Catholic University of Louvain - EPL

LINFO2275

Data Mining and Decision Making

Gestures Recognition

Group 7

Project made by:
- João Gomes, j.gomes@student.uclouvain.be
- Miguel Vicente, miguel.vicente@student.uclouvain.be
- Jacopo Palombarini, jacopo.palombarini@student.uclouvain.be

2022/2023
2ºSemester

Contents

1 Introduction **2**

1.1 Data description 2

1.2 Objective and analysis plan 2

2 Theorical notes **3**

2.1 Features 3

2.2 Dinamic Time Warping 3

2.3 $1 Recognizer for User Interface Prototypes 4

3 Implementation **4**

3.1 Exploratory analysis 4

3.2 Data Pre-processing 5

3.3 Main procedure 6

4 Model comparison and Results evaluation **6**

4.1 Comparison between different strategies 6

5 Conclusion **8**

6 Bibliography **8**

1 Introduction

In recent years there has been increasing interest in smart interfaces. Recent development in technology is push driving the integration of natural and intuitive sensors into everyday consumer devices such as Kinect in our homes. The use of such devices in commercial applications is setting the scene for the development of intuitive, natural, and people-centric applications over the next decades. Such people-centric devices combined with the ability of computational pipelines that can process information emanating from such devices in real-time to provide natural interaction portends a revolution for next-generation interfaces. Natural and intuitive interfaces based on hand gestures in the form of sketches and symbols can replace WIMP (windows, icons, menus, pointer)-based user interfaces.

This project aims to develop a hand gesture recognition system using three-dimensional tracking of hand movements with two different models. The system will be implemented and tested using the Python programming language.

The recorded hand movements are represented as a sequence of three-dimensional coordinates, denoted as the position vector r(t) = [x(t), y(t), z(t)]', where x(t), y(t), and z(t) represent the coordinates at a given time t. These sequences capture the user's execution of symbols using freehand drawings, which are then recorded and recognized by the system. The sketches in this project specifically represent the numbers from 0 to 9 and some geometrical shapes. Notice that the recorded time t has been disregarded for simplicity, assuming constant time steps (t = 1, 2, 3, ...) during gesture execution.

1.1 Data description

A database of sketches was used to evaluate the performance of the system. In total, 2000 sketch samples have been used.

The dataset consists of two domains. In the first one, ten users were asked to draw each of the numbers 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ten times, resulting in a total of 1000 matrices. The second one involves the repetition of drawing three-dimensional figures by ten subjects. The considered shapes are: cone, cuboid, cylindre, cylindrical pipe, hemisphere, pyramid, rectangular pipe, sphere, tetrahedron and toroid. Also for this second dataset we have a total of 1000 matrices.

1.2 Objective and analysis plan

Our goal in this study is to use 3D sketches coordinates drawn in the air to tune an efficient model able to classify symbols belonging to the two different domains (numbers and solid shapes). At first, we split the sketches database into a train set (70%) and a test set (30%). Then, two different models have been trained on the train set using the features computed on the sketch samples. Finally, the test set was used to obtain an objective assessment of the performance.

The work begins with the analysis and pre-processing of the data, including exploratory analysis and steps such as signal standardization and principal component analysis (PCA) to facilitate classification and comprehension of data.

For baseline techniques, the project incorporates dynamic time warping (DTW). The nearest-neighbour technique is then employed for classification. We also used, as a comparison, the $1 Recognizer method.

To assess the performance of the developed hand gesture recognition system, the classification accuracy has been evaluated using two different settings: user-independent and user-dependent gesture identification. Using cross-validation, these evaluations aim to measure the system's ability to identify gestures from new users and differentiate between different gestures

within the same user.

2 Theorical notes

2.1 Features

The most important factor in the success of any machine learning task is the selection of features. Features are quantifiable properties of the observed data that map it into an informative and discriminating feature space. Classification accuracy can be substantially improved and the amount of training data can be significantly reduced by using a well-chosen feature set. In fact, a good algorithm with real-time performance should extract a small set of relevant features and concurrently provide good classification accuracy (above 98%). Hence, it is important to choose a set of features, where each of them provides some new and relevant information.

