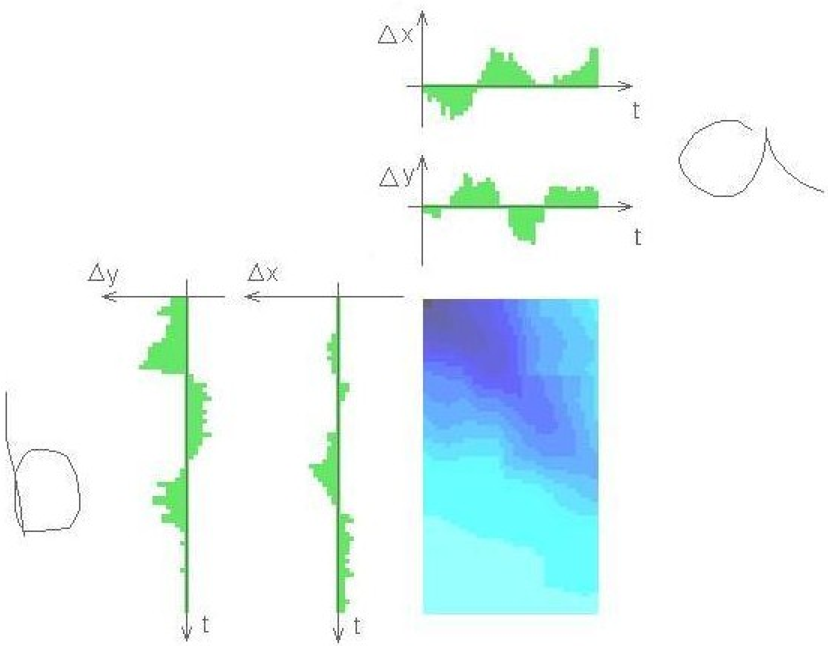
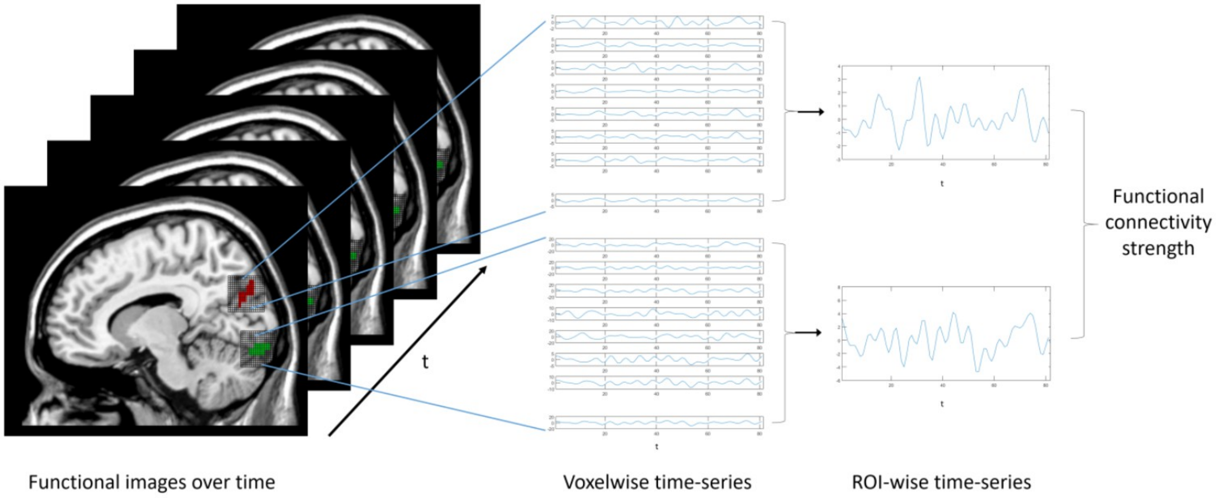
Tutorial: Time Series Classification and its Applications

