UCLouvain

École Polytechnique de Louvain

Academic year 2021 – 2022

LINFO2275 - Data Mining and Decision Making

Project: Gesture Recognition

Nepper Nathan - 1428 1800
Warnauts Aymeric - 8703 1800

May 17, 2022

Contents

1 Introduction

The main goal of this project is to implement a hand gesture recognition system which work with a recording of a sketch as input and that identify the category of the sequence. We use a three-dimensional tracking of the position vector $r(t) = [x(t), y(t), z(t)]^T$ which represent the spatial coordinates of the user hand in function of the time to train our model. For this project we will classify dynamic 3D hand movements of users.

As data set to train our implementation we will use the combined symbol experimentation of ten users each of them sketching ten times the numbers $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Based on the times series classification theories we will perform an analysis of the signals by applying the Dynamic time warping algorithm. This method is usually needed for pre-processing the data and is used to tackle sound pattern recognition or stock market issues.The DTW is a technique that allows to compare time series data in situation where the time indices between comparison sequences do not synchronize up perfectly.

The SVM implemented in the second part of this report has been chosen to use an optimal number of features based on the trade-off between classification accuracy and real-time performance. The SVM model with kernel function will be trained before turning on the cross-validation to tune the parameters C (the regularization parameter) and γ (the Kernel coefficient) to their optimum values. Then, the model will classify the samples in the test set. Finally, we will evaluate the mean test set accuracy of our model to interpret the ability of the SVM classifier to handle with the intra-class variation in drawing habits of users.

2 Dynamic time warping

2.1 Motivations

Comparing two time series is usually performed by using a distance metric between the sequences after the vector quantization of the data matrix. This algorithm is very useful in measuring the similarity of two temporal signals and this even if the speed of observations isn't linear over the time. Matching and comparing index of two sequence the method aims to find an optimal match, denoted by a minimal cost that make reference to the sum of absolute differences between the spatial coordinates of the sequences.

2.2 Distance computation

The algorithm performs a non-linear warping on data to ensure potential alignment of signals in addition to the similarity measure. We will use this warping over time to warp the compared series so that we can match them together.

Listing 1: Dynamic Time Warping algorithm

```
def distance(s1: array[1..n], s2: array[1..m]):
    DTW = array[0..n, 0..m]
    for i in 1 to n:
        for j in 1 to m:
            DTW[i, j] = Infinity
    DTW[0, 0] = 0
    for i in 1 to m:
        for j in 1 to m:
            cost = d(s1[i], s2[j])
            DTW[i, j] = cost + min(DTW[i-1, j], DTW[i, j-1], DTW[i-1, j-1])
    return DTW[n, m]
```

The DTW develop a one-to-many match so that the mapping of the peaks of similar signals are well displayed over time through shifting time. However, we encounter a problem because the algorithm allows that one element of a sequence matches an unlimited number of elements from the other sequence. So, in order to quantify a threshold for our index comparison we will impose a window parameter as constraint. That is a condition that ensure that if $a[i]$, which is the index i of the first sample, and $b[j]$, which is the index j of the second sample, are matched, then $|i - j| < w$ where w is the window threshold. In fact, this guarantees all indices can be matched up.

(graphe de la marge d'erreur en fonction de la taille de la window)

(graphe de la vitesse de mouvement en fonction de l'accuracy)

2.3 Modelling

As our classifier, we decided to implement a standard k-nearest neighbors classifier that exploits the Dynamic Time Warping rather than the Euclidean distance as the distance measure to find them. Therefore, we computed the pair-wise distance of each sequence of the training set to the sequence we wan't to predict the label.

Once the pair-wise distance matrix is computed, we reach the $n\_neighbors$ closest sequences from the training set to the test set, and we apply a voting process that will return the most frequent label in the neighborhood.

This process is guaranteed to be efficient as long as the training set is large and the distance metric is efficient, the DTW guarantees the efficiency because two similar sequences observations will result in a small distance. In that way, we construct a powerful classifier robust to the variability in sequence length.

2.4 Results

We expect to observe a falling in the accuracy in the predictive classification of the DTW-KNN algorithm. In fact, if k selected is too large, the constructed model becomes too generalized and fails to predict with high accuracy the sequences points in both train and test set.