2.2 Dinamic Time Warping

Dynamic Time Warping (DTW) is widely used in various applications, including speech recognition, gesture recognition, human motion analysis, biomedical signal processing, and more. Due to its flexibility in aligning temporal sequences of different lengths and shapes, the DTW algorithm has proven to be highly valuable in addressing pattern comparison and recognition problems in sequential data. DTW is an algorithm used to compare and align temporal sequences, commonly employed in data analysis and pattern recognition. It is an alignment technique that measures the similarity between two temporal sequences that may be non-linearly deformed or stretched/compressed. This algorithm computes the optimal alignment between two temporal sequences by minimizing a distance or dissimilarity measure. This alignment is achieved by searching for a "path" that connects elements of the two sequences, considering admissible constraints such as continuity and monotonicity.

In some cases, when the sequences are long or the computational resources are limited, it may not be feasible to perform DTW on the entire dataset. This is where the K-nearest neighbours (k-NN) approach comes into play. The k-NN algorithm is a simple and effective classification algorithm that is based on the idea of finding the k-closest training examples to a given test example and making predictions based on the labels of those examples. It is a non-parametric algorithm that does not make any assumptions about the underlying data distribution. In the context of DTW, the k-NN approach involves using DTW to compute the distances between a test sequence and a subset of training sequences (k nearest neighbours) instead of the entire training dataset. The k-NN algorithm then uses these distances to make predictions based on the majority vote or weighted voting of the labels of the nearest neighbours. By using the k-nearest neighbours in DTW, we can reduce the computational complexity and memory requirements of the algorithm while still leveraging the power of DTW for sequence comparison and classification. It allows us to make efficient use of the training data and achieve good classification performance even with large datasets.

The total cost c_p (X,Y) of a warping path p between X and Y with respect to the local cost measure c is defined as

$$c_p(X, Y) := \sum_l c(x_{nl}, y_{ml}), l = 1, ..., L \tag{1}$$

Furthermore, an optimal warping path between X and Y is a warping path $p*$ having minimal total cost among all possible warping paths. The DTW distance DTW(X,Y) between X and Y is then defined as the total cost of $p*$:

$$DTW(X,Y) = c_{p^*}(X,Y) = min\{c_p(X,Y) \mid p\ is\ an\ (N,M) - warping\ path\} \quad (2)$$

2.3 $1 Recognizer for User Interface Prototypes

We have a set of points C, as a user's gesture recording results. This algorithm determines which set of previously recorded template points Ti most closely matches with these points. Using Equation 3, a candidate C is compared to each stored template Ti to find the average distance di between corresponding points:

$$d_i = \frac{\sum_k \sqrt{(C[k]_x - T_i[k]_x)^2 + (C[k]_y - T_i[k]_y)^2}}{N}, k = 1, ..., N \quad (3)$$

Equation 3 defines di, the path-distance between C and Ti. The template Ti with the least path-distance to C is the result of the recognition. This minimum path-distance di* is converted to a [0..1] score using:

$$score = 1 - \frac{d_i^*}{\frac{1}{2}\sqrt{size^2 + size^2}} \quad (4)$$

In Equation 4, size is the length of a side of the reference square to which all gestures were scaled.

3 Implementation

3.1 Exploratory analysis

We have a dataset which is a list of 1000 matrices, each of them treated as a dataframe of 4 numeric variables, the coordinates x, y, z, and the time t. Each matrix has a variable length (number of observations).

Table 1

x	y	z	t
0.042075	0.036799	0.258380	6
0.041904	0.037187	0.258505	37
0.041739	0.037219	0.258619	67
0.041446	0.037573	0.258731	101
0.041303	0.037331	0.258830	132
0.040491	0.037559	0.258930	180

Table 2: An example of the variables' ranges

x	y	z	t
-0.050106	-0.069746	0.201506	28
0.030009	0.044953	0.217482	2219

Due to the ranges of the variables, a preliminary standardization of data will be necessary.