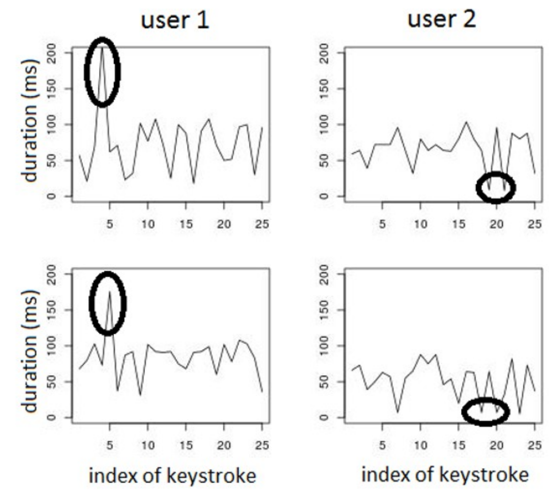


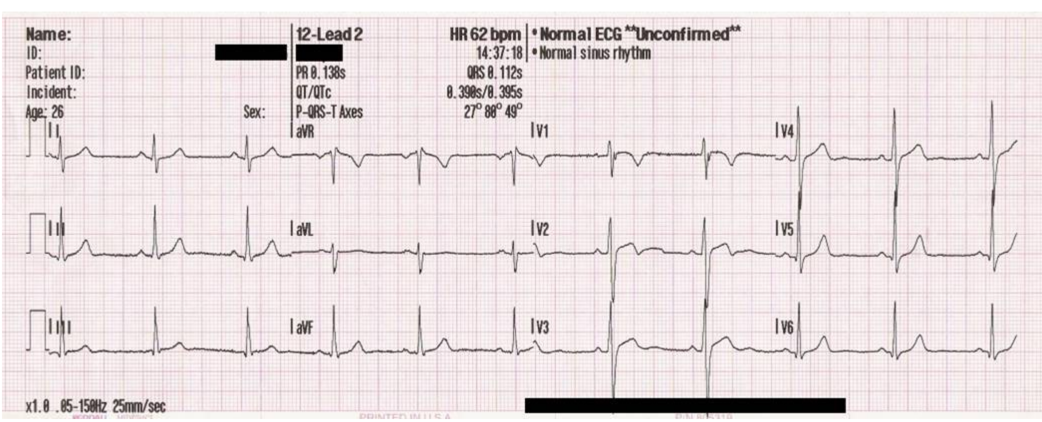
Krisztian Buza
buza@biointelligence.hu

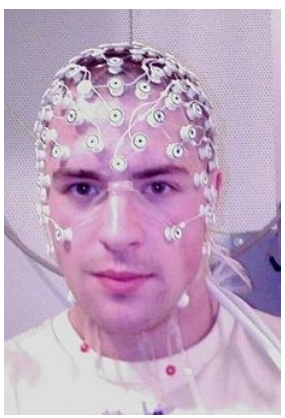
8th International Conference on Web Intelligence, Mining and Semantics.
June 25 – 27 2018, Novi Sad, Serbia

Time Series Classification – Examples









Images in the bottom, from left to right:
By MoodyGroove - 2007-01-24 (original upload date) Original uploader was MoodyGroove at en.wikipedia, Public Domain, https://commons.wikimedia.org/w/index.php?curid=5266589
By Thuglas at English Wikipedia - Transferred from en.wikipedia to Commons by Sreejithk2000 using CommonsHelper., Public Domain, https://commons.wikimedia.org/w/index.php?curid=10827060
By JSquish - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=16181727

Outline

- Categorisation of Time Series
- Quick Overview of Time Series Data Mining
- Time Series Classification Tasks
- (Some of the) Preprocessing Techniques
- Time Series Classification Techniques
  - Deep Neural Networks, DTW, Nearest Neighbor and its extensions
- Evaluation of Time Series Classifiers
- Selected Applications

Categorisation of Time Series

Categorisation of Time Series

time series
(w.r.t. type of observations)

- univariate
  (temperature, salary...)
- multivariate
  (EEG, ECG, coordinates on a touch screen,...)
- Time series of complex instances
  (e.g. spatiotemporal data: fMRI)

time series
(w.r.t. sampling frequency)

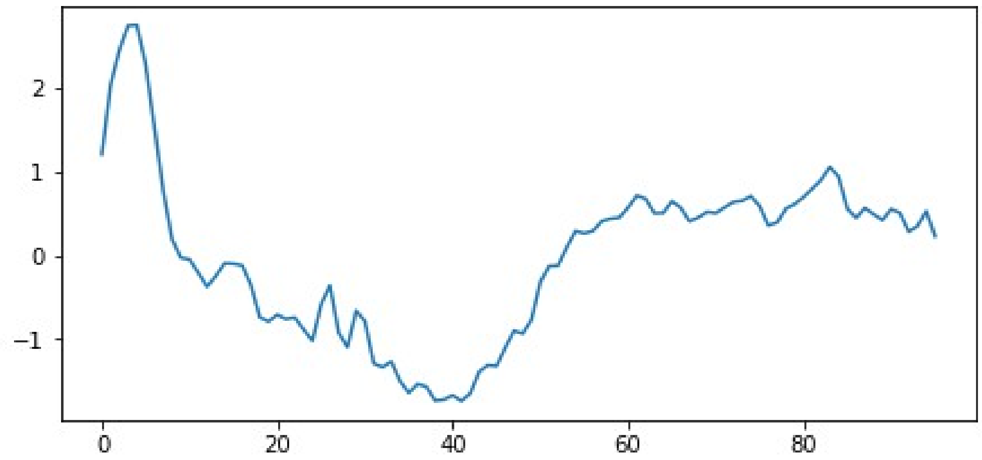
- „standard“
  (evenly sampled)
- unevenly sampled

Univariate Time Series

- Sequence of numbers
  (measurements in subsequent moments of time)

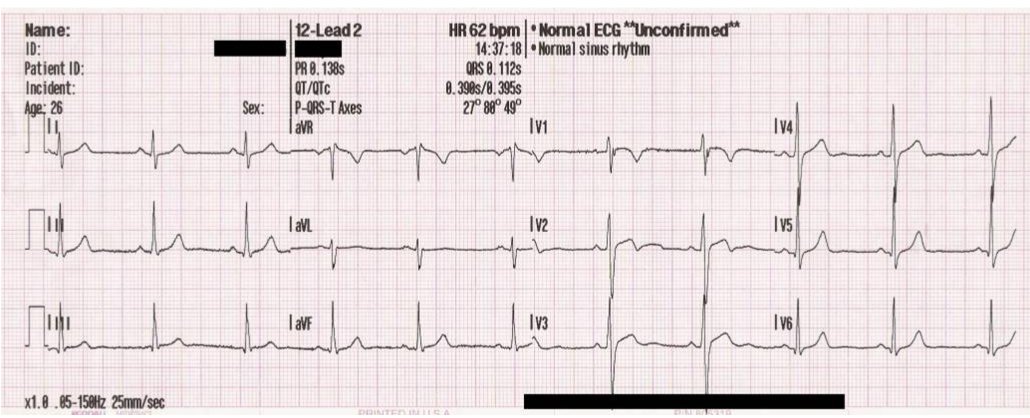
$$T = (x_1, \ldots, x_n) \qquad x_i \in \mathbb{R}$$

