Semi-Supervised Autoencoder for Human Activity Recognition

Denis Deronjic†

***Abstract*—Human Activity Recognition (HAR) is a field which is still in growing and gains a lot of interests in academic community and in industrial applications. As can be seen by the growing number of weareable devices such as smartwatch, the importance of studying HAR for Sport and health applications became critical for a commercial success of such devices. The advent of Deep Learning made possible to classify the signals directly from raw sensors data from IMU sensors. This approach is called Automatic Feature extraction. In this paper different algorithms are proposed for HAR. A 1D Convolutional Neural Network is proposed as the starting point for classification task. A more advanced architecture is InceptionTime [1] where the Inception mechanism is exploited for Time-Series recognition. This architecture is proposed with the objective to show that a complex architectures for HAR do not necessary lead to drastically better results compared with simpler Neural Networks. The last architecture showed is a Semi-Supervised autoencoder (SSAE) where, the labels of the dataset are utilized to help the network reduce the reconstruction error. Results will show that SSAE can learn very well the latent representation of the data and thus classify and reconstruct correctly the data. Even though CNN returns the best results, SSAE returns interesting results with a F1-score greater to 0.9 for the majority of classes and benefits of a lower training time. Furthermore SSAE can be utilized successfully for data reconstruction since it operates also as a Denoising Autoencoder and can be extended to a variational version for data genereration.**

***Index Terms*—Human Activity Recognition, Semi-Supervised Learning, Neural Networks, Convolutional Neural Networks, Autoencoder**

I. Introduction

Automatic Human Activity Recognition (HAR) is an interesting area in which companies and universities are investing researching funds. In particular timeseries HAR is taken into consideration thankfully to the growing number of portable devices with IMU sensors equipped, whose vision based HAR is less explored due the difficulties and constraints for placing cameras in the environment. Sensor based activity recognition has the advantage to be easily developed and integrated in the aformentioned systems, thanks to the increasing and low energy computing power of the portable devices. The main industrial applications for HAR are sports and healthcare. In particular, in this paper, we are focusing on sport activity recognition where multiple human activities such as running, walking and jumping are recorded. Automatic feature extraction for HAR is an interesting growing field that, with the help of the novelty Deep Learning techniques, human intervention for modeling a classificator is almost reduced to zero. Automatic feature extraction has the advantage that doesn't need a very deep domain knowledge of the considered problem. This helps on a faster development and a better generalization of the data. In HAR the data is highly variable due to the different individuals from which data are collected from. The data may vary by the height of the person, if the person has a diseases such as parkinson, from the different place where the device is disposed etc.

In this work an automatic feature extraction is computed from the raw data. Deep learning techniques such as Autoencoders(AE) and Convolutional neural networks for classification are implemented and tested. The classification goal consists in correctly classifying the following labels: JUMPING, STANDING, SITTING, WALKING, FALLING, LYING, RUNNING from the accelerometer's and gyroscope's signals of the IMU sensor. Different models are developed to achieve this goal, with a special care at memory requirements, computational speed and accuracy of the predictions. In particular for HAR, networks with low memory requirements and a low computational power needed is preferable due to the hardware limitations of the portable devices.

The structure of this paper is as it follows: In section II related works for HAR are discussed for automatic feature extraction utilizing machine learning models; in section III an high level description of this work is presented; in section IV the data pre-processing and transformation is discussed; in section V the models presented are described in more details; in section VI the results of the models are presented and in section VII the conclusions are discussed.

II. Related Work

Human Activity Recognition (HAR) is widely studied in literature. The conventional approach in HAR uses a human domain knowledge to extract some useful features from data. Those hand-crafted features utilize signal combination and norm calculation [2] or statistical data such as mean, variance and standard deviation [3]. Hand crafted features are usually fed into classical machine learning algorithms such as SVM, decision trees, random forest, k-nearest neighborhood or bayesian networks. Unfortunately, hand-crafted features extraction are strongly correlated with the domain of the problem and the sensor type, number of sensors, position of sensors giving limited generalization performance [4]. More modern approaches utilize Deep Learning (DL) models which have the ability to extract relevant informations from data. In particular in this case, neural network for signal processing

†Mat. 1231829, email: denis.deronjic@studenti.unipd.it

are utilized such as Convolutional Neural Networks (CNNs) [5], Recurrent Neural Networks (RNNs) [6] and AutoEncoders (AES) [7].

