EIVIA 2025: Deep Learning for Time Series and Applications to Healthcare

Gonzalo Uribarri
KTH Royal Institute of Technology & SciLifeLab

@gonzauri

Course Plan

1/ INTRODUCTION

- Challenges of ML for healthcare
- Introduction to ML for time series

2/ TIME SERIES: Standard Algorithms

- Basic ML for TS Classification
- Deep Learning for TS Classification

3/ TIMES SERIES: State-of -the-art

- SOTA TS Classification models
- ROCKET and InceptionTime

4/ APPLICATIONS

- Case studies in healthcare: Eye-tracking and EEG data
- Proper Evaluation

5/ LARGE MODELS

- TS models for large datasets
- Transfer learning: Foundational models for TS?

Machine Learning for Time Series

Sequential data

Sequential data is data arranged in sequences where order matters. Data points are conditioned on other data points in the sequence.

Sequential Data

Discrete Sequences	Time Series
Series of discrete tokens where order matters, but there is no explicit time variable	Quasi-continuous numerical values that evolve as a function of time.
E.g., Text, DNA, List of events	**E.g., ECG, Sound, Temperature**

Transformers

Linear RNNs: xLSTM, Mamba, RWKV

Focus

Tasks for Time Series Data

Machine Learning for Time Series

Supervised

Focus
- Classification
- Extrinsic Regression
- Forecasting

Unsupervised

- Clustering
- Compression
- Generative
- Outlier Detection

ML for Time Series Classification

There are two standard archives for benchmarking:

- UCR

  142 Univariate Time Series Datasets

- UEA

  30 Multivariate Time Series Datasets

Great reviews:

SPRINGER NATURE Link

Find a journal | Publish with us | Track your research | Search

Home > Data Mining and Knowledge Discovery > Article

Bake off redux: a review and experimental evaluation of recent time series classification algorithms

Open access | Published: 19 April 2024

Volume 38, pages 1958–2031, (2024) Cite this article

SPRINGER NATURE Link

Find a journal | Publish with us | Track your research | Search

Home > Data Mining and Knowledge Discovery > Article

The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances

Open access | Published: 18 December 2020

Volume 35, pages 401–449, (2021) Cite this article

Time Series Data

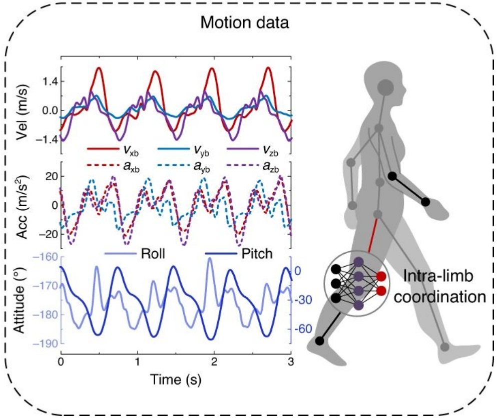
Depending on their nature (domain), time series can be **VERY** different.

Financial

Time Series Data

Depending on their nature (domain), time series can be **VERY** different.

Financial

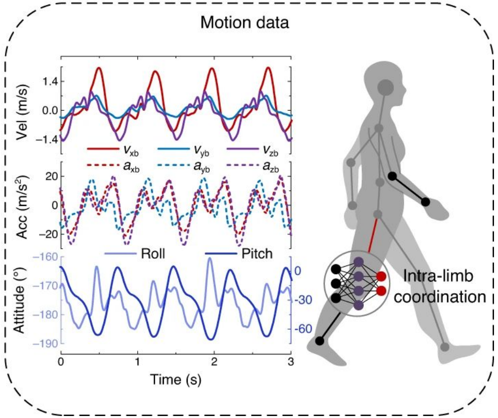


Accelerometer

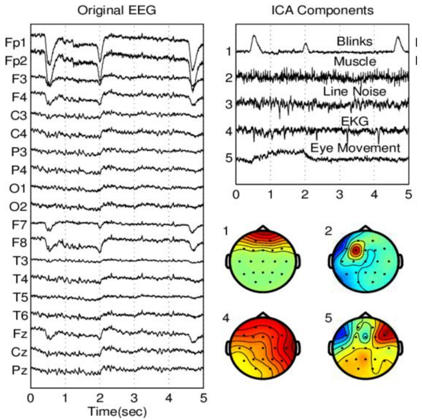
Time Series Data

Depending on their nature (domain), time series can be **VERY** different.

Financial



Accelerometer

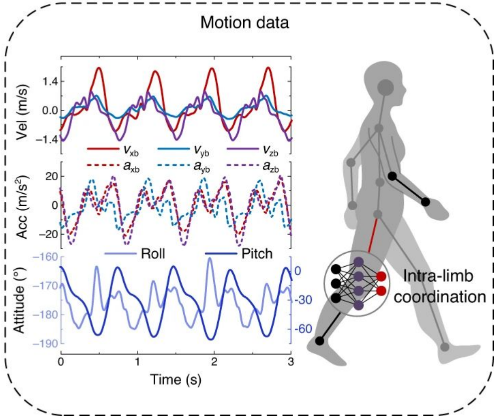


Brain Activity

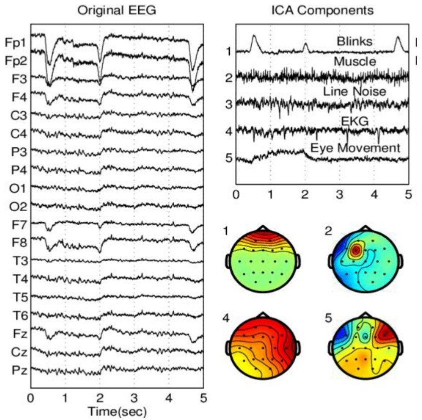
Time Series Data

Depending on their nature (domain), time series can be **VERY** different.

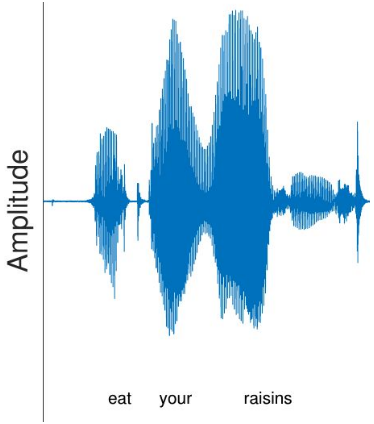
Financial



Accelerometer



Brain Activity



Sound

Time Series Classification Models

Time Series Data

Why is it hard to directly classify on the **raw** time series?

Time Series Data

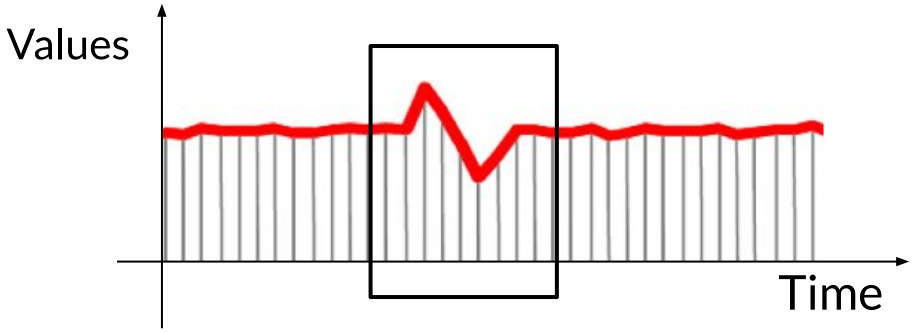
Why is it hard to directly classify on the **raw** time series?

If we use **each timestep as a feature**:

- Large number of features
- Don't exploit the regularity & invariance of the data

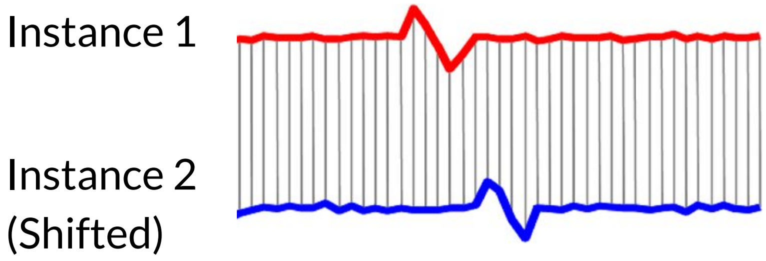
Time Series Data

Why is it hard to directly classify on the **raw** time series?



