Adaptive Resolution Loss: An Efficient and Effective Loss for Time Series Hierarchical Contrastive Self-Supervised Learning Framework

Kevin Garcia
Department of Computer Science
The University of Texas Rio Grande Valley
Edinburg, TX 78539
kevin.garcia09@utrgv.edu

Juan Manuel Perez
Department of Computer Science
The University of Texas Rio Grande Valley
Edinburg, TX 78539
juan.m.perez02@utrgv.edu

Yifeng Gao
Department of Computer Science
The University of Texas Rio Grande Valley
Edinburg, TX 78539
yifeng.gao@utrgv.edu

Abstract

Time series data is a crucial form of information that has vast opportunities. With the widespread use of sensor networks, large-scale time series data has become ubiquitous. One of the current state-of-the-art SSL frameworks in time series is called ts2vec. ts2vec specially designs a hierarchical contrastive learning framework that uses loss-based training, which performs outstandingly against benchmark testing. However, the computational cost for ts2vec is often significantly greater than other SSL frameworks. In this paper we present a new self-supervised learning loss, named adaptive resolution loss. The proposed solution reduces the number of resolutions used for training the model via an adaptive selection score, leading to an efficient adaptive resolution loss based learning algorithm. In the experiment, we demonstrate that the proposed method preserves the original model's integrity while significantly enhancing its training time.

1 Introduction

Time series data is a crucial form of information that has vast opportunities [6, 14, 19, 12, 1, 15, 18] . Recently, with the widespread use of sensor networks, large-scale time series data have become ubiquitous. Such data gives us a dense amount of valuable information. The task of mining time series could help us harvest important trends, patterns, and crucial behaviors, which ultimately benefit various applications. One of the most prominent problems in time series data mining is representation learning: transforming time series into low-dimensional representations that can represent their semantic similarity while benefiting various downstream tasks[20]. Recently, with the introduction of self-supervised learning frameworks (SSL) for image, video, and natural language representation learning [21, 13, 10, 2, 9, 3, 8], numerous research has focused on designing an effective SSL for time series data. One of the current state-of-the-art SSL frameworks in time series is called ts2vec [20]. ts2vec specially designs a hierarchical contrastive learning framework that uses loss-based training, which performs outstandingly against benchmark testing.

While ts2vec outperforms existing state-of-the-art models, the model's computational cost is much heavier than other self-supervised learning frameworks. It utilizes hierarchical enumeration to compute the loss in each resolution of the time series, which significantly increases the computational burden. In this work, on top of the existing ts2vec framework, we propose an adaptive single resolution-based loss function to train the model. We observe that the loss generated by each time

37th Conference on Neural Information Processing Systems. Self-Supervised Learning: Theory and Practice Workshop

series resolution are highly correlated. Therefore, we proposed a strategy to adaptive selects the most important resolutions throughout the training, which can optimize the overall loss while reducing the computation cost. Our empirical findings indicate the overall improvement of data sets with the proposed implementation.

2 Methodology

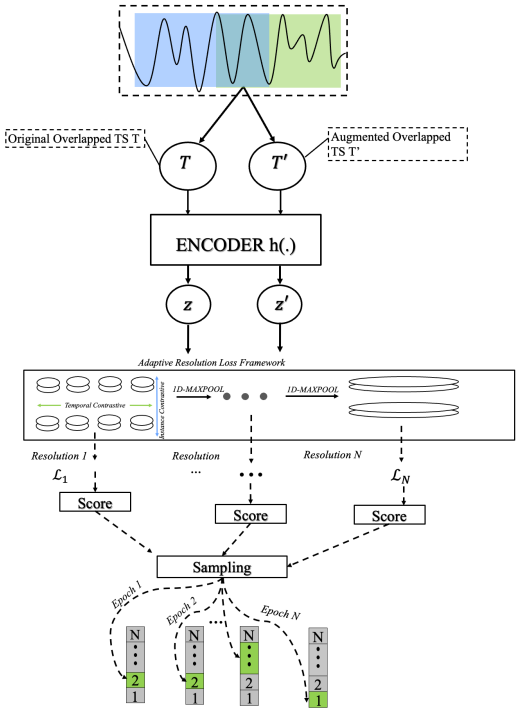


Figure 1: Proposed Framework: Starting with original and augmented time series T and T', the data is encoded into representations z and z'. The framework computes loss across multiple resolutions using Temporal Contrastive methods combined with 1-Dimensional MaxPooling. The framework calculates losses $\boldsymbol{\mathcal{L}_1}$ to $\boldsymbol{\mathcal{L}_N}$ across multiple resolutions. Each loss is associated with a score that evaluates its significance. Based on these scores, the adaptive sampling decides which resolution's loss to focus on for a specific training epoch.

