Machine Learning in Applications
GU02 Project Report - Bioinspired Temporal Data

Stefano Rainò **s282436**, Alberto Castignanò **s281689**, Lorenzo Mercurio **s290903**
Politecnico di Torino

27/04/2022

Contents

List of Figures

List of Tables

Abstract

Human activity recognition, or HAR for short, is a broad field of study concerned with identifying the specific movement or action of a person based on sensor data. In this project we want to verify whether the classification performances, in terms of parameters, energy required and accuracy, related to temporal data collected by wearable devices, can improve following the application of meta-classification techniques. They refer to the possibility of using clustering in order to better separate our data, and subsequently carry out the classification through the use of RNN networks such as sLMU. In order to carry out this type of study, we relied on the use of the WISDM dataset and carried out the optimization of the hyperparameters through NNI, a Neuromorphic algorithm for enabling the usage of compatible neuromorphic HW, and modular ML architectures for enabling the heterogeneous composition of our ML model.

1 Introduction

Human Activity Recognition or **HAR** is the process of interpreting human motion using computer and machine vision technology. Human motion can be interpreted as activities, gestures, or behaviors which are recorded by sensors.

Thanks to the wearable devices increasingly present in our daily life, **Body Sensor Networks (BSNs)** consisting of a number of body-worn sensor nodes wirelessly collaborating can be shrunk down to a single device. Typically, however, devices of this type are affected by important limitations related to their power. Consequently, one of the aspects considered within this study was the possibility of reducing the computational effort, in order to predict the possibility that they can process the data received, in real time.

Considering this aspect, **Spiking Neural Networks (SNNs)**, thanks to their event-based asynchronous operations, could represent a valuable candidate for energy-efficient solutions. Therefore, in order to carry out the final classification process, we have chosen to rely on the **Legendre Memory Unit (LMU)** in its spiking version. It represents a type of RNN capable of carrying out a typical approximation of temporal cells, modelling in an appropriate way the transmission and filtering of spikes through the synapses of our network.

In addition, in this project we want to explore **meta-classification techniques** for the single analyzed data. It means that instead of having an ensemble model made of homogeneous components, we would like to explore heterogeneous components to perform part of the classification.

The dataset used is **Wireless Sensor Data Mining (WISDM)**, which contains accelerometer and gyroscope time-series sensor data collected from a smartphone and smartwatch as 51 test subjects perform 18 activities for 3 minutes each.

Instead, for parameter optimization, we used the **Neural Network Intelligence (NNI)** framework, which provides tuners to speed up the process of finding best hyperparameter set, and also, a web portal to monitor training progress, to visualize hyperparameter performance, to manually customize hyperparameters, and to manage multiple experiments.

2 Related works and Background

2.1 Human activity recognition: suitability of a neuromorphic approach

HAR is a time-dependent task whose application is extended to all the aspects of human life. In [**1**] the authors, starting from the HAR problem, proposed a strategy to identify the optimal solution given a target application. They shown how a neuromorphic approach can be adopted to deal with time-varying inputs. After used multiple neural networks, the authors demonstrate that using the spiking implementation of the LMU carried out the optimal solution to achieve accuracies with low energy consumption.

2.2 Cancer miRNA biomarkers classification using a new representation algorithm and evolutionary deep learning

Several studies have verified that there is a correlation between miRNAs and cancer, consequently in [**2**], what has been done is the incorporation of machine learning methods to extract knowledge, which can be significant, from cancer genomic datasets. Therefore, to do this, they proposed a deep learning **superclass/meta-label** approach consisting of three different phases:

- **Partitioning data into super-classes**: the cancers in the same super- class would have a maximum of separability and minimum of class overlap with the cancers from other superclasses.
- **Meta-data creation**: for each classifier, the samples belong to the category of interest, are labeled as positive and the others have negative labels.
- **Super-classes classification**: After the features extraction from the meta-label, the appropriate CNN model is used to predict the actual label of the sample.

This approach therefore induces each sample to be first classified in its corresponding super class, and subsequently a CNN is used to predict the label associated with the sample. In this project we based on this approach, wanting to verify if performance could be improved in accordance with the previously mentioned use of sLMU.

