DenseNets for Time Series Classification: towards automation of time series pre-processing with CNNs

Guillaume Richard[*+], Georges Hébrail[*], Mathilde Mougeot[+] and Nicolas Vayatis[+]

[*]EDF Lab Paris-Saclay [+]ENS Paris-Saclay

Introduction

It is well known that data normalization is a fundamental pre-processing step for learning using Convolutional Neural Networks (CNN). Multiple normalization techniques have been proposed and finding an appropriate one is not an easy task.

Motivated by applications in the energy consumption field, we study Time Series Classification (TSC) with deep learning techniques. We adapt DenseNets to a new convolutional architecture for TSC.

We conduct an experimental study the impact of different data normalization techniques on this architecture. We propose a solution to mitigate different pre-processing methods and show its applicability across various fields.

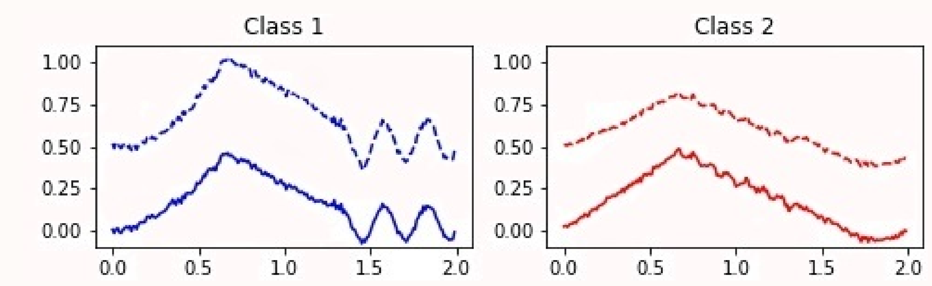
Normalization techniques

Given n time series $\{X_1, ..., X_n\} \in \mathbb{R}^T$. For $i \in \{1, ..., n\}$ and $j \in \{1, ..., T\}$:

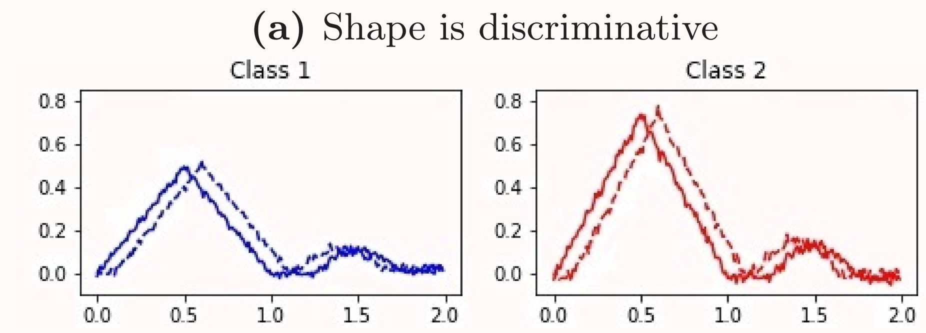
	Min-Max	Standardization
Global	$GN(X_i^j) = \frac{X_i^j - min(\mathbf{X})}{max(\mathbf{X}) - min(\mathbf{X})}$	$GS(X_i^j) = \frac{X_i^j - mean(\mathbf{X})}{std(\mathbf{X})}$
Instance	$IN(X_i^j) = \frac{X_i^j - min(X_i)}{max(X_i) - min(X_i)}$	$IS(X_i^j) = \frac{X_i^j - mean(X_i)}{std(X_i)}$

Table 1: Normalization methods

1. **Scaling:** mostly used to limit the influence of outlier points in the time series.
2. **Normalization:** necessary step to train neural networks. Standardization makes data distribution closer to normal and normalization rescales it to $[0, 1]$.
3. **Instance vs Global:** instance (or z-) normalization removes offset and variance information from time series.