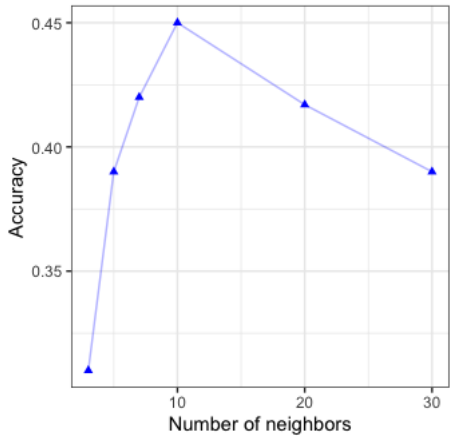

Figure 1: Accuracy variation with the number of neighbors

As we predicted previously, we denote the beginning of the accuracy score falling for more than 10 nearest neighbors computed. Increasing the number of k, the area predicting each class will be more smoothed because the algorithm refers to the majority of the k-nearest neighbors for classification. That's why when increasing the number of neighbors we decrease the complexity until the model faces underfitting.

3 Support Vector Machine

3.1 Motivations

The so-called support vectors are the coordinates of the individual temporal observations and the SVM classifier allows to find the non-linear hyperplane that best segregates classes by maximizing the distances (Margin) between the nearest data point and this hyperplane. We will train the SVM on our train set to classify new spatial sequences using the features selected. In that way, the SVM algorithm solve the dimentionality issue of non-linear hyperplanes by using a kernel function to export the points displayed in a low dimensional input space to a higher one.

3.2 Preprocessing

In order to perform the sketch classification algorithm, the recorded points of sketches were preprocessed to eliminate noise and resample the sketch with equidistant points. Performing this smoothing on the whole sketch is not considered because of the sharp turns of the sequences defining the aspect of the sketches. We will filter selected parts of the signals which are considered as outliers and unwanted self-interactions zones. In that way, we will perform dehooking to avoid human response time error by removing the first 3 points of the sequence and Gaussian filtering on the start and the end of the sketch in order to smooth this sequence. We will use the same filter to eliminate outliers. The threshold to determine if a spatial coordinate is an outlier is computed using a percentage of the diagonal of the PCA-based bounding box enclosing the sketch.

We will set the percentage of the distance represented by the length of this diagonal as a hyper-parameter in our model to compare our model accuracy in function of the threshold chosen.

(graphe threshold)

We will also use linear interpolation, that's a method of curve fitting to construct new data points, between the points to re-sample our spatial coordinates sequences. This is done to transform the non-uniformly spaced points in a N-equidistantly spaced points. This algorithm is performed by adapting the "1$ recognizer" algorithm and we will determine N, the number of resampled points prior to the resampling.

3.3 Features extraction

The smoothed sketch is then used to extract feature vectors; for instance speed, curvature, distance and angle features. After this extraction, the SVM classifier is trained using the features on the training set and so the feature vector obtained is used as base for the learning of the SVM classifier. The feature-based classification is used to ensure margin robustness of the classifier and thus reduce the chance of miss-classification.

So, the goal is to find the features that give the smallest expected generalization error we will keeping in mind to avoid over-fitting and reduce the training time. In fact, discriminating the sketches in an optimized feature space will be easier for our classifier than using the raw data at this end. In order to be able to cope with generalized sketches, we will be looking for features that are invariant to translation, rotation and scale of the sketch but we have to keep in mind that the features are dependent on drawing direction.

- Speed and curvature features; This two features are correlated because of the link that exists between the drawing speed of sketches and the sharpness of corners.

  The speed is smoothed with a Gaussian filter to remove noise and we compute the features $f_{s1} = \sigma_s/\mu_s$, $f_{s2} = s_{per90}/\mu_s$, $f_{s3} = s_{per10}/\mu_s$ after normalization.

  The curvature at a point of the sketch is the inverse of the radius of the fitted arc in the local neighborhood of that point and as we consider the similarity of curvature in identical shapes this features aren't normalized.

- Distance features; Assuming that the points of a sketch are ordered, we will compute a robust descriptor, the Distance Matrix D with $D_{i,j} = ||p_i - p_j|| \quad i,j \in 1,..,N$ where p_i represent the spatial coordinates (x_i, y_i, z_i). After this construction, we compute averaging portions of the matrix to reduce the redundancy. So we obtain for the ADM[1] $\bar{D}_{uv} = \frac{1}{M^2} \sum_{i=(u1)M+1}^{uM} \sum_{j=(v1)M+1}^{vM} D_{i,j}$ where $u,v \in 1,...K$ if we know that the DM is subdivided into $K \times K$ sub-matrices of size $M \times M$. Once again it's not necessary to normalize the features because it has been done in the scaling step of the preprocessing and so we stay in a range {0,1}.
- Angle features; We use the direction vector joining the current to the next point to calculate the $N \times N$ Angle Matrix A as $A_{ij} = \frac{1}{\pi} cos^{-1}(\frac{q_i.q_j}{||q_i - q_j||}) \quad i,j \in 1,..,N$ where q_i represents the vector joining i th point to its next point. For the last point, the vector at the penultimate point is used. Once again, we compute the $\bar{A}$ (AAM)[2] in the same way that for the Distance features to avoid redundancy.