Our proposed framework is illustrated in Fig. 1. Given B number of D-dimensional multivariate time series data of length N ($X \in R^{B \times D \times N}$), following the ts2vec training framework [20], we first perform cropping based augmentation operation. Specifically, the augmentation operator randomly crops two views T and T' where $T \cap T' \neq \emptyset$ from every time series X (shown in blue and green in the Fig. 1.top). T and T' are then passed through an encoder $h(.)$ to generate embedding sequence ($z = h(T)$ and $z' = h(T')$, where $z \in R^{M \times D' \times n}$ and $z' \in R^{M \times D' \times n'}$). Then the hierarchical contrastive learning force to push the latent representation of the *overlapping region* of T and T' to be similar (the overlapped region embedding denoted as d and d' respectively). Specifically, given a resolution r, the latent embedding sequence d and d' are first down-sample to resolution r (e.g. d downsampled to a sequence only consists r time stamps) and perform 1) **temporal-wise contrastive loss**, which consider the embedding at differing timestamps as negative samples and 2) **instance-wise contrastive loss**, which consider other time series instances as negative samples:

$$\ell_r^{temp} = \sum_i \sum_t -log \frac{exp(d_{i,t} \cdot d'_{i,t})}{\sum_{t'} \Big(exp(d_{i,t} \cdot d'_{i,t'}) + \mathbb{1}_{[t \neq t']} exp(d_{i,t} \cdot d_{i,t'}) \Big)} \tag{1}$$

$$\ell_r^{inst} = \sum_i \sum_t -log \frac{exp(d_{i,t} \cdot d'_{i,t})}{\sum_{j=1}^{B} \Big(exp(d_{i,t} \cdot d'_{j,t}) + \mathbb{1}_{[i \neq j]} exp(d_{i,t} \cdot d_{j,t}) \Big)} \tag{2}$$

Afterwards, both losses are then added together to create an overall loss value:

$$\mathcal{L}_r = \alpha \ell_r^{temp} + (1-\alpha)\ell_r^{inst} \tag{3}$$

where α is a hyper parameter control the importance of temporal and instance loss. In ts2vec model, the training loss is defined as

$$\mathcal{L}^{hier} = \sum_{i=1, i<\log m} \mathcal{L}_{2^i} \tag{4}$$

.

where m denotes the length of input embedding sequence d or d', in order to capture the semantic similarity of time series in multi-resolutions.

However, each $\mathcal{L}_r$ is highly correlated because the losses are aggregation from same resources. In fact, we found that during the training, by training $\mathcal{L}_r$ of a fixed resolution, the loss in multiple resolutions are improved as well. Meaning, by optimizing one loss value, the aggregation is then changed thus changing it's entire loss value for the subsequent epoch to base itself from. Therefore, we propose an adaptive loss selection approach to adaptive choose the loss computed in a specific resolution to train, instead of training loss of every resolution in every epoch. By doing so, we enhance the subsequent training time of the model while maintaining the same model performance as optimizing $\mathcal{L}^{(hier)}$.

Specifically, we measure the **loss evolving trend**, as an indicator of whether the loss is indirectly co-trained in a specific epoch when optimizing loss of another resolution, and utilizes it to determine which $\mathcal{L}_r$ will be picked in the next epoch. Intuitively, throughout the training, the loss tends to trend downward when the $\mathcal{L}_r$ is indirectly trained and the loss is either plateaus or it increases when $\mathcal{L}_r$ cannot be indirectly trained. Specifically, given the loss value of $\mathcal{L}_{r',e}$ in the e_{th} epoch of resolution i, we measure this co-training behavior based on:

$$s_i = \frac{\exp(\mathcal{L}_{i,e} - \mathcal{L}_{i,e-1})}{\sum_{r\in\mathcal{D}} \exp(\mathcal{L}_{r,e} - \mathcal{L}_{r,e-1})} \tag{5}$$