2.3 Legendre Memory Unit

The **Legendre Memory Unit (LMU)** is a novel memory cell for recurrent neural networks that dynamically maintains information across long windows of time using relatively few resources. The main property of the LMU network is the capability of decoding a delayed signal $u(t-\theta)$, contained within a sliding window of length θ, through a high-dimensional projection of the input $u(t)$ that is orthogonalized using the shifted Legendre polynomials.

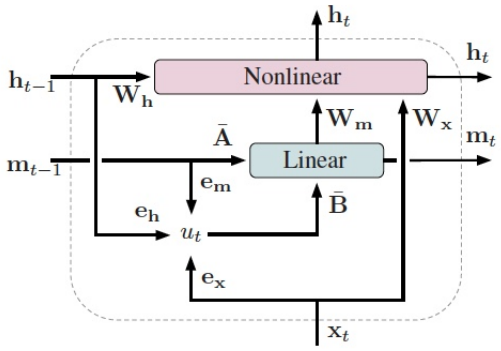

Figure 1: Time-unrolled LMU layer. See [**3**] for more

The main component of the Legendre Memory Unit (LMU) is a memory cell that orthogonalizes the continuous-time history of its input signal, $u(t) \in R$, across a sliding window of length $\theta \in R_{>0}$. The cell is derived from the linear transfer function for a continuous-time delay, $F(s) = e^{-\theta s}$, which is best-approximated by d coupled ordinary differential equations (ODEs):

$$\theta m'(t) = Am(t) + Bu(t)$$

where $m(t) \in R^d$ is a state-vector with d dimensions. The ideal state-space matrices, **(A, B)**, are derived through the use of **Padé approximants**:

$$A = [a]_{ij} \in R^{d*d},$$

$$a_{ij} = (2i+1) \begin{cases} -1 & i < j \\ (-1)^{i-j+1} & i \geq j \end{cases}$$

$$B = [b]_i \in R^{d*1}, \quad b_i = (2i+1)(1)^i, \quad i, j \in [0, \text{d}-1]$$

In this project we wanted to analyze the behavior of the network, on clustered data, both in its spiking version and in its non-spiking version, to verify if this version could actually be a candidate to implement solutions that are energy efficient. More details about LMU networks could be found in [3]

3 Dataset configuration and KMEANS clustering

3.1 Dataset

The dataset used for this project is **WISDM** [**4**], which contains accelerometer and gyroscope time-series sensor data collected from a smartphone and smartwatch. Each activity was performed for 3 minutes, so that each subject contributed 54 minutes of data. These activities include **basic ambulation-related activities** (e.g., walking, jogging, climbing stairs), **hand-based activities of daily living** (e.g., brushing teeth, folding clothes), and various **eating activities** (eating pasta, easting chips). The data set contains the low level time-series sensor data from the phone's accelerometer, phone's gyroscope, watches' accelerometer, and watches' gyroscope. All of the time-series data is tagged not only with the activity that was being performed, but with a subject identifier, which means that the data be used for building and evaluating biometrics models, as well activity recognition models. The 18 activities presented in the dataset are: Walking, Jogging, Stairs, Sitting, Standing, Typing, Brushing Teeth, Eating Soup, Eating Chips, Eating Pasta, Drinking from Cup, Eating Sandwich, Kicking (Soccer Ball), Playing Catch w/Tennis Ball, Dribbling(Basketball), Writing, Clapping and Folding Clothes.

3.2 KMEANS clustering

Once we collected the data associated with each single class, we tried to cluster the entire dataset in order to identify optimal separations for the distinctions between the various super classes. To do this we have chosen to use the **KMEANS clustering** algorithm, which is a method of vector quantization, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster. By default we have chosen as possible values, relating to the number of reference clusters, a list that includes: 4, 5, 6, 7, 8. This is because, once clustering has been carried out, we were able to judge the quality of clustering through an evaluation of **Pareto front** type, based on two evaluation values:

- **Global score**: The sum of the differences between how much a class is represented in a cluster, compared to the most represented class in the same cluster
- **Global median**: The median value of the individual differences between how much a class is represented in a cluster, compared to the most represented class in the same cluster

After carrying out the clustering process, the evaluation through Pareto front found that the best values to be associated with the number of clusters (among those proposed) are 5 and 6, as can also be seen from Table 1. On the other hand, the visualization of the cluster operation, can be appreciated through Figure 3. After the clustering we have noticed that the assignments for each class are the same even if we have different numbers of K.