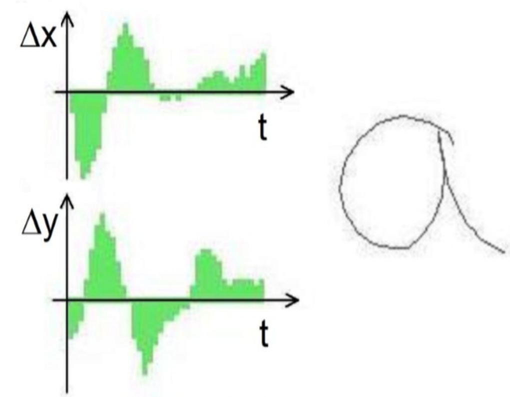
- E.g. temperature, speed of a car, salary...



Multivariate Time Series

- Sequence of vectors
- E.g. measurements describing weather conditions, ECG, EEG, (x,y) coordinates...

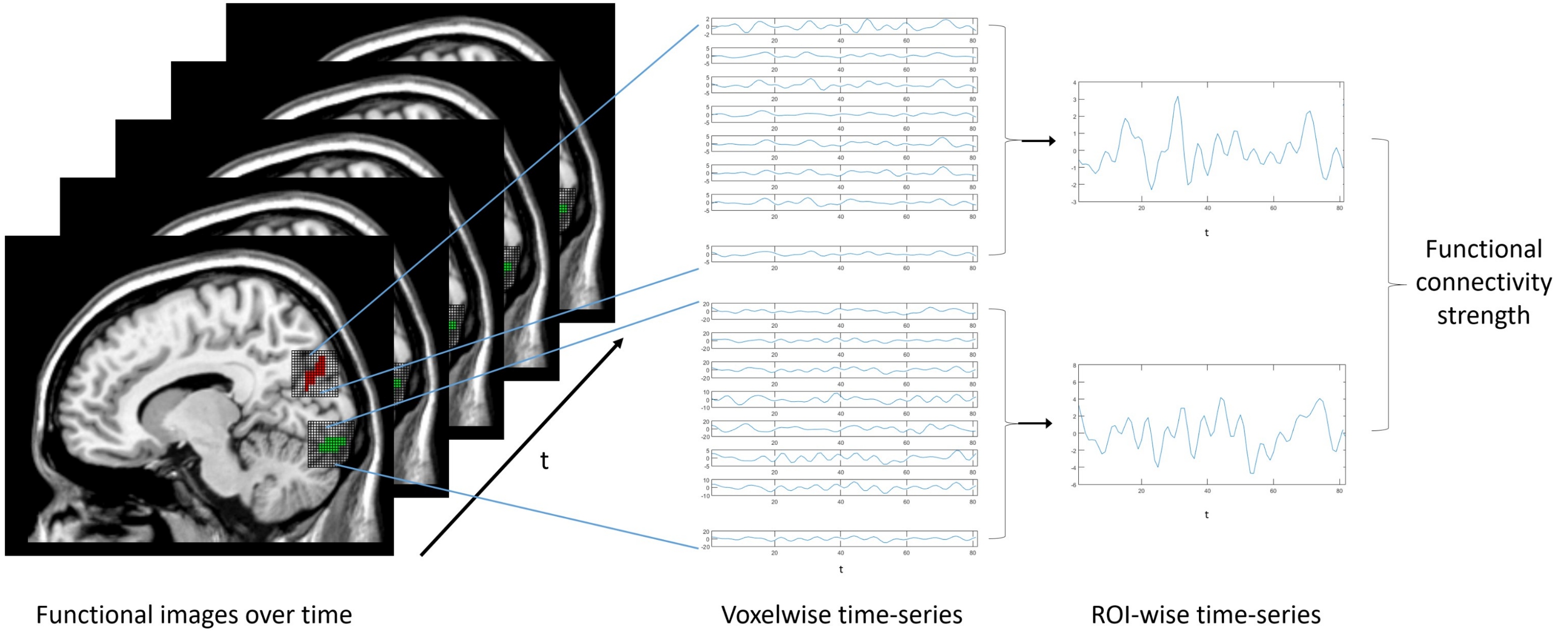




Images from left to right:
By MoodyGroove - 2007-01-24 (original upload date) Original uploader was MoodyGroove at en.wikipedia, Public Domain, https://commons.wikimedia.org/w/index.php?curid=5266589
By Thuglas at English Wikipedia - Transferred from en.wikipedia to Commons by Sreejithk2000 using CommonsHelper, Public Domain, https://commons.wikimedia.org/w/index.php?curid=10827060
K. Buza (2011): Fusion methods for time series classification, http://www.ismll.uni-hildesheim.de/pub/pdfs/Buza_thesis.pdf

Time Series of Complex Instances

- E.g. functional magnetic resonance imaging (fMRI) data
- May be transformed to simpler time series for analysis



