CSI4107 Human and Computer Interfaces

Assignment 1

Due: End of 11, November (Saturday) 2023

Interaction Scenario 1

You are trying to optimize the best-performing touchscreen keyboard layout for the stylus pen. The performance of a specific keyboard layout is evaluated based on the average movement time to reach a key, with lower times being considered superior. Touch screen users are assumed to use only one stylus pen when using the keyboard, and no time is taken into account for cognitive processes such as deciding which key to press during keyboard usage. This optimization process is conducted using a `large-scale English corpus`, and it is assumed that no errors occur when typing.

Your task is to implement a `Python` based program to optimize the keyboard layout based on `word_frequency.csv` data file and `Fitts' law`. `word_frequency.csv` file is a large-scale corpus analysis dataset from the University of Leeds. This file ranks words based on their frequency of occurrence per million words and provides corresponding frequencies.

The keyboard layout follows a 5 x 6 structure. The keys in the layout are all of the same size, measuring 1cm x 1cm, and arranged in a square shape. There are 26 characters placed on the keyboard layout, with the remaining 4 cells left empty. The initial keyboard layout is shown in Figure 1.

Interaction Scenario 2

In Interaction Scenario 1, the keyboard layout was optimized for a `large-scale corpus`. In this scenario, we will optimize the keyboard layout for `writing.txt` rather than a large-scale corpus. The content of the `writing.txt` file is the 138th Anniversary Commemorative Address of President Seoung Hwan Suh of Yonsei University. The text has been edited in some parts for keyboard layout optimization, and it contains only alphabets, whitespace characters, hyphens (-), and commas (,).

The initial keyboard layout is a 5 x 6 structure, with key sizes of 1cm x 1cm, as shown in Figure 2. Unlike the keyboard layout in Interaction Scenario 1, this keyboard layout includes a space key, a backspace key, a hyphen (-) key, and a comma (,) key.

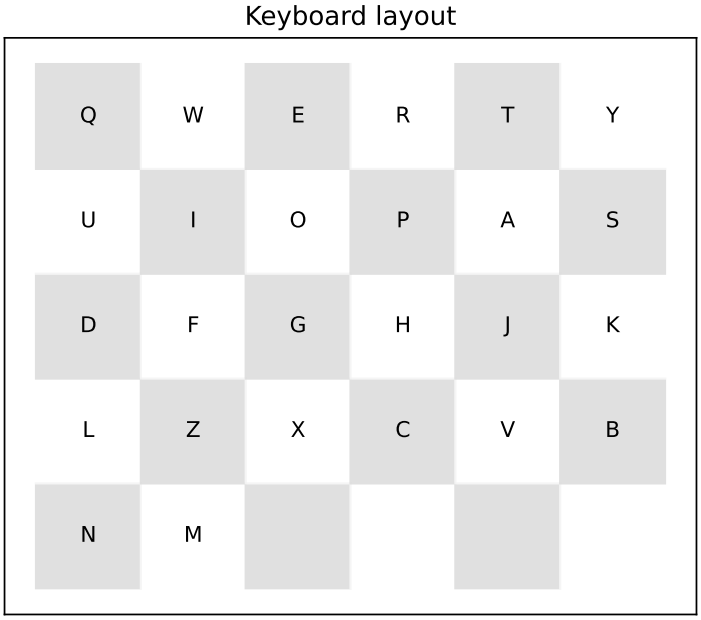

Figure 1: Interaction Scenario 1 - Initial keyboard layout

Besides the keyboard layout structure, another difference is there can be `Fat finger errors` in the `writing.txt` file. `Fat finger error` is a human error caused by pressing the wrong key when using a keyboard to input data.

If an error occurs, assume that the user pressed an unintended key instead of the intended key. The incorrectly entered key is very close in distance to the intended key. Thus, the calculation for Fitts' Law is based on the distance (D) to the intended key, rather than the incorrectly entered key. In such cases, the user presses the backspace key to delete the incorrect letter and then presses the correct letter. The error occurrence probability is `4%` for all letters except backspace.