where $\mathcal{D}$ denotes all resolutions used to compute $\mathcal{L}^{hier}$ Eq. 5 measures whether the loss can be indirectly trained and normalize the value scale. In the proposed work, we essentially pick the loss value that cannot be indirectly trained in current epoch to optimize. Specifically, the algorithm will sample one resolution r' from a multinomial distribution:

$$r' \sim Multinomial(s) \tag{6}$$

Finally, the training loss used in the $e+1$ epoch is:

$$\mathcal{L}^a = \mathcal{L}_{r'} \tag{7}$$

In summary, in each epoch, we use the current loss trend to pick a new resolution to train in the next epoch. By adaptively selecting the resolution and by fully utilizing the correlation between losses to only train the model, we are able to efficiently train the overall hierarchical loss $\mathcal{L}^{hier}$.

3 Experiment

3.1 Experiment Setup

We compare our proposed method with the original ts2vec framework through classification accuracy performance, as well as training time comparisons. Following the evaluation protocol adopted in ts2vec, we use the trained model $h(.)$ to convert the multivariate time series into K dimension representation. Then, the latent embedding is applied with a logistical regression classifier to perform the classification task. We use ten longest UEA/UCR multi-variate time series data sets excluding the longest data set, Eignworm due to memory issue. The experiments conducted in Google Colab with an NVIDIA T4 in most of the data and A100 GPU was used if T4 cannot fulfilled the computation resource. Both models are trained in the same environment. We evaluate the effective by classification accuracy and evaluate the efficiency by the training time. For all comparison experiments, we repeated experiments five times for each dataset and reported the average performance.

3.2 Comparison Evaluation

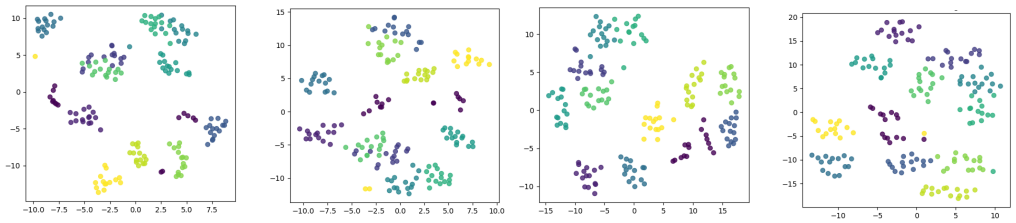
The comparison results is shown in Table 1. From the table, training through both loss functions achieve very similar accuracy performance in all datasets. This indicate the proposed loss $\mathcal{L}^a$ can achieves similar downstream task performance compared with using original loss $\mathcal{L}^{hier}$. Besides, we also observed an increased efficiency throughout all of tested data sets. On average, we improved the original model's training time by on average 40.21%. Moreover, we observed a slightly more improvements when analyzing data sets of larger length. Overall, the result shows that our proposed approach is successful at improving original model's efficiency, while maintaining the integrity of its accuracy.

		Proposed $\mathcal{L}^a$		Original $\mathcal{L}^{hier}$	
Dataset	Length	Execution Time (s)	acc.	Execution Time (s)	acc.
HandMovementDirecton	400	**28.87**	0.26	43.24	**0.28**
HeartBeat	405	**32.55**	**0.74**	52.92	0.73
AtrialFibrilation	640	**22.65**	**0.29**	35.12	**0.29**
SelfRegulationSCP1	896	**84.94**	**0.80**	171.22	**0.80**
PhoneMe	1024	**336.85**	0.14	506.38	**0.15**
SelfRegulationSCP2	1152	**86.40**	**0.54**	173.86	0.53
Cricket	1197	**55.84**	**0.91**	100.40	**0.91**
EthanolConcentration	1751	**389.92**	**0.28**	624.04	0.27
StandWalkJump	2000	**45.90**	**0.51**	65.75	0.44
MotorImagery	3000	**665.68**	**0.53**	1297.92	0.52
Total Time	-	**1749.6**	-	3070.85	-