An interesting approach is the application of an Autoencoders ensemble [8] where each autoencoder is trained for each activity. An approach consist in the use of a pre-trained CNN as a feature extractor [9] in the domain of HAR and Audio signals as a case study, hoping that in future pre-trained models could also be used for time-series data as it happens in computer vision applications.

A very useful approach [10] to overcome weakly labeled data in the Internet of HealthCare Things (IoHT) is the use of Reinforcement learning; more specifically a Deep Q-Network (DQN) for an intelligent auto-labeling and a LSTM network for the classification of activity signals.

III. Processing Pipeline

The utilized dataset is collected by the German Aerospace Center(DLR) [11] and it is composed by more than ten hours of labeled activities. The data is collected by 3D measurements of the accelerometer, gyroscope and magnetometer in a series time format. In signal processing applications one important task is to make the data with a fixed length. The main challenge with those signals is their variability. The input for a neural network can be different for each signal and this issue makes learning very difficult. One way to overcome this issue is to apply a framing step on the data with a fixed window size and stride in order to maintain the data with the same length. For HAR applications, magnetometer signals can be removed considering that to obtain good results it is sufficient to use only the accelerometer and gyroscope signals without leading to significant improvements as stated in the literature. Another important aspect on removing magnetometer is due to the fact that most of the devices don't have magnetometer sensors. This is a good way to generalize the results. Data normalization is an important pre-processing step since the data may be influenced by the height of individuals, position of the sensors, type of sensors etc. The whole processing pipeline can be seen in Fig. 1.

In this paper different models are utilized for classification. A 1D Convolutional neural network which is the first starting point for signal classification. An Inception architecture designed for time series classification and a Semi-Supervised autoencoder.Every model is trained on different version of signals, for instance raw signals, DCT signals and a concatenation of raw signal and DCT of the accelerometer norm. These types of networks are analyzed to show on how differently complex networks behaves from the simpler ones in the field of HAR.

IV. Signals and Features

The dataset is composed by accelerometer, gyroscope and magnetometer activities taken from the IMU sensor Xsens MTx-28A53G25 in a time-series format taken by 4.5 hours of labelled activities from 16 individuals which are in between 23 and 50 years old. The data is taken with respect of sensor frame (SF) which can be converted to global frame (GF) utilizing the cosine matrix provided in this dataset.

The data is relabeled in order to simplify and reduce the quantity of labels that are originally sixteen. The acceleration labels are removed due to the fact that those are not related to any useful activity. The jumping labels JUMPBCK, JUMPFWD and JUMPVRT (jumping back, jumping forward and jumping vertical) are grouped into JUMPING. The same cluster is done with WALKDWS and WALKUPS (walking down and walking up) grouped into WALKING. The resulting labels are FALLING, JUMPING, RUNNING, SITTING, STNDING, WALKING and XLYINGX.

The direction cosine matrix provided by the dataset is utilized to transform the collected data from SF to GF applying a simple vector-matrix multiplication. The framing step is suddenly computed utilizing a window size of 128 and stride step of 8. The data in this stage has the dimension of 128x6 where 6 are the x,y,z acceleration and x,y,z orientation. In this way the dataset is now ready to be fed in a neural network without the concern of having different signal dimensions.

The data is divided at 70% as a training set and 30% as test set maintaining the ratio classes of the original dataset. Training and test sets are hence rescaled. Different type of data rescaling technique may result in different networks performance. Min-max normalization and standardization with 0 mean and 1 variance were used. Standardization is the most effective in terms of neural network performance. During the analysis, the training data is divided with a 20% of validation.

Analyzing the available data after the preprocessing step (for instance the data framing in 128 windows and stride 8) can be noticed that not all the classes are well represented in the dataset. As can be seen in Fig. 2 the majority of the labels are the ones corresponding with STNDING followed by WALKING and SITTING.