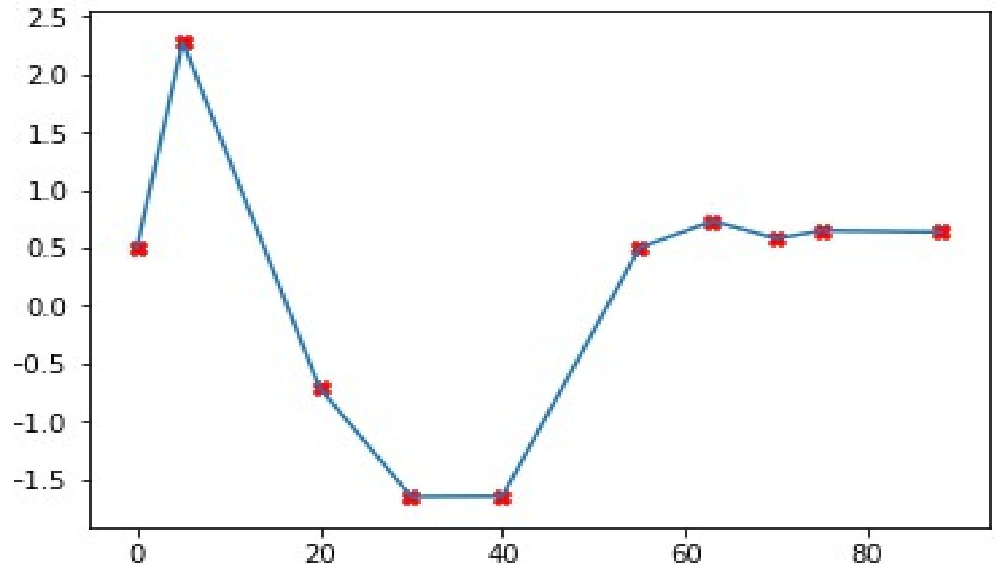
A. Szenkovits, R. Meszlényi, K. Buza, N. Gaskó, R.I. Lung, M. Suciu (2018): Feature Selection with a Genetic Algorithm for Classification of Brain Imaging Data, in U. Stanczyk, B. Zielosko, L.C. Jain: Advances in Feature Selection for Data and Pattern Recognition, Springer

Unevenly Sampled Time Series

- E.g. blood pressure of patient is measured irregularly
- Each observation x_i is associated with a time stamp t_i

$$T = (\ t_1 : x_1,\ t_2 : x_2, \ldots, t_n : x_n\ )$$

- Note: observation x_i may be a value, vector or complex instance
- Interpolation may be necessary

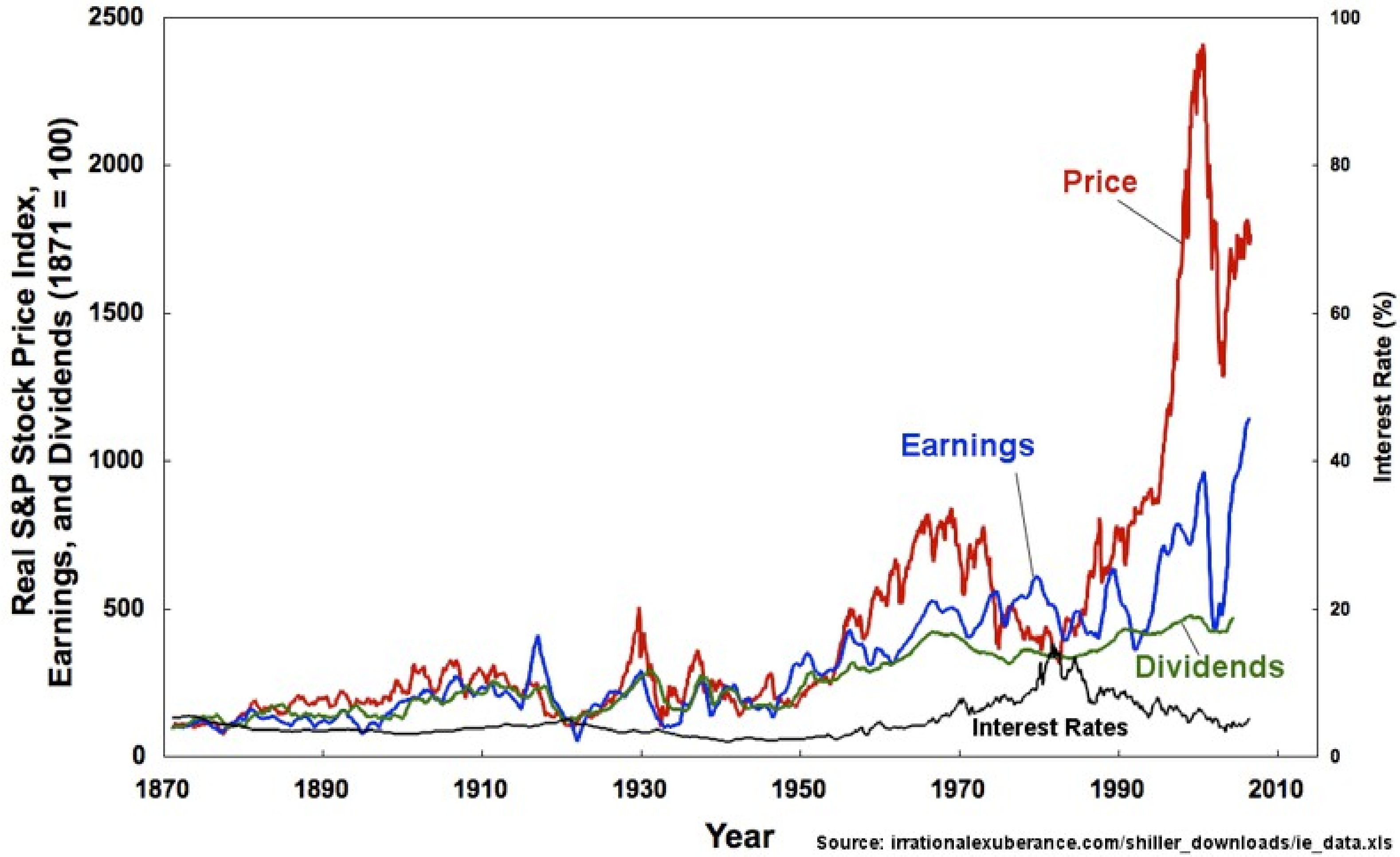


Quick Overview of Time Series Data Mining

Time Series Data Mining

- Time Series Forecasting
- Store Time Series Efficiently
- Similarity Search
- Clustering
- Anomaly Detection in Time Series Data
- **Time Series Classification**
- ...