When typing `'yonsei'`, the correct order of input with no error is y→o→n→s→e→i. If an error occurs at `'n'`, user should type y→o→n→backspace→n→s→e→i. If the first input, `'y'`, has an error, user should type y→backspace→y→o→n→s→e→i. Therefore, considering the `Fat finger error probability`, the typing process may not follow the sequence of y→o→n→s→e→i, but instead become y→backspace→y→o→n→s→e→backspace→e→i or y→o→n→backspace→n→s→e→i.

In the keyboard layout optimization for `Interaction Scenario 2`, you must consider the probability of `Fat finger error` when typing the contents of `writing.txt`. The remaining conditions, other than the differences explained earlier, are all the same as the `Interaction Scenario 1`.

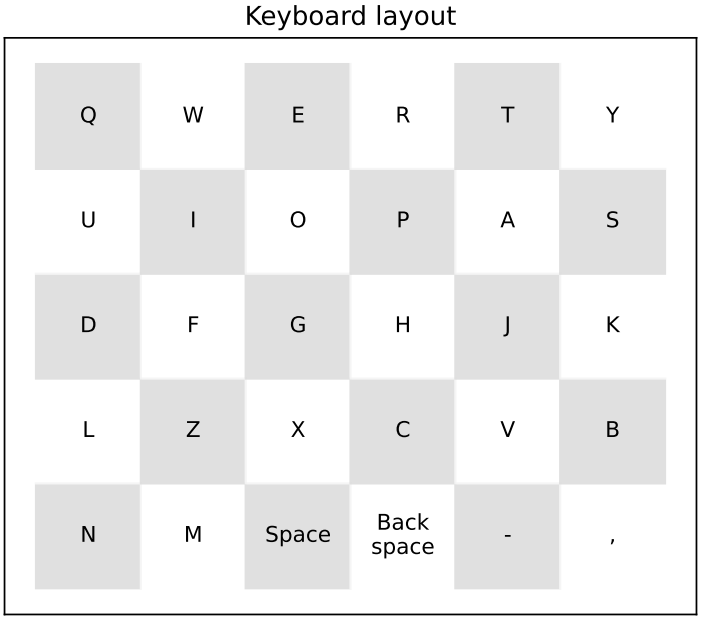

Figure 2: Interaction Scenario 2 - Initial keyboard layout

Your task is to implement a `Python` based program to optimize the keyboard layout based on the `writing.txt` file, `Fat finger error`, and `Fitts' law` (In the same way as `Interaction Scenario 1`).

The keyboard layout to be optimized follows a 5 x 6 structure. The keys in the layout are all of the same size, measuring 1cm x 1cm, and they are arranged in a square shape. There are a total of 30 characters (26 Roman letters, hyphen (-), comma (,), backspace, whitespace) placed on the keyboard layout. The initial keyboard layout before the optimization is shown in Figure 2, and it must be used for the optimization process.

Optimization Method

Our assignment's optimization metric is the average movement time to reach a key. It is possible to estimate the average text entry time on a given keyboard layout by summing up Fitts' law movement times between every pair of letters, weighted by the transitional frequency from one letter to another. According to Fitts' law, D is the distance from the center of key i to the center of key j and W is the key width (1cm). In this assignment, a is 0.083 and b is 0.127 for Fitts' law calculation. All other conditions follow each `Interaction Scenario`.

