Semi-unsupervised Learning for Time Series Classification

Padraig Davidson
davidson@informatik.uni-wuerzburg.de
Chair of Computer Science X
University of Würzburg
Würzburg, Germany

Michael Steininger
steininger@informatik.uni-wuerzburg.de
Chair of Computer Science X
University of Würzburg
Würzburg, Germany

André Huhn
andre.huhn@stud-mail.uni-wuerzburg.de
Chair of Computer Science X
University of Würzburg
Würzburg, Germany

Anna Krause
anna.krause@informatik.uni-wuerzburg.de
Chair of Computer Science X
University of Würzburg
Würzburg, Germany

Andreas Hotho
hotho@informatik.uni-wuerzburg.de
Chair of Computer Science X
University of Würzburg
Würzburg, Germany

ABSTRACT

Time series are ubiquitous and therefore inherently hard to analyze and ultimately to label or cluster. With the rise of the Internet of Things (IoT) and its smart devices, data is collected in large amounts any given second. The collected data is rich in information, as one can detect accidents (e.g. cars) in real time, or assess injury/sickness over a given time span (e.g. health devices). Due to its chaotic nature and massive amounts of datapoints, timeseries are hard to label manually. Furthermore new classes within the data could emerge over time (contrary to e.g. handwritten digits), which would require relabeling the data.

In this paper we present *SuSL4TS*, a deep generative Gaussian mixture model for semi-unsupervised learning, to classify time series data. With our approach we can alleviate manual labeling steps, since we can detect sparsely labeled classes (semi-supervised) and identify emerging classes hidden in the data (unsupervised). We demonstrate the efficacy of our approach with established time series classification datasets from different domains.

KEYWORDS

semi-unsupervised learning, time series classification, sparsely labeled data

ACM Reference Format:
Padraig Davidson, Michael Steininger, André Huhn, Anna Krause, and Andreas Hotho. 2022. Semi-unsupervised Learning for Time Series Classification. In *Milets'22: 8th SIGKDD International Workshop on Mining and Learning from Time Series – Deep Forecasting: Models, Interpretability, and Applications (Milets'22).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.48550/arXiv.2207.03119

Milets'22, August 2022, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.48550/arXiv.2207.03119

1 INTRODUCTION

Autoencoders (AE) have become the de-facto standard for anomaly detection within deep learning. In this pair of neural networks, one trains an encoder to map the features of the input data (i.e. time series) into the latent space. The decoder reconstructs this representation as well as possible, with the constraint that the latent space is much smaller than the input domain. After training on data without anomalies (e.g. normal data), predictions of anomalies can be done by defining a threshold on the anomaly score (e.g. reconstruction loss) to predict abnormalities. In recent years, variational AEs [15] (VAE), have gained popularity, since they encode the data distribution in the latent space, rather than the raw features. This allows training on all variations of data and thus reliefs the burden of filtering data beforehand. Furthermore one can see anomaly detection as a probability rather than a raw score [3].

Since time series are ubiquitous and present in a myriad of types for classification, we are interested in models beyond this binary classification task. With the development of semi-supervised generative models [14], we are able to classify time series data, while only having to label a smaller amount of data. But, we still need to know all manifestations of classes beforehand. On the other hand, we could cluster the data, needing no label information at all [1]. This however, often comes with the drawback of lower classification accuracy and the need to manually annotate the found clusters.

To combine the benefits of the high classification accuracy in semi-supervised models with the ability to detect new classes, the hybrid approach of *semi-unsupervised* learning [7, 28] has emerged. In this paper we present *SuSL4TS*, a convolutional Gaussian mixture model for semi-unsupervised learning on time series data. Figure 1 visualizes the basic principle of our approach. Our contributions are twofold: (1) We present a model capable of semi-unsupervised time series classification from raw time series, partially on par with state of the art models, while only needing a limited amount of labels, (2) We show the efficacy of our approach on several benchmark datasets, and perform extensive experiments in this new domain for time series.