Table 1: Comparsion Experiment Result in 10 long multivariate time series data

3.3 Embedding Visualization

To better understand the performance of our proposed loss, we visualize both the training data embeddings, and testing data embeddings generated by the model. We uses two types of embedding - average pooling and flattened embedding. The t-Distributed Stochastic Neighbor Embedding (t-SNE)[17] visualization is shown in Fig. 2. The color of each points indicates its class label. From the figure, the proposed loss can obtain similar data embeddings as the original $\mathcal{L}^{hier}$



(a) Proposed avg. pooling (b) Original avg. pooling (c) Proposed flattened (d) Original flattened

Figure 2: t-SNE visualization of Embedding Space for Cricket Dataset

3.4 Conclusion

In this paper, we proposed a method to improve the computational efficiency of the ts2vec model for time series representation learning. Our method involves the use of an adaptive resolution setting, in the model's loss function which allows us to reduce the computational load of the training process without sacrificing the model's performance. Our experimental results confirm our proposed method is effective. Our model achieved similar classification accuracy in a range of 10 UCR/UEA datasets while consistently reducing the training time. These findings suggest that our method can be a valuable tool for researchers and practitioners working with large-scale time series data.

4 Acknowledge

This work is supported by the National Science Foundation (NSF) under grant 2318682

References

[1] Stefan Baisch and Götz HR Bokelmann. Spectral analysis with incomplete time series: an example from seismology. *Computers & Geosciences*, 25(7):739–750, 1999.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[3] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[4] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. *arXiv preprint arXiv:2106.14112*, 2021.

[5] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

[6] André Gensler, Janosch Henze, Bernhard Sick, and Nils Raabe. Deep learning for solar power forecasting—an approach using autoencoder and lstm neural networks. In *2016 IEEE international conference on systems, man, and cybernetics (SMC)*, pages 002858–002865. IEEE, 2016.

[7] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[8] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[9] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[10] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in neural information processing systems*, 32, 2019.

[11] Eric Jang, Coline Devin, Vincent Vanhoucke, and Sergey Levine. Grasp2vec: Learning object representations from self-supervised grasping. *arXiv preprint arXiv:1811.06964*, 2018.

[12] Argyro Kampouraki, George Manis, and Christophoros Nikou. Heartbeat time series classification with support vector machines. *IEEE transactions on information technology in biomedicine*, 13(4):512–518, 2008.

[13] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020.

[14] Hussein Sharadga, Shima Hajimirza, and Robert S Balog. Time series forecasting of solar power generation for large-scale photovoltaic plants. *Renewable Energy*, 150:797–807, 2020.

[15] Tetsuo Takanami and Genshiro Kitagawa. Estimation of the arrival times of seismic waves by multivariate time series model. *Annals of the Institute of Statistical mathematics*, 43(3):407–433, 1991.

[16] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. *arXiv preprint arXiv:2106.00750*, 2021.

[17] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[18] Panayiotis Varotsos, Nicholas V Sarlis, and Efthimios S Skordas. *Natural time analysis: the new view of time: precursory seismic electric signals, earthquakes and other complex time series*. Springer Science & Business Media, 2011.

[19] Jin Wang, Ping Liu, Mary FH She, Saeid Nahavandi, and Abbas Kouzani. Bag-of-words representation for biomedical time series classification. *Biomedical Signal Processing and Control*, 8(6):634–644, 2013.

[20] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8980–8987, 2022.

[21] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1476–1485, 2019.

Supplementary Materials

A Related Work

Recently, the self-supervised learning (SSL) framework [21, 13, 10, 2, 9, 3, 8] is introduced for vision representation learning in the research domain of computer vision. The goal of SSL is to train a deep learning model to understand the semantic-level invariance characteristic through carefully designed pretext tasks from high-level semantic understanding related to image data (e.g. learning rotation-invariant representation for images) [21, 11, 7]. Recently, an increasing amount of time series representation learning research have been focused on designing the self-supervised deep learning framework [20, 5, 16, 4]. Most models are designed based on the unsupervised contrastive learning framework SimCLR [2].