It means that, if we put:

- $K = 5$: there is one cluster, among the five present, for which no class is represented in a majority manner
- $K = 6$: there are two clusters, among the six present, for which no class is represented in a majority manner

Also, the remaining clusters have the same assignments for either K=4, K=5 or K=6. For this reason we have choose to set $K = 4$.

Taking into account the considerations made so far, the assignments made for each individual cluster include:

- **Cluster 1**: Sitting, Eating Chips, Clapping
- **Cluster 2**: Jogging, Eating Pasta, Kicking (Soccer Ball), Playing Catch w/Tennis Ball, Dribbling (Basketball)
- **Cluster 3**: Drinking from Cup, Eating Sandwich, Writing, Folding Clothes

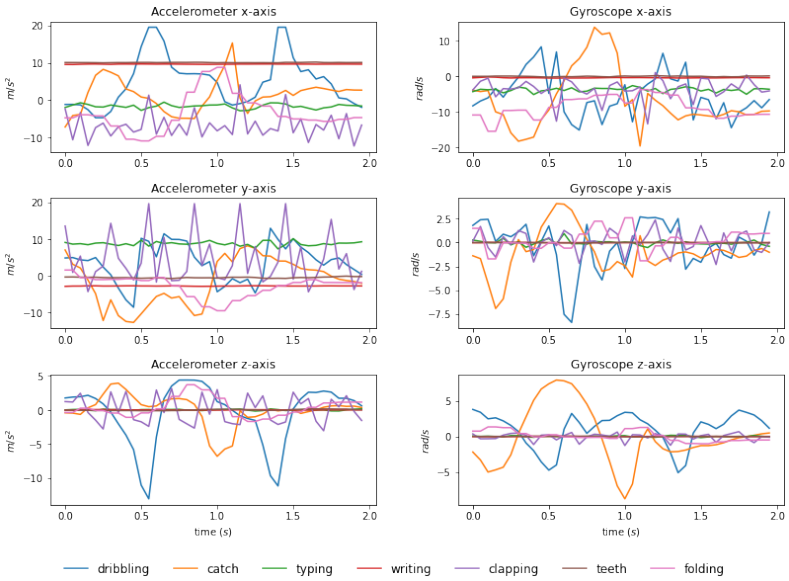

Figure 2: Temporal behaviour on each sensor axis of a random sample for some classes of WISDM

K (n. clusters)	**Global score**	**Global median**
4	5.80	0.36
5	6.16	**0.38**
6	**6.44**	0.37
7	6.20	0.34
8	6.04	0.34

Table 1: Global score and global median for each number of clusters after the KMEANS clustering applied on WISDM

- **Cluster 4**: Walking, Stairs, Standing, Typing, Brushing Teeth, Eating Soup

4 LSTM and Dimensionality Reduction

4.1 LSTM

After applying the clustering, we chose to run an **LSTM (Long Short-Term Memory)** ANN to our data, in order to verify if it will be able to separate the new data well in the right supercluster, and then insert it correctly in the right final classifier.

Unlike standard feedforward neural networks, LSTM has feedback connections. Such a recurrent neural network (RNN) can process not only single data points (such as images), but also entire sequences of data.

Therefore, before applying the LSTM to the training and test data, we built the labels associated with the individual samples on the basis of the belonging cluster associated with the reference class of the sample. In this way, for each single label, the number 1 will be positioned in correspondence with the cluster to which a specific class associated with the sample belongs, and the number 0 otherwise. The fitting of the network on the data was done taking into consideration 10 epochs and a batch size equal to 100. The model is made up of five distinct layers, that is, two blocks formed by the LSTM and Dropout pair and a final Dense layer. The shape output is equal to four, which is the number of clusters taken into consideration.