If we use **each timestep as a feature**:

- Large number of features
- Don't exploit the regularity & invariance of the data



Two “similar” time series present different features. Very hard to generalize.

Classification Strategies

Time Series Classification

Feature Based

- Predefined Features
- Spectral Features
- Intervals / Dictionary

Distance Based

- Elastic distance
- Representation Learning

End-to-end

- CNNs
- RNNs
- Transformers

Classification Strategies

Time Series Classification

Feature Based

- Predefined Features
- Spectral Features
- Intervals / Dictionary

Distance Based

- Elastic distance
- Representation Learning

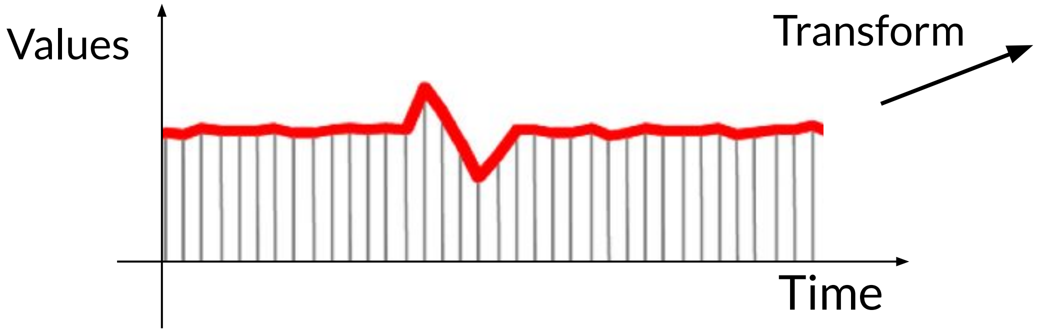
End-to-end

- CNNs
- RNNs
- Transformers

Feature Based: Predefined features

	f1	f2	f3	f4
x1	1.2	0.4	0.1	-1
x2	…	…	…	…
x3	…	…	…	…
x4	…	…	…	…

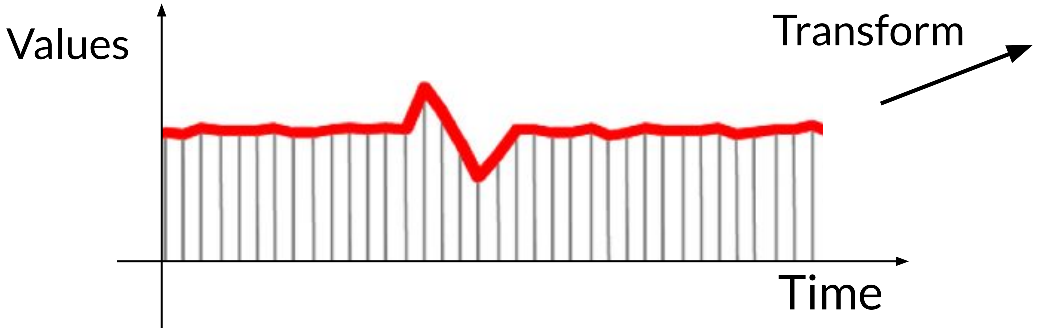
Feature Based: Predefined features



	f1	f2	f3	f4
x1	1.2	0.4	0.1	-1
x2	…	…	…	…
x3	…	…	…	…
x4	…	…	…	…

Is there a **UNIVERSAL** set of good features?

Feature Based: Predefined features



	f1	f2	f3	f4
x1	1.2	0.4	0.1	-1
x2	…	…	…	…
x3	…	…	…	…
x4	…	…	…	…

Is there a **UNIVERSAL** set of good features?

NO, depends on both the **task** and the **domain** of the data (TS can have a **VERY** different properties).

Feature Based: Predefined features

However, there are attempts to define a comprehensive set of features that can work reasonably well in many scenarios.

catch22

22 Features

catch22: CAnonical Time-series CHaracteristics

DOI 10.5281/zenodo.6673597 | License GPLv3 | Follow @compTimeSeries

catch22 is a collection of 22 time-series features coded in C that can be run from Python, R, Matlab, and Julia, licensed under the GNU GPL v3 license (or later). The *catch22* features are a high-performing subset of the over 7000 features in *hctsa*.

Feature Based: Predefined features

However, there are attempts to define a comprehensive set of features that can work reasonably well in many scenarios.

catch22

22 Features

catch22: CAnonical Time-series CHaracteristics

DOI 10.5281/zenodo.6673597 | License GPLv3 | Follow @compTimeSeries

catch22 is a collection of 22 time-series features coded in C that can be run from Python, R, Matlab, and Julia, licensed under the GNU GPL v3 license (or later). The *catch22* features are a high-performing subset of the over 7000 features in *hctsa*.

ts fresh

1200 Features

tsfresh

docs passing | Test Default Branch passing | codecov 94% | license MIT | launch binder | downloads 17M

This repository contains the *TSFRESH* python package. The abbreviation stands for

"Time Series Feature extraction based on scalable hypothesis tests".

Feature Based: Predefined features

However, there are attempts to define a comprehensive set of features that can work reasonably well in many scenarios.

- Fast & Easy Training (no HP)
- Interpretable
- + Scaling, + Deployment
- Good Benchmark
- Not SOTA Performance
- Not (Really) Multivariate

catch22

22 Features

catch22: CAnonical Time-series CHaracteristics

DOI 10.5281/zenodo.6673597 | License GPLv3 | Follow @compTimeSeries

catch22 is a collection of 22 time-series features coded in C that can be run from Python, R, Matlab, and Julia, licensed under the GNU GPL v3 license (or later). The *catch22* features are a high-performing subset of the over 7000 features in *hctsa*.

tsfresh

1200 Features

tsfresh

docs passing | Test Default Branch passing | codecov 94% | license MIT | launch binder | downloads 17M

This repository contains the *TSFRESH* python package. The abbreviation stands for

"Time Series Feature extraction based on scalable hypothesis tests".

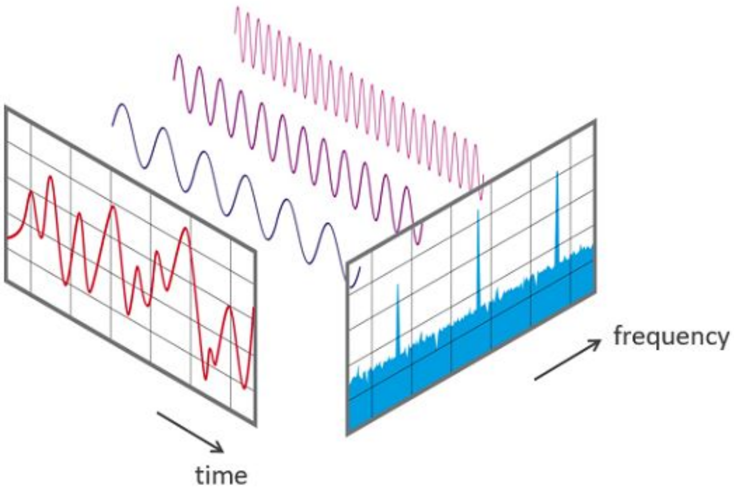
Feature Based: Spectral features

Another popular strategy is to characterize the time series by its spectral content.

Feature Based: Spectral features

Another popular strategy is to characterize the time series by its spectral content.