(a) Shape is discriminative



(b) Scale is discriminative

Figure 1: Toy examples

- In Figure 1a: z-normalization is necessary
- In Figure 1b: z-normalization would be ill-advised

DenseNet and normalization

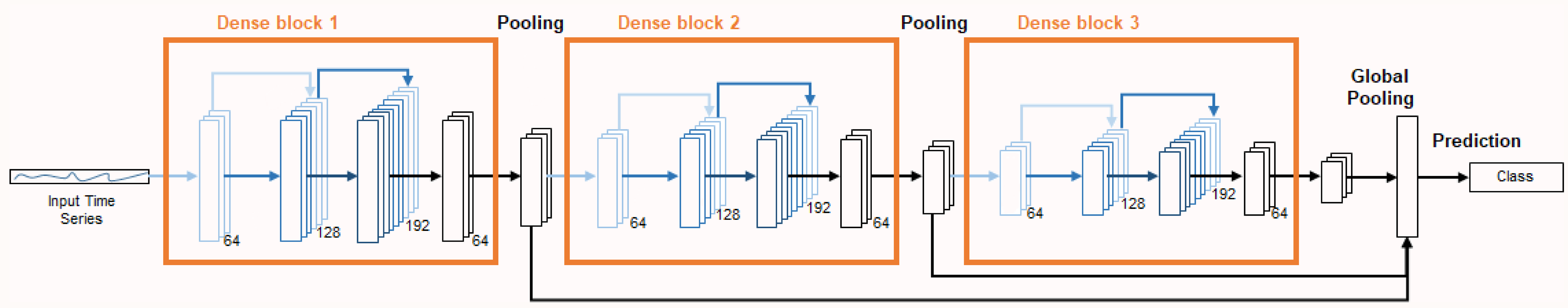


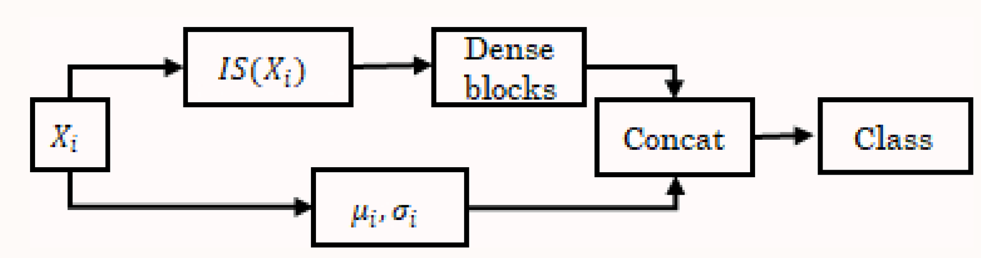
Figure 2: DenseNet: each block is made of successive convolutions and skip connections ended by a bottleneck convolution

Layers	Input shape	Output shape	Filter shape
Input	Time series of length l		
Conv (1)	$l \times 1$	$l \times 64$	1×7
Conv (2)	$l \times 64$	$l \times 64$	64×5
Conv (3)	$l \times 128$	$l \times 64$	128×3
Conv (4)	$l \times 192$	$l \times 64$	192×3
Pooling (1)	$l \times 64$	$l/2 \times 64$	2

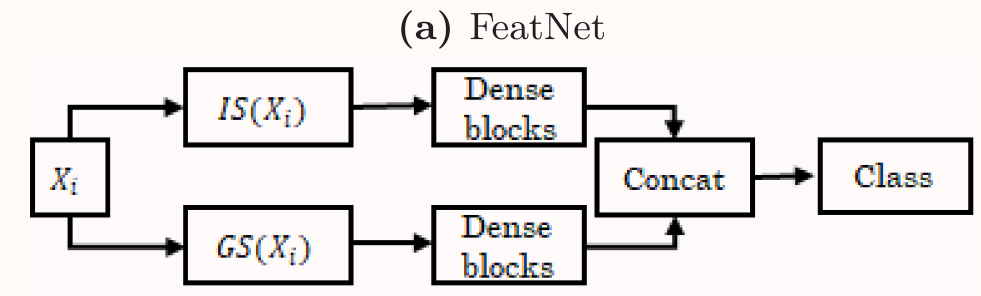
Table 2: Dense block ; 3 consecutive dense blocks are used in the final architecture

FeatNet One entry corresponds to the instance-normalized time series, which is passed into a convolutional architecture. At the fully-connected level, the output of the convolutional blocks is concatenated with the other entry, containing the scale information from the time series

EnsNormNet Different entries corresponding to the input time series, normalized and scaled differently for each entry. Each entry is passed into convolutional blocks with no weight sharing. The outputs are then concatenated into a fully connected layer that gives the final prediction.