Time Series Forecasting



By Frothy (Own work) [GFDL (http://www.gnu.org/copyleft/fdl.html) or CC BY-SA 4.0-3.0-2.5-2.0-1.0 (https://creativecommons.org/licenses/by-sa/4.0-3.0-2.5-2.0-1.0)], via Wikimedia Commons

Store Time Series Efficiently

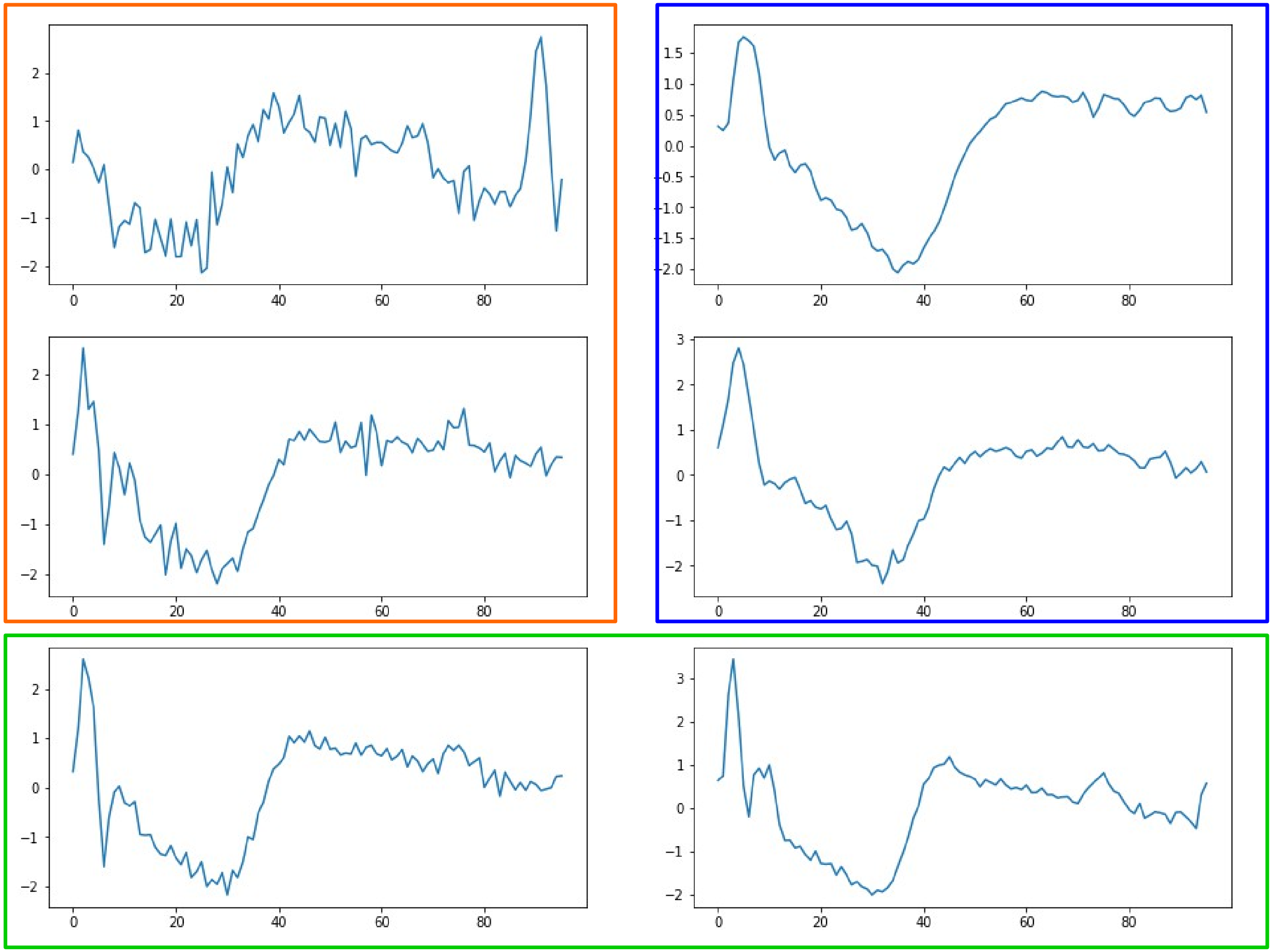
Time	Temp. (°C)	Hum. (%)	Press. (Pa)	Wind (v) (km/h)	Wind (dir.)	Radiation	Outlook
10:21	15	20	100 200	5	SW	low	
10:22	16	20	100 200	5	SW	low	
10:38	16	30	100 100	5	SW	low	
10:40	17	30	100 100	5	SW	medium	
10:43	18	30	100 100	10	SW	medium	
10:44	18	30	100 100	15	W	medium	
10:51	18	20	100 200	15	W	medium	