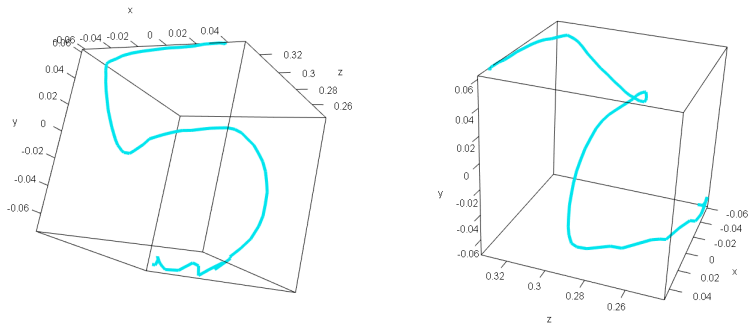


Figure 1: 3D representation of digit "5" from two different prospectives

3.2 Data Pre-processing

Each data point of the sketch comprises of three spatial coordinates (xi, yi, zi) and one temporal coordinate ti, where i denotes the index of the data points. Before computing the models, we passed the raw sketch data through some pre-processing steps. We first verified that there were any missing values in the data and, based on the graphic visualization of the coordinates' paths, we assured that an outliers analysis was not necessary. The sketches have been made by ten participants with ages between 20 years to 29 years and having an engineering background supervised while sketching the first symbol for each domain. Among the ten participants, five had no experience, using depth-sensing cameras in the past. From the following graphics in Figures 2, 3 and 4 we can distinctly see the difference between the experienced participants and the beginners (despite in the first graph, the one that considers the axes x and y, we can see that the subjects did quite the same).

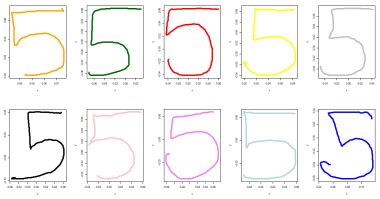


Figure 2: Axes x and y

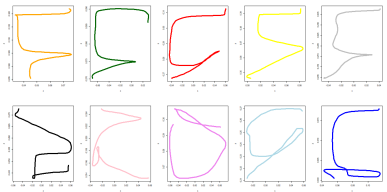


Figure 3: Axes x and z

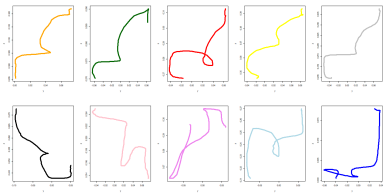


Figure 4: Axes y and z

In this particular case, we expect that, with a PCA, the most important variables in terms of "explained variance" are axes y and z (each digit has a different way to be drawn and, with this, also the importance of the variables in describing how that digit varies between subjects and tries, changes).

As we can see in Figures 5 and 6, for the first 5 subjects (the more experienced) the first 2 components have a higher relevance because they have been more "precise" in the movement. For the second domain analyzed, the one with 3D shapes, all three coordinates will have a similar relevance, while, for the digits, x and y axes are clearly more important. And that's also why, as we will see, we obtained better results with the first domain using the $1 Recognizer algorithm.

Then, we computed a z-score standardisation (dividing each variable by its mean and then dividing by his SD). Standardizing the data before PCA helps to ensure that variables with different scales or units are treated equally. It can also improve the interpretation of principal components and make them comparable.