Various approaches can be applied to optimize keyboard layouts. In this assignment, you

have to use `Random search algorithm` and `Metropolis algorithm` for keyboard layout optimization. This approach consists of a finite number of iterations. In each iteration, the `Metropolis algorithm` picks up two random keys on the keyboard and swaps their positions to reach a new configuration. The text input time of a new configuration is then estimated based on Fitts' law. Whether the new configuration is kept as the starting position for the next iteration depends on the following `Metropolis function`:

$$W(O \to N) = \begin{cases} e^{-\frac{\Delta t}{kT}} & (\Delta t > 0) \\ 1 & (\Delta t \leq 0) \end{cases} \tag{1}$$

In Equation 1, W($O \to N$) is the probability of changing from configuration O (old) to configuration N (new); $\Delta t = t_{new}$ - t_{old}, where t_{new}, and t_{old} are mean times for typing a character on the new and old keyboard configurations, estimated by Fitts' law; k is a coefficient; T is 'temperature', which can be interactively adjusted. Key to the algorithm is the fact that the search does not always move toward a lower energy state. It occasionally allows moves with positive energy changes to climb out of a local minimum.

Result & Submission

For Interaction Scenario 1, The function `optimization` in the `optimization.py` performs the optimization of the keyboard layout based on the `word_frequency.csv` file. Similarly, for Interaction Scenario 2, the function `optimization` in the `optimization_error.py` performs the optimization of the keyboard layout based on error occurrence probability. For each scenario, you need to write the `optimization` function and provide detailed explanations through comments and docstrings in your written code. Each `optimization` function returns two lists as follows: (refer to the `Python` code example in `optimization.py` and `optimization_error.py`).

1. Optimized keyboard layout result (`Python list format`)
2. Changes of the Fitts' law movement time value (average movement time to reach a key) recorded during the optimization process (`Python list format`)

In the list storing Fitts' Law movement time values, the first element represents the movement time value for the initial keyboard layout. All Fitts' law result values should be rounded to 6 decimal places. (You have to use the `round()` function for rounding.)

When `main.py` (or `main_error.py`) is executed, the function `optimization` is executed, and this function returns two lists. `main.py` (or `main_error.py`) then saves the two returned lists as a binary file using `pickle`. So, to submit the binary file, you must execute `main.py` after writing the `optimization` function code. You can use the following libraries: `numpy`, `pandas`, and any `Python` standard libraries. If you want to use any other third-party library, please contact the TA. You should zip and submit your source codes and binary files (`optimization.py`, `optimization_error.py`, `main.py`, and `main_error.py` must be

included, and do not change the function name and structure) following the given file hierarchy. The filename and folder name should include your student ID.

```
YourStudentID.zip
└── YourStudentID
    ├── scenario_1
    │   ├── dataset
    │   │   └── word_frequency.csv
    │   ├── optimization.py
    │   ├── main.py
    │   ├── YourStudentID_keyboard.pkl  // Keyboard layout
    │   └── YourStudentID_time.pkl  // Fitts' law time changes
    └── scenario_2
        ├── dataset
        │   └── writing.txt
        ├── optimization_error.py
        ├── main_error.py
        ├── YourStudentID_keyboard_error.pkl  // Keyboard layout
        └── YourStudentID_time_error.pkl  // Fitts' law time changes
```

Important Note

1. In Interaction Scenario 1, You don't need to consider non-alphabetic characters. For instance, in the word 'long-term' the sequential letter pairs are 'l-o', 'o-n', 'n-g', 't-e', 'e-r', and 'r-m'.

2. Even if you submit an optimized keyboard layout, you will not receive a score if you have not implemented the optimization process correctly. (For example, keyboard layout results through only random search, modifying someone else's optimized keyboard layout and submitting it, etc.)

3. The implemented optimization function code is also subject to grading. Therefore, please provide detailed comments and docstrings to allow the code to be graded.

4. Since it is a probability-based optimization method, the results of all optimized keyboard layouts may not be the same. If you want to maintain a specific optimized result, please implement the function `optimization` by hard-coding a seed.

5. The empty keys on the keyboard layout can be placed anywhere. There is no specific key that needs to be filled first, and you are free to design the keyboard layout.

6. All submissions are checked for plagiarism. Once detected, measures will be taken for

all students involved in the plagiarism incident (including the "source" of the plagiarized code).