Phase – 1 Report
CSE 515 – Multimedia and Web Databases

Group Members:

- Karan Ajith (kajith@asu.edu, 1217144448)
- Sandeep Sunny (ssunny2@asu.edu, 1217181914)
- Shubham Mehta (smehta24@asu.edu, 1213240039)
- Daniar Tabys (dtabys@asu.edu, 1213757322)
- Jay Nankani (jnankani@asu.edu, 1217122205)
- Monil Nisar (mnisar2@asu.edu, 1217111805)

Abstract:

This report is based on Phase 1 for the CSE 515 course project. The aim was to experiment and understand multivariate time-series data sets and how to manipulate them to generate various vector models. The data set provided was the Z – axis values of twenty sensors attached to various body parts of the test subjects recorded as they performed different gestures. The data was documented for 60 different gestures. The database provided was converted into a univariate time-series data sets and several different vector were generated based on TF(Term Frequency), TF-IDF(Inverse Document Frequency) and TF-TDF2 values for the words occurring in each gesture. The phase examines the different ways to convert the existing data to make it easier to perform other tasks on it. It also explores how different vectors of the dataset can be used to understand the composition of each object in the dataset and helps in comparison between different objects.

Keywords:

Word, Vectors, Term Frequency, Inverse Document Frequency, Heatmap, Similarity, Normalization, Quantization, Gesture, Sensor

Introduction:

- **Terminology:**
  - **Normalization** - To change the range of the values given. In this case bringing all values to [-1, 1].
  - **Quantization** - To make continuous values discrete, according to Guassian band lengths.
  - **Wordification** - To slide a window across a data record, to group the values in the window to make words.
  - **Shift length** - The shift made by the window while wordification
  - **Resolution** - The value of how fine, small movements we want to capture by the gaussian band.
- **Goal Description:**
  This phase was divided into four tasks on the provided data set.
  - **Task 1:**
    Implement a program that would take the directory of the given data set as input and convert the dataset to a univariate time-series dataset. The initial data set values must be normalized and quantized. Wordification is done on the quantized values to generate word files for each gesture. The final data after manipulation will be stored in a file named 'f.wrd' in the same directory .
  - **Task 2:**
    Implement a program that will take the univariate time series data for each gesture file and create three vectors based on their TF, TF-IDF and TF-IDF2 values. These vectors must be stored in a separate file named 'vectors.txt' in the same directory.
  - **Task 3:**
    Implement a program that lets the user select any gesture in the existing database and generate a heat map (grey scale) based on the TF, TF-IDF and TF-IDF2 values.
  - **Task 4:**
    Implement a program that will allow the user to take any gesture from the existing dataset and find the 10 most similar gestures to the query gesture in the database. The comparison is also done between the query gesture and the others based on TF, TF-IDF and TF-IDF2 values.
- **Assumptions:**
  The assumptions considered for the code are:
  - All gesture files in the dataset are of the type '.csv'
  - There are 20 sensors
  - The first task is always run before running the other tasks (This assumption was made since all other tasks rely on the output of the first task as their input)

Implementation:

Each task was written as separate python programs. The python version used for development and testing was Python 3.7.4. Implementation of each task will be explained in detail individually.

- **Task 1 Implementation:**
  The data set consisted of 60 gesture files. Each gesture file contained 20 rows which showed the Z-axis values of each sensor attached to the body. These values were recorded across time. However,

each gesture did not have the same time span. As a result, some had more values per gesture that many of the other gestures. The aim of the first task was to convert this data into a univariate time-series dataset for all the gestures. The first step required was to normalize the data between -1 to 1. This was done since the values to gain better consistency and remove redundant values for each row of values.

