Paper Review Seminar

(2022. 11. 01)

Self-Supervised Learning (SSL) with Time Series (TS) Data

통계데이터사이언스학과통합과정 5학기 이승한

Papers

Unsupervised Scalable Representation Learning for Multivariate Time Series

Jean-Yves Franceschi*
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
jean-yves.franceschi@lip6.fr

Aymeric Dieuleveut
MLO, EPFL, Lausanne CH-1015, Switzerland
CMAP, Ecole Polytechnique, Palaiseau, France
aymeric.dieuleveut@polytechnique.edu

Martin Jaggi
MLO, EPFL, Lausanne CH-1015, Switzerland
martin.jaggi@epfl.ch

Abstract

Time series constitute a challenging data type for machine learning algorithms, due to their highly variable lengths and sparse labeling in practice. In this paper, we tackle this challenge by proposing an unsupervised method to learn universal embeddings of time series. Unlike previous works, it is scalable with respect to their length and we demonstrate the quality, transferability and practicability of the learned representations with thorough experiments and comparisons. To this end, we combine an encoder based on causal dilated convolutions with a novel triplet loss employing time-based negative sampling, obtaining general-purpose representations for variable length and multivariate time series.

https://arxiv.org/pdf/1901.10738.pdf

Unsupervised Representation Learning for Time Series with Temporal Neighborhood Coding

Sana Tonekaboni*
University of Toronto & Vector Institute
The Hospital for Sick Children
stonekaboni@cs.toronto.edu

Danny Eytan
The Hospital for Sick Children
biliary.colic@gmail.com

Anna Goldengerg
University of Toronto & Vector Institute
The Hospital for Sick Children
anna.goldenberg@utoronto.ca

Abstract

Time series are often complex and rich in information but sparsely labeled and therefore challenging to model. In this paper, we propose a self-supervised framework for learning generalizable representations for non-stationary time series. Our approach, called Temporal Neighborhood Coding (TNC), takes advantage of the local smoothness of a signal's generative process to define neighborhoods in time with stationary properties. Using a debiased contrastive objective, our framework learns time series representations by ensuring that in the encoding space, the distribution of signals from within a neighborhood is distinguishable from the distribution of non-neighboring signals. Our motivation stems from the medical field, where the ability to model the dynamic nature of time series data is especially valuable for identifying, tracking, and predicting the underlying patients' latent states in settings where labeling data is practically impossible. We compare our method to recently developed unsupervised representation learning approaches and demonstrate superior performance on clustering and classification tasks for multiple datasets.

https://arxiv.org/pdf/2106.00750.pdf

Papers

Unsupervised Scalable Representation Learning for Multivariate Time Series

Jean-Yves Franceschi*
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
jean-yves.franceschi@lip6.fr

Aymeric Dieuleveut
MLO, EPFL, Lausanne CH-1015, Switzerland
CMAP, Ecole Polytechnique, Palaiseau, France
aymeric.dieuleveut@polytechnique.edu

Martin Jaggi
MLO, EPFL, Lausanne CH-1015, Switzerland
martin.jaggi@epfl.ch

Abstract

Time series constitute a challenging data type for machine learning algorithms, due to their highly variable lengths and sparse labeling in practice. In this paper, we tackle this challenge by proposing an unsupervised method to learn universal embeddings of time series. Unlike previous works, it is scalable with respect to their length and we demonstrate the quality, transferability and practicability of the learned representations with thorough experiments and comparisons. To this end, we combine an encoder based on causal dilated convolutions with a novel triplet loss employing time-based negative sampling, obtaining general-purpose representations for variable length and multivariate time series.

https://arxiv.org/pdf/1901.10738.pdf

UNSUPERVISED REPRESENTATION LEARNING FOR TIME SERIES WITH TEMPORAL NEIGHBORHOOD CODING

Sana Tonekaboni*
University of Toronto & Vector Institute
The Hospital for Sick Children
stonekaboni@cs.toronto.edu

Danny Eytan
The Hospital for Sick Children
biliary.colic@gmail.com

Anna Goldengerg
University of Toronto & Vector Institute
The Hospital for Sick Children
anna.goldenberg@utoronto.ca

ABSTRACT

Time series are often complex and rich in information but sparsely labeled and therefore challenging to model. In this paper, we propose a self-supervised framework for learning generalizable representations for non-stationary time series. Our approach, called Temporal Neighborhood Coding (TNC), takes advantage of the local smoothness of a signal's generative process to define neighborhoods in time with stationary properties. Using a debiased contrastive objective, our framework learns time series representations by ensuring that in the encoding space, the distribution of signals from within a neighborhood is distinguishable from the distribution of non-neighboring signals. Our motivation stems from the medical field, where the ability to model the dynamic nature of time series data is especially valuable for identifying, tracking, and predicting the underlying patients' latent states in settings where labeling data is practically impossible. We compare our method to recently developed unsupervised representation learning approaches and demonstrate superior performance on clustering and classification tasks for multiple datasets.

https://arxiv.org/pdf/2106.00750.pdf

Unsupervised Scalable Representation Learning with MTS (2019)

1. Introduction
2. Triplet Loss
3. Triplet Loss with MTS
4. Encoder Architecture
5. Experiment

1. Introduction

Challenges in Time Series Data :

- (1) **highly variable lengths**
- (2) **sparse labeling** → need for UNSUPERVISED learning

1. Introduction

Challenges in Time Series Data :

- (1) **highly variable lengths**
- (2) **sparse labeling** → need for UNSUPERVISED learning

This paper proposes **“Unsupervised method to learn universal embeddings of time series”**

- **scalable w.r.t length**
- proposes …
  - (Architecture) Encoder based on **Causal Dilated Convolutions**
  - (Loss Function) **Novel Triplet Loss** for Time Series (via time-based negative sampling)
- demonstrate transferability of the learned representations

2. Triplet Loss

compare distance between (**Anchor** & **Positive**) and (**Anchor** & **Negative**)

$$\mathcal{L}(A, P, N) = \max\left(\| \mathrm{f}(A) - \mathrm{f}(P)\|^2 - \| \mathrm{f}(A) - \mathrm{f}(N)\|^2 + \alpha, 0\right)$$