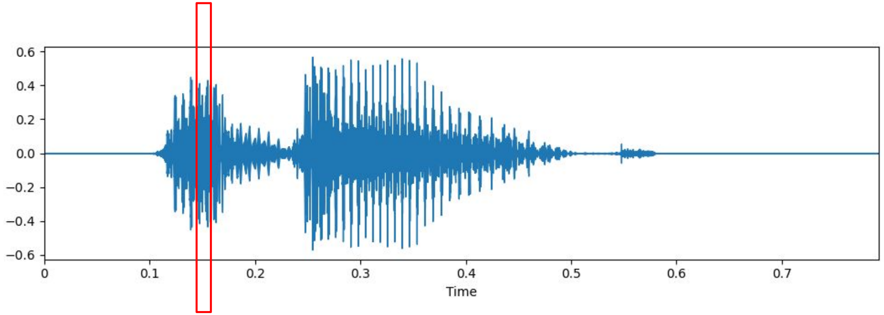
The simplest spectral representation is the **Fourier transform.**

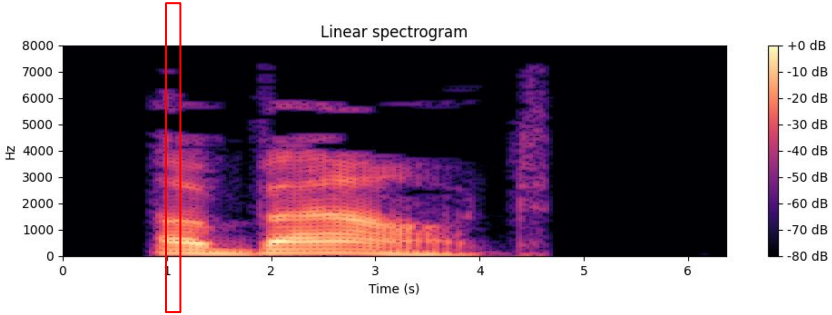


It is not ideal for non-stationary time series. We lose information about where in time things are happening.

Feature Based: Spectral features

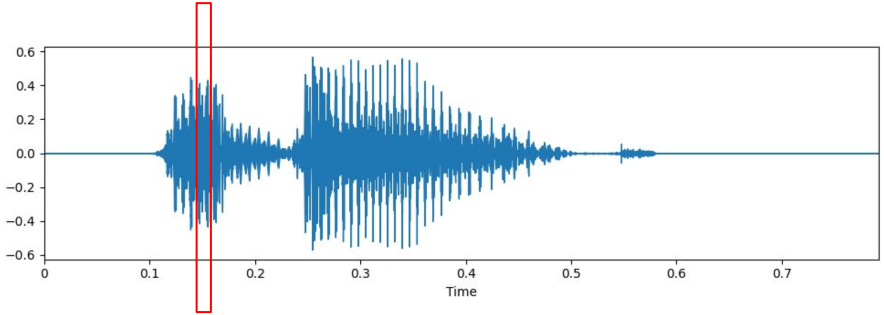
Instead, we can create a spectrogram using the Short-Time Fourier Transform (STFT) in windows of the time series.

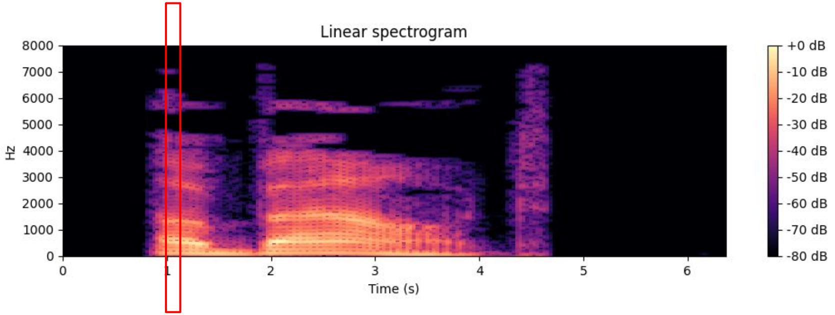




Feature Based: Spectral features

Instead, we can create a spectrogram using the Short-Time Fourier Transform (STFT) in windows of the time series.





We can then use the same strategies (architectures) we use for classifying **images**.

But this is not the best representation for a time series.

Classification Strategies

Time Series Classification

Feature Based

- Predefined Features
- Spectral Features
- Intervals / Dictionary

Distance Based

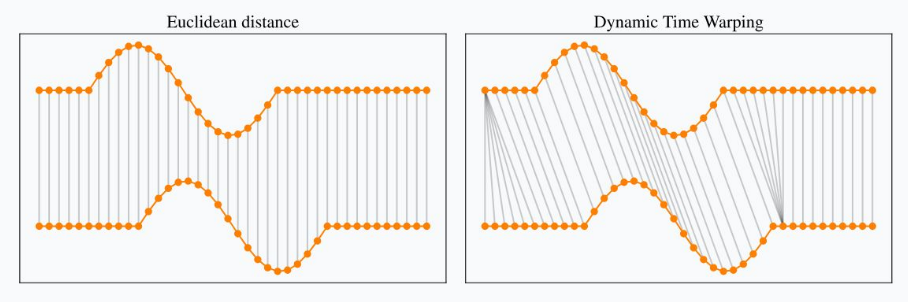
- Elastic distance
- Representation Learning

End-to-end

- CNNs
- RNNs
- Transformers

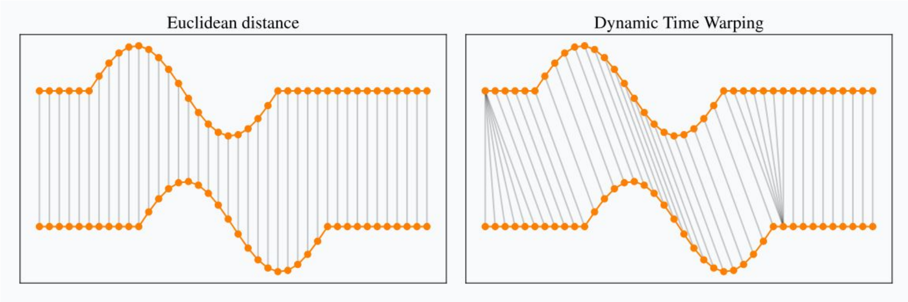
Distance Based: Elastic Distance

Compute a “proper distance” between the time series, and then classify using a distance based algorithm (K Nearest Neighbours).



Distance Based: Elastic Distance

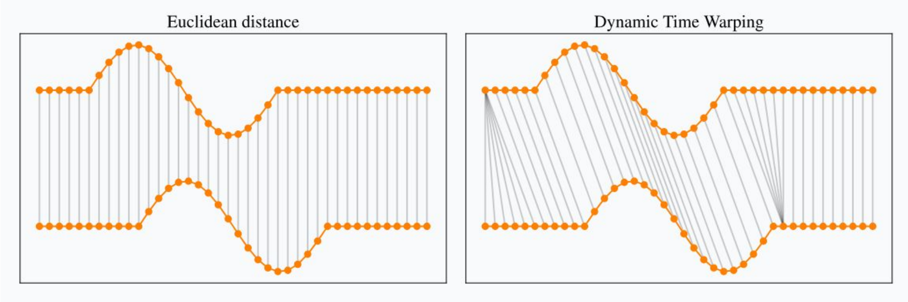
Compute a “proper distance” between the time series, and then classify using a distance based algorithm (K Nearest Neighbours).



Distances:
_ DTW,
_ Edit Distance
_ Move–split–merge
_ etc…

Distance Based: Elastic Distance

Compute a “proper distance” between the time series, and then classify using a distance based algorithm (K Nearest Neighbours).



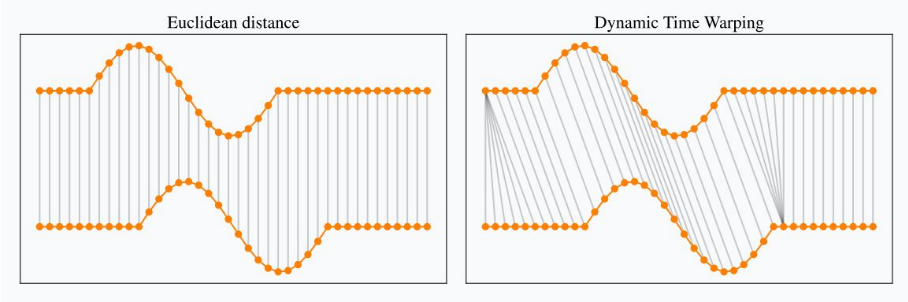
Distances:
_ DTW,
_ Edit Distance
_ Move–split–merge
_ etc…

- No “Training” required
- Useful for semi-supervised (and unsupervised)
- Somehow “Interpretable”

- Not SOTA Performance
- Does Not scale well with number of instances and timepoints
- Slow Inference

Distance Based: Elastic Distance

Compute a “proper distance” between the time series, and then classify using a distance based algorithm (K Nearest Neighbours).