The data was normalized for each sensor's values across all gestures. This means that the highest and the lowest values of sensor 1 was mapped to other 1 and -1 respectively and the other values in the row were scaled accordingly. The reason behind normalizing based on the sensor for each gesture was to ensure that each sensor was considered as a unique item(since the sensors are connected to various parts of the body, the range of values for each sensor should be considered uniquely).

$$normalizedX = -1 + \frac{2 * (X - minimum)}{(maximum - minimum)}$$

Once the values are normalized, quantization is performed on the dataset. Quantization refers to converting the real value present in the dataset to discrete values that are easier to understand. The values are represented in a normal distribution chart between the normalized range of -1 to 1.

$$length_i = 2 \times \frac{\int_{(i-r-1)/r}^{(i-r)/r} Gaussian_{(\mu=0.0,\sigma=0.25)}(x)\,\delta x}{\int_{-1}^{1} Gaussian_{(\mu=0.0,\sigma=0.25)}(x)\,\delta x}$$

$$Gaussian(x) = \frac{1}{\sigma \times \sqrt{2\pi}} \times e^{\frac{-(x-\mu)^2}{2 \times \sigma^2}}$$

The graph is divided into (2 * r) bands, where 'r' is the resolution considered. Each band is given a number between 1 and (2 * r). All normalized values are plotted across the graph and their quantized value is the value of the band that they are found in. The graph was divided into (2 *r) bands in a way to ensure sensitivity and make extreme values stand out.

The final step in the task was to create words(wordification) from the final quantized dataset. Using the shift-length and the window-length (both of which are defined along with the resolution in a file called 'param.json'). These values specify the length of the word. These words are stored as part of a pair along with an 'idx' for each word. This 'idx' was an identifier that contained the gesture the word was found in, the sensor which recorded the words and the time in which it was recorded. The format in which the final data is recorded in 'f.wrd' is (where 'f' is the 'gesture _id'):
{f, sensor_id, time} [q_1, q_2, q_3]

- **Task 2 Implementation:**
  The input for task 2 is the univariate dataset that is returned as the output from Task 1. The aim of this task is to use the given data set and find the TF, TF-IDF and TF-IDF2 values for each word that occurs for each gesture. To maintain consistency and for ease of calculation, the length for all the vectors were the set of distinct words that occurred in the complete dataset, regardless of the gesture or the sensor. Each value in the vector corresponds to the word. The words were arranged in numerical order to ensure consistency in every execution.

  TF (Term Frequency) refers to the number of times a particular word occurs in a file with respect to the total number of files in the file. For this case, we found the ratio of the number of occurrences of each word in a particular sensor in a gesture file with respect to the total number of words in the sensor in the gesture file. The final TF vector was made for each gesture ID and sensor ID combination.

$$TF = \frac{n}{N}$$

$$where\ n = number\ of\ occurances\ of\ the\ word\ and$$
$$N\ = Total\ number\ of\ words$$

  The calculated TF values for each word with respect to the (gesture, sensor) pair was saved in a file named ‘tf_vectors.txt’.

  IDF (Inverse Document Frequency) is a value used to identify and remove irrelevant words from the dataset. This is done to make sure that certain words that are not essential are not given the same weight as other critical words. TF-IDF and TF-IDF values for each word is used to calculate this importance.

$$IDF = \log_e \frac{f}{n_f}$$

$$where\ n_f = number\ of\ files\ the\ word\ occurs\ in\ and$$
$$f\ = Total\ number\ of\ files$$

  IDF values are calculated for each word that occurs in a sensor across all the gesture files. For this case, files refer to the number of gesture files in the dataset. IDF values are stored for each word with respect to the sensor in the file ‘idf_vectors.txt’. Whereas IDF2 values are calculated for each word in a gesture across different sensors. The point to note is that each word occurring in a sensor is considered unique to the sensor. This means that even if the same word occurs in a different sensor, it is considered as a separate word. Due to this, idf2 values for each gestures across 20 sensors would be log(20) for all the words that occur in the gesture. The values for the words that do not occur are considered as 0.