Franceschi et al.[5] introduces an unsupervised contrastive learning framework by introducing a novel triplet selection approach based on segment's context. Similarly, Tonekaboni et al. proposed a framework named Temporal Neighborhood Coding (TNC) [16]. TNC aims to utilizes the temporal correlation along neighboring segments to learn the representation. Eldele et al. [4] introduces a Temporal and Contextual Contrast (TS-TCC) based framework. In TS-TCC, two types of augments, strong augmentation and weak augmentation are used to perform contrastive learning. Zhang et al. [**?**] proposes a time-frequency consistent loss for contrastive learning where temporal and frequency of the same neighborhoods pushed closer together for optimal loss accuracy. Yue et al. [20] proposed a framework named ts2vec. The proposed framework introduces a random cropping based augmentation and a hierarchical loss to stabilize the obtained embedding. It achieved significantly better performance compared with previous methods.However, we found the computational burden for ts2vec is also higher than existing works.

B Additional Details: Adaptive Resolution Loss

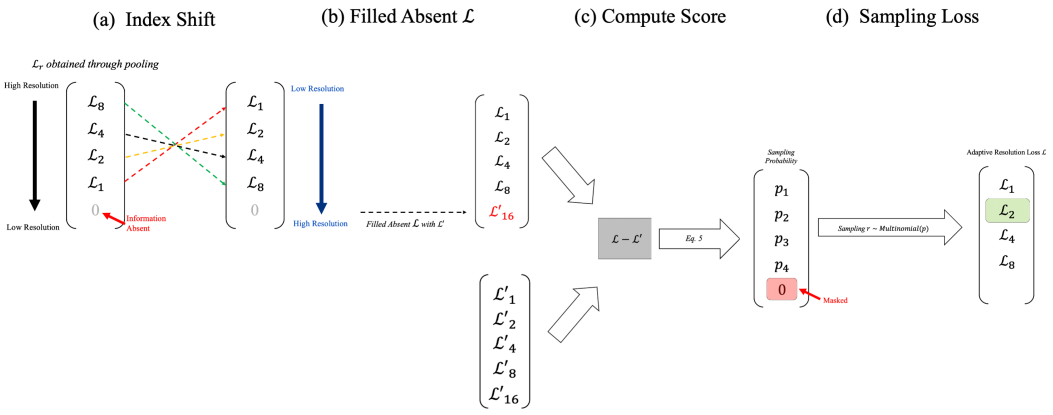


Figure 3: Adaptive Resolution Loss Overview: **(a)**: aligned resolution loss $\mathcal{L}$. **(b)**: Every absented loss value is replaced by its prior recorded loss $\mathcal{L}'$. **(c)**: Each loss is passed through Eq. 5 and generate probability. **(d)**:Sampling from the generated distribution.

B.1 Index shift function:

Since the overlapping area between T and T' can be an arbitrary length between $[1, L-1]$, the resolution indices r in different epochs do not represent the same resolution. Therefore, the proposed algorithm will first align each resolution across different epochs. In order to accurately compute the score, we design a flip and shift function that properly reorders the resolutions. Firstly, the algorithm reorders the resolutions and obtains the number of resolutions from the given dataset. Secondly, we interpolate the most current loss value for any resolution absent, as the respective resolutions must have a score assigned to them later in the process. By properly pre-aligning the array of resolutions, we can now suitably assign them a score value.

B.2 Score Function

Once the resolutions are aligned with each epoch, a score function is used to evaluate the importance of each resolution. Intuitively, at a given epoch, the algorithm computes the loss difference between current and previous epoch. Then the algorithm will compute the score via a softmax function based off of the respective resolution's loss. The score is correlated to the loss value given, the greater the loss the greater the score assigned. This generates a probability array that we can then use to select the most important resolution based on a random multinomial function. Note that this function ignores any absented resolution in the original loss array, which means this array is usually smaller than $\log L$ size.

B.2.1 Probability function:

The probability function is the final part of the adaptive resolution setting. It utilizes the probability array produced by the score function via softmax, to select the most weighted resolution. It does so through a multinomial distribution random variable sampling function, which makes a selection based on the given probabilities. Finally, the chosen value is then used to update the weights of the resolutions in the model. Giving the chosen resolution priority over the rest.

C Datasets

Each Datasets used for experimentation's characteristics are shown below.