The remainder of this paper is structured as follows: after presenting related work in the field, we present the used datasets in Section 3. Section 4 illustrates the foundations of the used model, while

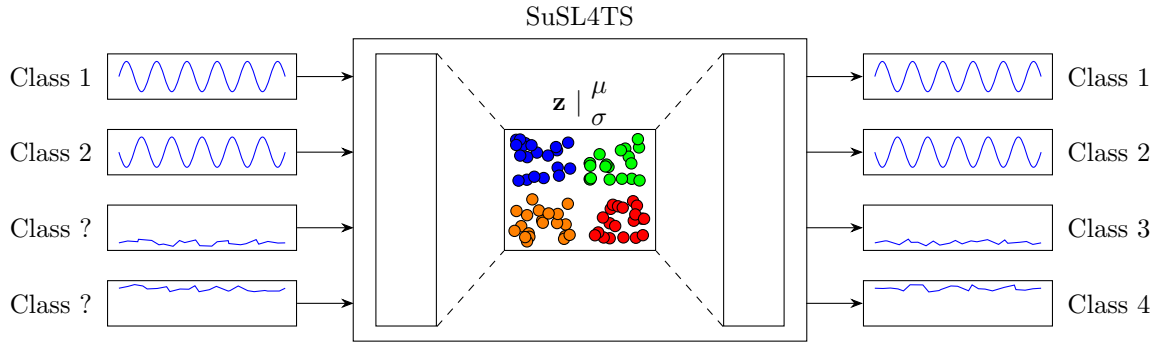


Figure 1: Semi-unsupervised learning for time series data (SuSL4TS). The model is tasked to classify the data on the left hand side (a single multivariate time series), with only limited labels available for classes 1 and 2, and two completely unknown classes. The output on the right hand side is the classified data with all four classes found. Within the model we can see four distinct cluster automatically found when zooming in on the latent space.

Section 5 outlines our experiments. We conclude with a discussion (Section 6) of the experiments and depict future work in Section 7.

2 RELATED WORK

Related work in time series classification is manifold since most datasets and solutions are customary. A larger review and benchmark of different algorithms and datasets can be found in [6, 23]. The authors present algorithms suited for time series classification task in an univariate [6] and a multivariate [23] setting. Univariate refers to datasets in which only a single sensor is used for the classification, whereas multivariate considers multiple sensor readings at the same time. The best performing algorithms are often *BOSS* [24], *COTE* [17] and *TS-Chief* [26] in the univariate problems, whereas *ROCKET* [9], *WEASEL* [25] and *INCEPT-NET* [10] show great performance in the multivariate problems. We refer the reader to the mentioned papers for a detailed explanation of the algorithms and specific datasets. A systemic analysis of related work can be found in [18], in which the authors list time series classifiers based on their technique: distance based, features based, ensemble approaches and tree approaches. In general, most algorithms aim for fully supervised classification, whereas we aim to reduce the time spent for labeling datasets.

Since the field of semi-unsupervised learning is relatively new, there is limited related work in this domain. In [27], the authors present a network capable of semi-unsupervised time series classification on human activity recognition. They compare their approach to the semi-supervised M2-model [14], and show great performance even with classes hidden. Their approach uses extracted features from the time series and use them within the fully connected network. We are however interested in the use of the raw time series signals, and therefore focus on the convolutional approach presented in [7].

3 DATASETS

The following datasets were used for the experiments. Since we are especially interested in the learning paradigms of semi-supervised and semi-unsupervised learning, we only hand-select some datasets for our purposes. The main strength of our approach are most prominent, if we have a large dataset that would require huge efforts to label, and only a limited amount of known classes in comparison to the whole dataset. Therefore we use three datasets to perform our experiments, both in the univariate and multivariate setting, stemming from different domains of data acquisition, which meet our requirements. The datasets are introduced in the following. A short tabular view is available in Table 1.

3.1 Human Activity Recognition

The Human Activity Recognition (HAR) dataset [4] consists of data collected from accelerometer and gyroscope sensors in smartphones. The subjects (30), aged 19 to 48, were tasked with performing various Activities of Daily Living (ADL) while carrying a smartphone on their waist. The subjects were instructed to perform six distinct ADL adhering to a defined protocol outlining the order of activities. The selected activities were *standing, sitting, laying down, walking, walking downstairs* and *walking upstairs*. Each activity was performed for 15 s, except walking up- and downstairs which only lasted 12 s. Each activity was performed twice throughout the routine and 5 s pauses separated activities. Linear acceleration and angular velocity in three axes was recorded at a sampling rate of 50 Hz. The data was then pre-processed for noise reduction. Additionally gravitational and body motion was separated using a low-pass filter. A total of 9 signals were sampled with a window of 2.56 s with 50 % overlap (i.e. input is of size $\mathbb{R}^{9\times128}$). A feature vector was obtained from each sampling window. A total of 561 features were extracted with measures common in HAR literature like mean, correlation, signal magnitude area and autoregression coefficients as well as energy of different frequency bands, frequency skewness and the angle between vectors like mean body acceleration and the y vector. The data was randomly divided into a 70/30 training/test split containing 7352 and 2947 samples respectively as shown in Table 4a. The authors also used a multiclass SVM to achieve a 96 % classification accuracy on the dataset [4].