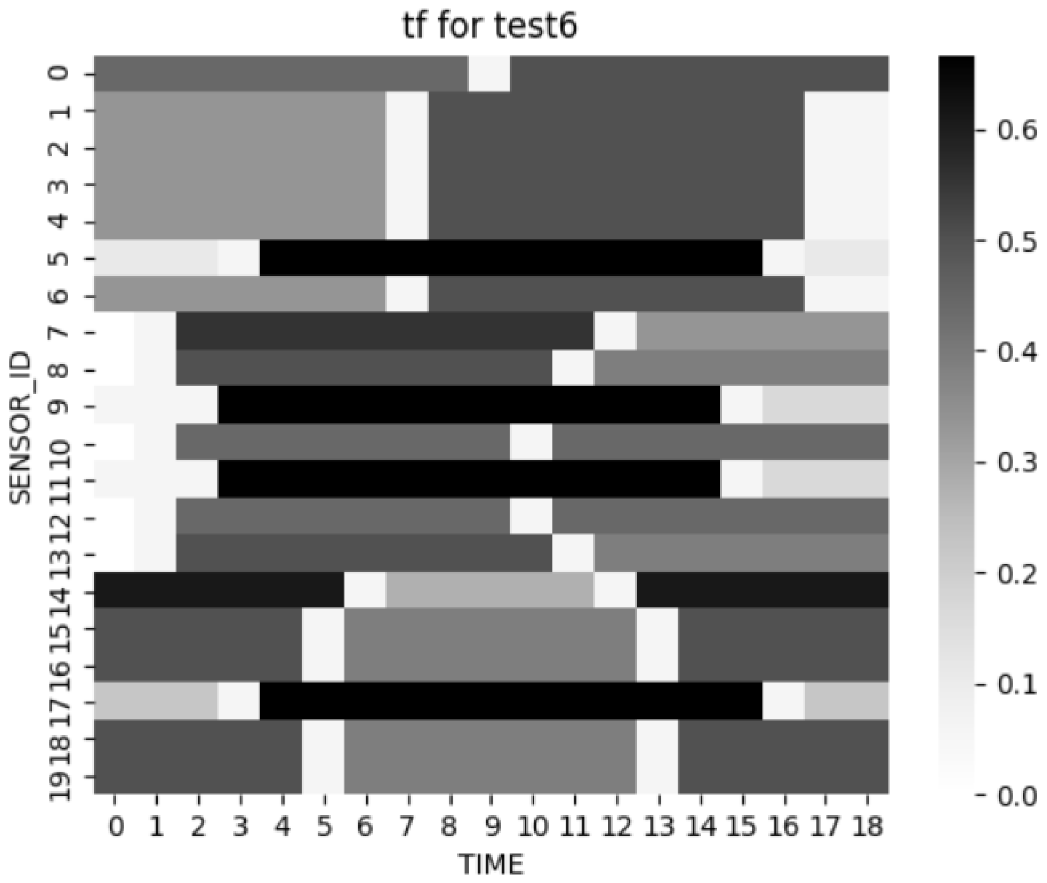
$$TF\ IDF = TF * \ IDF$$
$$TF\ IDF2 = TF * \ IDF$$

TF-IDF and TF-IDF2 values were calculated for each word with respect the corresponding (gesture, sensor) pair was saved in files named 'tf-idf_vectors.txt' and 'tf-idf2_vectors.txt'. All these files are saved in the same directory as the code for reference purposes.