A : Anchor input

P : Positive input

N : Negative input

α : Margin (between **positive pair** & **negative pair**)

- positive pair : (A, P)
- negative pair : (A, N)

f : Embedding function

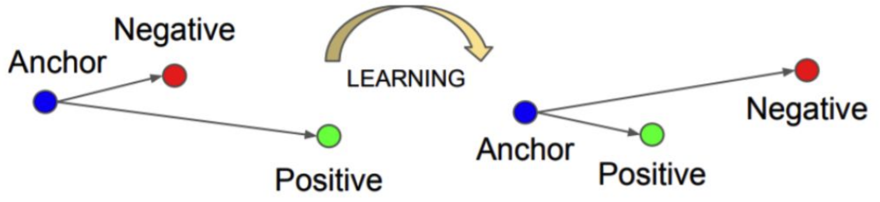


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

3. Triplet Loss with MTS

How to choose POS / NEG samples ?

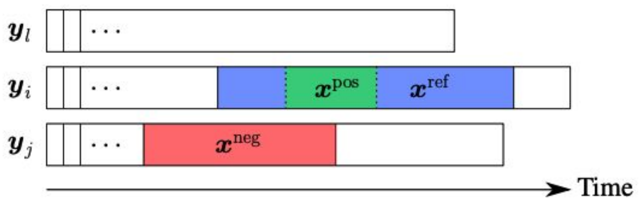


Figure 1: Choices of x^{ref}, x^{pos} and x^{neg}.

Unsupervised : no need for label of each TS

- originated from same TS (of anchor) : **POSITIVE**
- originated from different TS (of anchor) : **NEGATIVE**

3. Triplet Loss with MTS

How to choose POS / NEG samples ?

Algorithm 1: Choices of $\boldsymbol{x}^{\text{ref}}$, $\boldsymbol{x}^{\text{pos}}$ and $(\boldsymbol{x}_k^{\text{neg}})_{k \in [\![1,K]\!]}$ for an epoch over the set $(\boldsymbol{y}_i)_{i \in [\![1,N]\!]}$.

```
for i ∈ ⟦1, N⟧ with s_i = size(y_i) do
    pick s^pos = size(x^pos) in ⟦1, s_i⟧ and s^ref = size(x^ref) in ⟦s^pos, s_i⟧ uniformly at random;
    pick x^ref uniformly at random among subseries of y_i of length s^ref;
    pick x^pos uniformly at random among subseries of x^ref of length s^pos;
    pick uniformly at random i_k ∈ ⟦1, N⟧, then s_k^neg = size(x_k^neg) in ⟦1, size(y_k)⟧ and finally
      x_k^neg among subseries of y_k of length s_k^neg, for k ∈ ⟦1, K⟧.
```

3. Triplet Loss with MTS

Triplet Loss Function

$$-\log\left(\sigma\left(\boldsymbol{f}\left(\boldsymbol{x}^{\mathrm{ref}}, \boldsymbol{\theta}\right)^{\top} \boldsymbol{f}\left(\boldsymbol{x}^{\mathrm{pos}}, \boldsymbol{\theta}\right)\right)\right) - \sum_{k=1}^{K} \log\left(\sigma\left(-\boldsymbol{f}\left(\boldsymbol{x}^{\mathrm{ref}}, \boldsymbol{\theta}\right)^{\top} \boldsymbol{f}\left(\boldsymbol{x}_k^{\mathrm{neg}}, \boldsymbol{\theta}\right)\right)\right)$$

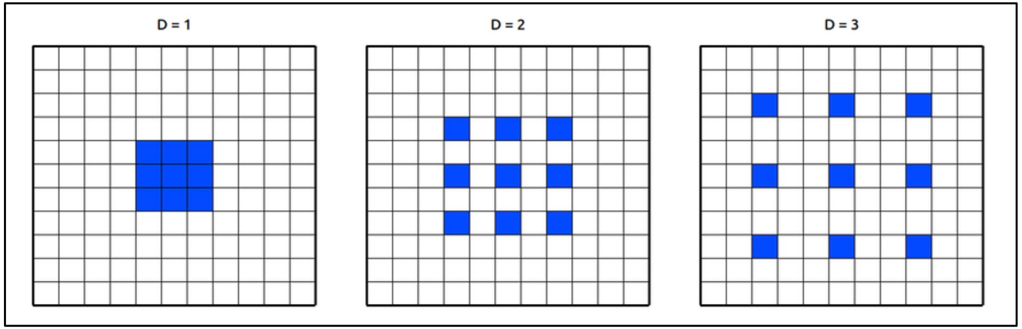
1 Positive sample

K Negative sample

4. Encoder Architecture

Dilated Causal Convolution

- Dilated Convolution
  - **make receptive field larger ! (with less computation)**
  - ex) filter size = (3,3) & dilation factor = 1 / 2 / 3



4. Encoder Architecture

Dilated **Causal** Convolution

- **Causal** Convolution
  - make convolution filter consider the **“time order”**

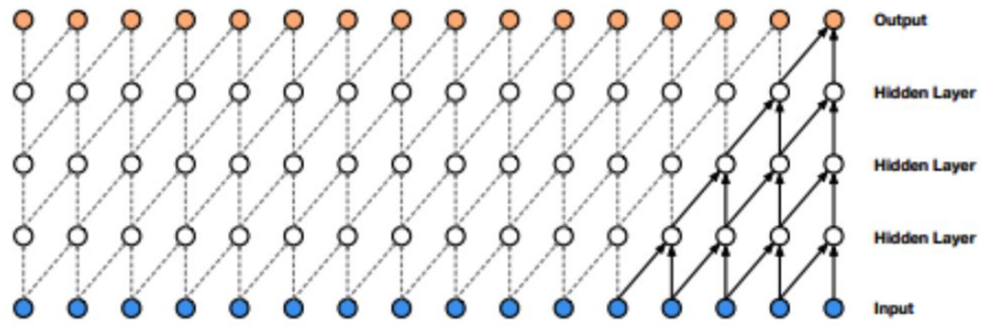


Figure 2: Visualization of a stack of causal convolutional layers.