3.2 ECG Heartbeat Classification

The second dataset used was the MIT-BIH Arrythmia Dataset [11]. It consists of electrocardiogram (ECG) recordings from 47 subjects

Table 1: Datasets used for the experiments

Dataset	Features	Classes	Samples from	Sampling rate	Input size	Best accuracy
HAR	accelerometer & gyroscope data	6	30 subjects	50 Hz	$\mathbb{R}^{9\times128}$	96 % (Multiclass SVM [4])
ECG	electrocardiogram recording	5	47 subjects	360 Hz	$\mathbb{R}^{1\times186}$	93 % (CNN [11])
El. Devices	electrical consumption	7	251 households	every 2 min	$\mathbb{R}^{1\times96}$	80 % (BOSS [26])

recorded at a 360 Hz sampling rate. The recordings are grouped in five categories based on annotations by cardiologists. As can be seen in Table 4b the class frequency is skewed towards the N class, which is to be expected since it includes normal heart behavior. Further explanation of the individual classes can be found in [11]. Furthermore each entry in the set consists of a single heartbeat padded with zeroes to ensure consistent length (i.e. input is of size $\mathbb{R}^{1\times186}$). The authors of [11] propose a CNN architecture to classify the dataset in a fully supervised fashion, achieving an accuracy of 93 %.

3.3 Electric Devices

The Electric Devices dataset contains measurements from households observing their electrical consumption. Samples are taken every 2 min from 251 households. After pre-processing and resampling to 15 min averages, the samples have a length of 96 values (i.e. input is of size $\mathbb{R}^{1\times96}$). We use the version from [5][1], regrouping the originally ten classes to seven: kettle; immersion heater; washing machine; cold group; oven/cooker; screen group and dishwasher. The authors of [26] report the best accuracy by using BOSS at 80 %.

4 METHODOLOGY

Gaussian Mixture Models. A variational autoencoder in general consists of an encoder $\Phi(x) : \mathbb{R}^n \to \mathbb{R}^d$, mapping the input data of dimensions n into the latent space of dimensions d [15]. The decoder $\Theta(z) : \mathbb{R}^d \to \mathbb{R}^n$ on the other hands inverts this mapping, recreating the input data from the (compressed) latent encoding. This compressed space is often used for other downstream tasks in a second training step, for example classification or other information extraction tasks. The first step in this process is trained unsupervised, as it requires no annotated data, whereas the latter task is trained fully supervised.

This two-step process can be merged into one, by adapting the joint probability distribution p_Θ, resulting in a Gaussian Mixture Deep Generative model (GMM) capable of learning semi-supervised classification [14]. With some further modification we can use the inductive bias requirement to perform semi-unsupervised classification tasks with GMMs [7, 27, 28].

GMM for Semi-unsupervised Classification. In this work, we built on the work presented in [7, 28] and adapt it to perform time series classification on raw sensor signals. That is, we are interested in the improved pattern recognition and performance shown in [7]. Therefore we adapt their work and replace the 2d convolutional networks for image classification, with 1d convolutions for raw time series. Additionally we use the work shown in [28] in two ways. Firstly, we use it a reference in performance for the presented convolutional model. Secondly, we adapt their idea of the Gaussian L_2 regularization with the standard L_2 term provided by the Adam optimizer [13]. Our overall loss function can now be described as [7]

$$\begin{aligned}\mathcal{L} \coloneqq & \mathop{\mathbb{E}}_{x,y\in D_l}\left[\mathcal{L}_l(x,y) - \alpha\cdot\log q_\Phi(y|x)\right] \\ & + \mathop{\mathbb{E}}_{x\in D_u}\left[\mathcal{L}_u(x) - \gamma\cdot\lambda\cdot\sum_{c\in C} q_\Phi(c|x)\cdot\log q_\Phi(c|x)\right] \\ & + w\cdot\Theta_t.\end{aligned}$$

D_l refers to the labeled subset of the data, thus containing samples x and their corresponding class y (one-hot encoded). On the other hand, D_u contains all unlabeled data. That is all data that is to be mapped to the known classes (semi-supervised classification), and data stemming from possibly new classes (unsupervised clustering). Θ_t holds the trainable weights at epoch t. α, γ, λ are hyperparameters weighting the entropy regularization, whereas the loss terms $\mathcal{L}_l, \mathcal{L}_u$ measure the evidence lower bound (ELBO) from the GMM model. All other loss terms, the network architecture and further details can be seen in [7]. This approach allows us to analyze our experiments in four learning regimes: unsupervised, semi-unsupervised, semi-supervised and fully supervised.