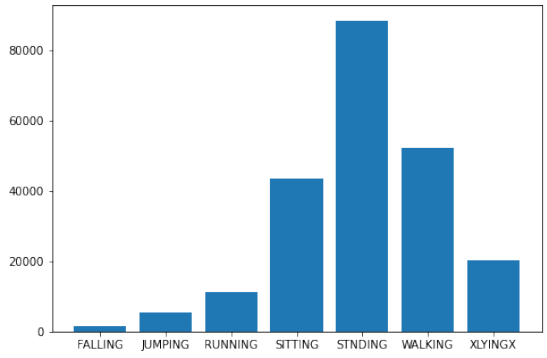

Fig. 2: Class Distribution

A data augmentation technique which adds Jitter, Rotation etc to the least represented labels was tested. Unfortunately classification results were not effective during classification. This may be the case where data augmentation worsening the results because the network doesn't learn from a real data distribution but from the ones we processed leading to a bias inducted by ourselves. For that reason in this paper it's not applied any data augmentation technique. Fortunately as the

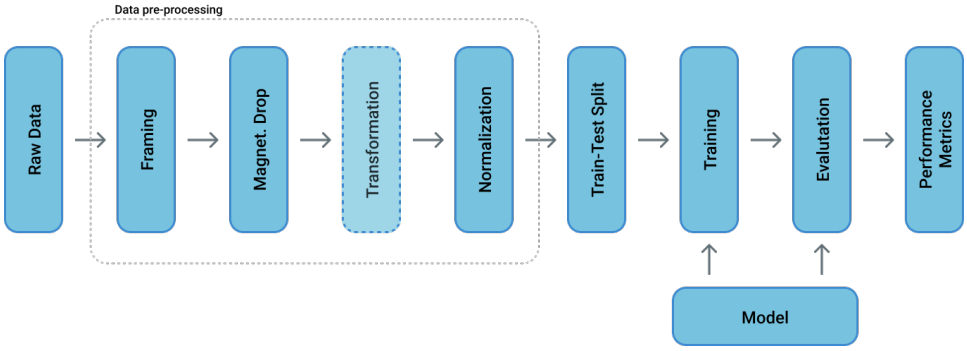

Fig. 1: Processing Pipeline. Raw data passes through the pre-processing step and is splitted in Training, Validation and Test set. A model then is trained and evaluated until the best performing parameters are found and finally tested on the test set showing the performances achieved. The transformation block inside the pre-processing block is an optional step which applies some transformations on the data such DCT transformation.

results will prove, the less represented data are learnt by the networks giving also good results in some cases.

V. Learning Framework

A. Convolutional Neural Network

The proposed architecture is composed by 5 blocks which each block contains 1D convolutional layer, batch normalization layer and a MaxPooling1D layer. All the layers have ReLU as activation function and MaxPooling1D has pool size of 2 and stride of 2. The convolutional layers in each block has number of filters respectively 16,32,64,128,256 and kernel size repectively 9,7,5,3,3 all with stride 1. At the end of the network a flattening operation is performed and fully connected output layer is connected. Between the flattening and the output layer a dropout layer with rate 0.4 is added as regularization.

B. Inception-Like Neural Network

This network utilizes the concept of Inception from the computer vision domain applied in the field of HAR and, in general, in Time-Series classification applications. This network is called InceptionTime [1]. For every 3 inception blocks, a residual connection is added. In the residual connection a convolution operation followed by batch normalization is performed. The inception block is composed as in Fig. 3. For each inception block the number of filters chosen is 32 (i.e. the number of filters of each convolutional operation inside inception block) which in the output of the inception block the number of filters is 128 thanks to the concatenation operation. The architecture of this network can be seen in Fig. 4.

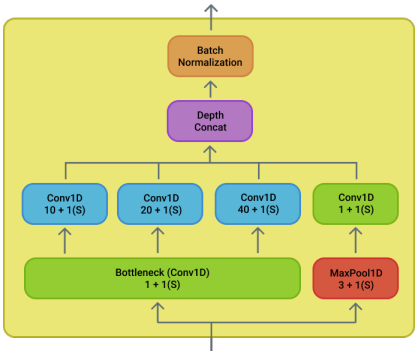

Fig. 3: Inception Block. In convolutional blocks and MaxPooling block the leftmost number represents the kernel size, the rightmost number the stride and (S) represents *same* padding.