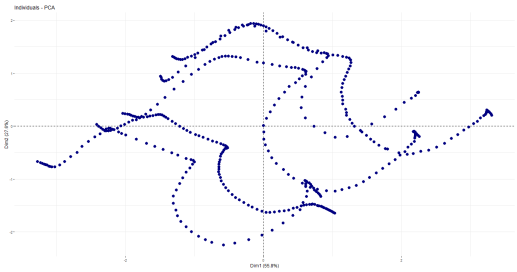


Figure 5: PCA for the first 5 Users (Axes x and z resulted more important components)

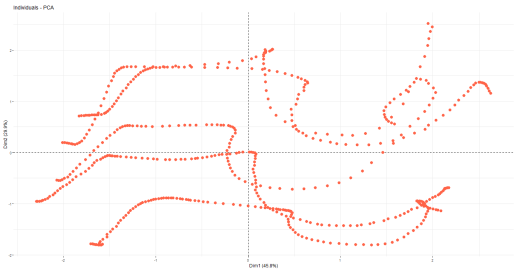


Figure 6: PCA for the first 5 Users (Axes y and z resulted more important components)

3.3 Main procedure

1. Splitting the dataset: We started by splitting the dataset into a train set and a test set. We used a 70%/30% split (70% of the data is used for training and 30% for the testing).

2. Cross-validation on the training set: We performed cross-validation on the train set using two different techniques:

   - The leave-one-user-out procedure (user-independent): We split the train set into a smaller train set (9 out of 10 users) and a smaller test set (1 out of 10 users). The model has been evaluated on each single user.
   - The leave-one-gesture-sample-out procedure (user dependent): For each user, we split the data into a smaller trainset (90% of the user's data) and a smaller test set (10% of the user's data). The model has been evaluated on different subsets of the training data for each user.

   We used the user-independent procedure to find the best model (based on accuracy level) and once we found the one with the best score we compared the two procedures to understand which one gave us better results.

3. Testing the model: After performing cross-validation and training the model, we tested it to evaluate the model's performance. This allowed us to assess how well the model generalizes to new, unseen data.

4 Model comparison and Results evaluation

For both domains, we compared the two models using some different approaches in order to determine the best strategy to adopt. What follows is a comparison between these approaches and, as a result, the best model to use. Then, with this model, we compared both the user-dependent and user-independent cross-validations. We reported a table for the comparison and a confusion matrix, that better helped to understand the model's accuracy. Note that for the $1 recognizer, we always used PCA before tuning the model because the algorithm itself is made to work on 2D data. (reformulate looking at the theory in the paper).

4.1 Comparison between different strategies

For every approach, we obtained a table like the one that follows (Table 3).

Table 3: Cross Validation (70% of Data) and Model Result (30% of Data) for Domain 1, using DTW with standardization and PCA before computing the algorithm

Domain 1							
User Independent				**User Dependent**			
User 1	0.91	**User 6**	0.64	**User 1**	0.91	**User 6**	0.64
User 2	0.89	**User 7**	0.93	**User 2**	0.89	**User 7**	0.93
User 3	1.0	**User 8**	0.94	**User 3**	1.0	**User 8**	0.94
User 4	1.0	**User 9**	1.0	**User 4**	1.0	**User 9**	1.0
User 5	0.91	**User 10**	0.9	**User 5**	0.91	**User 10**	0.9
Model Score							
0.97							

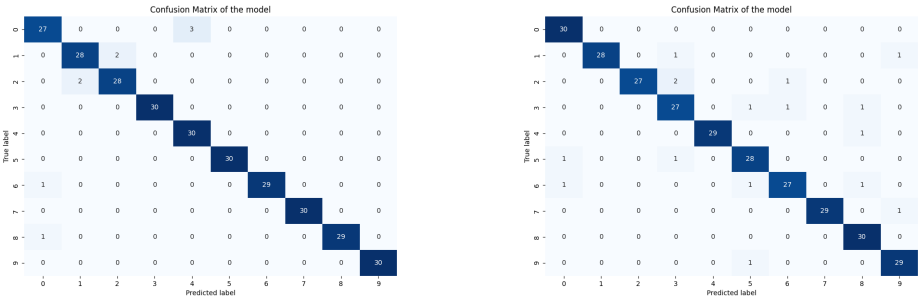
1. We report only the best results, we tried some more combinations with Standardization or not and with PCA or not for both domains and for both the models.
2. The percentage for the cross-validations represented in the following table is the average of the results we obtained for all 10 Users and for simplicity we report here only the score for the user-independent method

We resume here the results for every strategy.