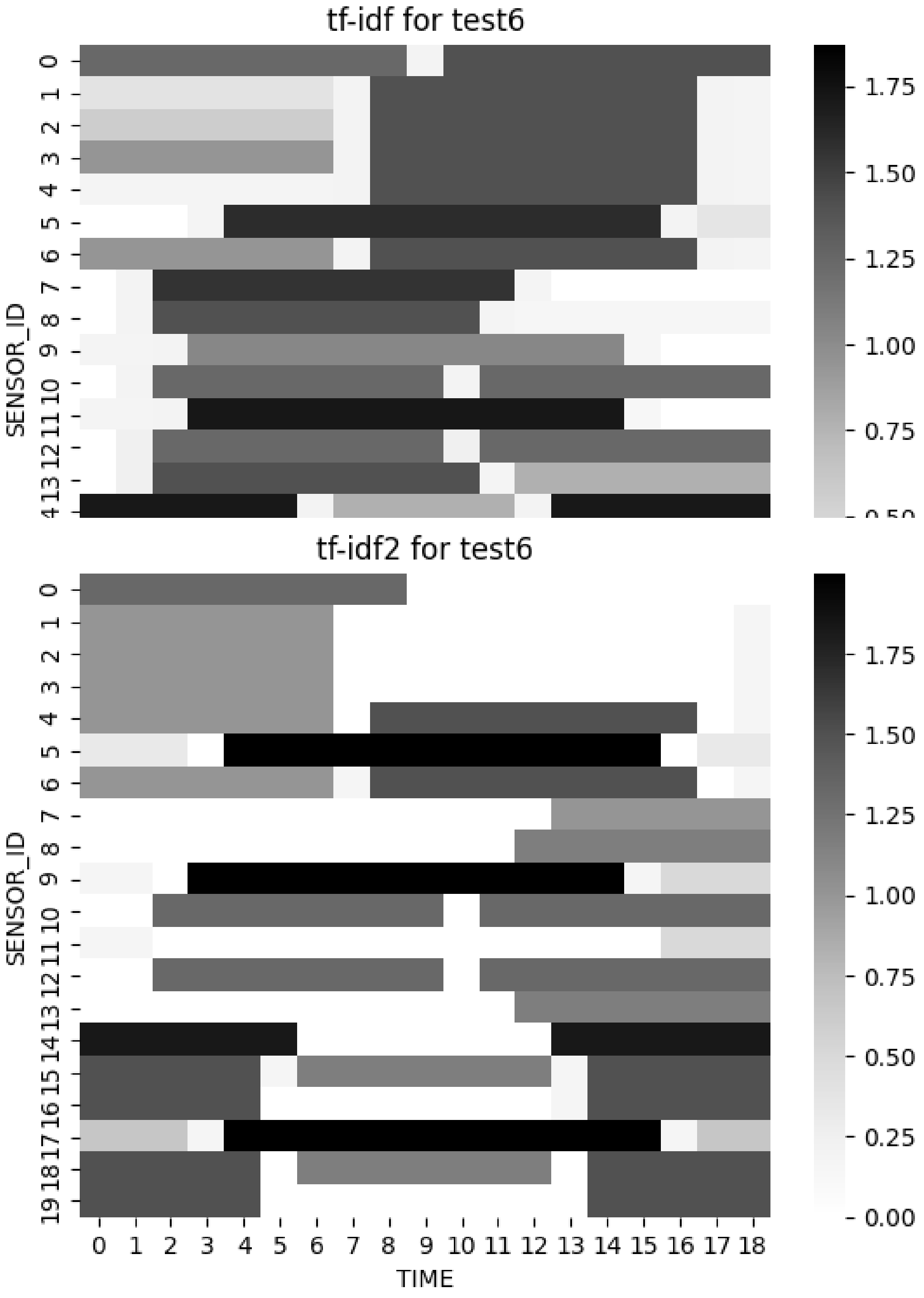
The final consolidated vectors for each gesture is stored in a file named 'vectors.txt'. This is saved in the same location as the dataset. The format in which the data is stored is of the form:

```
Gesture_ID: {
        TF Vectors: {tf(sensor_1), tf(sensor_2),….}
        TF-IDF Vectors: {tf-idf(sensor_1), tf-idf(sensor_2),….}
        TF-IDF2 Vectors: {tf-idf2(sensor_1), tf-idf2(sensor_2),….}
    }
```

- **Task 3 Implementation:**
  The aim of task 3 was to generate heat maps for a selected gesture file. The gesture file was to be selected from the dataset provided. The heat map was a graph with the x – axis as time and y-axis as the sensor ID. Each cell in the heatmap corresponded to the word that occurred in the sensor at that time in the gesture. The intensity of the heat map was based on the TF, TF-IDF or TF-IDF2 value of the word.



The above image is an example heatmap for where the TF values for all the words in the gesture file named ‘test6.csv’ are considered. The sensor numbers are labelled from ‘0’ to ‘19’ in the graph. Each cell corresponds to the word that occurred at sensor ID in the y – axis at time in the x – axis. The scale on the side shows the corresponding intensity to the TF value of each word. The intensity of the grey increases as the TF value increases. Sections containing long bars of the same color indicate the same word occurring for that sequence of time.