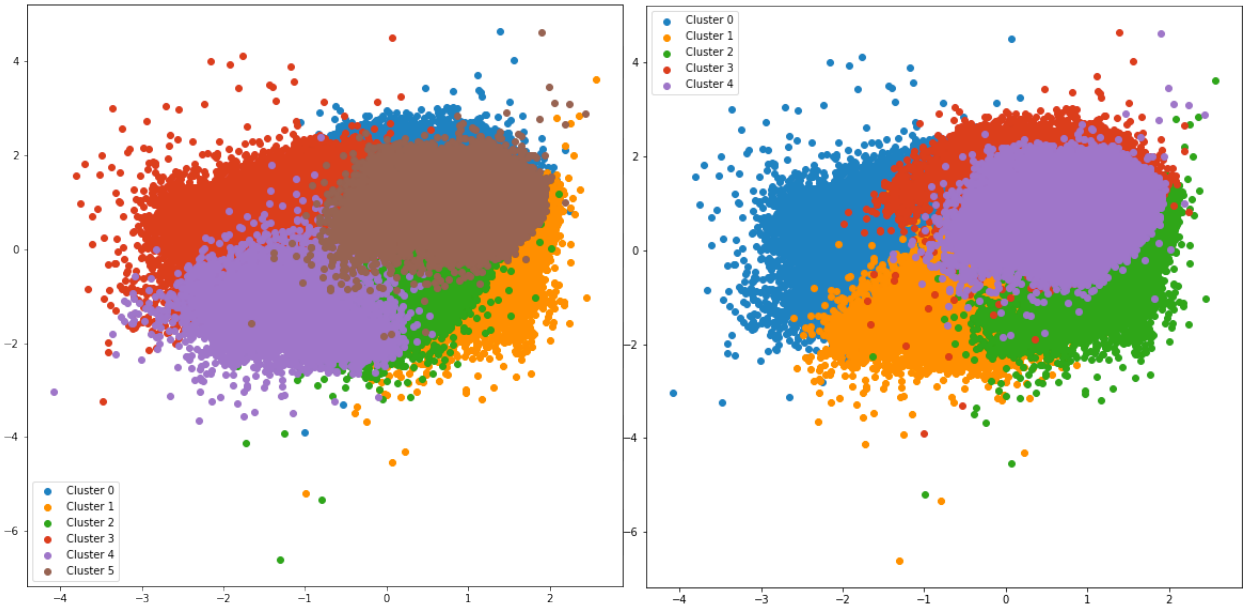

Figure 3: 2D representation of the first two components, of the data associated with the individual clusters, after the application of LDA, for K = 6 (left) and K = 5 (right)

Target	Final loss	Final accuracy
Training set	0.28	**89.59**%
Validation set	0.34	88.16%
Test set	0.33	88.18%

Table 2: Final loss and final accuracy, for each target set, after the application of LSTM with 10 epoch and batch size equal to 10

The result of applying the LSTM can be appreciated both in the Table 2 and in the Figure 4

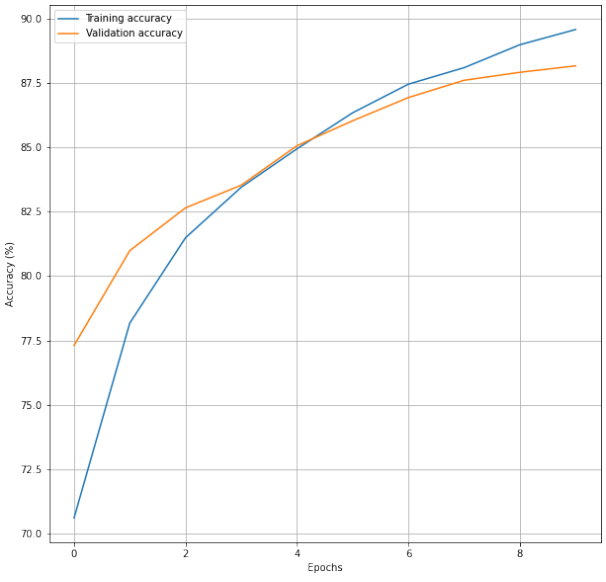

Figure 4: Accuracy trend, for each epoch, after LSTM application on validation and training set