Time	Hum. (%)	Press. (Pa)
10:21	20	100 200
10:38	30	100 100
10:51	20	100 200

Time	Temp. (°C)	Wind (v) (km/h)	Wind (dir.)	Radiation	Outlook
10:21	15	5	SW	low	
10:22	16	5	SW	low	
10:40	17	5	SW	medium	
10:43	18	10	SW	medium	
10:44	18	15	W	medium	

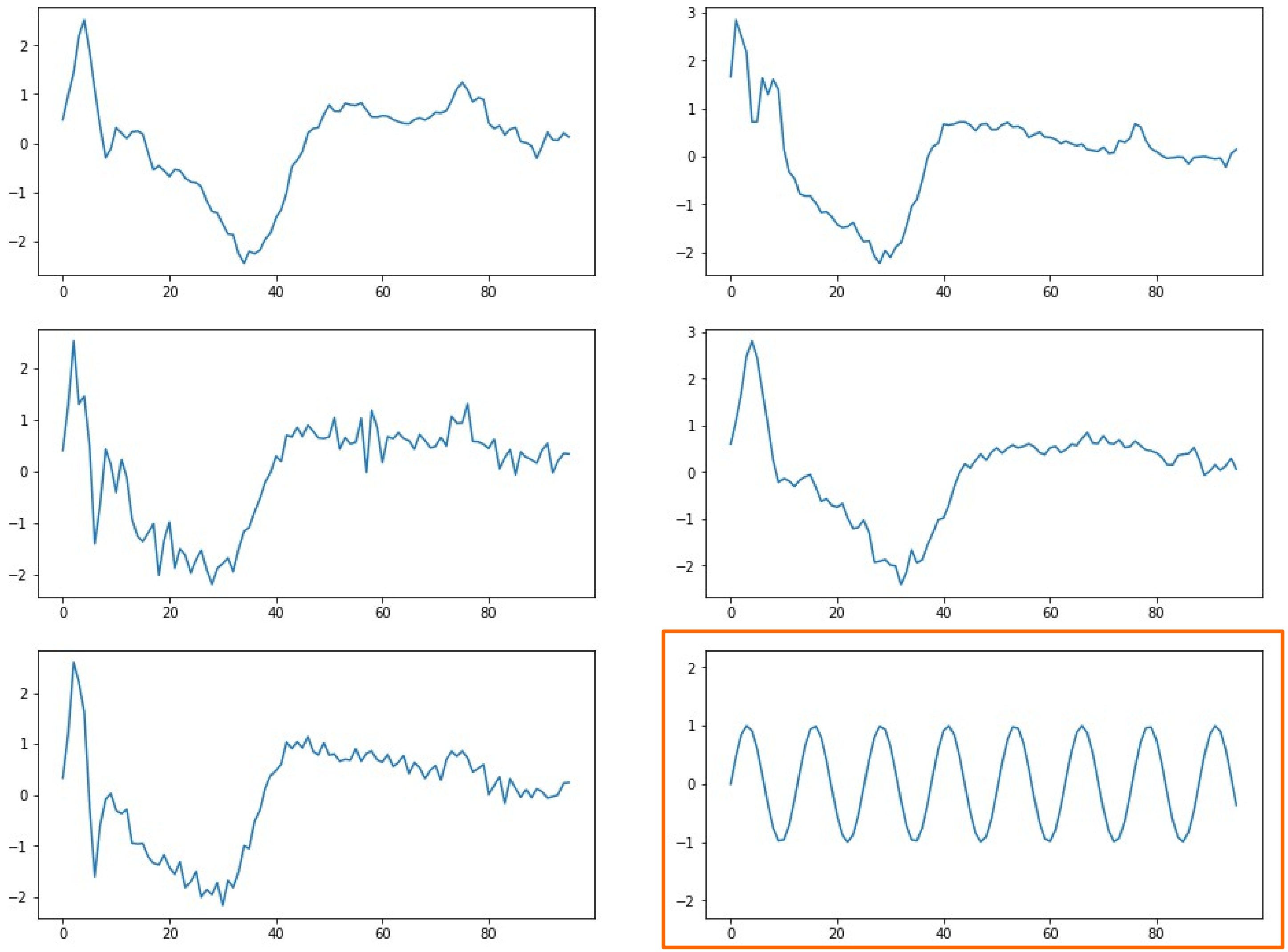
K. Buza, G. Nagy, A. Nanopoulos (2014): Storage-Optimizing Clustering Algorithms for High-Dimensional Tick Data, Expert Systems with Applications, 41, pp. 4148-4157