TF-IDF
and

TF-IDF2 are values that give weight to the words by reducing the values for insignificant, frequently occurring words. In the TF-IDF heatmap, the sections shown in dark are words that are more significant in this gesture file as they do not occur frequently in other gesture files. In the TF-IDF2 heatmap, the sections should be considered for each sensor. Darker sections relate to words that occur in the respective sensor for that gesture, but do not occur as frequently in the same sensor for the other gestures. These heatmap values can be used to highlight more important words and sections in the gesture file with respect to sensor and time.

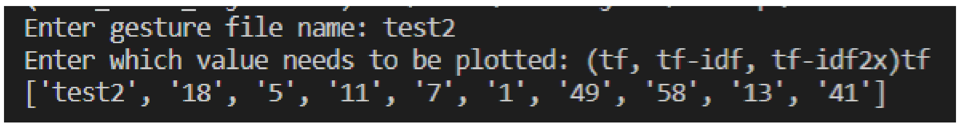
- **Task 4 Implementation:**
  The objective of task 4 was to take a gesture file from the provided dataset as input and print out the 10 most similar gesture files from the dataset. The similarity can be calculated based on TF, TF-IDF or TF-IDF values. The user could select the value type of their choice and the output would be a list of the ten most similar gestures. The gestures are represented using their file names in the dataset.

$$difference \sum_{1}^{s} \sum_{1}^{w} abs(query_{val} - ges_{val})$$

$$where\ s = number\ of\ sensors\ and\ w = number\ of\ words$$

$$similarity\ = \frac{1}{difference}$$

The difference between two gestures is calculated by taking the absolute value of the difference between the vector values of the words in the query gesture to those of the gesture being used for comparison. If there is a word that occurs in one of the gestures but not in the other, their corresponding value is added directly into the difference. All the differences between the words for each sensor is cumulatively added to give a total difference value. The similarity value between the two is the reciprocal of the difference value. The similarity values of the query gesture with respect to all the other gestures are calculated and a list is generated that stores the ten gestures with the highest similarity values in descending order.



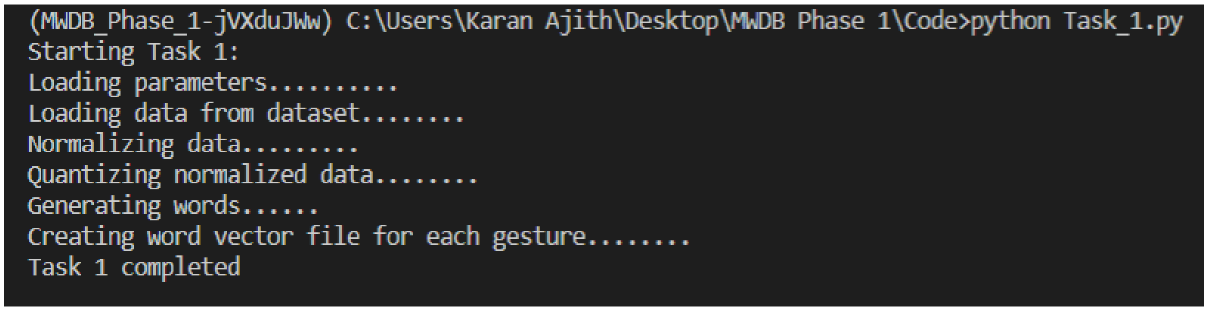
```
Enter gesture file name: test2
Enter which value needs to be plotted: (tf, tf-idf, tf-idf2x)tf
['test2', '18', '5', '11', '7', '1', '49', '58', '13', '41']
```

The above example uses the gesture file ‘test2.csv’ as the query and finds the most similar gesture files based on their TF values. Since test2 is a part of the dataset, it will come on top of the list as the gesture with the highest similarity.