4. Encoder Architecture

Dilated Causal Convolution

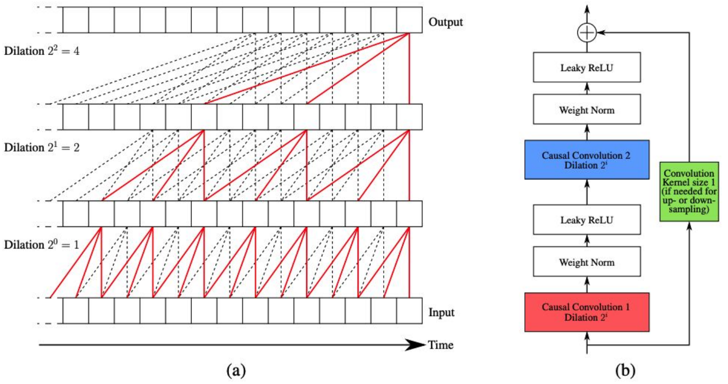


Figure 2: (a) Illustration of three stacked dilated causal convolutions. Lines between each sequence represent their computational graph. Red solid lines highlight the dependency graph for the computation of the last value of the output sequence, showing that no future value of the input time series is used to compute it. (b) Composition of the i-th layer of the chosen architecture.

Dilated = allow LONG sequence input

Causal = consider TIME ORDER (causality)

Global Max Pooling (GMP) = allow VARIABLE-LENGTH input

- output of Dilated Causal Convolution : given to a GMP
  → squeeze the temporal dimension &
  aggregate all temporal information in a **fixed-size vector**

4. Encoder Architecture

Dilated Causal Convolution

```
B = 64
C_in = 8
L = 100

input = torch.randn((B, C_in, L))
output = cau_cnn(input)
print(input.shape)
print(output.shape)
```

```
torch.Size([64, 8, 100])
torch.Size([64, 20])
```

```
class CausalCNNEncoder(torch.nn.Module):
    """
    Encoder of a TS using a causal CNN
    - (1) causal_cnn
    ----- (B, C_in, L) -> (B,C_out,L)
    - (2) adaptive max pooling ( makes TS to fixed size )
    ----- (B, C_out, L) -> (B,C_out, 1)
    - (3) squeeze
    ----- (B,C_out, 1) -> (B,C_out)
    """
    def __init__(self, in_channels, mid_channels, depth, reduced_size,
                 out_channels, kernel_size):
        super(CausalCNNEncoder, self).__init__()

        causal_cnn = CausalCNN(in_channels, mid_channels, depth, reduced_size, kernel_size)
        reduce_size = torch.nn.AdaptiveMaxPool1d(1)
        squeeze = SqueezeChannels(squeeze_dim = 2) # Time dimension
        linear = torch.nn.Linear(reduced_size, out_channels)

        self.network = torch.nn.Sequential(causal_cnn, reduce_size, squeeze, linear)

    def forward(self, x):
        return self.network(x)
```

regardless of Input Length!

5. Experiment

Investigate the relevance of the “learned representations”

- Experiment 1) Time Series Classification
- Experiment 2) Evaluation on Long Time Series

5. Experiment

Experiment 1) **Time Series Classification**

- test the quality of learned representations on supervised tasks
- **K (# of negative samples)** : significant impact on the performance

  → present a **combined version** of our method
    - representations trained with **different values of K are concatenated**
    - enables the representations with different parameters to complement each other & remove some noise in the classification scores

5. Experiment

S2.1 Influence of K

As mentioned in Section 5, K can have a significant impact on the performance of the encoder. We notably observed that $K = 1$ leads to statistically significantly lower scores compared to scores obtained when trained with $K > 1$ on the UCR datasets, justifying the use of several negative examples during training. We did not observe any clear statistical difference between other values of K on the whole archive; however, we noticed important differences between different values of K when studying individual datasets. Therefore, we chose to combine several encoders trained with different values of K in order to avoid selecting it as a fixed hyperparameter.

Experiment 1) **Time Series Classification**

- test the quality of learned representations on supervised tasks
- **K (# of negative samples)** : significant impact on the performance

→ present a **combined version** of our method

  - representations trained with **different values of K are concatenated**
  - enables the representations with different parameters to complement each other & remove some noise in the classification scores

5. Experiment

Experiment 1) **Time Series Classification**

1-1) Univariate TS : accuracy for all 128 datasets of UCR archive

Table 1: Accuracy scores of variants of our method compared with other supervised and unsupervised methods, on some UCR datasets. Results for the whole archive are available in the supplementary material, Section S3, Tables S1, S2 and S4. Bold and underlined scores respectively indicate the best and second-best (when there is no tie for first place) performing methods.

Dataset	Unsupervised					Supervised			
	Ours				DTW	ST	BOSS	Ensemble	
	$K = 5$	$K = 10$	Combined	FordA				HIVE-COTE	EE
DiatomSizeReduction	**0.993**	0.984	**0.993**	0.974	0.967	0.925	0.931	0.941	0.944
ECGFiveDays	**1**	**1**	**1**	**1**	**1**	0.984	**1**	**1**	0.82
FordB	0.781	0.793	<u>0.81</u>	0.798	0.62	0.807	0.711	**0.823**	0.662
Ham	0.657	**0.724**	<u>0.695</u>	0.533	0.467	0.686	0.667	0.667	0.571
Phoneme	0.249	0.276	0.289	0.196	0.228	<u>0.321</u>	0.265	**0.382**	0.305
SwedishLeaf	0.925	0.914	<u>0.931</u>	0.925	0.792	0.928	0.922	**0.954**	0.915

trained with another dataset (= FordA), with K=5

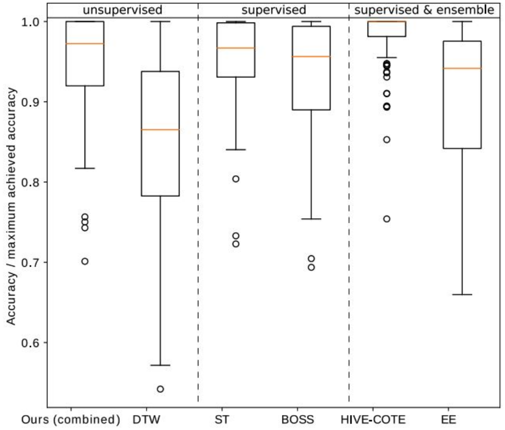


Figure 3: Boxplot of the ratio of the accuracy versus maximum achieved accuracy (higher is better) for compared methods on the first 85 UCR datasets.