C. Semi-Supervised Autoencoder

A semi supervised AutoEncoder is implemented in this paper. It is semi-supervised since it uses data labels to reduce the reconstruction error. Inspired by [12] the autoEncoder is composed by a encoder which maps the input in a latent representation. It is composed by 4 1D-convolutional layers with kernel sizes 32, 64, 128 and 256 correspondigly. A dropout of 0.4 is added to the input of the encoder for simulating corrupted and noised data for a better regularization. The latent space is developed by a dense layer with 64 neurons. To the encoder are attached two different networks which both take benefit from the latent representation as can be seen in Fig. 5. The networks attached are:

- **Decoder** which tries to reconstruct the input from the latent representation. It is composed by 1D convolutional layers stacked one on top of another. The architecture is specular with encoder to maintain the simmetry. The decoder tries to minimize the mean squared error defined as:

$$J_{rec}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(X_i - \hat{X}_i\right)^2 \tag{1}$$

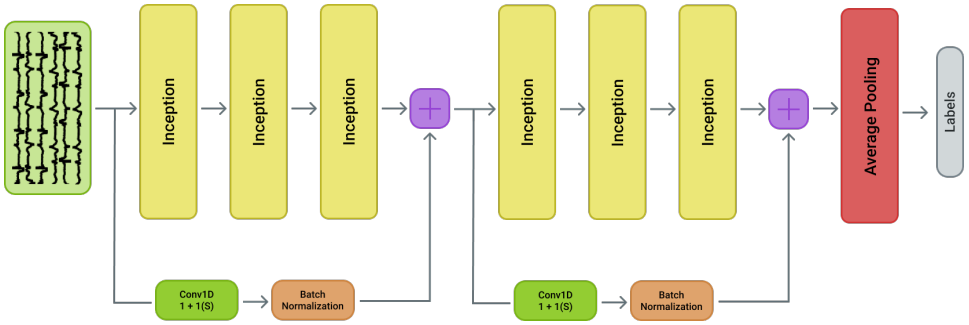

Fig. 4: The InceptionTime architecture. Residual connections are followed by a Convolutional and Batch Normalization operation.

where X_i is the input signal passed to the encoder and $\hat{X}_i$ is the reconstruction of the input signal.

- **Classifier** is made by a simple feed forward layers of 512 neurons and an output layer with 7 neurons as the number of labels. The classifier tries to minimize the sparse categorical cross entropy loss defined as:

$$J_{cl}(\theta) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\log p_\theta(y_{nk}) \tag{2}$$

where t_{nk} is the the target class of a sample n which belongs to a class k, y_{nk} is the prediction label of the sample n in a class k by the model θ and K is the number of classes.

Given the losses (1) and (2) the resulting loss function minimized by the model is:

$$J_{SSAE}(\theta) = \alpha J_{rec}(\theta) + \beta J_{cl}(\theta) \tag{3}$$

where α and β are tuning hyperparameters for loss regularization. For instance the value selected for α is 0.1 and β is 1.0.

The advantage of this network is the ability to have both the benefits of unsupervised and supervised networks. The cross entropy loss (2) in the network loss (3) behaves as a sort of regularization of the MSE loss. It helps the overall generalization of the unsupervised task resulting with a well clustered hidden latent representation. The unsupervised model in this way finds more easily patterns of HAR task, in particular it can better recognize the activities for whose the sensors activities are not too much different from each other. For instance XLYINGX, SITTING and STDING have similar patterns which a model with only the MSE error has difficulties to categorize. The dropout layer at the beginning of the Encoder acts as a regularizer and a noise simulation during the inference. In this way the AutoEncoder is more robust from noises in the input signal.

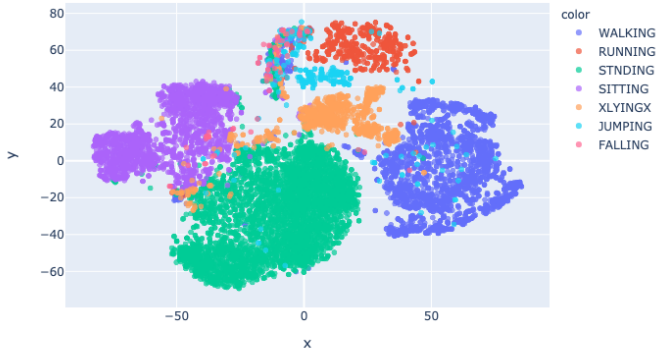

Fig. 6: 2D visualization of the latent space learnt by the semi-supervised AutoEncoder using *rawdata* set. As can be easily seen the classes are well clustered in the latent representation.