Interface Specifications:

The interface for all four tasks are simple and basic. The interface for each task is explained below:

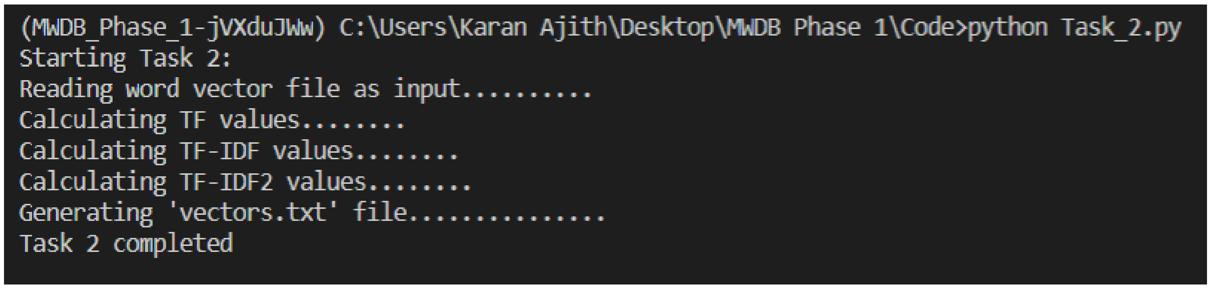
- **Task 1:**
  Task 1 will take the file 'param.json' as imput. This file contains the directory for the dataset, the resolution for the Gaussian bands, the window length for the words and the shift length for the next window.
  The data is loaded from the dataset for all gestures. The values are then normalized and quantized for each gesture and stored in intermediate files called 'normalized.csv' and 'quantized.csv'. These files are stored in the 'Output' folder. The window length and shift length are used on the 'quantized.csv' file to generate the word vectors. These final word vectors for each gesture are stored in the Dataset directory with the file name as 'f.wrd', where f is the gesture ID.
  Task 1 also creates a file named 'word_vector_dictionary.txt' in a folder called 'Extras' in the 'Code' folder. This stores all the words for all the gestures in a consolidated file that can be used by other programs.



```
(MWDB_Phase_1-jVXduJWw) C:\Users\Karan Ajith\Desktop\MWDB Phase 1\Code>python Task_1.py
Starting Task 1:
Loading parameters..........
Loading data from dataset........
Normalizing data.........
Quantizing normalized data........
Generating words......
Creating word vector file for each gesture........
Task 1 completed
```

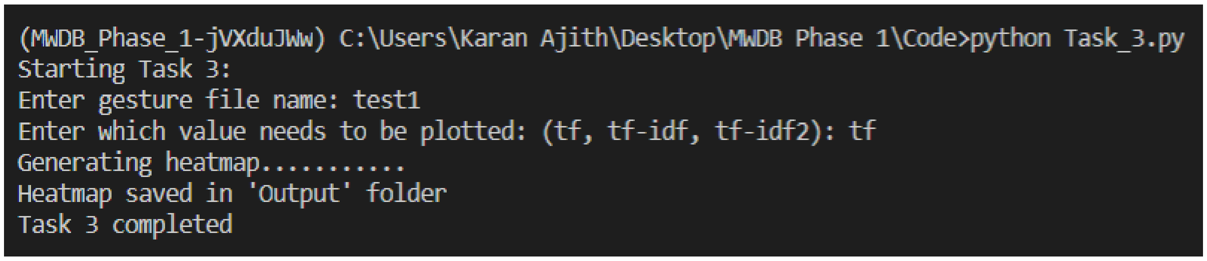
- **Task 2:**
  Task 2 uses the file named 'word_vector_dictionary.txt' to extract the words for each gesture. This is for ease of use as the file makes searching for the words easier. Once task 2 gets the data for all gestures, it starts to calculate the values.
  The program initially calculates the TF values for all (gesture_id, sensor_id) combination and stores them in a dictionary. The TF values are generated for each word and stored as a dictionary. The same is done when calculating the IDF and IDF2 values for all words for each (gesture_id, sensor_id) combination.
  These values are used to calculate and store the TF-IDF and TF-IDF2 values for each word. The final consolidated vectors for each gesture is stored in a file named 'vectors.txt'. This is saved in the same location as the dataset. The format in which the data is stored is of the form:

```
        Gesture_ID: {
                TF Vectors: {tf(sensor_1), tf(sensor_2),….}
                TF-IDF Vectors: {tf-idf(sensor_1), tf-idf(sensor_2),….}
                TF-IDF2 Vectors: {tf-idf2(sensor_1), tf-idf2(sensor_2),….}
            }
```