Clustering

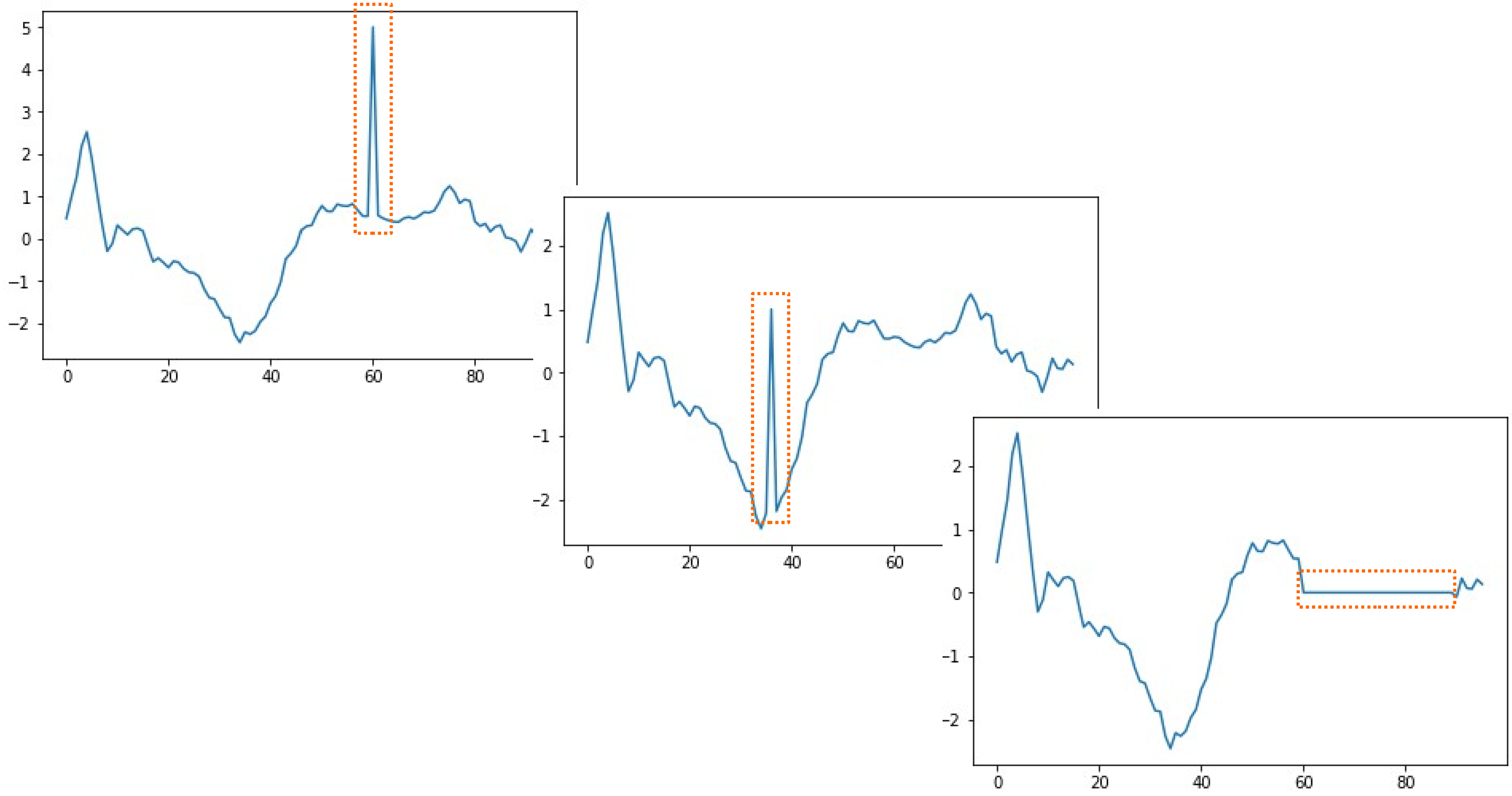


T. Warren Liao (2005): Clustering of time series data – a survey. Pattern recognition 38,11, pp. 1857–1874.

Anomaly Detection



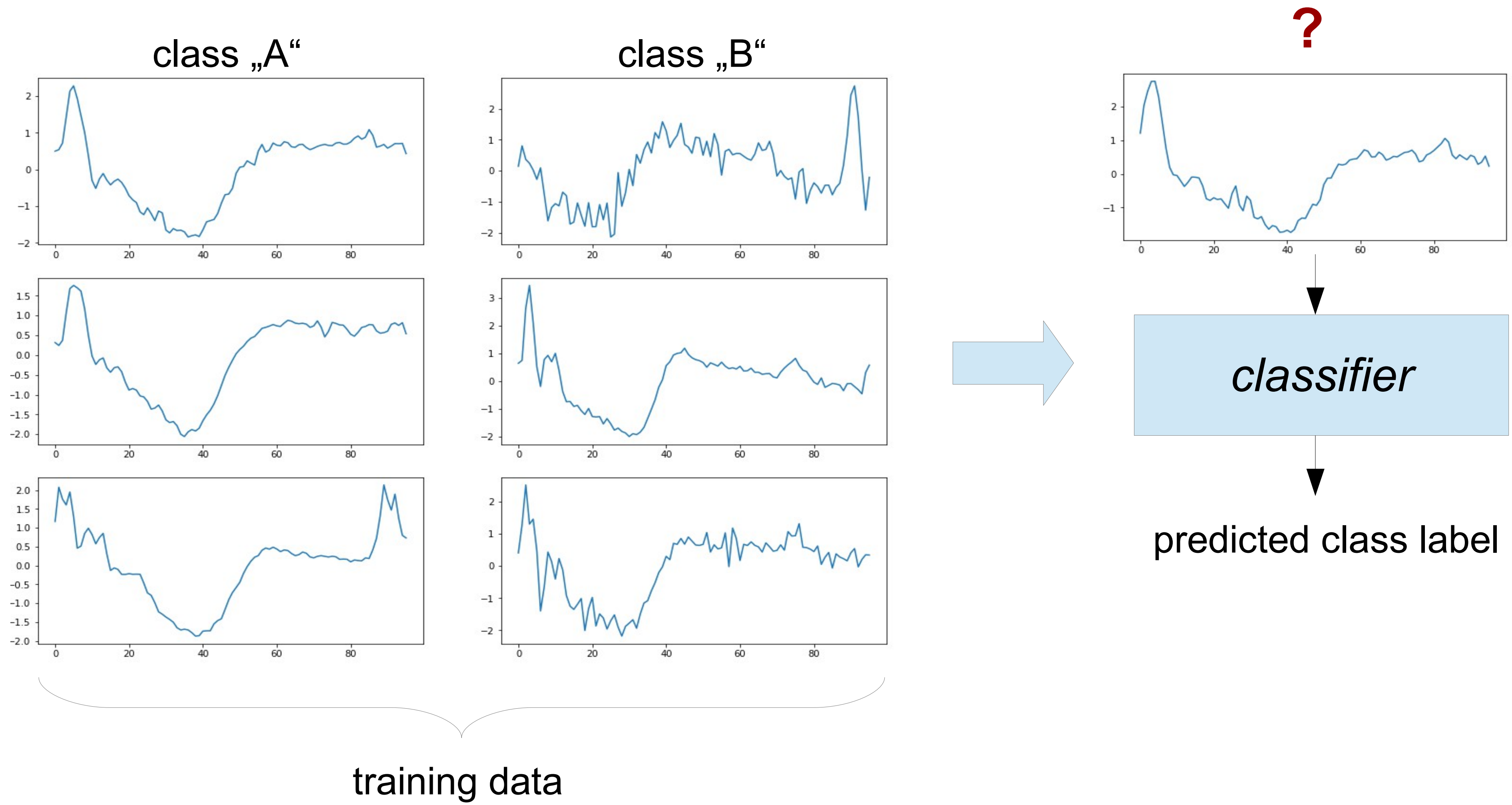
Anomaly Detection:
Point Anomaly, Contextual Anomaly, Collective Anomaly



