UNIVERSITÉ CÔTE D'AZUR

Final Assignement

Student name: *Jeremy Gomes, Mohamed Adly Zaroui*

Course: *Statistical Computational Methods* – Professor: *Dr. Martinet Jean*
Due date: *December 08th, 2022*

Task presentation

Our data consists of 320 event files. An event file is a dataframe where rows are events, and an event is 4-tuple (x, y, p, t), where (x, y) denotes the pixel position related to the event, p is the polarity of the event (= 1 or 0 whether the brightness has increased or decreased) and t is the timestamp : time (in μs) since the begining of the recording, at which the event has been detected.
This dataset is provided by Prophesee Event camera. This type of sensor is also known in literature as Dynamic Vision Sensor, and has large application in computer vision, specially where refresh rate has to be high (of the order of μs) We have 10 classes, 32 examples for each class, and our task is to classify new examples, by any mean.

Data pre-processing

For us, one example is a dataframe of 4 columns and a variable number of rows. The main problem is to format this dataframe such that all exemples have same number of entries.
First, we have aggregated the data[1] : each 200.000 μs, we have stacked the pixels positions of the events that occured in that interval of time (with respect to their polarity) in a matrix of shape $(640, 480)$. By this way, one example is now an array of shape $(30, 640, 480)$.
Then, in order to reduce dimension, we resized each $(640, 480)$ array into shape $(160, 120)$ using openCV and considering it as a conventional fame.
Finally, we vectorized this $(30, 160, 120)$ array into a vector of length $30 \times 160 \times 120 = 576\,000$ and we put each one of the 320 examples as rows in an array of shape $(320, 576000)$.

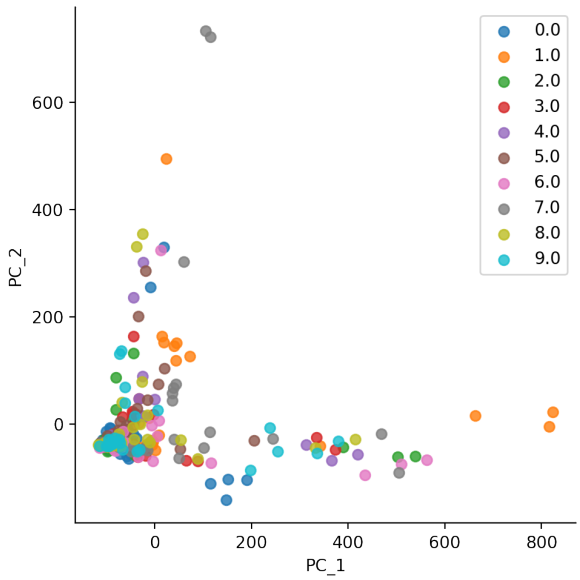
That's how we built X_{train}, X_{test}, y_{train} and y_{test}.

Clustering

Before applying PCA, we can note that the first 2 components summarise 0.25 (cf. ipynb file) of the variance of the data, and that therefore, a 2D visualisation of a draft clustering is possible.

[1] We have used and modified a function that we found on this kaggle notebook : https://www.kaggle.com/code/gogo827jz/generating-superframes-from-dvs-event-data/notebook

Here is how the examples are distributed on the plane whose base is formed by the first 2 principal components:



PCA Dimension Reduction

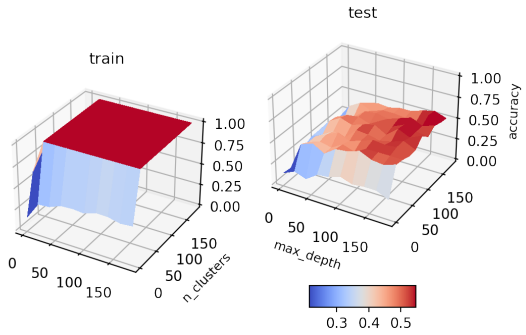
As the outcome of pre-processing, one example is for us en element of a 576 000-dimensional (vectorial) space. For dimension reduction purposes, we applied a PCA on our subsets. We selected the number of components such that it explains 0.99 of the variance. we have gone from a 576 000-dimensional space to a 214-dimensional space, a reduction of 2700 on the dimensions.

KMeans Clustering

We performed a KMeans clustering on our reduced data with $k = 10$. We obtained a Kmeans score of -3.772.379. It is the opposite of the K-means objective (positive sum of Euclidean distances with respect to the centroids, also called inertia). Accuracy is 0.125. However, it can be improved by other classifiers, which we will see in the following sections.

Random Forest Classifier

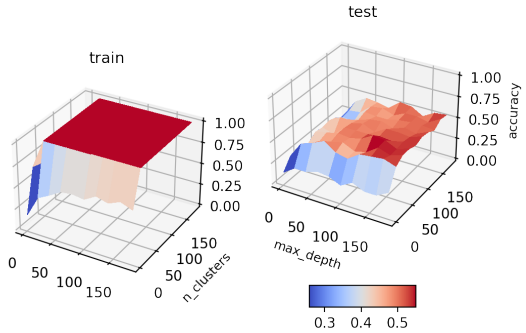
We trained a Random Forest model on raw X_{train} (not transformed by PCA).
We can see in the plot that $max\_depth \geq 150$, $n\_clusters \geq 100$ is sufficient for a 0.6 accuracy.
We suggest to keep the smallest values of these parameters that realizes that maximum accuracy, to avoid the risk of overfitting.



$(max\_depth_{opt}, n\_clusters_{opt}) = (150, 100)$ and realizes an accuracy of 0.6
Training time : 1159s (20min). It includes hyperparameters tuning : max_depth and $n\_clusters$ both varies from 1 to 200, with a step of 20.

Random Forest Classifier on reduced data

We then trained a Random Forest model on the PCA reduced data.
We can see in the plot that $max\_depth \geq 100$, $n\_clusters \geq 150$ is sufficient for a 0.65 accuracy.
We suggest to keep the smallest values of these parameters that realizes that maximum accuracy, to avoid the risk of overfitting.



$(max\_depth_{opt}, n\_clusters_{opt}) = (100, 150)$ and realizes an accuracy of 0.65
Here, training time is much more smaller : 40s. It includes hyperparameters tuning : max_depth and $n\_clusters$ both varies from 1 to 200, with a step of 20.

Comments and conclusion

This work gives us the opportunity to carry out a quantitative study of a classification task with huge dimensional data and gives us a first step into event data handling.
In particular:

- The events data is not video and conventional image/video classification algorithms do not work, unless a heavy pre-processing.
- Dimension reduction using PCA, although simple to implement, allows a strong compression rate (therefore a much faster computational time), and slightly improves performance.

- As PCA, despite its simplicity of implementation, Random Forest is an efficient classifier.
- To go further, we could have considered treating the event data as they are: multivariate time series : $t \mapsto (x, y, p)$ and our problem would amount to a multivariate time series classification problem.

Thank you for reading