5. Experiment

Experiment 1) **Time Series Classification**

1-1) Univariate TS : accuracy for all 128 datasets of UCR archive

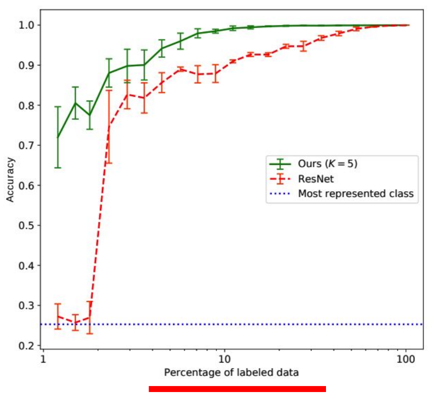


Figure 4: Accuracy of ResNet and our method with respect to the ratio of labeled data on TwoPatterns. Error bars correspond to the standard deviation over five runs per point for each method.

[Sparsely Labeled]

Green : SVM, trained on our representations of a randomly chosen labeled set

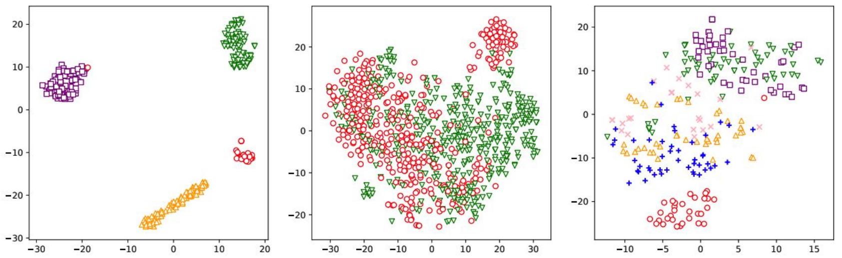
Red : ResNet, trained on a labeled set of the same size

5. Experiment

Experiment 1) **Time Series Classification**

1-1) Univariate TS : accuracy for all 128 datasets of UCR archive

[Representations metric space]



(a) DiatomSizeReduction. (b) FordB. (c) OSULeaf.

Figure 5: Two-dimensional t-SNE (Maaten & Hinton, 2008) with perplexity 30 of the learned representations of three UCR test sets. Elements classes are distinguishable using their respective marker shapes and colors.

5. Experiment

Experiment 1) **Time Series Classification**

1-2) Multivariate TS (classification task)

: accuracy for 30 datasets of UEA archive

$\mathrm{DTW_D}$

- dimension-Dependent DTW
- extension of DTW in the MTS setting
- best baseline studied by Bagnall et al. (2018).

Dataset	Unsupervised				
	Ours				$\mathrm{DTW_D}$
	$K = 5$	$K = 10$	$K = 20$	Combined	
ArticularyWordRecognition	0.967	0.973	0.943	**0.987**	**0.987**
AtrialFibrillation	**0.2**	0.067	0.133	0.133	**0.2**
BasicMotions	**1**	**1**	**1**	**1**	0.975
CharacterTrajectories	0.986	0.99	0.993	**0.994**	0.989
Cricket	0.958	0.972	0.972	0.986	**1**
DuckDuckGeese	0.6	**0.675**	0.65	**0.675**	0.6
EigenWorms	0.87	0.802	0.84	**0.878**	0.618
Epilepsy	**0.971**	**0.971**	**0.971**	0.957	0.964
Ering	**0.133**	**0.133**	**0.133**	**0.133**	**0.133**
EthanolConcentration	0.289	0.251	0.205	0.236	**0.323**
FaceDetection	0.522	0.525	0.513	0.528	**0.529**
FingerMovements	0.55	0.49	**0.58**	0.54	0.53
HandMovementDirection	0.311	0.297	**0.351**	0.27	0.231
Handwriting	0.447	0.464	0.451	**0.533**	0.286
Heartbeat	**0.756**	0.732	0.741	0.737	0.717
InsectWingbeat	0.159	0.158	0.156	**0.16**	-
JapaneseVowels	0.984	0.986	**0.989**	**0.989**	0.949
Libras	0.878	**0.883**	**0.883**	0.867	0.87
LSST	0.535	0.552	0.509	**0.558**	0.551
MotorImagery	0.53	0.54	**0.58**	0.54	0.5
NATOPS	0.933	0.917	0.917	**0.944**	0.883
PEMS-SF	0.636	0.671	0.676	**0.688**	0.711
PenDigits	**0.985**	0.979	0.981	0.983	0.977
Phoneme	0.216	0.214	0.222	**0.246**	0.151
RacketSports	0.776	0.836	0.855	**0.862**	0.803
SelfRegulationSCP1	0.795	0.826	0.843	**0.846**	0.775
SelfRegulationSCP2	0.55	0.539	0.539	**0.556**	0.539
SpokenArabicDigits	0.908	0.894	0.905	0.956	**0.963**
StandWalkJump	0.333	**0.4**	0.333	**0.4**	0.2
UWaveGestureLibrary	0.884	0.869	0.875	0.884	**0.903**

5. Experiment

Experiment 2) **Evaluation on Long Time Series**

- UCR, UEA : mostly SHORT TS
- IHEPC dataset (from UCI) : **LONG single TS** (length = 2,075,259)

→ train / test = 5 x 10^5 / remaining

→ single Nvidia Tesla P100 GPU in **no more than a few hours**

5. Experiment

Experiment 2) **Evaluation on Long Time Series**

use learned encoder on **2 regression tasks** (with 2 different input scales)

Task : for each time step, predict the discrepancy between mean value of the series …..