Distances:
_ DTW,
_ Edit Distance
_ Move–split–merge
_ etc…

Good review paper:

SPRINGER NATURE Link

Find a journal | Publish with us | Track your research | Search

Home › Knowledge and Information Systems › Article

A review and evaluation of elastic distance functions for time series clustering

Review | Open access | Published: 07 September 2023

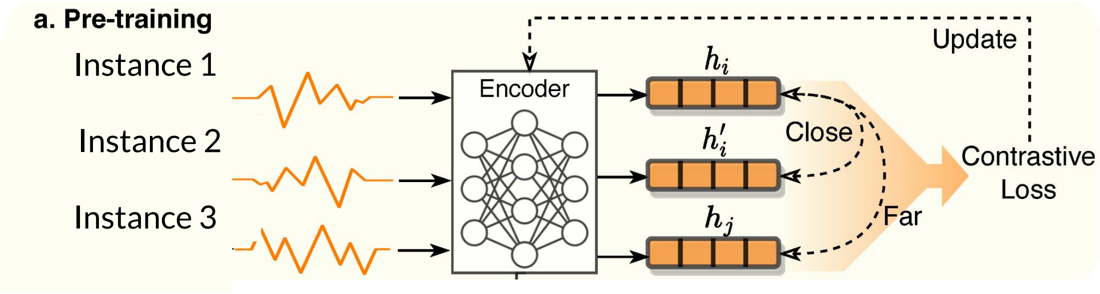
Volume 66, pages 765–809, (2024) Cite this article

- No “Training” required
- Useful for semi-supervised (and unsupervised)
- Somehow “Interpretable”

- Not SOTA Performance
- Does Not scale well with number of instances and timepoints
- Slow Inference

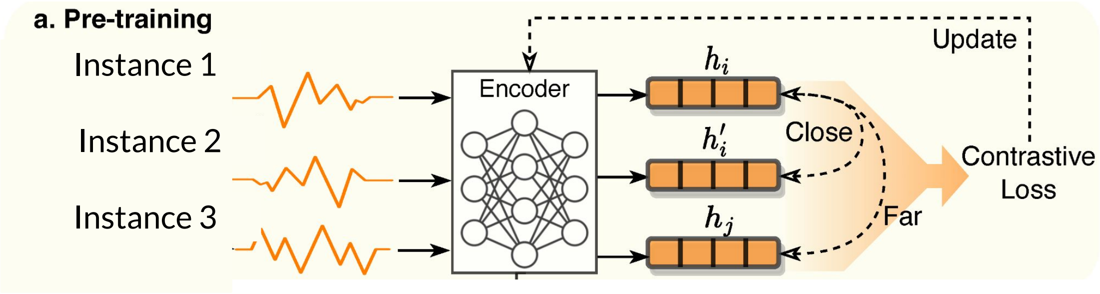
Distance Based: Representation Learning

Instead of imposing a given distance, we can try to “learn” a representation space with a “meaningful” metric.



Distance Based: Representation Learning

Instead of imposing a given distance, we can try to “learn” a representation space with a “meaningful” metric.



We will discuss this more in depth this strategy later.

- Achieves SOTA in some cases
- Fast inference

- Complex training
- Requires a large dataset
- Not obvious pretrain task

Classification Strategies

Time Series Classification

Feature Based

- Predefined Features
- Spectral Features
- Intervals / Dictionary

Distance Based

- Elastic distance
- Representation Learning

End-to-end

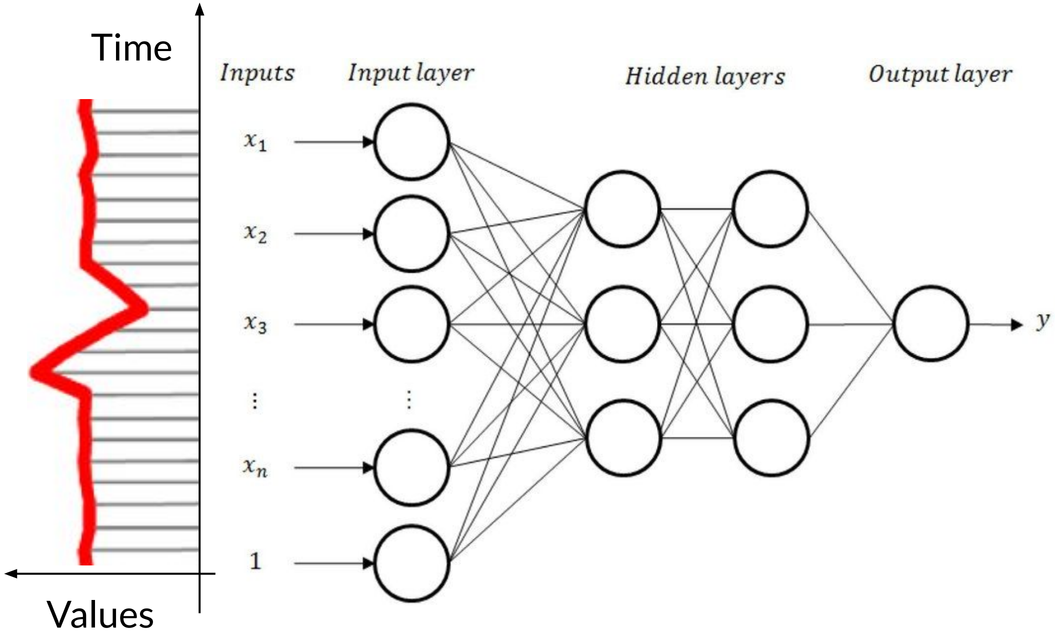
- CNNs
- RNNs
- Transformers

Focus

Neural Nets:
Naive MLP approach

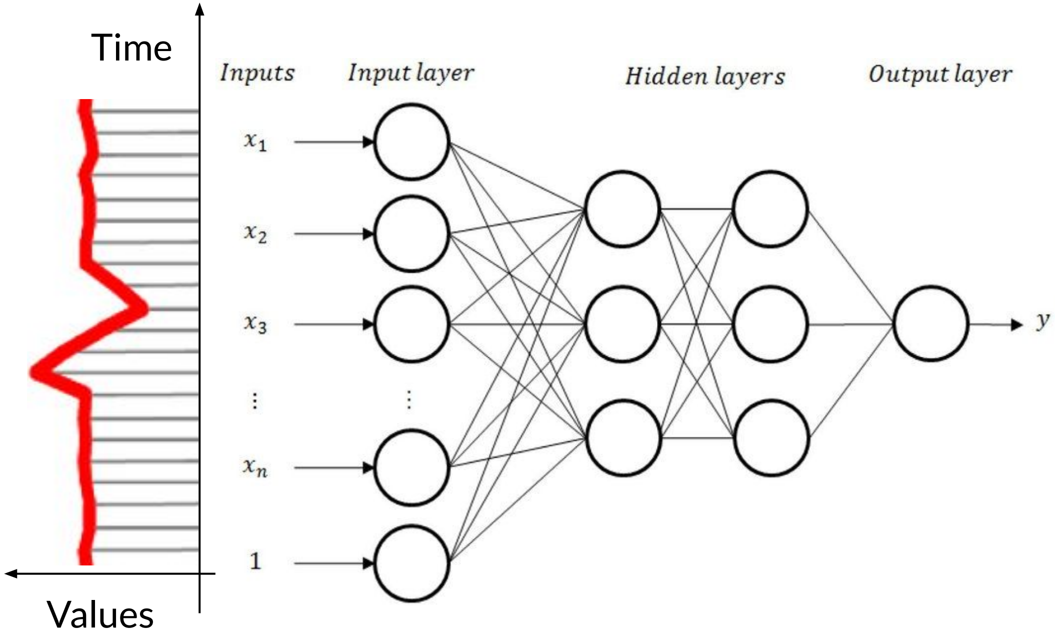
Naive MLP Approach

Why can't we use MLPs to classify?