4.2 Feature Selection and Cluster Separation

After applying the model associated with LSTM to our clustered data, we focused on some dimensionality reduction techniques. In particular, we decided to select features based on the application of **PCA (Principal Component Analysis)** on the data associated with our individual clusters. The purpose of

Cluster	**N. Components** (95% variance)
Cluster 1	5
Cluster 2	82
Cluster 3	69
Cluster 4	113

Table 3: Number of components for each cluster needed in order to preserve 95% variance on the data

data pre-processing methods, and more specifically data reduction models, is to clean up and simplify input data. In particular, PCA is commonly used for dimensionality reduction by projecting each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible. The first principal component can equivalently be defined as a direction that maximizes the variance of the projected data. The i_{th} principal component can be taken as a direction orthogonal to the first $i-1$ principal components that maximizes the variance of the projected data.

In our specific case, the PCA application has planned to select, for each cluster, the number of components such that the amount of variance that needs to be explained is greater than 95%, in order to preserve the variance despite the reduction in terms of dimensionality.

The results obtained for each single cluster have been collected and can be appreciated in the Table 3 and Figure 5. In particular, the graph allows you to observe the trend of the percentage of:

- **Individual explained variance**: which describes how much variation in a dataset can be attributed to each of the principal components.

- **Cumulative explained variance**: which shows the accumulation of variance for each principal component number.

5 Classification

5.1 LMU classification

Legendre Memory Units (LMUs), as seen in 2.3, are a novel memory cell for recurrent neural networks. LMUs have achieved state of the art performance on complex RNN tasks, and in this project, as final step, we built an LMU classifier using **NengoDL** [5].

NengoDL represents a simulator able to provide sophisticated networks featuring cognitive abilities starting from single neuron models. The three NEF principles (representation, transformation and dynamics) are translated by Nengo into the fundamental units for networks construction, defining three core objects called **ensemble, node and connection**. In addition, NengoDL could translate deep learning models by replacing standard activation functions with Nengo's spiking neurons.

For the construction of an LMU cell, we calculated the matrices A and B relating to the mathematical derivation of LMU, then we created the objects corresponding to the variables x, u, m and h, and finally we calculated u_t (see Figure 1 for more details). In this stage, we have removed e_h and e_m as they are not needed in this task. Our implementation for the calculation of m_t, requires the need to make A and B not trainable, but they could also be optimized in the same way as the other parameters. At this point, for each connection, we have set the synapse parameter of the Nengo connection for A equal to 0, because Nengo automatically adds a one-timestamp delay, so we can think of any connections with $Synapse = 0$ as representing $value_{t-1}$.

At the end, we have constructed a simple network consisting of an input node, a single LMU cell, and a dense linear readout. For the training we trained for 10 epochs both for the entire dataset and for the single clusters. The results could be visualized in Table 4

5.2 NNI experiments on sLMU

Neural Network Intelligence NNI is an easy to use toolkit, that allows to configure an experiment to be carried out through a Neural Network, and then ex-