VI. Results

In this section the results of the described networks are going to be discussed giving the comparisons between the models. In this paper, the models have been implemented utilizing *Keras* using *Tensorflow* as backend. Both are open-source projects widely utilized by the machine learning community. The workstation on which the models were implemented and tested is the newest Mac-Mini with M1 architecture and 16 GB of RAM.

Due to the highly imbalanced data, the accuracy metric is not sufficient to test models accuracy. To overcome this, two metrics are used: PRECISION and RECALL. Precision is defined as

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

and recall is defined as

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

where TP = *True Positives*, FP = *False Positives* and FN = *False Negatives*.

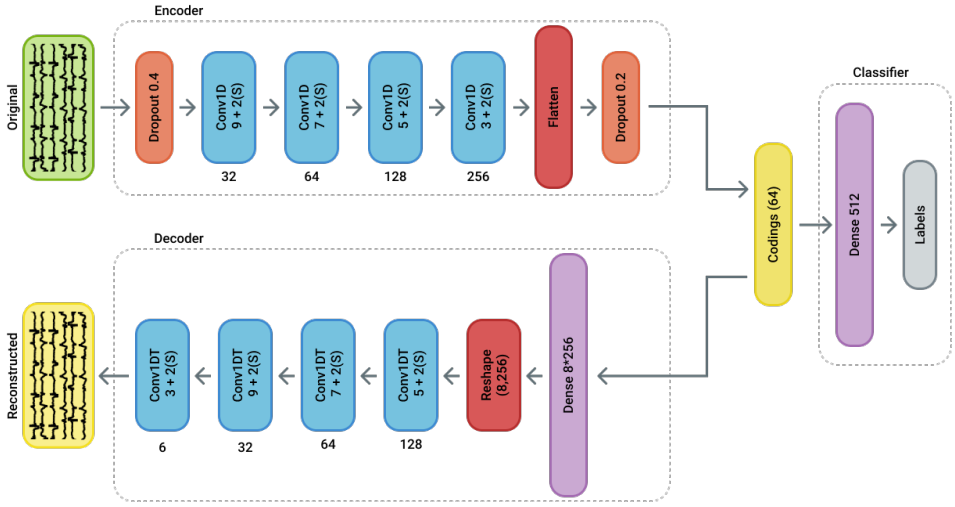

Fig. 5: The Semi-Supervised Autoencoder architecture. The numbers below Conv1D and Conv1DT are the number of filters.

For the simplicity of reading and plotting, F1-score is utilized. This metric combines both precision and recall and is defined as it follows:

$$F1\text{-}score = 2\frac{Precision \cdot Recall}{Precision + Recall} \tag{6}$$

The F1-score is measured for each label giving the metric on how well the model is able to classify a specific label.

The training of each model is made with 100 epochs with the help of *early stopping* to avoid overfitting. Adam optimizer is chosen.

In the following Tables the results are going to be discussed utilizing different features of test set. The tested models are those described in Section V. For a better comparison with SSAE, a simple AutoEncoeder with MSE loss is also tested. This autoencoder is Composed by stacked 1-D convolutional layers identical with the SSAE's Encoder and Decoder but without the Dropout layer in the input of the Encoder.

In the following Tables, the F1-score is showed for every HAR label in correspondence of the column of each model. The models are summarized as follows:

- CNN: Convolutional Neural Network
- INC: InceptionTime network
- AE: AutoEncoder
- SSAE: Semi-Supervised AutoEncoder

	CNN	INC	AE	SSAE
FALLING	0.92	0.90	0.49	0.79
JUMPING	0.96	0.95	0.79	0.94
RUNNING	0.99	0.99	0.96	0.99
SITTING	0.98	0.95	0.61	0.96
STNDING	0.99	0.97	0.79	0.98
WALKING	0.99	0.99	0.96	0.99
XLYINGX	0.98	0.93	0.07	0.95

TABLE 1: F1-scores on *rawdata* test set

As can be noticed in Table 1 it turns out that the CNN architecture gives the best results. The basic autoencoder gives the worst results. During the unsupervised pre-training, the autoencoder was able to cluster well the labels RUNNING and WALKING as highlighted in Fig. 7 and thus classify better these two classes.