- (1) for the **next period** (either a day or quarter)
- (2) for the **previous period**

induce only a **"slightly degraded performance"**

but provide a **"large efficiency improvement"**

(due to their small size compared to the raw TS)

Table 2: Results obtained on the IHEPC dataset.

Task	Metric	Representations	Raw values
Day	Test MSE	$\mathbf{8.92 \times 10^{-2}}$	$\mathbf{8.92 \times 10^{-2}}$
	Wall time	**12s**	3min 1s
Quarter	Test MSE	7.26×10^{-2}	$\mathbf{6.26 \times 10^{-2}}$
	Wall time	**9s**	1h 40min 15s

Papers

Unsupervised Scalable Representation Learning for Multivariate Time Series

Jean-Yves Franceschi*
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
jean-yves.franceschi@lip6.fr

Aymeric Dieuleveut
MLO, EPFL, Lausanne CH-1015, Switzerland
CMAP, Ecole Polytechnique, Palaiseau, France
aymeric.dieuleveut@polytechnique.edu

Martin Jaggi
MLO, EPFL, Lausanne CH-1015, Switzerland
martin.jaggi@epfl.ch

Abstract

Time series constitute a challenging data type for machine learning algorithms, due to their highly variable lengths and sparse labeling in practice. In this paper, we tackle this challenge by proposing an unsupervised method to learn universal embeddings of time series. Unlike previous works, it is scalable with respect to their length and we demonstrate the quality, transferability and practicability of the learned representations with thorough experiments and comparisons. To this end, we combine an encoder based on causal dilated convolutions with a novel triplet loss employing time-based negative sampling, obtaining general-purpose representations for variable length and multivariate time series.

https://arxiv.org/pdf/1901.10738.pdf

UNSUPERVISED REPRESENTATION LEARNING FOR TIME SERIES WITH TEMPORAL NEIGHBORHOOD CODING

Sana Tonekaboni*
University of Toronto & Vector Institute
The Hospital for Sick Children
stonekaboni@cs.toronto.edu

Danny Eytan
The Hospital for Sick Children
biliary.colic@gmail.com

Anna Goldengerg
University of Toronto & Vector Institute
The Hospital for Sick Children
anna.goldenberg@utoronto.ca

ABSTRACT

Time series are often complex and rich in information but sparsely labeled and therefore challenging to model. In this paper, we propose a self-supervised framework for learning generalizable representations for non-stationary time series. Our approach, called Temporal Neighborhood Coding (TNC), takes advantage of the local smoothness of a signal's generative process to define neighborhoods in time with stationary properties. Using a debiased contrastive objective, our framework learns time series representations by ensuring that in the encoding space, the distribution of signals from within a neighborhood is distinguishable from the distribution of non-neighboring signals. Our motivation stems from the medical field, where the ability to model the dynamic nature of time series data is especially valuable for identifying, tracking, and predicting the underlying patients' latent states in settings where labeling data is practically impossible. We compare our method to recently developed unsupervised representation learning approaches and demonstrate superior performance on clustering and classification tasks for multiple datasets.

https://arxiv.org/pdf/2106.00750.pdf

Unsupervised Representation Learning for TS with Temporal Neighborhood Coding (2021)

1. Introduction

Challenges in Time Series Data : **sparse labeling** → need for UN/SELF-SUPERVISED learning

This paper proposes …..

"Self-supervised method to learn generalizable representations for non-stationary TS"

proposes **Temporal Neighborhood Coding (TNC)**

- takes advantages of **"local smoothness" of signal's generative process** to define neighborhood
- distinguish (1) & (2)
    - (1) distn of signals from **NEIGHBORHOOD**
    - (2) distn of signals from **NON-NEIGHBORHOOD**

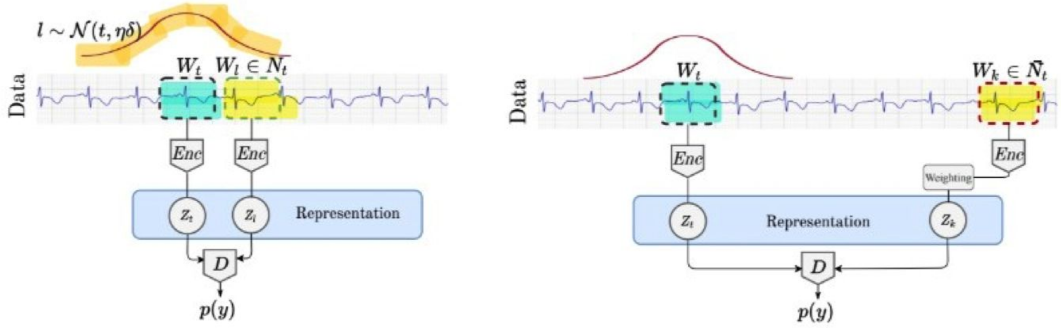
2. Temporal Neighborhood Coding (TNC)

(a) Overall Architecture

- **Self-supervised Framework** for learning representations for complex **Non-stationary MTS**
- Temporal Settings : **latent distribution** of the signals **changes over time**
- Goal : capture the **progression of the underlying temporal dynamics**
- Characteristics :
  - **(1) efficient**
  - **(2) scalable to high dimensions**
  - **(3) can be used in different TS settings**

2. Temporal Neighborhood Coding (TNC)

(a) Overall Architecture



(a) Neighborhood samples (b) Non-neighboring samples

Figure 1: Overview of the TNC framework components. For each sample window W_t (indicated with the dashed black box), we first define the neighborhood distribution. The encoder learns the distribution of windows sampled from N_t and $\bar{N}_t$, in the representation space. Then samples from these distributions are fed into the discriminator alongside Z_t, to predict the probability of the windows being in the same neighborhood.