5 EXPERIMENTS

Experimental Setup. Since SuSL4TS is capable of handling all learning paradigms, we have conducted experiments in any setting. That is, we performed a parameter search for 60 trials in the settings unsupervised learning (UL, 0 % labeled), semi-supervised (SSL, 20 % and 50 % labeled), semi-unsupervised (SuSL, 20 % and 50 % labeled) and supervised (SL, 100 % labeled). In the semi-unsupervised setting, we hid different classes. For the HAR dataset we tested three settings: (1) hiding all *walking* classes, (2) hiding all stationary classes, and (3) hiding one movement (walking) and one stationary (laying) class, while using the remainder semi-supervised. For the ECG dataset we used two hiding schemes: (1) omitting all normal heart beats, and (2) omitting classes Q and V. Within the electric devices dataset, we used two settings: (1) hiding classes 1–3, and (2) hiding classes 4–7. This was chosen arbitrarily, since there is no inherent split from the data[2]. When hiding classes with different sizes, we used 20 % and 50 % of each class. That is, we did not use subsampling. Finally, each time series was scaled via standard scaling/z-normalization.

For all settings and datasets we tested different types of networks. The first one is the fully convolutional approach, using a convolutional feature extractor and decoder (SuSL4TS). In the second setting, we tested a fully connected model, using MLPs in

[1] Downloaded at https://timeseriesclassification.com

[2] Classes are only labeled 1–7, while the mapping to the named version is missing.

Table 2: Hyperparameter search spaces. Optimization is done with *optuna* [2] for 60 runs in each experiment.

Parameter	Search space
$\|C_a\|$	randint(0,100,10)
w	{0}∪ 10**randint(-10,0)
lr	loguniform(10^{-6}, 10^{-1})
α	10**randint(0,10)
z	randint(10,100,10)
γ	10**randint(0,10)
layers	randint(1,3)
filters	2**randint(5,7)
units	2**randint(5,11)
kernel size	[3, 5, 7]
clipping	10**randint(-10, 0)

the encoder, as well as the decoder (MLP SuSL). We also experimented with a mixture (convolutional encoder and linear decoder) but found its performance to consistently lay between the other two settings.

Implementation. All networks were implemented using the PyTorch [20] framework. We chose the Adam optimizer [13] for training and performed a Bayesian hyperparameter search using optuna [2] for each learning paradigm and dataset. The search space for each parameter can be seen in Table 2. The batch size was fixed to 512, meaning that each batch contained 512 labeled examples and the same amount of unlabeled examples. In case of a size mismatch of labeled and unlabeled data, we re-sampled the smaller subset to fill the batches. We used a cosine annealing learning rate scheduler and trained for 100 epochs. Predictions on the test set were done using weights of the last epoch resulting in the best accuracy on the validation set (= 20 % of the training set). We increase λ every epoch with a step size of .1, with a maximum of 1.

6 RESULTS & DISCUSSION

The results of our experiments described in Section 5 can be seen in Table 3. Some tables are available in the online appendix [3] as they only quantify the depicted observations.

Human Activity Recognition. When taking a closer look at the HAR dataset, we see that our approach is not able to perform on par with the fully supervised baseline (SVM, 92 vs 96). However, the SVM performs on extracted features of the signal, while we directly use the raw signal. The main difference in performance can be attributed to the classes sitting and standing, as can be seen in Table 6f. They are easily confused with each other.

But even with fewer labels (SSL, 20 % and 50 %), we achieve almost the same classification performance as with all labels (91 vs 92). Surprisingly the accuracy is higher with fewer labeled samples (92 vs 93), which can be attributed to the better recall with the class standing.

[3] https://github.com/LSX-UniWue/SuSL4TS

In the unlabeled setting (i.e. time series clustering), the performance drops significantly (64 vs 92). Furthermore, there is no difference between the fully connected approach and the convolutional one.

