sample with an accurate representation of typing patterns as the dataset for our Total Cost Function. This function takes in the list of words and the effort grid dictionary of each key position, and it cumulatively determines the total cost of each layout.

Using this function we tested the Genetic Algorithm layout, the TF-IDF layout, and for comparison the QWERTY layout. Taking into account that we seek to minimize the cost value, it is visible as shown in the chart below, that the TF-IDF layout is slightly more efficient than the GA layout by a difference of 38.7 efficiency points, however, both generated keyboards show a significant improvement from the optimization level tested with the QWERTY layout.

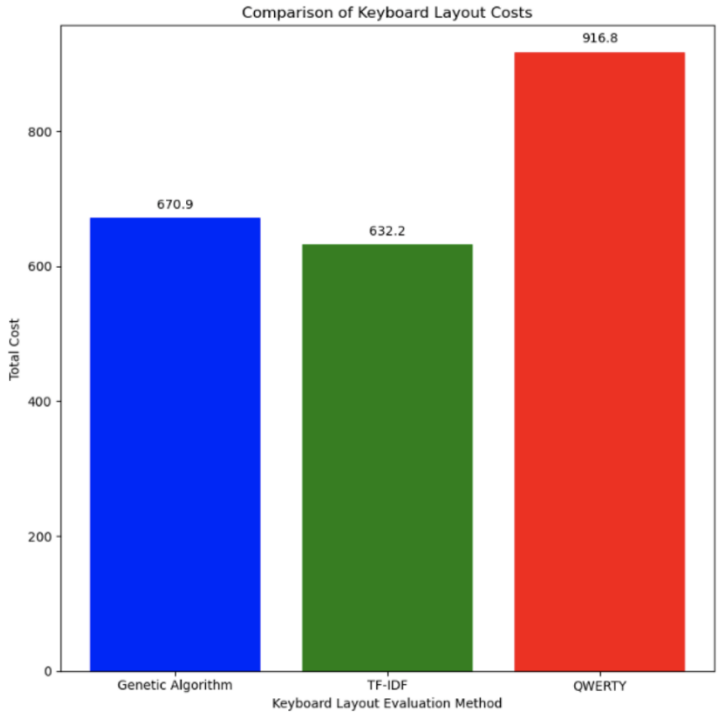

Figure 5: Keyboard Cost Comparison

4 Conclusion

4.1 Discussion and learning

The exploration into optimizing the Spanish keyboard layout through this study has concluded with the design of two more efficient keyboard layouts using the GA and TF-IDF models, indicating the potential of algorithmic optimization in keyboard design.

As observed in Figure 5, the GA (VUCK) and TF-IDF (WBTM) have demonstrated improvements over the conventional QWERTY, with the VUCK layout showing a 32% increase in efficiency. Such advancements underscore the importance of incorporating machine learning techniques in keyboard design, which can leverage datasets to create layouts for specific languages. In order to enhance accuracy, expanding the analysis to include a broader dataset is crucial for including a wider range of typing patterns.

Nonetheless, while the data points towards a compelling case for the implementation of these optimized layouts, it is crucial to consider the real-world application. Factors such as the learning curve associated with new layouts, user adaptability, and the entrenched position of QWERTY as a universal standard are significant. The potential benefits of adopting alternative layouts such as VUCK or WBTM must be weighed against these considerations.

4.2 Next Steps

The next steps for this project involve several key actions: Firstly, using a GPU or more computing power to handle larger datasets for more accurate modeling and analysis, leading to further improvements in keyboard layout optimization. Secondly, training the model for specific use cases, such as English, German, and other languages or specialized domains like programming languages, to create optimized keyboard layouts tailored to different linguistic requirements. Thirdly, developing a physical keyboard prototype based on the optimized layout and conducting user testing to evaluate its effectiveness and ergonomic benefits in real-world typing scenarios and how willing are people to move away from traditional keyboards. By focusing on these next steps, the project can advance from theoretical optimization to practical implementation, potentially leading to a valuable innovation in keyboard design for Spanish and other languages.

5 References

- Colemak mod-DH. Colemak Mod-DH - Keyboard effort grid. (n.d.). `https://colemakmods.github.io/mod-dh/model.html`
- Cardellino, C. (2019, August). Spanish Billion Words Corpus and Embeddings.

Retrieved March 2024, https://huggingface.co/datasets/spanish_billion_words

- Codes, A. (2022, September 23). Using AI to create the perfect keyboard. YouTube/@adumb_codes. ttps://www.youtube.com/watch?v=EOaPb9wrgDY

6 Appendix

.ipynb files