2. Temporal Neighborhood Coding (TNC)

(a) Overall Architecture

Notation

- $X \in R^{D \times T}$: MTS
- $X_{[t-\frac{\delta}{2}, t+\frac{\delta}{2}]}$: window .... refer as W_t
- N_t : temporal neighborhood of window W_t
  - set of all windows, with centroids t^*, where $t^* \sim N(t, \eta \cdot \delta)$
    - η : range of neighborhood
  - how to set η ?
    - (1) domain experts
    - (2) determined by analyzing the stationarity properties of the signal for every W_t
- $\bar{N}_t$: non-neighborhood of window W_t

(considered as negative samples)

Value of η

- too SMALL : many samples within neighborhood will **OVERLAP**
- too BIG : the neighborhood would span over multiple ounderlying states

(fail to distinguish among these states)

2. Temporal Neighborhood Coding (TNC)

(a) Overall Architecture

Notation

- $X \in R^{D \times T}$: MTS

Value of η

- too SMALL : many samples within neighborhood will **OVERLAP**

Non-neighborhood : far from window (anchor / reference)

→ *is it always NEGATIVE samples?*

- N_t : non-neighborhood of window W_t

(considered as negative samples)

2. Temporal Neighborhood Coding (TNC)

(b) Sampling Bias & PU-Learning

Sampling Bias

- Why does it occur?

  → randomly drawing negative samples from data distn **MAY NOT result in negative samples !!**

  (may be actually SIMILAR to the reference)

- Solution

  → consider samples from NON-neighborhood as **"UN-labeled samples" (not NEGATIVE)**

  → **"PU Learning"**

2. Temporal Neighborhood Coding (TNC)

(b) Sampling Bias & PU-Learning

PU Learning (Positive-Unlabeled Learning)

- classifier is learned, using…
  - (1) **Positive** samples (**P**)
  - (2) **Unlabeled** samples (**U**)
    - mixture of **Positive (P)** & **Negative (N)** (with a positive class prior π)
- falls into 2 categories
  - (1) **identify negative samples** from the unlabeled cohort
  - (2) treat the unlabeled data as negative samples, with **"smaller weights"**
    - unlabeled samples should be **properly weighted** to make an **unbiased classifier**

2. Temporal Neighborhood Coding (TNC)

(b) Sampling Bias & PU-Learning

PU Learning (Positive-Unlabeled Learning)

Samples from...

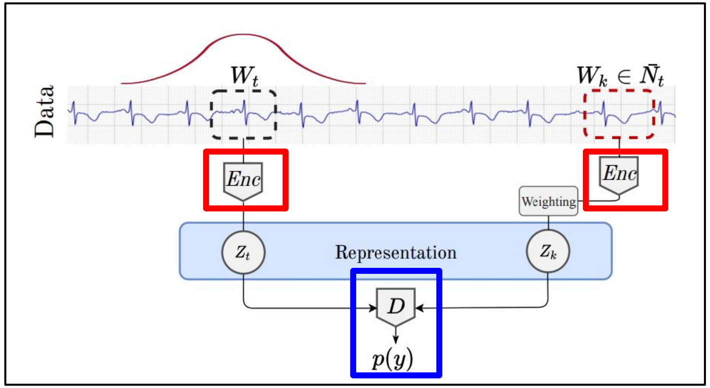
- (1) neighborhood (N_t) : positive
- (2) non-neighborhood ($\bar{N}_t$) : combination of positive (weight : w) & negative (weight : $1-w$)
  - weight (w) : probability of having samples similar to W_t in $\bar{N}$
    - (1) can be approximated using the prior knowledge
    - (2) or tuned as hyperparameter

2. Temporal Neighborhood Coding (TNC)

(c) 2 main components of TNC

(1) Encoder : $Z_t = Enc(W_t)$

- maps $W_t \in R^{D \times \delta}$ to $Z_t \in R^M$



(2) Discriminator : $D(Z_t, Z)$

- approximates the probability of Z being the representation of a window in N_t
- predicts the probability of samples belonging to the **same temporal neighborhood**

2. Temporal Neighborhood Coding (TNC)

(d) Objective Function

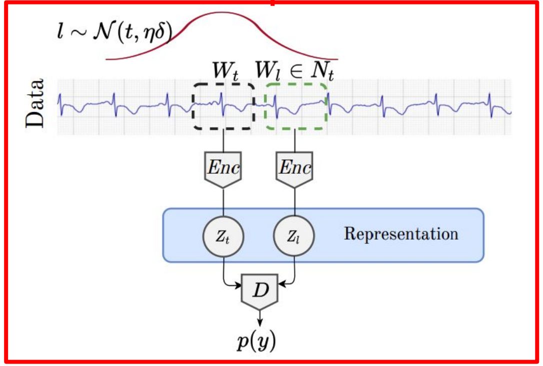
$$\mathcal{L} = -\mathbb{E}_{W_t \sim X}\left[\mathbb{E}_{W_l \sim N_t}\left[\log \underbrace{\mathcal{D}\left(\mathrm{Enc}(W_t), \mathrm{Enc}(W_l)\right)}_{\mathcal{D}(Z_t, Z_l)}\right] + \mathbb{E}_{W_k \sim \bar{N}_t}\left[(1 - w_t) \times \log \underbrace{\left(1 - \mathcal{D}\left(Z_t, Z_k\right)\right)}_{\mathcal{D}(\mathrm{Enc}(W_t), \mathrm{Enc}(W_k))} + w_t \times \log \mathcal{D}\left(Z_t, Z_k\right)\right]\right]$$

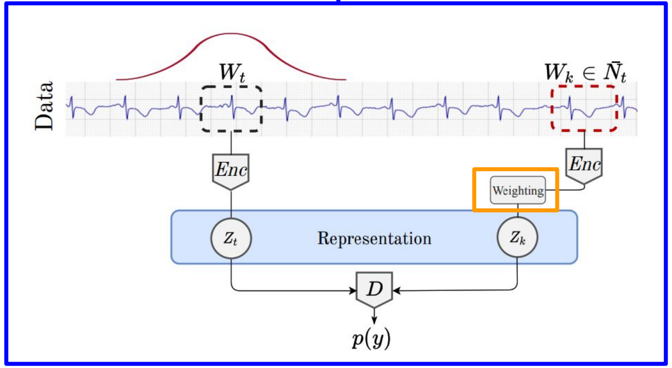
2. Temporal Neighborhood Coding (TNC)