In the first semi-unsupervised setting (walking classes hidden), we can see worse performance in both (20 % and 50 %) settings than compared to the unsupervised clustering. This drop in accuracy is due to the fact that all walking related classes are classified as walking, and even standing is classified as walking (see Table 6b). In contrast to the unsupervised model within the semi-unsupervised task, both parts of the encoder, the labeled and unlabeled leg, have to be trained, implying more weights need to be tuned for the network with only few labeled samples left. This setting is thus more complicated than the unsupervised task and the models tend to group unknown classes into one unknown super-class.

On the other hand, when hiding all stationary classes (i.e. standing, laying and sitting), performance increases again (51 vs 88). Wrongfully assigned classes are mostly confusion of sitting and standing (see Table 6c).

In the last semi-unsupervised setting (hiding walking and laying), performance is a little lower than hiding all static classes (77 vs 88). The drop in performance is due to the fact that the class standing is completely missed and most samples are classified as walking (see Table 6d).

All discussed observations for the HAR dataset are also visible in the visualization of the latent space shown in Figure 2. We used UMAP [19](min_dist=0.99, n_neighbors=10, metric=cosine) to plot a two dimensional manifold of the learned embedding.

ECG Heartbeat Classification. The classification results for the univariate ECG dataset are displayed in Table 3. When comparing with the fully supervised CNN architecture presented in [11], both semi-unsupervised approaches outperform the baseline (98 vs 96).

In the semi-supervised setting (20 % and 50 %, all classes known), there is almost no difference in terms of accuracy compared to the supervised settings. Again, accuracies are slightly higher when using fewer labels. Compared to the fully connected SuSL model, the convolutional feature extractor fares slightly better (97 vs 98).

In the unsupervised setting we can observe a performance drop, although not as high as for the HAR dataset (83 vs 63). Due to the highly skewed nature of the dataset, the unsupervised classification task is not much better than the majority vote (82.8 %), suggesting that only the normal class was detected (see Table 8a).

In the first semi-unsupervised setting (classes Q and V hidden), we can see a slight increase in performance in comparison to the unsupervised setting (84 vs 85). Both hidden classes are missed in the test set classification, where the differences in accuracies are founded in better recall of classes F and S (see Tables 8b and 8c).

On the other hand, when hiding the normal class, performance increases above the majority vote (88 vs 83). The amount of available labels (20 % and 50 %) does not impact the predictions largely (88 vs 90). However, with 50 % available labels, the class F is not missed, class V is missed in both settings (see Tables 8d and 8e).

Electric Devices. The classification results for the univariate electric devices dataset are displayed in Table 3. When comparing with the best performing model (BOSS) in the supervised settings, both

Table 3: Results. Each block describes one dataset, and is subdivided with the baseline methods. For reference, we include the majority vote baseline, a fully supervised baseline, and a semi-unsupervised baseline with an MLP on the raw time series. For SuSL4TS, we provide performance of our model in the different learning paradigms, with two versions of the semi-supervised and semi-unsupervised. We report the accuracy on the test set for unsupervised (UL), semi-supervised (SSL), semi-unsupervised (SuSL) and supervised (SL) learning paradigms.

Dataset	Model	UL	SuSL		SSL		SL
		0.00 %	20.00 %	50.00 %	20.00 %	50.00 %	100.00 %
HAR	Majority Vote						18.22
	Baseline (SVM,[4])						96.40
	Baseline (MLP SuSL,[27, 28])	64.20			83.30	87.82	89.82
	SuSL4TS	65.38			92.70	91.72	92.33
	SuSL4TS (movement (h))		50.93	51.74			
	SuSL4TS (stationary (h))		87.98	87.81			
	SuSL4TS (walking,laying (h))		60.84	77.63			
ECG	Majority Vote						82.76
	Baseline (CNN,[11])						96.40
	Baseline (MLP SuSL,[27, 28])	84.56			96.94	97.60	98.21
	SuSL4TS	83.28			97.51	97.28	97.61
	SuSL4TS (q,v (h))		84.70	84.25			
	SuSL4TS (n (h))		88.26	90.41			
El. Devices	Majority Vote						24.23
	Baseline (BOSS,[26])						79.92
	Baseline (MLP SuSL,[27, 28])	53.33			51.42	54.21	58.34
	SuSL4TS	52.72			69.69	68.02	69.96
	SuSL4TS (1–3 (h))		54.59	59.14			
	SuSL4TS (4–7 (h))		49.51	59.61			

neural network approaches perform worse (80 vs 70), where most mis-classification are done within the classes 3–5 (see Table 7h).