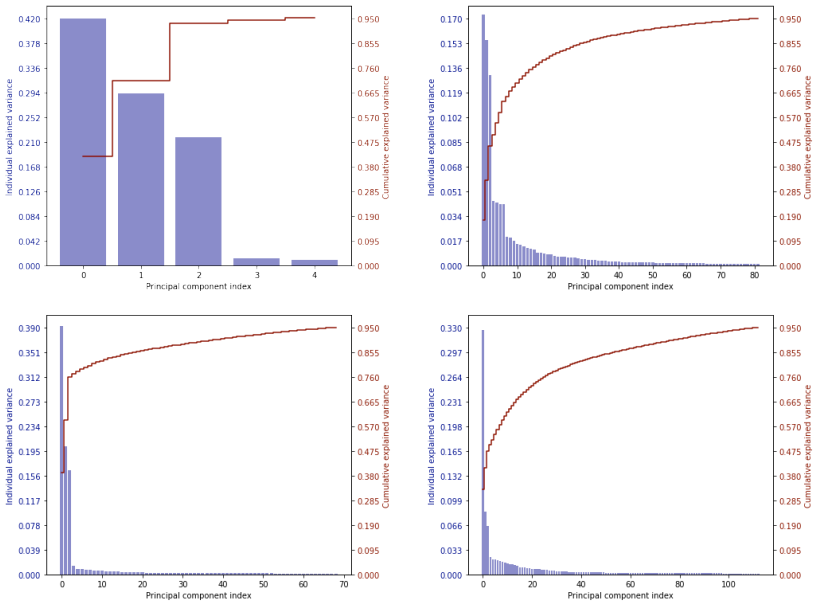

Figure 5: Cumulative explained variance vs Individual explained variance graphs for each cluster detected

Target set	Initial accuracy	Final accuracy	Initial loss	Final loss
Entire dataset	5,99	63.11	3,43	1,23
Cluster 1	36,71%	**82,92%**	1,22	**0,45**
Cluster 2	20,68%	81,18%	1,93	0,55
Cluster 3	23,76%	79,62%	1,72	0,52
Cluster 4	16,53%	73,87%	2,24	0,82

Table 4: Comparison between initial/final accuracy and initial/final loss for each test set associated with individual clusters or the entire dataset

ecuting the experiment many times (trials) by applying a strategy to optimize the hyperparameters, while maintaining personalized performance counters for each trial [6]. NNI offers a web interface that allows to keep track of the trials and their performance in a GUI. To exploit this interface, in this project NNI has been used through **ngrok**, a reverse proxy that offers a front-end [7]. Leaving the execution to a proxy allows to interact with NNI to control (pause, resume, stop) the experiment, and save its state even during the execution. The experiment output contains the specified performance counters, allowing to elaborate them as needed. Thanks to the Nengo framework, we can transform the LMU model into a Spiking version (sLMU), by replacing standard activation functions with Nengo's spiking neurons, as anticipated in LMU section.

Spiking Neural Networks SNNs are a new paradigm of neural networks, based on bio-inspired neurons. As a direct product of neuromorphic computing, they work well with neuromorphic hardware, and thus are designed to be low-energy and memory optimal too. They are based on event-based asynchronous operations, which means that their activation functions

Hyperparameter	Description
order	Order of LMU polynomial
theta	Length of LMU sliding window
$\mathrm{n}_n eurons$	Size of sLMU neuron ensambles
$\mathrm{synapse}_i n$	LMU input time constant
$\mathrm{synapse}_o ut$	LMU output time constant
$\mathrm{synapse}_a ll$	sLMU time constant between neurons
max_rate	sLMU neurons firing rate
minibatch	batch size
tau	LMU time constant for memory connections
lr	learning rate

Table 5: Optimized hyperparameters through NNI

Target set	Test acc.	Probe acc.	Loss	Params	Mem(MB)	neuron	SOPs	μJ
Entire dataset	68.1	67.84	0.9952	77292	0.3092	20520	225120	114,2
Cluster 1	77.5%	77.2%	0.5475	70.416	0.282	19440	9123	4.63
Cluster 2	**78.05%**	**78.15%**	**0.5443**	**67.000**	**0.268**	32160	94918	48.06
Cluster 3	68.1%	68.28%	0.7358	72.254	0.289	12740	58469	29.64
Cluster 4	54.46%	53.63%	1.1029	40508	0.162	22800	44892	22.76

Table 6: Results of NNI experiments on the each cluster vs the entire dataset

act by firing temporally sparse binary spikes, just as real neurons spread information through spikes (action potentials) [1].

Through NNI, an sLMU is rebuilt for each trial, by exploring a search space with hyperparameters in Table 5. In our case, prediction accuracy and loss are tracked by NNI, and its output can be used to simulate the energy consumption and the memory footprint. The experiment results can be seen in Table 6, where we understand that accuracy is generally great taking into account that both number of network parameters and memory **footprint are much lower with respect to traditional known ANN**. Also, in Figure 6 we can see prediction error. Here we can notice that generally classes are well classified, with exceptions for activities that are significantly similar (e.g. eating pasta or eating soup, brushing teeths or clapping, drinking or eating sandwich, walking or climbing stairs, dribbling or playing catch).