Data Sets						
Name	Abbrev.	Train Size	Test Size	Length	No. of Classes	Type
HandMovementDirection	HMD	160	74	400	4	EEG
HeartBeat	HB	204	205	405	2	AUDIO
AtrialFibrilation	AF	15	15	640	3	ECG
SelfRegulationSCP1	SCP1	268	293	896	2	EEG
PhoneMe	PM	214	1896	1024	39	SOUND
SelfRegulationSCP2	SCP2	200	180	1152	2	EEG
Cricket	Cr	108	72	1197	12	HAR
EthanolConcentration	EC	261	263	1751	4	OTHER
StandWalkJump	SWJ	12	15	2500	3	ECG
MotorImagery	MI	278	100	3000	2	EEG

Table 2: Tested Datasets Information

D Additional Encoder Details

Our model's base architecture is adopted from the original TS2Vec model architecture, which comprises of three main components: an Input Projection Layer, Random Cropping Augmentation, and Time Stamp Masking Module.

D.1 Encoder $h(.)$

Unlike classical contrastive learning other field[2], the function $h(.)$ generates an embedding series z_i (i.e. defined as another time series with K-dimensional observations in each time step), instead of simple K-dimensional vector. The generated series will be used to compute the hierarchical contrastive loss used in TS2Vec. Note that only the overlapping region shared by z_i and z_i' are used to compute loss. Following the architecture in TS2Vec, $h(.)$ function is modeled through a dilated convolution neuron network without the final pooling and FCN layers.

D.2 Random Cropping Augmentation

Given an input time series, the model first generates two augmentations based on random cropping. Intuitively, the model generates two overlapped sub-sequences T and T' where $T \cap T' \neq \emptyset$.

D.3 Time Stamp Masking Module

: A random masking is applied to generate an augmented context view by masking latent vectors at randomly selected timestamps (via dropout). It essentially hides some of the information by creating a slightly different version of the data from the original, allowing the model to learn more robust representations.

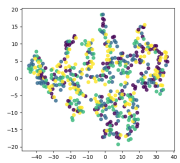
Both the cropped-and-masked sub-sequences will pass through encoder to obtain the embedding $z = h(T)$ and $z' = h(T')$, where $z \in R^{M \times D' \times n}$ and $z' \in R^{M \times D' \times n'}$)

D.4 Detailed Implementation Parameters

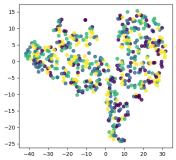
Throughout the experiment, we set embedding size $K = 16$. The number of dilated convolution layer to 2, and learning rate is $1e - 3$. The final embedding is computed through global average pooling across all timestamps during the comparison evaluation.

E Additional Result: Embedding Visualization results

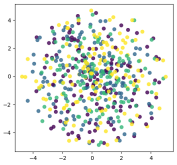
We provided t-SNE visualization of all the tested dataset as following:



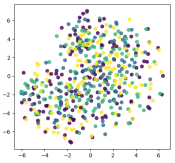
(a) EC TS2Vec (avg pool)



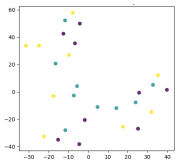
(b) EC Proposed (avg pool)



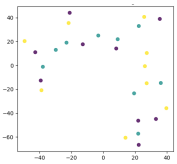
(c) EC TS2Vec (flatten)



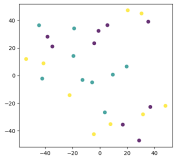
(d) EC Proposed (flatten)



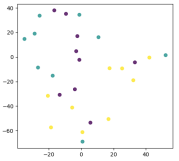
(e) SWJ TS2Vec (avg pool)



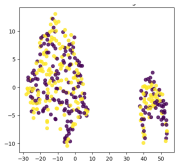
(f) SWJ Proposed (avg pool)



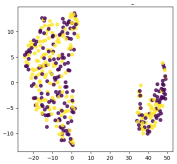
(g) SWJ TS2Vec (flatten)



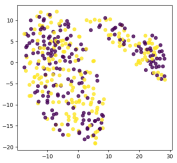
(h) SWJ Proposed (flatten)



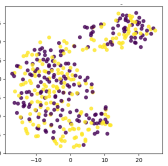
(i) MI TS2Vec (avg pool)



(j) MI Proposed (avg pool)

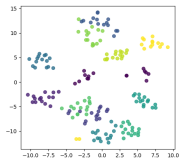


(k) MI TS2Vec (flatten)

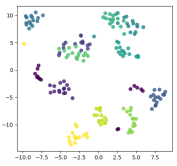


(l) MI Proposed (flatten)