In the semi-supervised setting (20 % and 50 %, all classes known), we observe the same behavior as in the ECG datasets, classification accuracy remains similar to the fully supervised setting (68 vs 70). Since class 7 is completely missed (20 %), the model likely overfitted, since this class is detected in the validation set (see Tables 7f and 7g).

In the unsupervised classification task, we can observe that two classes are missed in the test set (1, 7), but even predictions within the other classes are not very clear (52 vs 70). The missed classes only represent smaller portions of this dataset, suggesting the parameter search found models yielding higher accuracy when predicting mostly the larger classes (see Table 7a).

In the first semi-unsupervised setting (classes 1–3 hidden), we observe a larger dip in accuracy compared to the semi-supervised setting (55/59 vs 70). Within this dataset we can see an increase in performance when comparing the setting with 20 % available labels, in contrast to the 50 % configuration (55 vs 59, see Tables 7b and 7c). In both settings, one class is completely missed (1 or 7), but the main difference in performance is the higher precision at all other classes with 50 % labels available.

On the other hand, when hiding classes 4–7, we can observe a similar decline in performance (50/60 vs 70). With 20 % labels available, the model assigns mostly all missed samples to the same hidden class (5), while missing classes 1, 4 and 6–7 completely (see Table 7d). When presenting the model more labels, accuracy once again increases (similar to classes 1–3 hidden), and only class 1 is completely missed (see Table 7e).

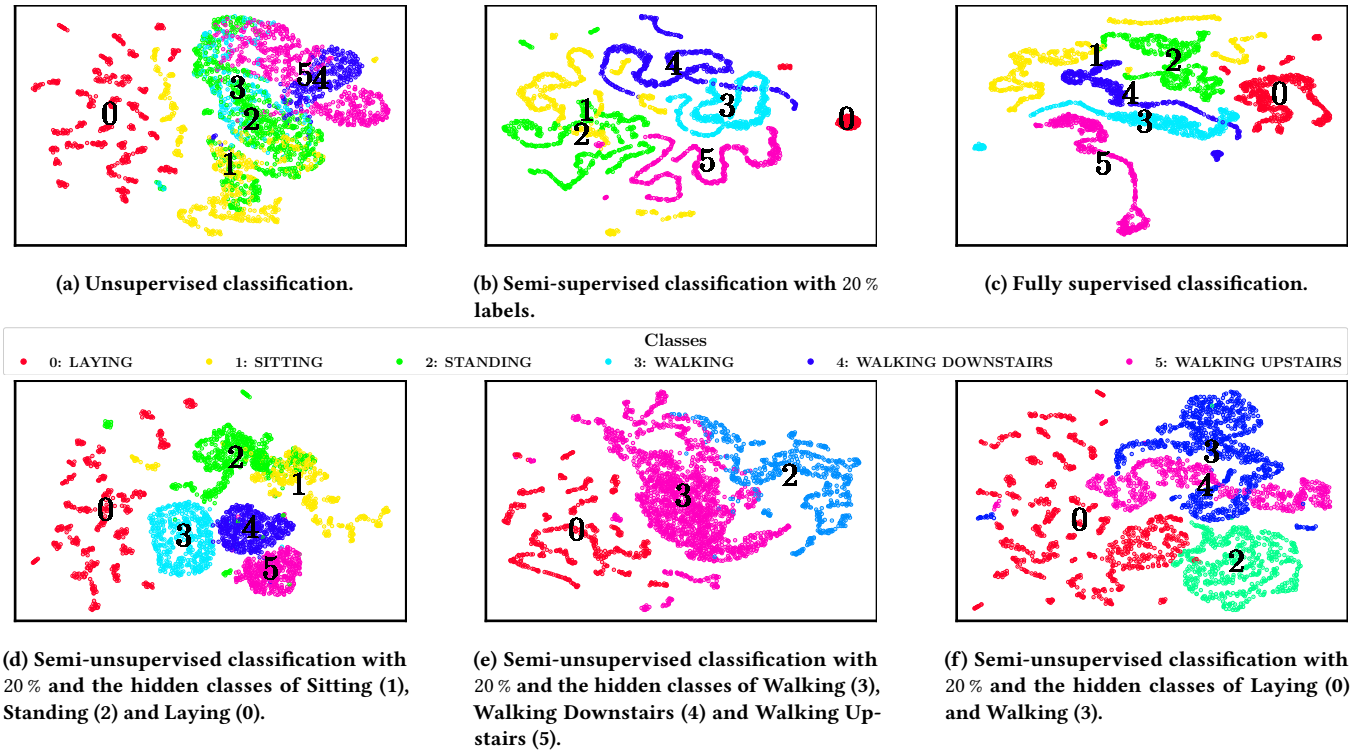
General Discussion. Throughout all datasets and settings, we have made some observations applicable in general, which we will discuss now.