(a) FeatNet



(b) EnsNormNet

Figure 3: Proposed architectures

Electrical Appliance Recognition

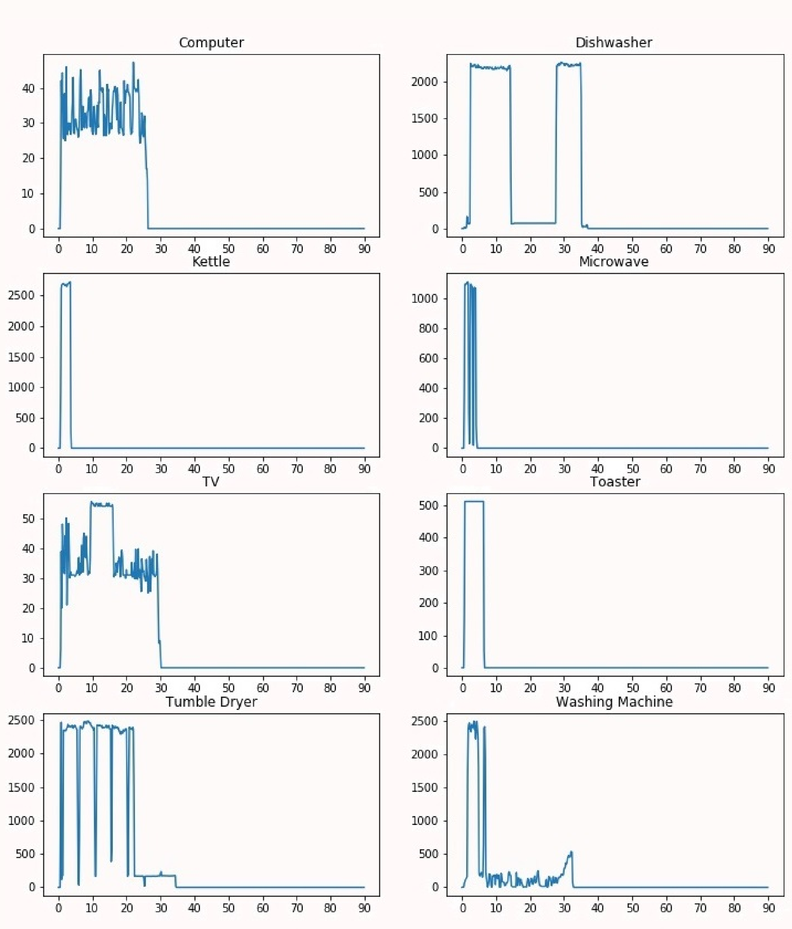


Figure 4: Appliance signatures (x-axis: minutes ; y-axis: W)

REFIT [2]: dataset of smart meter measurements of household electricity consumption. Our goal is to automatically recognize home appliances based on their electric consumption profiles. Namely we transform create a time series classification problem with the following characteristics:

- Extraction of single signatures from 6 devices and get an appliance identification task
- Signatures are padded to length of 540, corresponding to 90 minutes (10s sampling)
- Out of 20 houses, we extract $28,890$ signatures

Results

Procedure

- Global Standardization (**GS**)
- Global min-maxNormalization (**GN**)
- Instance Standardization (**IS**)
- Instance min-max Normalization (**IN**)
- Box-Cox Transformation + Global Standardization (**BC-GS**)
- Box-Cox Transformation + Instance Standardization (**BC-IS**)
- Instance Standardization + FeatNet (**IS-Feat**)
- Instance Standardization - Global Standardization + EnsNormNet (**IS-GS-ENN**)

Using K-fold validation and keeping houses used for training separate from houses used for testing, we compare the different normalization techniques. We ran the experiments 10 times for each classifier with $K = 5$ and compare the macro F1-score to take into account class imbalance.

Results

The best performance is achieved by the ensemble of transformations (**EnsNormNet**). One can see that instance standardization has a bad impact on classification performance, especially for discriminating devices similar in shape such as Toaster/Kettle/Microwave or Computer/TV. At the same time, **GS** is not optimal for separating Dishwashers from Washing Machines for instance. For this application, we highlight that **IS-GS-ENN** gets the best of both worlds.

As assumed in [1], z-normalization is not necessarly the best choice for time series classification using convolutional networks. Mixing different transformation of time series in a Convolutional Network offer a solution to balance shapes and levels in time series and leads to more robust classifiers.

GS	GN	IS	IN	BC-GS	BC-IS	IS-Feat	**IS-GS-ENN**
78,37 (0,63)	77,37 (0,58)	75,69 (0,89)	75,48 (0,76)	77,99 (0,43)	75,83 (0,73)	76,11 (0,97)	**83,39 (0,54)**

Table 3: F1 score (%) for different architectures with standard deviation over 10 runs

References

[1] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. 09 2018.

[2] David Murray and Lina Stankovic. REFIT: Electrical load measurements. University of strathclyde. 2015.