3.4 Modelling

Our SVM will split data into n-classes using the binary {-1,1} label that indicates if the point is in or out the set that we want to learn to recognize and as said previously, using kernels we can handle with non-linear separation of data. For training and evaluating the performance of the SVM classifier we will use a leave one-user-out procedure for our cross-validation on the data by using 90% of the data from 9 users for fitting the model and we will test the model on the remaining 10% for generalization. We will repeat this procedure 10 times, leaving each time a different user out of the training set to avoid user dependence. We will use the results to obtain a accuracy measure of the performance of the system.

Unfortunately, the SVM doesn't perform as expected, with a performance slightly better than a random classifier.

3.5 Results

As pointed out, the SVM based methods doesn't perform as intended; the low classification accuracy can come from several sources.

First of all, our preprocessing may have issues, with a bad heuristic regarding the outlier detection and smoothing methods used. However, when comparing the initial signal to the preprocessed one, the sequence seems to be in overall a good depiction of the symbol the user tried to represent. In fact, we cleaned the two tails of the sequence in order to avoid the hooking phenomenon and removed outlier in regard to the overall shape, without removing inflicting point that define particular (angular) shape.

Then, the problem may be due to the feature creation, which is the keystone of the whole algorithm. We defined geometric features that aimed at characterising the shape of the sequence based on curvature, speed, distances and angles. Without digging in the math behind, those features are intuitively interesting because a symbol (such as a numeral) is nothing more than carefully timed angle and curvature in a sequence of points. However, those features don't result in a gain of information for the classification.

At the end, the classifier isn't efficient but we may improve the result by experimentation of novel techniques. In fact, we may try to project the sequences on the two first principal components in order to remove the last dimension, which is intuitively not useful for characterising a 2D symbol. By doing so, we remove the complexity of doing numerical geometry in 3D, which as said previously, may be our source of problems.

Finally, we may also assume the 3D as necessary and therefore investigate the 3D numerical geometry realm in order to tackle this challenging yet interesting problem.

[1] ADM = Averaged Distance Matrix
[2] AAM = Averaged a Angle Matrix

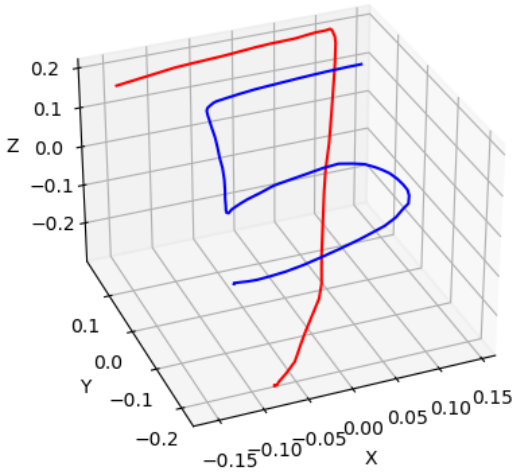

Figure 2: Comparison of sequences

The visualisation of several numbers and symbols in 3 dimensions is useful to understand our algorithm issues and to confirm that the alternative methodical approaches will ensure us more optimal results.

4 Conclusion

To conclude, we have a working Dynamic Time Warping implementation that satisfies our expectation regarding the efficiency. By cross-validation, we found a optimal number of k-neighbors of 10 that maximize the accuracy score.

As said in the result of our two algorithms, our result are interesting and we have had the ability to build a classifier tackling the preprocessing, the features selection and the classification. Unfortunately, we haven't succeeded in the SVM implementation due to the problems previously explored.

Finally, there is a lot of area of improvement in the implementations, starting with the DTW computing time. In fact, we can parralelize the computation of the distance matrix in order to speed-up the prediction time of our classifier, which isn't practicable when treating a lot of sequences. Finally, there is a possible improvement in the SVM classifier regarding the creation of the matrix; those improvements may be outside the scope of the course but are nevertheless interesting and meaningfull regarding the sequence embedding.