Naive MLP Approach

Why can't we use MLPs to classify?

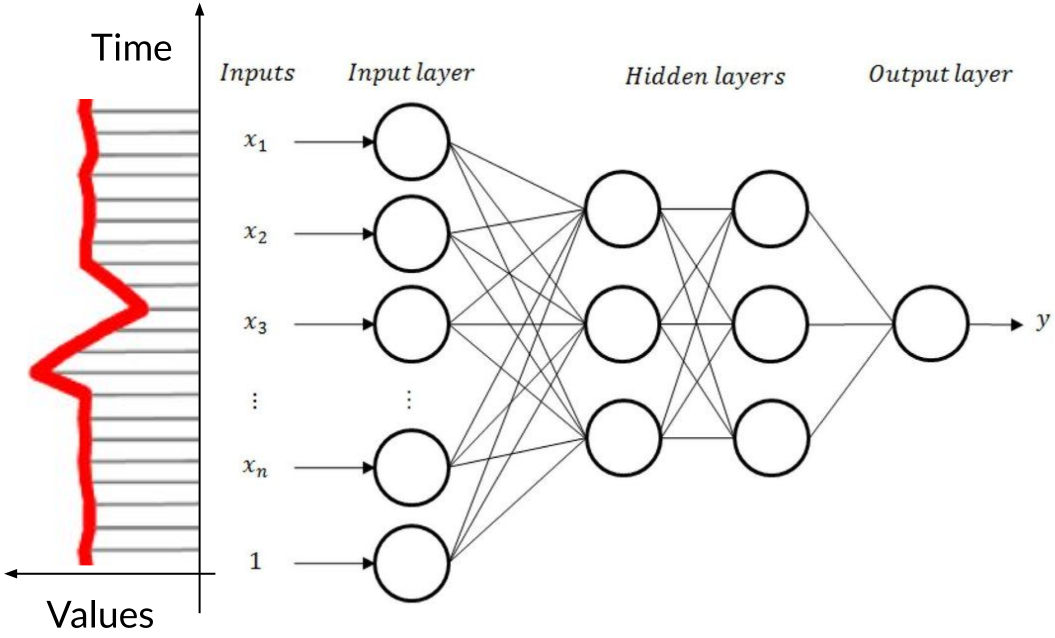


If we use **each timestep as a feature**:

- The number of parameters needed scale poorly with the input length
- Don't exploit the regularity & invariance of the data
- Two “similar” time series present different features.

Naive MLP Approach

Why can't we use MLPs to classify?



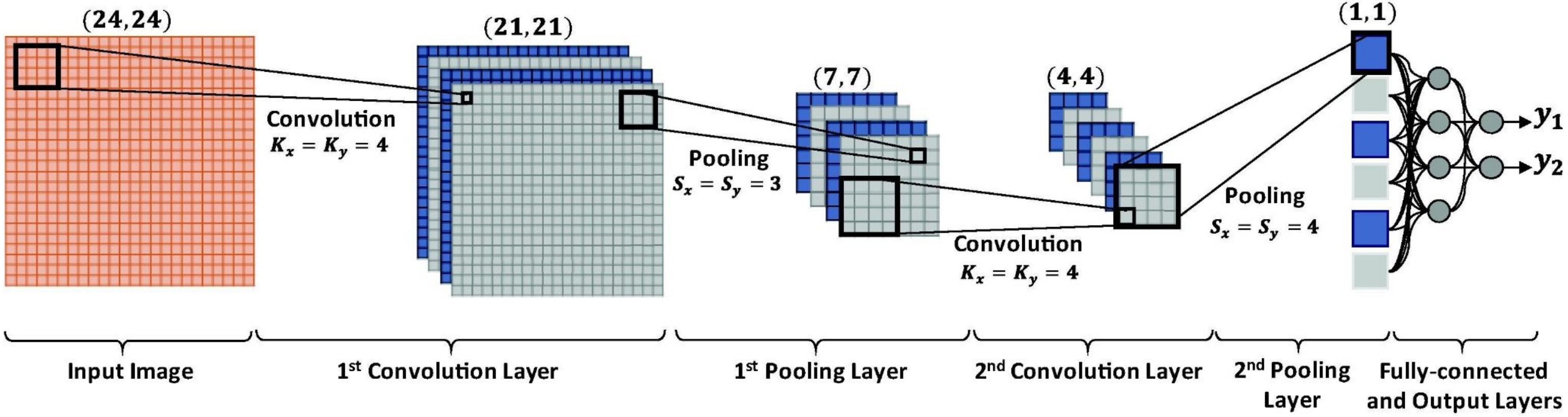
If we use **each timestep as a feature**:

- The number of parameters needed scale poorly with the input length
- Don't exploit the regularity & invariance of the data
- Two "similar" time series present different features.

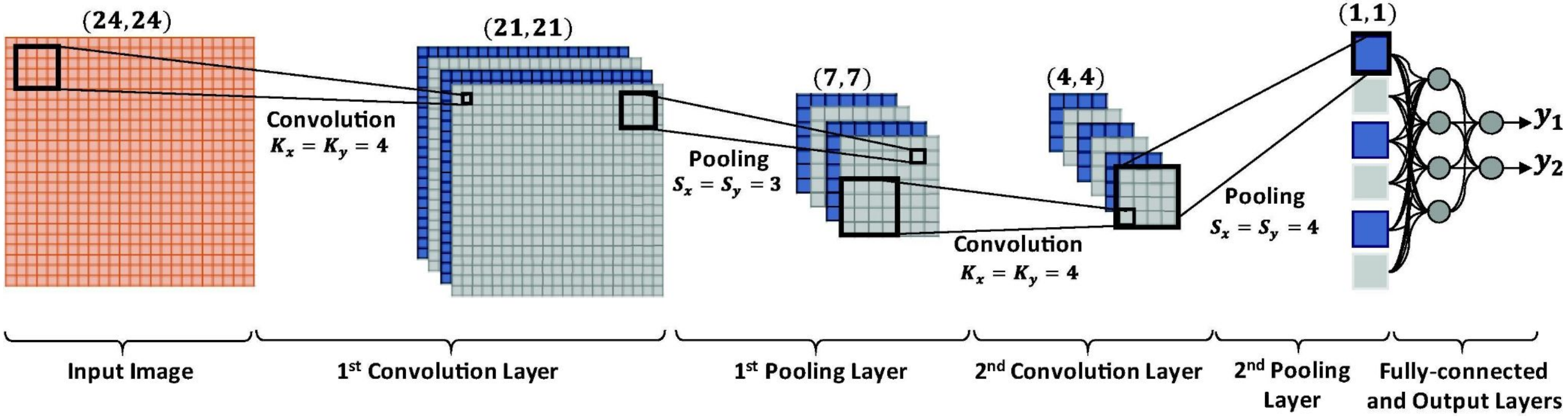
We would need a very large dataset to properly genelize.

Neural Nets: Convolutional Neural Networks (CNNs)

CNNs for Images Classification [Refresh]



CNNs for Images Classification [Refresh]



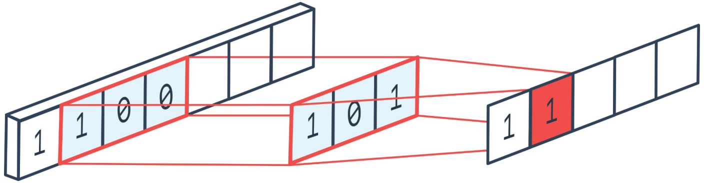
Key Concepts about CNNs

- **Weight Sharing:** Less parameters
- **Translational Invariance:** Exploit data regularities
- **Local Receptive Fields:** Spatial Hierarchies
- **Pooling Operations:** Proper Data Downsampling

CNNs for Time Series Classification

For time series we use 1D-convolutions.

Convolution moves in the time dimension.

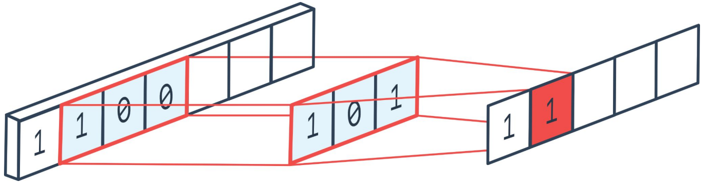


The kernel size is **(c x k)**, where c is the number of channels and k the number of elements in the kernel. In this example is **(1 x 3)**.