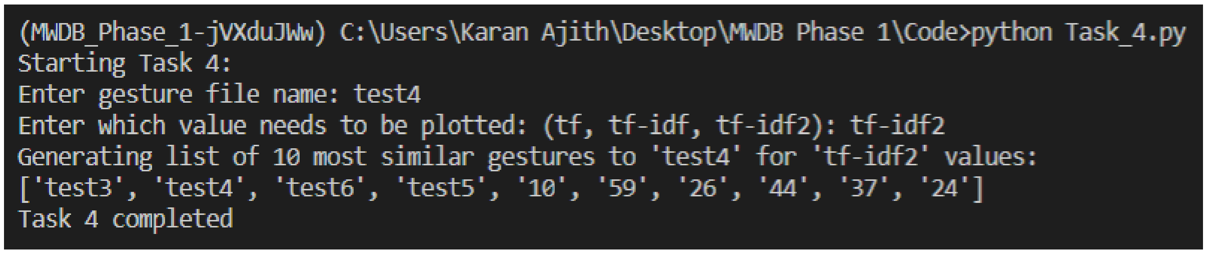
```
(MWDB_Phase_1-jVXduJWw) C:\Users\Karan Ajith\Desktop\MWDB Phase 1\Code>python Task_2.py
Starting Task 2:
Reading word vector file as input.........
Calculating TF values........
Calculating TF-IDF values........
Calculating TF-IDF2 values........
Generating 'vectors.txt' file..............
Task 2 completed
```

- **Task 3:**
  Task 3 takes a gesture file (f.csv) from the dataset directory and asks the user what value they would like to use for the heatmap. The program calculates the required value that is specified by the user for that gesture and uses the ‘seaborn’ python library to generate the final heatmap.



```
(MWDB_Phase_1-jVXduJWw) C:\Users\Karan Ajith\Desktop\MWDB Phase 1\Code>python Task_3.py
Starting Task 3:
Enter gesture file name: test1
Enter which value needs to be plotted: (tf, tf-idf, tf-idf2): tf
Generating heatmap..........
Heatmap saved in 'Output' folder
Task 3 completed
```

- **Task 4:**
  Task 4 also take the gesture file from the dataset as input and asks the user which value they would like to use to calculate the similarity with respect to the other gestures. Once these inputs are given, the program first extracts all the words for that particular gesture and gets the specified values for the respective gesture. This is then compared with all the other gestures in the dataset, including itself in the dataset using the similarity formula specified in the Implementation section of the report. The ten gestures with the highest similarity scores are saved and stored in a list. This list is finally displayed as the output for the program.



```
(MWDB_Phase_1-jVXduJWw) C:\Users\Karan Ajith\Desktop\MWDB Phase 1\Code>python Task_4.py
Starting Task 4:
Enter gesture file name: test4
Enter which value needs to be plotted: (tf, tf-idf, tf-idf2): tf-idf2
Generating list of 10 most similar gestures to 'test4' for 'tf-idf2' values:
['test3', 'test4', 'test6', 'test5', '10', '59', '26', '44', '37', '24']
Task 4 completed
```

System Requirements:

All for tasks were developed on Python 3.7.4.

During execution, the libraries required are:

- Pandas
- Numpy
- Sklearn
- Json
- Math
- Seaborn
- Matplotlib

Installation and Execution Instructions

Each task can by run as regular python files using a python interpreter. Task 1 and Task 2 do not require any console input and get their input from the files in the directory. The resulting output from both programs are files stored in the directory where the dataset is present. Task 3 and task 4 require the user to specify the file name of the gesture they wish to use as the query object and the values which they wish to use (TF, TF-IDF or TF-IDF2). Task 3 will generate the heat maps and save them in the 'Output' folder. Each heat map is named according to the gesture name and the values type used. Task 4 will only display a list in the terminal where it is run showing the file names of ten most similar gestures regarding the query gesture selected and the values considered.