		CV accuracy (%)	**Test accuracy (%)**
Domain 1	Std and no PCA with DTW	0.91	0.97
	PCA and no Std with $1 recognizer	0.91	0.95
Domain 4	Std and no PCA with DTW	0.87	0.96
	PCA and no Std with $1 recognizer	0.74	0.73

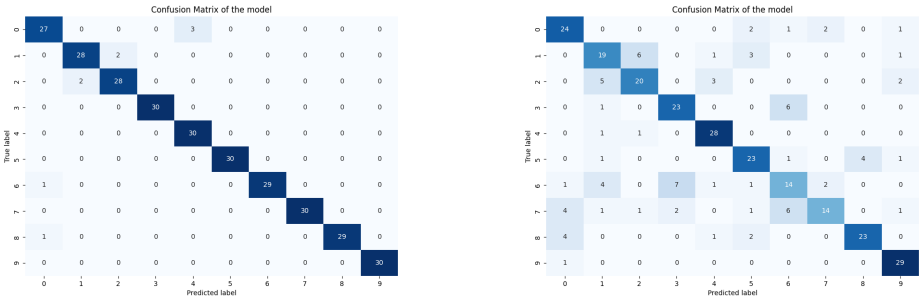
Given these results we decided to proceed with the DTW algorithm for both the domains and we computed the cross-validation for both the user-dependent and independent methods, obtaining the exact same result.

Here are the confusion matrices for both methods.



(a) DTW model (b) $1 Recognizer model

Figure 7: Domain 1 Confusion Matrix



(a) DTW model (b) $1 Recognizer model

Figure 8: Domain 4 Confusion Matrix

5 Conclusion

Before concluding, we have to mention the limits of the analysis. We didn't obtain a real distinction, in terms of results, for the two cross-validation methods, it would have been interesting discriminate which one of the two works better. Some procedures of data smoothing (with a Gaussian filter, for example) and other pre-processing procedures like, for example, eliminating the very first and very last values of the time series (that are not really part of the hand movements) could have helped to obtain more precise results. We also have to remember the limitations of the $1 Recognizer, which is a simple technique. The $1 recognizer is a geometric template matcher, which means that candidate strokes are compared to previously stored templates, and the result produced is the closest match in 2-D Euclidean space. To facilitate pairwise point comparisons, the default $1 algorithm is rotation, scale, and position invariant. While this provides tolerance to gesture variation, it means that $1 cannot distinguish gestures whose identities depend on specific orientations, aspect ratios, or locations.

Upon analyzing both models, it has been determined that, in the first domain, the DTW model yielded slightly superior results. However, it is worth noting that the DTW model also requires significantly more time to execute, resulting in considerably higher computational overhead. Therefore, if prioritizing accuracy is paramount, selecting the first model would be advisable. On the other hand, if speed is of greater importance, opting for the second model would be more suitable. In domain 4, the first DTW model demonstrated significantly superior results despite its lengthier computational time. Therefore, it is strongly recommended to select this model due to its considerably better performance.

6 Bibliography

1. Gesture-based system for next generation natural and intuitive interfaces, Jinmiao Huang1, Prakhar Jaiswal2 and Rahul Rai2.

(https://www.cambridge.org/core/journals/ai-edam/article/gesturebased-system-for-next-generation-natural-and-intuitive-interfaces/F429C70029095C3AB32FFEB28F8F68CB)

2. Gestures without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes

(https://dl.acm.org/doi/10.1145/1294211.1294238)

3. Dynamic Time Warping

(https://www.ccs.neu.edu/home/yzsun/classes/2013Fall$_C$S6220/*slides*/*DTW.pdf*)