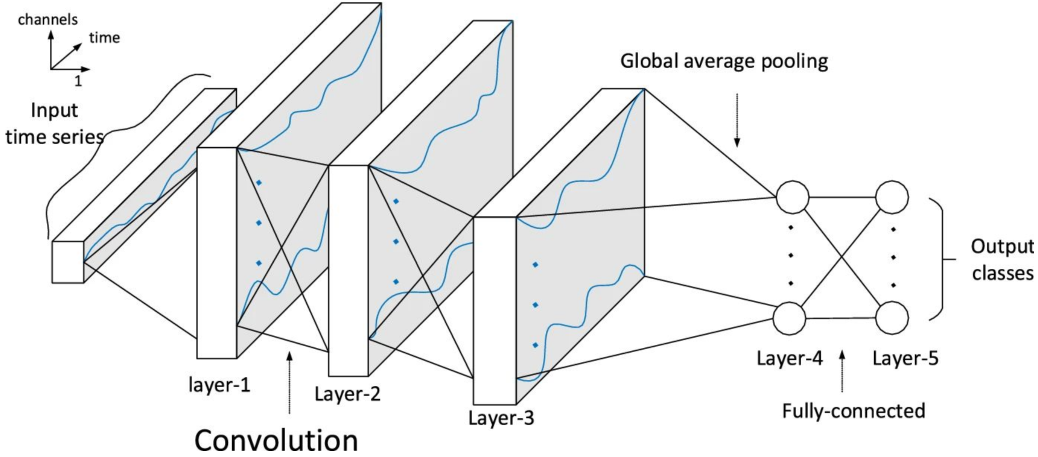
CNNs for Time Series Classification

For time series we use 1D-convolutions.

Convolution moves in the time dimension.



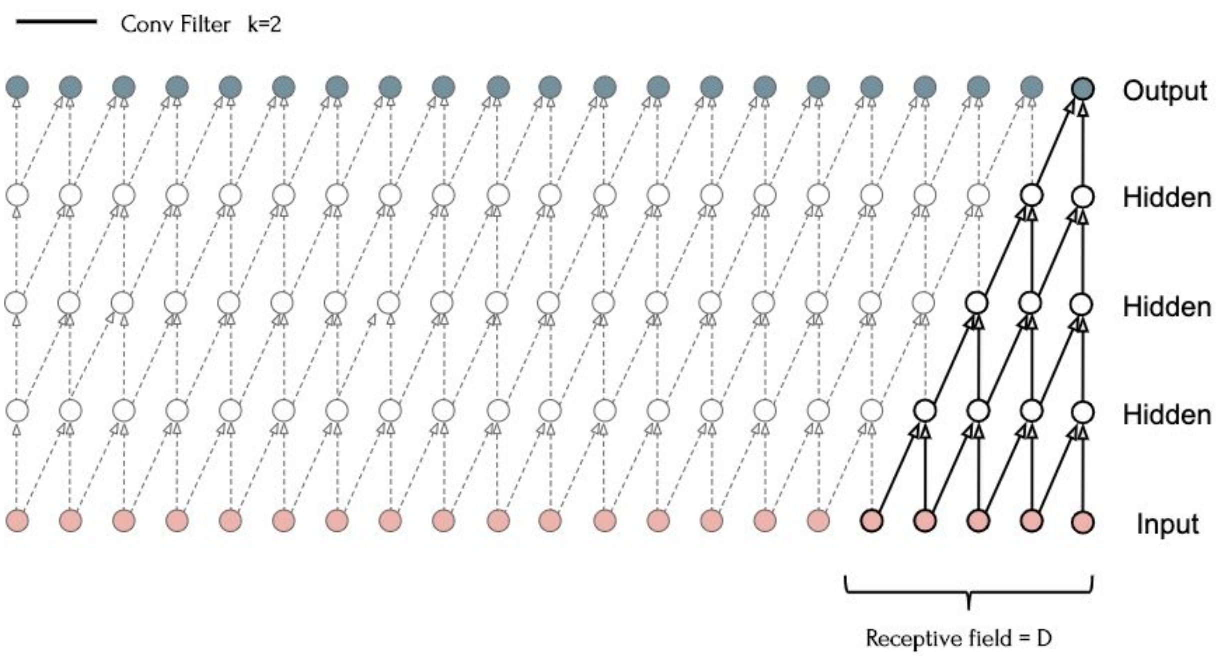
The kernel size is **(c x k)**, where c is the number of channels and k the number of elements in the kernel. In this example is **(1 x 3)**.



A **1-D CNN** with 3 conv layers anw 2 fully connected layers. Notice the time-pooling operation at the last convolutional layer.

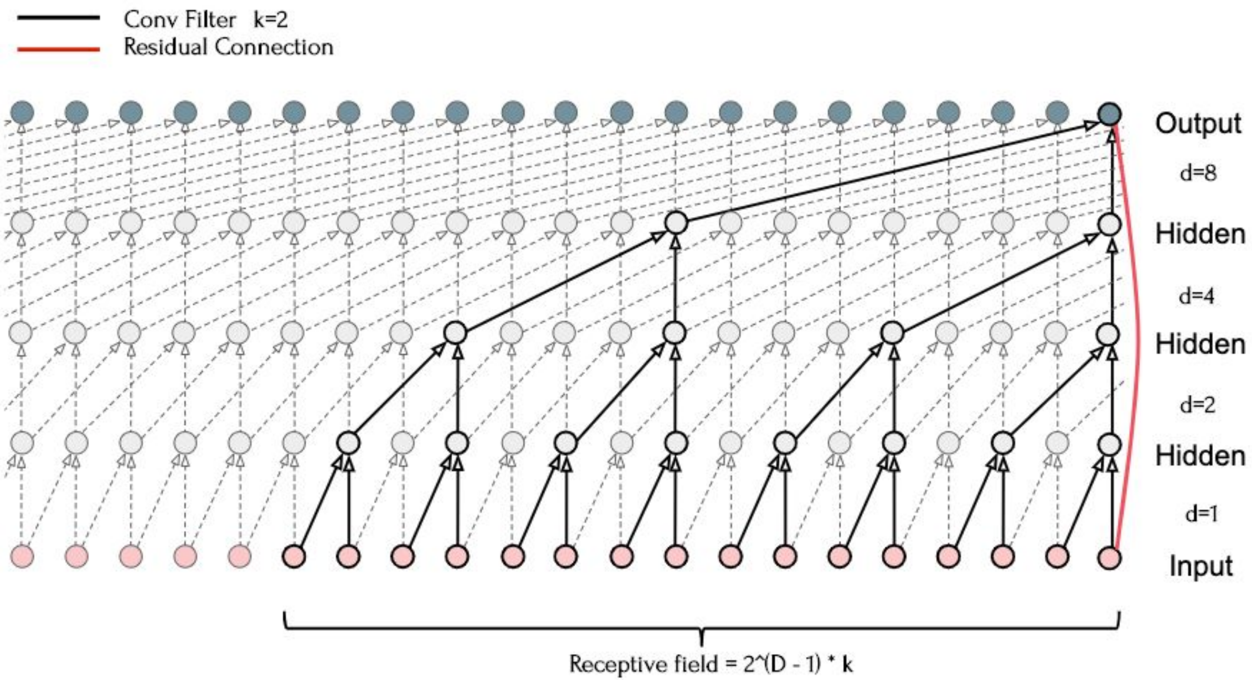
Receptive Field

To have a larger receptive field, we need a deeper network.