(1) In the multivariate dataset (i.e. HAR), the convolutional approach of SuSL4TS outperforms the fully connected version for semi-unsupervised learning. That is, it performs better in any setting we tested.

(2) In the univariate datasets, we can see a mixed picture. For the electric devices dataset, SuSL4TS performs better in any labeled setting by a large margin. Given the ECG dataset, both versions perform equally well, with no clear tendency.

(3) If specific classes are not known, we can see drastically different results in the classification. Most prominent in the HAR dataset, as hiding the walking classes collapses predictions to perform worse than the unsupervised setting.

(4) In general the semi-unsupervised setting, only when using the larger amount of 50 % labels available, we can see an increased performance compared to the unsupervised settings with no labels at all (except HAR with all movements hidden).



(a) Unsupervised classification.

(b) Semi-supervised classification with 20 % labels.

(c) Fully supervised classification.

(d) Semi-unsupervised classification with 20 % and the hidden classes of Sitting (1), Standing (2) and Laying (0).

(e) Semi-unsupervised classification with 20 % and the hidden classes of Walking (3), Walking Downstairs (4) and Walking Upstairs (5).

(f) Semi-unsupervised classification with 20 % and the hidden classes of Laying (0) and Walking (3).

Figure 2: Embedding visualization. UMAP dimensionality reduction of the learned latent space on the test set for the HAR dataset.

(5) Generally the semi-supervised (i.e. all classes known, only limited amount of labels) performance is as good as the fully supervised setting.

7 SUMMARY

In this paper we presented SuSL4TS, a convolutional Gaussian mixture model for performing semi-unsupervised time series classification. We showed the efficacy of our approach by comparing it with optimized methods on several benchmark datasets, while requiring no manual feature extraction. Especially in the semi-supervised settings, the model performs nearly as good as its fully supervised counterpart. When omitting specific classes (i.e. classes unknown a priori) , accuracy can highly deviate in certain combinations of labeled versus unlabeled data, showing lower performance than using no labels at all.

In future work, we will analyze the applicability of our approach in real world, large scale data. For example, we could test the highly skewed sensor data obtained from beehives [8, 16, 29], or other highly skewed anomaly detection datasets [22], alleviating the burden of having to manually discern the different types of anomalies. On the other hand, we could use the normal class completely unlabeled and only annotate a few anomaly classes. This dataset is similar to the presented ECG analysis. In a more complex setting of time series classification, we could try to classify audio files, either with extracted mel-spectrograms or the raw series [8, 12, 21].

As mentioned, SuSL4TS is a generative model, thus we can draw random samples resembling the learned classes from the latent space. That enables us to generate samples for a given class to be used for other tasks or augment the labeled set in the whole dataset.

REFERENCES

[1] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. 2015. Time-series clustering–a decade review. *Information Systems* 53 (2015), 16–38.
[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
[3] Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE* 2, 1 (2015), 1–18.
[4] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra Perez, and Jorge Luis Reyes Ortiz. 2013. A public domain dataset for human activity recognition using smartphones. In *Proceedings of the 21th international European symposium on artificial neural networks, computational intelligence and machine learning*. 437–442.
[5] Anthony Bagnall, Luke Davis, Jon Hills, and Jason Lines. 2012. Transformation based ensembles for time series classification. In *Proceedings of the 2012 SIAM international conference on data mining*. SIAM, 307–318.
[6] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. 2017. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery* 31, 3 (2017), 606–660.
[7] Padraig Davidson, Florian Buckermann, Michael Steininger, Anna Krause, and Andreas Hotho. 2021. Semi-unsupervised Learning: An In-depth Parameter Analysis. In *German Conference on Artificial Intelligence (Künstliche Intelligenz)*. Springer, 51–66.
[8] Padraig Davidson, Michael Steininger, Florian Lautenschlager, Konstantin Kobs, Anna Krause, and Andreas Hotho. 2020. Anomaly detection in beehives using deep recurrent autoencoders. *arXiv preprint arXiv:2003.04576* (2020).