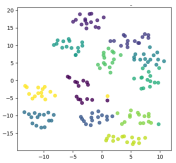
Figure 4: Visualizing embedding instances via t-SNE



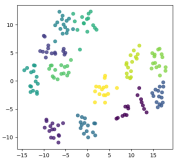
(a) Cr TS2Vec (avg pool)



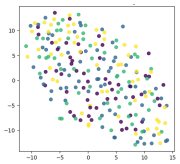
(b) Cr Proposed (avg pool)



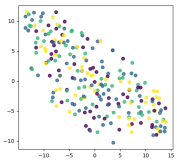
(c) Cr TS2Vec (flatten)



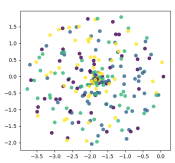
(d) Cr Proposed (flatten)



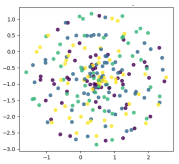
(e) HMD TS2Vec (avg pool)



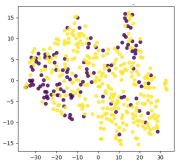
(f) HMD Proposed (avg pool)



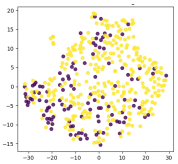
(g) HMD TS2Vec (flatten)



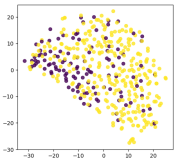
(h) HMD Proposed (flatten)



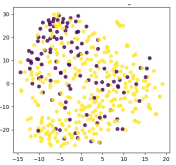
(i) HB TS2Vec (avg pool)



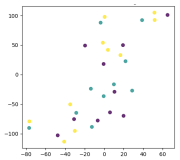
(j) HB Proposed (avg pool)



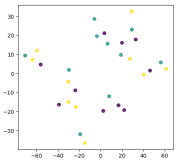
(k) HB TS2Vec (flatten)



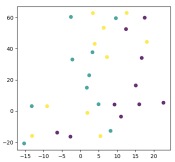
(l) HB Proposed (flatten)



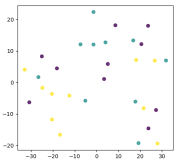
(m) AF TS2Vec (avg pool)



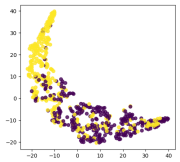
(n) AF Proposed (avg pool)



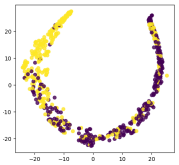
(o) AF TS2Vec (flatten)



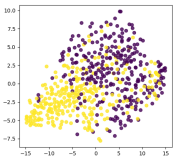
(p) AF Proposed (flatten)



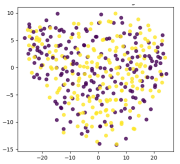
(q) SCP1 TS2Vec (avg pool)



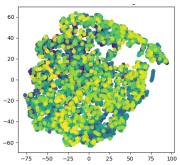
(r) SCP1 Proposed (avg pool)



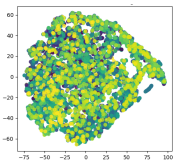
(s) SCP1 TS2Vec (flatten)



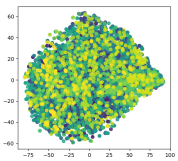
(t) SCP1 Proposed (flatten)



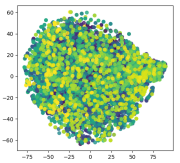
(u) PM TS2Vec (avg pool)



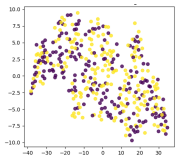
(v) PM Proposed (avg pool)



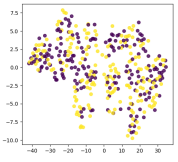
(w) PM TS2Vec (flatten)



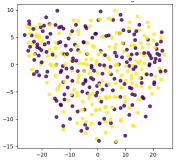
(x) PM Proposed (flatten)



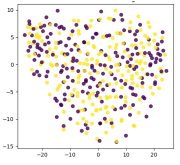
(y) SCP2 TS2Vec (avg pool)



(z) SCP2 Proposed (avg pool)



() SCP2 TS2Vec (flatten)



() SCP2 Proposed (flatten)

Figure 5: Visualizing embedding instances via t-SNE (CONT'D)