Receptive Field

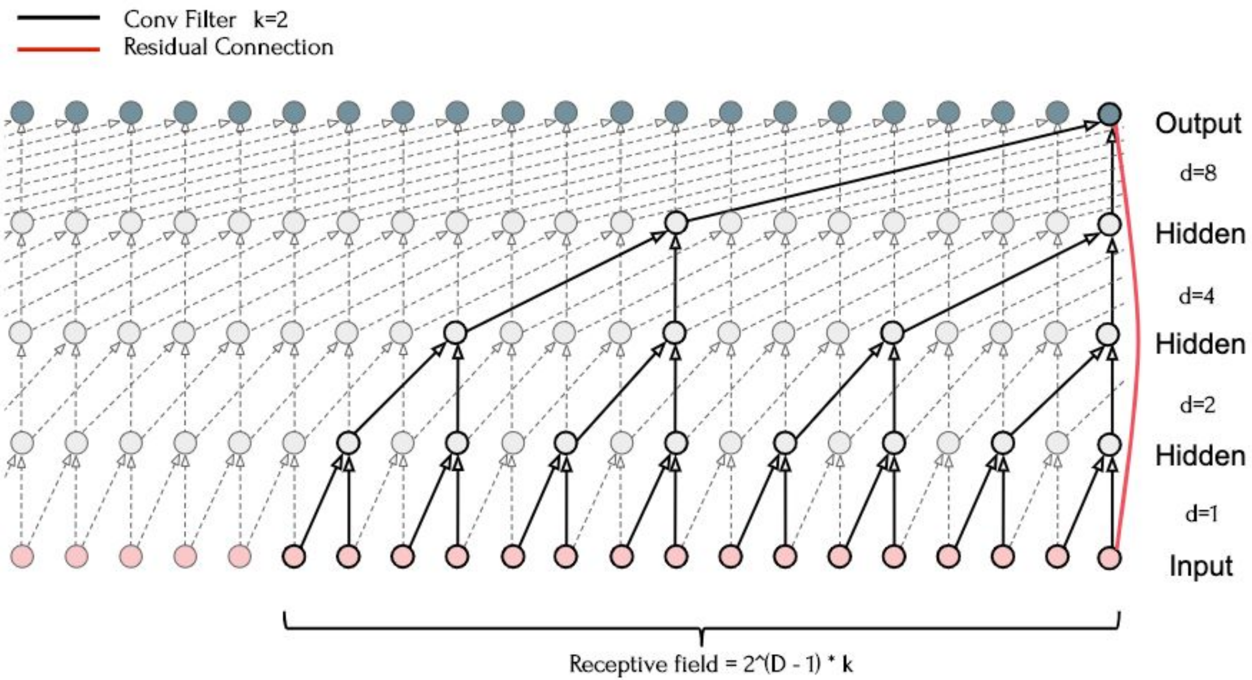
One way to solve this is using Dilated Convolutions.



Receptive Field

One way to solve this is using Dilated Convolutions.

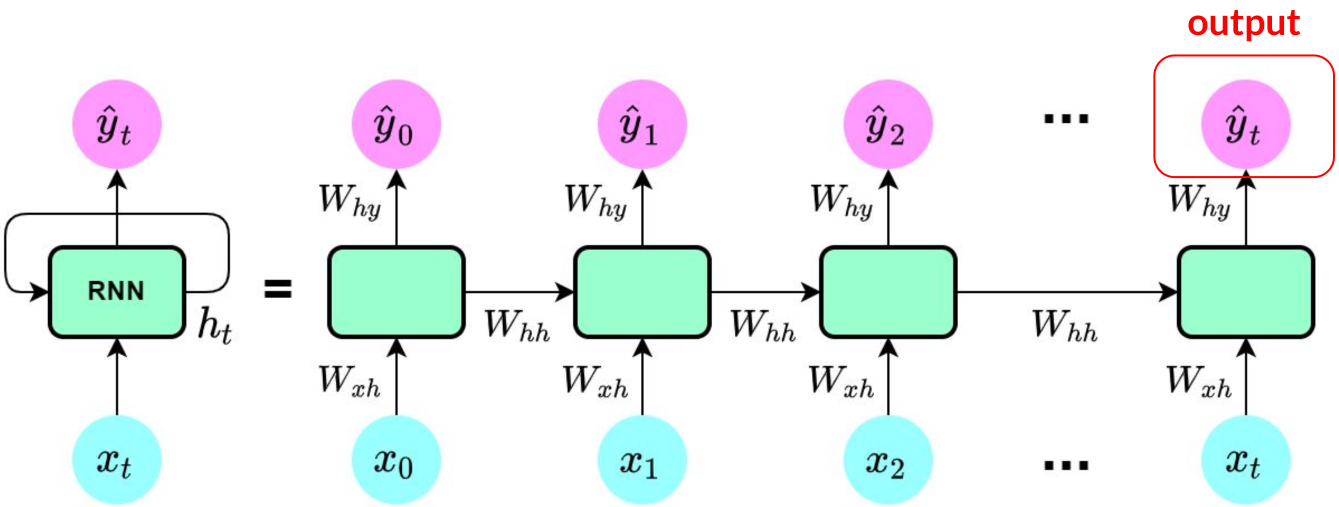
This strategy was proposed by Google Deepmind in 2016. The **WaveNet** architecture was the first capable of effectively dealing with raw audio time series.



Neural Nets: Recurrent Neural Networks (RNNs)

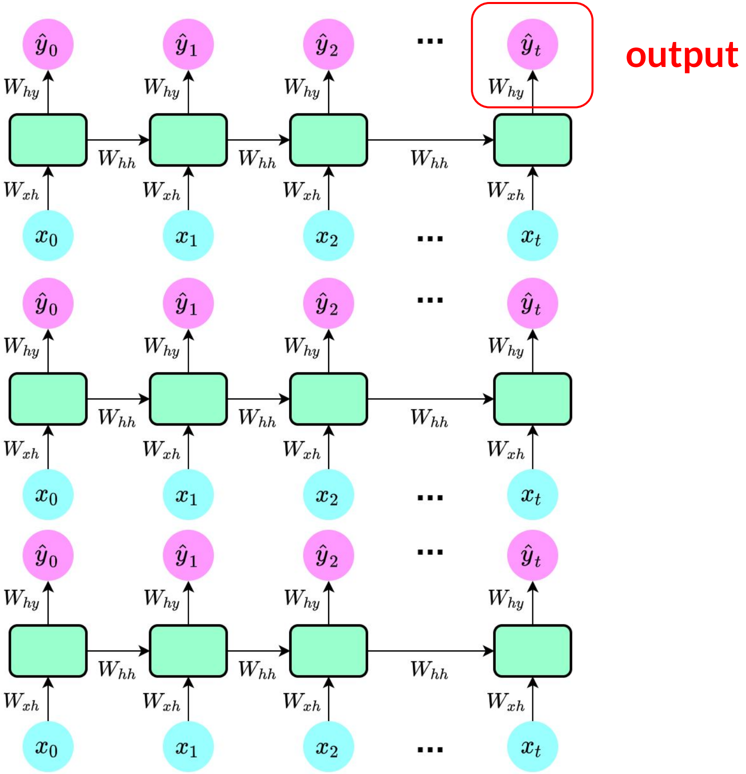
Time Series Data

Another strategy for sequential data is to recursively process each of the steps with the same network.

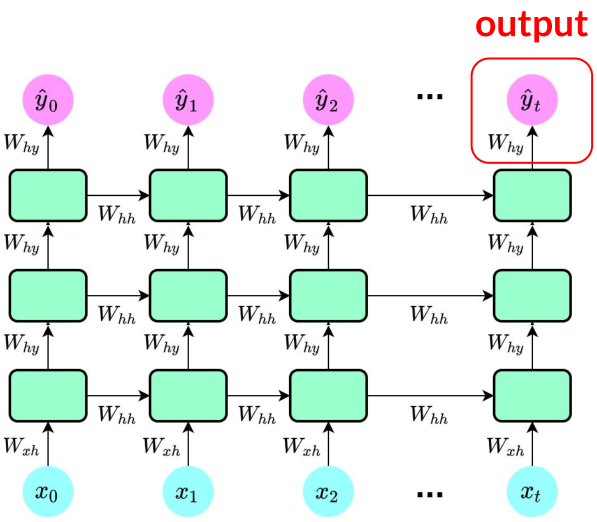


Unfolded RNN: Shared Weights!