[9] Angus Dempster, François Petitjean, and Geoffrey I Webb. 2020. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery* 34, 5 (2020), 1454–1495.
[10] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. 2020. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery* 34, 6 (2020), 1936–1962.
[11] Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh. 2018. ECG Heartbeat Classification: A Deep Transferable Representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE. https://doi.org/10.1109/ichi.2018.00092
[12] Stefan Kahl, Mary Clapp, W Hopping, Hervé Goëau, Hervé Glotin, Robert Planqué, Willem-Pier Vellinga, and Alexis Joly. 2020. Overview of birdclef 2020: Bird sound recognition in complex acoustic environments. In *CLEF 2020-11th International Conference of the Cross-Language Evaluation Forum for European Languages*.
[13] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[14] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. *Advances in neural information processing systems* 27 (2014).
[15] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).
[16] Armands Kviesis and Aleksejs Zacepins. 2016. Application of neural networks for honey bee colony state identification. In *2016 17th International Carpathian Control Conference (ICCC)*. IEEE, 413–417.
[17] Jason Lines, Sarah Taylor, and Anthony Bagnall. 2018. Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles. *ACM Transactions on Knowledge Discovery from Data* 12, 5 (2018).
[18] Benjamin Lucas, Ahmed Shifaz, Charlotte Pelletier, Lachlan O'Neill, Nayyar Zaidi, Bart Goethals, François Petitjean, and Geoffrey I Webb. 2019. Proximity forest: an effective and scalable distance-based classifier for time series. *Data Mining and Knowledge Discovery* 33, 3 (2019), 607–635.
[19] Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).
[20] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).
[21] Karol J Piczak. 2015. ESC: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*. 1015–1018.
[22] Hansheng Ren, Bixiong Xu, Yujing Wang, Chao Yi, Congrui Huang, Xiaoyu Kou, Tony Xing, Mao Yang, Jie Tong, and Qi Zhang. 2019. Time-series anomaly detection service at microsoft. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 3009–3017.
[23] Alejandro Pasos Ruiz, Michael Flynn, James Large, Matthew Middlehurst, and Anthony Bagnall. 2021. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery* 35, 2 (2021), 401–449.
[24] Patrick Schäfer. 2015. The BOSS is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery* 29, 6 (2015), 1505–1530.
[25] Patrick Schäfer and Ulf Leser. 2017. Multivariate time series classification with WEASEL+ MUSE. *arXiv preprint arXiv:1711.11343* (2017).
[26] Ahmed Shifaz, Charlotte Pelletier, François Petitjean, and Geoffrey I Webb. 2020. TS-CHIEF: a scalable and accurate forest algorithm for time series classification. *Data Mining and Knowledge Discovery* 34, 3 (2020), 742–775.
[27] Matthew Willetts, Aiden Doherty, Stephen Roberts, and Chris Holmes. 2018. Semi-unsupervised learning of human activity using deep generative models. *arXiv preprint arXiv:1810.12176* (2018).
[28] Matthew Willetts, Stephen Roberts, and Chris Holmes. 2020. Semi-unsupervised learning: clustering and classifying using ultra-sparse labels. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 5286–5295.
[29] Aleksejs Zacepins, Armands Kviesis, Egils Stalidzans, Marta Liepniece, and Jurijs Meitalovs. 2016. Remote detection of the swarming of honey bee colonies by single-point temperature monitoring. *Biosystems engineering* 148 (2016), 76–80.

APPENDIX

(a) Class distribution in the HAR dataset

Class	Training samples	Test samples	Σ
Walking	1226	496	1722
Walking up	1073	471	1544
Walking down	986	420	1406
Sitting	1286	491	1777
Standing	1374	532	1906
Laying	1407	537	1944
	7352	2947	10 299

(b) Class distribution in the MIT-BIH dataset

Class	Training samples	Test samples	Σ
N	72 471	18 118	90 589
S	2223	557	2780
V	5788	1448	7236
F	641	162	803
Q	6431	1607	8038
	87 554	21 892	109 446

(c) Class distribution in the electric devices dataset

Class	Training samples	Test samples	Σ
1	727	667	1394
2	2231	1956	4187
3	851	755	1606
4	1474	1165	2639
5	2406	1869	4275
6	509	743	1252
7	728	556	1284
	8926	7711	16 637