(d) Objective Function

Neighborhood | **Non-Neighborhood**

$$\mathcal{L} = -\mathbb{E}_{W_t \sim X}\left[\mathbb{E}_{W_l \sim N_t}\left[\log \underbrace{\mathcal{D}\left(\mathrm{Enc}(W_t), \mathrm{Enc}(W_l)\right)}_{\mathcal{D}(Z_t, Z_l)}\right] + \mathbb{E}_{W_k \sim \bar{N}_t}\left[(1 - w_t) \times \log \underbrace{\left(1 - \mathcal{D}\left(Z_t, Z_k\right)\right)}_{\mathcal{D}(\mathrm{Enc}(W_t), \mathrm{Enc}(W_k))} + w_t \times \log \mathcal{D}\left(Z_t, Z_k\right)\right]\right]$$





3. Experiments

- assess the **quality of the learned representations** on multiple datasets
- show that the representations are **general and transferable to many downstream tasks** (such as classification and clustering)
- outperforms existing approaches for **unsupervised representation learning**
- performs closely to **supervised techniques** in classification tasks

3. Experiments

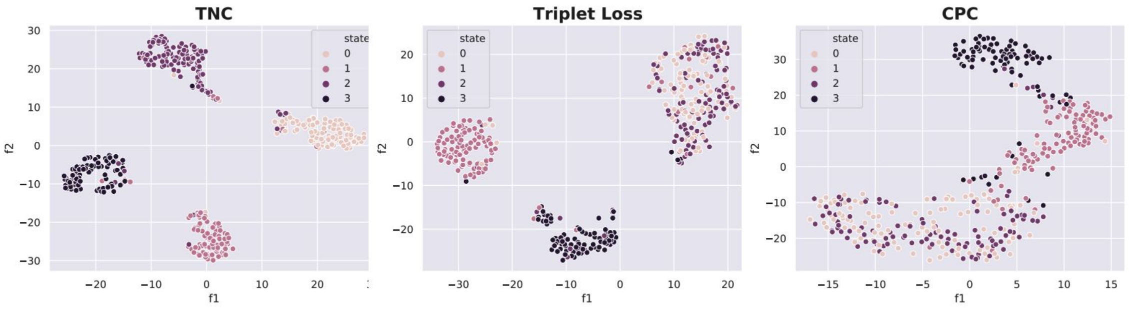
- test the "generalizability" of the representations, by…
  - comparing **(1) classification performance** & **(2) clusterability**
- with 2 SOTA for unsupervised representation learning for TS
  - a) **Contrastive Predictive Coding (CPC)**
  - b) **Triplet-loss (T-Loss)**
  - etc) **K-means** (for "clustering") & **KNN with DTW** (for "classification")

  (for fair comparison : use same encoder for all cases)
- Dataset)
  - (1) Simulated Data / (2) Clinical Waveform Data / (3) Human Activity Recognition (HAR) Data

3. Experiments

(1) Clusterability

- assess the distn of the representations in the encoding space
- ex) Simulated Data



(a) TNC representations (b) T-loss representations (c) CPC representations

Figure 2: T-SNE visualization of signal representations for the simulated dataset across all baselines. Each data point in the plot presents a 10-dimensional representation of a window of time series of size $\delta = 50$, and the color indicates the latent state of the signal window. See Appendix A.7 for similar plots from different datasets.

3. Experiments

(1) Clusterability

- 2 cluster validity indices :
  - **(1) Silhouette score**
    - measures the similarity of each sample to its own cluster, compared to other clusters
    - (range) -1 ~ 1 : greater score, better cohesion
  - **(2) Davies-Bouldin index**
    - measures intra-cluster similarity & inter-cluster differences
    - smaller values indicate “low within-cluster scatter” & “large separation btw clusters
- use K-means in the representation space to measure these scores

3. Experiments

CPC = Triplet Loss (on ECG Waveform)

CPC < Triplet Loss (on Simulation)

- signals are highly "non-stationary" & transitions are "less predictable"

(1) Clusterability

	Simulation		ECG Waveform		HAR	
Method	Silhouette ↑	DBI ↓	Silhouette ↑	DBI ↓	Silhouette ↑	DBI ↓
TNC	**0.71±0.01**	**0.36±0.01**	**0.44±0.02**	**0.74±0.04**	**0.61±0.02**	**0.52±0.04**
CPC	0.51±0.03	0.84±0.06	0.26±0.02	1.44±0.04	0.58±0.02	0.57±0.05
T-Loss	0.61±0.08	0.64±0.12	0.25±0.01	1.30±0.03	0.17±0.01	1.76±0.20
K-means	0.01±0.019	7.23±0.14	0.19±0.11	3.65±0.48	0.12±0.40	2.66±0.05

Table 1: Clustering quality of representations in the encoding space for multiple datasets.

3. Experiments

(1) Clusterability

CPC : perform well on HAR

- most activities are recorded in a specific order, empowering predictive coding.

	Simulation		ECG Waveform		HAR	
Method	Silhouette ↑	DBI ↓	Silhouette ↑	DBI ↓	Silhouette ↑	DBI ↓
TNC	**0.71±0.01**	**0.36±0.01**	**0.44±0.02**	**0.74±0.04**	**0.61±0.02**	**0.52±0.04**
CPC	0.51±0.03	0.84±0.06	0.26±0.02	1.44±0.04	0.58±0.02	0.57±0.05
T-Loss	0.61±0.08	0.64±0.12	0.25±0.01	1.30±0.03	0.17±0.01	1.76±0.20
K-means	0.01±0.019	7.23±0.14	0.19±0.11	3.65±0.48	0.12±0.40	2.66±0.05

Table 1: Clustering quality of representations in the encoding space for multiple datasets.