The main advantage of the semi-supervised Autoencoder over the basic Autoencoder as can be seen in Fig. 6, the clusters for each label are well defined, leading the semi-supervised AutoEncoder achieve good classification and reconstruction results with almost the same training time.

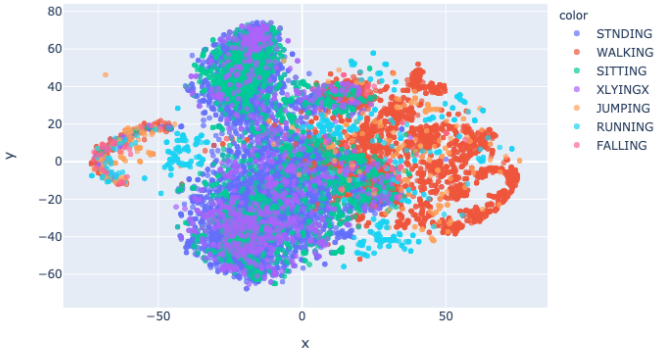

Fig. 7: 2D visualization of the latent space learnt by the AutoEncoder using *rawdata* set. The most relevant cluster is the orange one which represents WALKING. The light blue cluster which stands for RUNNING, although it's sparse in the latent space, can be easily recognized.

For DCT data set, as can be seen in Table 2 the results are far way worse than with *rawdata*. This could be cause by the similarities of the frequencies on activities. For all the models, best F1-scores are achieved wiht labels RUNNING and WALKING. It turns out that DCT transform for recogniz-

	CNN	INC	AE	SSAE
FALLING	0.77	0.65	0.57	0.56
JUMPING	0.87	0.85	0.69	0.68
RUNNING	0.98	0.97	0.93	0.92
SITTING	0.84	0.89	0.60	0.79
STNDING	0.92	0.94	0.78	0.88
WALKING	0.98	0.98	0.96	0.96
XLYINGX	0.77	0.80	0.27	0.63

TABLE 2: F1-scores on *DCT* test set

ing SITTING, STANDING and XLYINGX is not beneficial. Combining *rawdata* with the DCT of the aceleromenter norm doesn't lead noticeable improvements over the *rawdata* only models as can be seen in Table 3. Further, this type of dataset contains one more dimension making the dataset be at the dimension of 128x7.

	CNN	INC	AE	SSAE
FALLING	0.86	0.83	0.54	0.79
JUMPING	0.94	0.93	0.82	0.93
RUNNING	0.99	0.99	0.97	0.98
SITTING	0.93	0.94	0.63	0.96
STNDING	0.97	0.96	0.79	0.98
WALKING	0.99	0.99	0.96	0.99
XLYINGX	0.90	0.91	0.12	0.96

TABLE 3: F1-scores on *raw* + *DCT* test set

Looking Tables 1, 2 and 3 it's clear that whatever the type of data is, every model is able to classify well RUNNING and WALKING giving an F1-score greater than 0.9 in every model. This result is interesting because it shows very clearly what are the difficulties on modeling an HAR application. In HAR there are two main issues that make the prediction harder:

- **Less representation of data points** where there are activities with few labels such as FALLING and JUMPING.
- **Similar patterns between different classes** where some activities have similar patterns. With activities SITTING, STNDING and XLYINGX which are static activities, and due to their nature it's difficult to clusterize hence classify them without a good model and loss function.

The confusion Matrix in Fig. 8 shows the uncertainty of predictions showing the difficulties just mentioned.