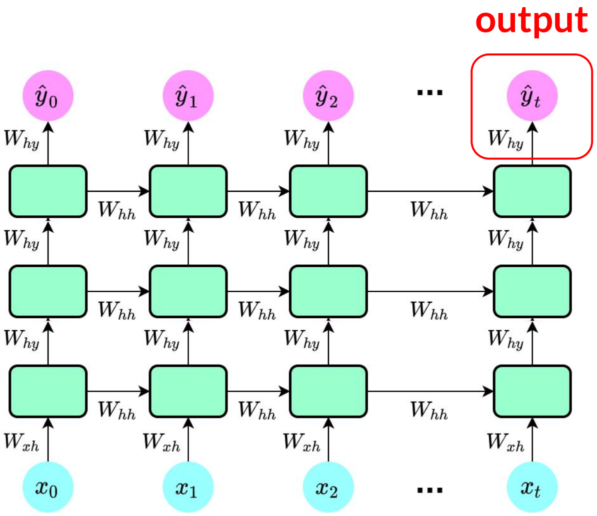
Time Series Data



Time Series Data



Time Series Data

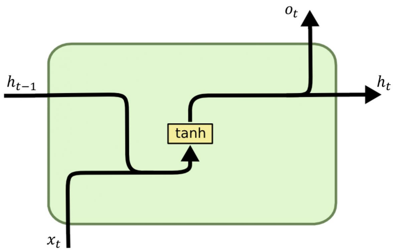


The cells can be stacked to create deeper and more complex models.

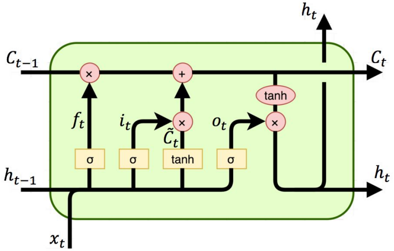
But they are **hard to train** (unstable and not parallelizable).

Time Series Data

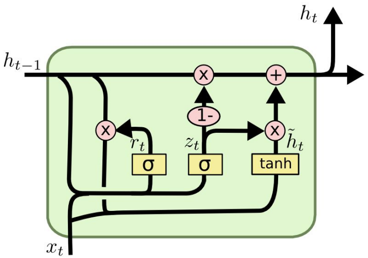
There are different types of RNNs cells:



Classical RNN



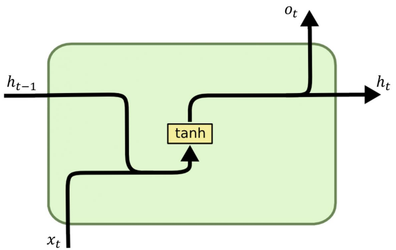
LSTM



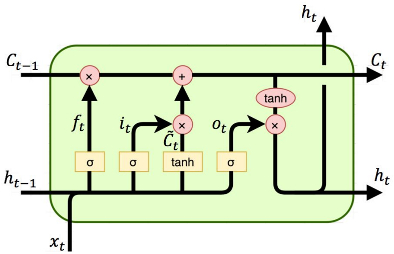
GRU

Time Series Data

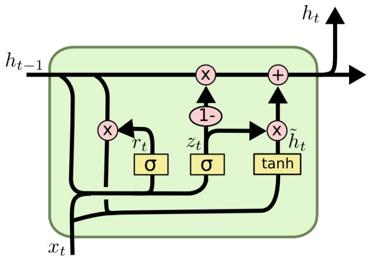
There are different types of RNNs cells:



Classical RNN



LSTM

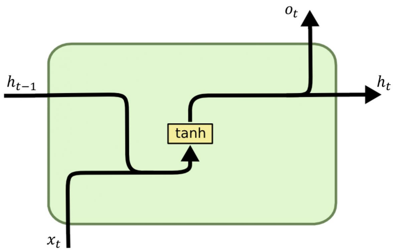


GRU

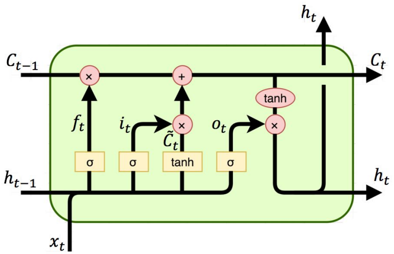
- Google's use of LSTMs in **Google Voice Search** in 2015 dramatically improved accuracy.
- In 2016, **Google Translate** started using neural networks (stacked LSTMs), having previously used statistical models.

Time Series Data

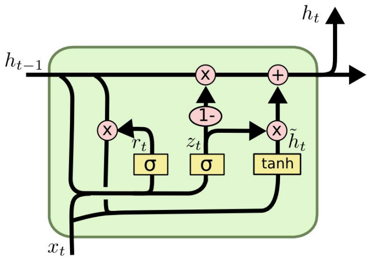
There are different types of RNNs cells:



Classical RNN



LSTM



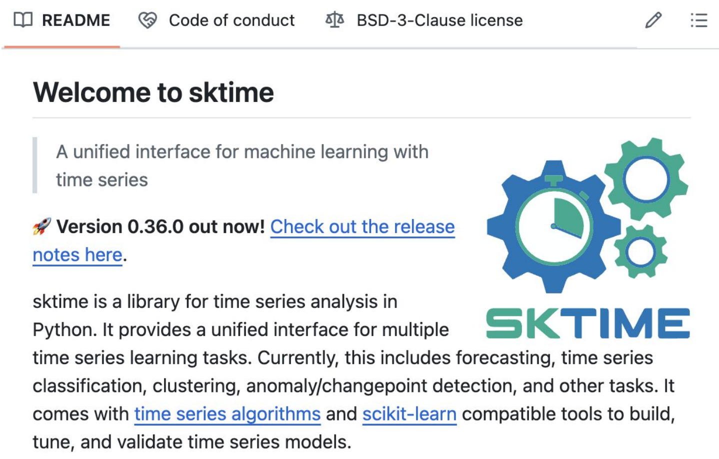
GRU

Best model for large sequential dataset before Transformers

- Google's use of LSTMs in **Google Voice Search** in 2015 dramatically improved accuracy.
- In 2016, **Google Translate** started using neural networks (stacked LSTMs), having previously used statistical models.

Python Libraries for Time Series

Time Series Data Libraries



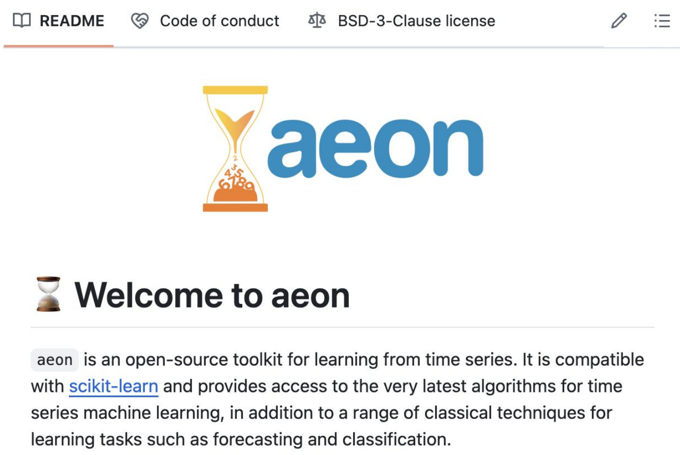
README Code of conduct BSD-3-Clause license

Welcome to sktime

> A unified interface for machine learning with time series

🚀 **Version 0.36.0 out now!** Check out the release notes here.

sktime is a library for time series analysis in Python. It provides a unified interface for multiple time series learning tasks. Currently, this includes forecasting, time series classification, clustering, anomaly/changepoint detection, and other tasks. It comes with time series algorithms and scikit-learn compatible tools to build, tune, and validate time series models.



README Code of conduct BSD-3-Clause license

⌛ Welcome to aeon

`aeon` is an open-source toolkit for learning from time series. It is compatible with scikit-learn and provides access to the very latest algorithms for time series machine learning, in addition to a range of classical techniques for learning tasks such as forecasting and classification.

Largest library for TS analysis.
A lot of implemented models and functions. (scikit-learn style)

Newer library for TS analysis.
Fewer models, but more curated and up to date. (scikit-learn style)

Hands-on Time: Notebook 2

Hands-on Repository

github.com/gon-uri/EIVIA2025