The directory for the input, the resolution for the Gaussian bands, the window length for each word in the univariate dataset and the shift length are all stored in a file named 'param.json' and can be edited in this file. Those parameters are the ones that are used in all the tasks.

Related Work:

In ChaLearn 2013 data challenge [7], multiple teams participated to recognize Italian gestures. The data for that challenge was also similar to our data where each gesture's data consists of a time series of 20 sensors. To segment continuous data sequences into candidate gesture intervals, the majority of participants used a sliding window just as we did in this phase. The winning team used the Gaussian Hidden Markov Model to recognize the gestures. The second-ranked team used KNN and randomized decision trees to get posteriors which then will be converted to gesture predictions using some heuristic.

Conclusion:

This aim of this phase was to understand how to convert certain types of dataset to different types and how that changes the ease of understanding the data. The input dataset was a multivariate time-series dataset that was hard to understand and difficult to perform other data related tasks. This was converted to a univariate time-series dataset with words. These words were then used to calculate various data vectors for the gesture. The vectors were based on TF, TF-IDF and TF-IDF2 values. These vectors gave more information on the dataset and made it easier to compare the different objects in the dataset. Heatmaps were generated for each gesture file for the different values to better understand the information for each gesture. The vectors were also used to identify the most similar gesture objects to a selected one.

References:

1. Patro, S. and Sahu, K.K., 2015. Normalization: A preprocessing stage. arXiv preprint arXiv:1503.06462.
2. Ze-Nian Li, Mark S. Drew, 2003, Fundamentals of Multimedia, 8.4.2 Non Uniform Scalar Quantization, pg 204-205.
3. Manning, C.D.; Raghavan, P.; Schutze, H. (2008). “Scoring, term weighting, and the vector space model”t; (PDF). Introduction to Information Retrieval. p. 128.
4. Maheswari, S., R, RAMAKRISHNAN (2015); Indian Journal of Science and Technology; Sports Video Classification Using Multi Scale Framework and Nearest Neighbor Classifier
5. Sergio Escalera, Jordi Gonzàlez, Xavier Baró, Miguel Reyes, Oscar Lopés, Isabelle Guyon, Vassilis Athitsos, Hugo J. Escalante, Multi-modal Gesture Recognition Challenge 2013: Dataset and Results, Chalearn Multi-Modal Gesture Recognition Workshop, International Conference on Multimodal Interaction, ICMI, 2013.
6. Das G., Gunopulos D., Mannila H. (1997) Finding similar time series. In: Komorowski J., Zytkow J. (eds) Principles of Data Mining and Knowledge Discovery. PKDD 1997. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), vol 1263. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-63223-9_109
7. Jordi Abella, Raúl Alcaide, Anna Sabaté, Joan Mas, Sergio Escalera, Jordi Gonzàlez, and Coen Antens, Multi-modal Descriptors for Multi-class Hand Pose Recognition in Human Computer Interaction Systems, Chalearn Multi-Modal Gesture Recognition Workshop, International Conference on Multimodal Interaction, ICMI, 2013.
8. Per Ahlgren, CristianColliander (2008); Document–document similarity approaches and science mapping: Experimental comparison of five approaches. https://doi.org/10.1016/j.joi.2008.11.003
9. O. Shahmirzadi, A. Lugowski and K. Younge, “Text Similarity in Vector Space Models: A Comparative Study,” 2019 18th IEEE International Conference On Machine Learning AndApplications (ICMLA), Boca Raton, FL, USA, 2019, pp. 659-666, doi: 10.1109/ICMLA.2019.00120.
10. Lei Chen, Raymond Ng; 2004; Proceedings of the 30th VLDB Conference, pg 792-796

Appendix:

- **Roles of each Group members:**

All tasks were expected to be completed all the group members individually. Each member was given the freedom to tackle the problem in the way they felt was ideal due to the flexible nature of the phase.