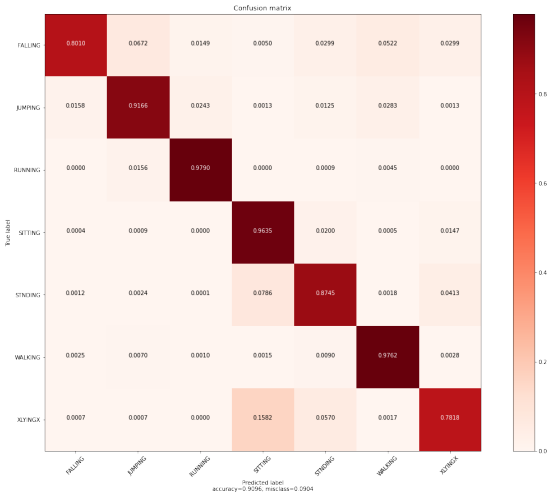

Fig. 8: Confusion Matrix of the ConvNN model using DCT test data. This figure shows the uncertainty of the model with FALLING, STDNING and XLYINGX activities.

Results also shows that in HAR much more complex networks don't achieve significant improvements over simpler networks. This result is satisfactory since the devices which runs those networks are constrained with resources.

VII. Concluding Remarks

The self-supervised autoencoder returns very good solutions compared with the basic autoencoder in terms of classification and reconstruction. Training the autoencoder in a semi-supervised way helps the autoencoder to learn a better representation of the data hence a very good classification with less computing power thus the time needed for prediction. The less demanding resource capability is important or crucial in certain devices, especially IoT devices where the computing and memory power is limited. An algorithm that can do classification and data compression/reconstruction at the same time could be very important in such fields. What could be done in a future work is the implementation of the Semi-Supervised Variational AutoEncoder which would have generative capability, in addition to classification and data compression capability. Data generation from this model could be useful for data augmentation techniques.

As a student I learnt how to tackle with time series applications utilizing Deep Learning techniques especially the data framing, how to search in the scientific literature, which are the requirements needed for writing a good research paper and how to write a scientific paper.

References

[1] H. Ismail Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean, "Inceptiontime: Finding alexnet for time series classification," *Data Mining and Knowledge Discovery*, vol. 34, p. 1936â€"1962, Sep 2020.

[2] K. Frank, M. J. Vera, P. Robertson, and M. Angermann, "Reliable real-time recognition of motion related human activities using mems inertial sensors," vol. 4, 09 2010.

[3] D. Figo, P. C. Diniz, D. R. Ferreira, and J. M. P. Cardoso, "Preprocessing techniques for context recognition from accelerometer data," *Personal and Ubiquitous Computing*, vol. 14, pp. 645–662, Oct 2010.

[4] A. A. Varamin, E. Abbasnejad, Q. Shi, D. Ranasinghe, and H. Rezatofighi, "Deep auto-set: A deep auto-encoder-set network for activity recognition using wearables," 2018.
[5] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Exploiting multi-channels deep convolutional neural networks for multivariate time series classification," *Frontiers of Computer Science*, vol. 10, pp. 96–112, Feb 2016.
[6] H. Zou, Y. Zhou, J. Yang, H. Jiang, L. Xie, and C. J. Spanos, "Deepsense: Device-free human activity recognition via autoencoder long-term recurrent convolutional network," in *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2018.
[7] L. Wang, "Recognition of human activities using continuous autoencoders with wearable sensors," *Sensors*, vol. 16, p. 189, Feb. 2016.
[8] K. D. Garcia, C. R. de SÃ¡, M. Poel, T. Carvalho, J. Mendes-Moreira, J. M. Cardoso, A. C. de Carvalho, and J. N. Kok, "An ensemble of autonomous auto-encoders for human activity recognition," *Neurocomputing*, vol. 439, pp. 271–280, 2021.
[9] F. Cruciani, A. Vafeiadis, C. Nugent, I. Cleland, P. McCullagh, K. Votis, D. Giakoumis, D. Tzovaras, L. Chen, and R. Hamzaoui, "Feature learning for human activity recognition using convolutional neural networks," *CCF Transactions on Pervasive Computing and Interaction*, vol. 2, pp. 18–32, Mar 2020.
[10] X. Zhou, W. Liang, K. I.-K. Wang, H. Wang, L. T. Yang, and Q. Jin, "Deep-learning-enhanced human activity recognition for internet of healthcare things," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6429–6438, 2020.
[11] P. Zausinger, "Human activity recognition with inertial sensors."
[12] Z. Chai, W. Song, H. Wang, and F. Liu, "A semi-supervised auto-encoder using label and sparse regularizations for classification," *Applied Soft Computing*, vol. 77, pp. 205–217, 2019.