6 Conclusions

Results obtained show that SNNs maintain good performances while offering an important reduction in power and memory consumption, with respect to a classic ANN, which is critical for applications like HAR and similar use cases. Also, having a metalabeling phase is evidently important as similar classes that have been separated in different clusters would have caused many more wrong predictions otherwise. In fact, in both cases separating classes into clusters gives more performance in each cluster, and even if training could take equally or more energy and memory than using the entire dataset, the actual usage of this method in a device real application would consider only one cluster at a time. Future works could concentrate on improve class separation with better clustering techniques and putting effort on time sequence data preprocessing. Surely, we have shown that a neuromorphic alternative is well suitable to HAR applications.

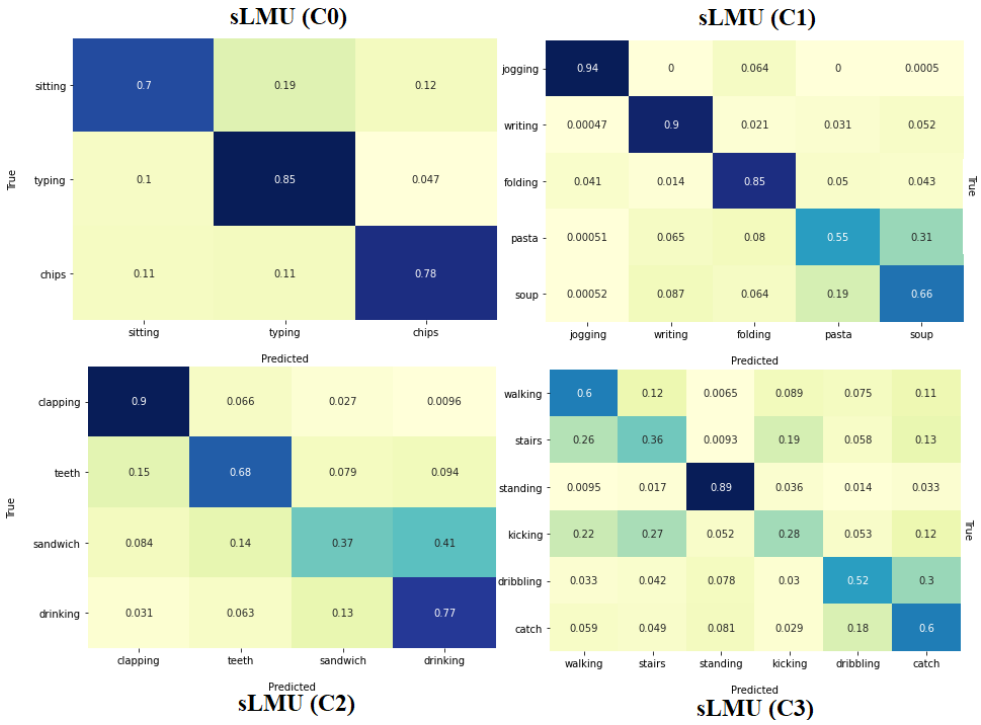

Figure 6: Confusion matrices of clusters classes

References

[1] V. Fra et al.; 2022; Human activity recognition: suitability of a neuromorphic approach for on-edge AIoT applications; Neuromorph.Comput. Eng. 2 014006

[2] N. B. Khoulenjani et al.; 2020; Cancer miRNA biomarkers classification using a new representation algorithm and evolutionary deep learning; Springer-Verlag GmbH Germany

[3] A. R. Voelker et al; 2019; Legendre Memory Units: Continuous-Time Representation in Recurrent Neural Networks; NeurIPS

[4] G. Weiss et al.; 2019; WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set; Computer and Information Sciences Department, Fordham University;

[5] NengoDL: https://www.nengo.ai/nengo-dl/introduction.html

[6] NNI: https://nni.readthedocs.io/en/stable/

[7] ngrok: https://ngrok.com/docs