3. Experiments

(2) Classification

- compared with (1) supervised classifier & (2) KNN with DTW metric
    - (1) supervised classifier : composed of an encoder & classifier
      (identical architectures with unsupervised model)
- metric : AUPRC (Area Under the Precision-Recall Curve)
    - better metric for “imbalanced classification settings” (ex. Waveform)

3. Experiments

(2) Classification

	Simulation		ECG Waveform		HAR	
Method	AUPRC	Accuracy	AUPRC	Accuracy	AUPRC	Accuracy
TNC	**0.99±0.00**	**97.52±0.13**	**0.55±0.01**	**77.79±0.84**	**0.94±0.007**	**88.32±0.12**
CPC	0.69±0.06	70.26±6.48	0.42±0.01	68.64±0.49	0.93±0.006	86.43±1.41
T-Loss	0.78±0.01	76.66±1.40	0.47±0.00	75.51±1.26	0.71±0.007	63.60±3.37
KNN	0.42±0.00	55.53±0.65	0.38±0.06	54.76±5.46	0.75±0.01	84.85±0.84
Supervised	**0.99±0.00**	**98.56±0.13**	**0.67±0.01**	**94.81±0.28**	**0.98±0.00**	**92.03±2.48**

Table 2: Performance of all baselines in classifying the underlying hidden states of the time series, measured as the accuracy and AUPRC score.

3. Experiments

(2) Classification

CPC (in HAR) : performs well

- inherent ordering usually exists in HAR

CPC (with increased non-stationarity) : performance drops

	Simulation		ECG Waveform		HAR	
Method	AUPRC	Accuracy	AUPRC	Accuracy	AUPRC	Accuracy
TNC	**0.99±0.00**	**97.52±0.13**	**0.55±0.01**	**77.79±0.84**	**0.94±0.007**	**88.32±0.12**
CPC	0.69±0.06	70.26±6.48	0.42±0.01	68.64±0.49	0.93±0.006	86.43±1.41
T-Loss	0.78±0.01	76.66±1.40	0.47±0.00	75.51±1.26	0.71±0.007	63.60±3.37
KNN	0.42±0.00	55.53±0.65	0.38±0.06	54.76±5.46	0.75±0.01	84.85±0.84
Supervised	**0.99±0.00**	**98.56±0.13**	**0.67±0.01**	**94.81±0.28**	**0.98±0.00**	**92.03±2.48**

Table 2: Performance of all baselines in classifying the underlying hidden states of the time series, measured as the accuracy and AUPRC score.

3. Experiments

(2) Classification

Triplet Loss

- samples positive examples from **overlapping windows of TS**
- vulnerable to map the overlaps into the encoding

  → fail to learn more general representations.

	Simulation		ECG Waveform		HAR	
Method	AUPRC	Accuracy	AUPRC	Accuracy	AUPRC	Accuracy
TNC	**0.99±0.00**	**97.52±0.13**	**0.55±0.01**	**77.79±0.84**	**0.94±0.007**	**88.32±0.12**
CPC	0.69±0.06	70.26±6.48	0.42±0.01	68.64±0.49	0.93±0.006	86.43±1.41
T-Loss	0.78±0.01	76.66±1.40	0.47±0.00	75.51±1.26	0.71±0.007	63.60±3.37
KNN	0.42±0.00	55.53±0.65	0.38±0.06	54.76±5.46	0.75±0.01	84.85±0.84
Supervised	**0.99±0.00**	**98.56±0.13**	**0.67±0.01**	**94.81±0.28**	**0.98±0.00**	**92.03±2.48**

Table 2: Performance of all baselines in classifying the underlying hidden states of the time series, measured as the accuracy and AUPRC score.

3. Experiments

TNC

- samples from a **wider distn (= temporal neighborhood)**
- thus, many of the neighboring signals do not necessarily overlap

(2) Classification

	Simulation		ECG Waveform		HAR	
Method	AUPRC	Accuracy	AUPRC	Accuracy	AUPRC	Accuracy
TNC	**0.99±0.00**	**97.52±0.13**	**0.55±0.01**	**77.79±0.84**	**0.94±0.007**	**88.32±0.12**
CPC	0.69±0.06	70.26±6.48	0.42±0.01	68.64±0.49	0.93±0.006	86.43±1.41
T-Loss	0.78±0.01	76.66±1.40	0.47±0.00	75.51±1.26	0.71±0.007	63.60±3.37
KNN	0.42±0.00	55.53±0.65	0.38±0.06	54.76±5.46	0.75±0.01	84.85±0.84
Supervised	**0.99±0.00**	**98.56±0.13**	**0.67±0.01**	**94.81±0.28**	**0.98±0.00**	**92.03±2.48**

Table 2: Performance of all baselines in classifying the underlying hidden states of the time series, measured as the accuracy and AUPRC score.

3. Experiments

TNC > **CPC & Triplet-Loss** ….reason ?

whether they consider **"sampling bias"** !

(2) Classification

(happens when randomly selected NEG samples are similar to the reference)

	Simulation		ECG Waveform		HAR	
Method	AUPRC	Accuracy	AUPRC	Accuracy	AUPRC	Accuracy
TNC	**0.99±0.00**	**97.52±0.13**	**0.55±0.01**	**77.79±0.84**	**0.94±0.007**	**88.32±0.12**
CPC	0.69±0.06	70.26±6.48	0.42±0.01	68.64±0.49	0.93±0.006	86.43±1.41
T-Loss	0.78±0.01	76.66±1.40	0.47±0.00	75.51±1.26	0.71±0.007	63.60±3.37
KNN	0.42±0.00	55.53±0.65	0.38±0.06	54.76±5.46	0.75±0.01	84.85±0.84
Supervised	**0.99±0.00**	**98.56±0.13**	**0.67±0.01**	**94.81±0.28**	**0.98±0.00**	**92.03±2.48**

Table 2: Performance of all baselines in classifying the underlying hidden states of the time series, measured as the accuracy and AUPRC score.

Thank You !