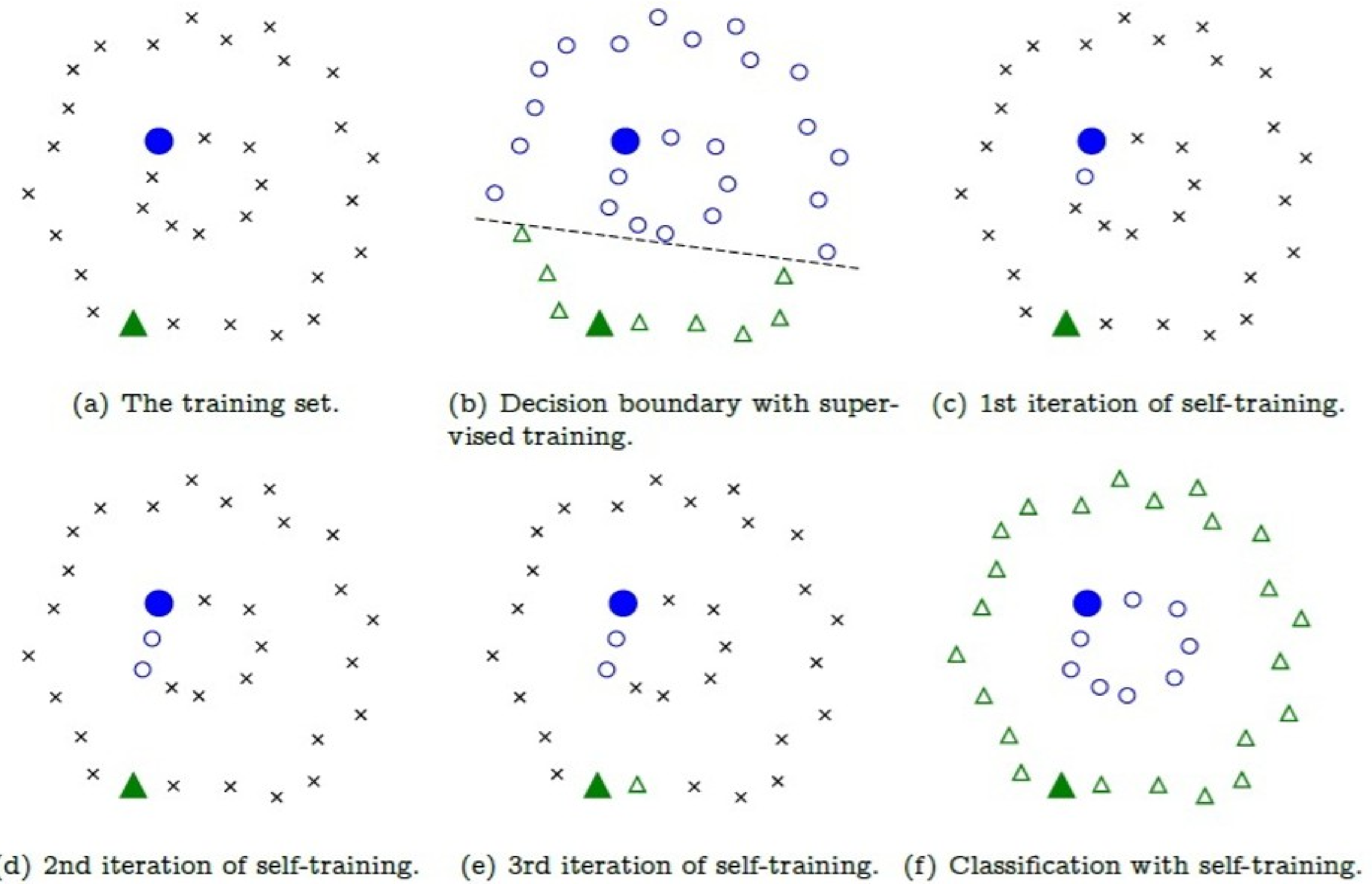
Time Series Classification Tasks

(<u>not</u> the solutions yet)

(Conventional) Time Series Classification Problem



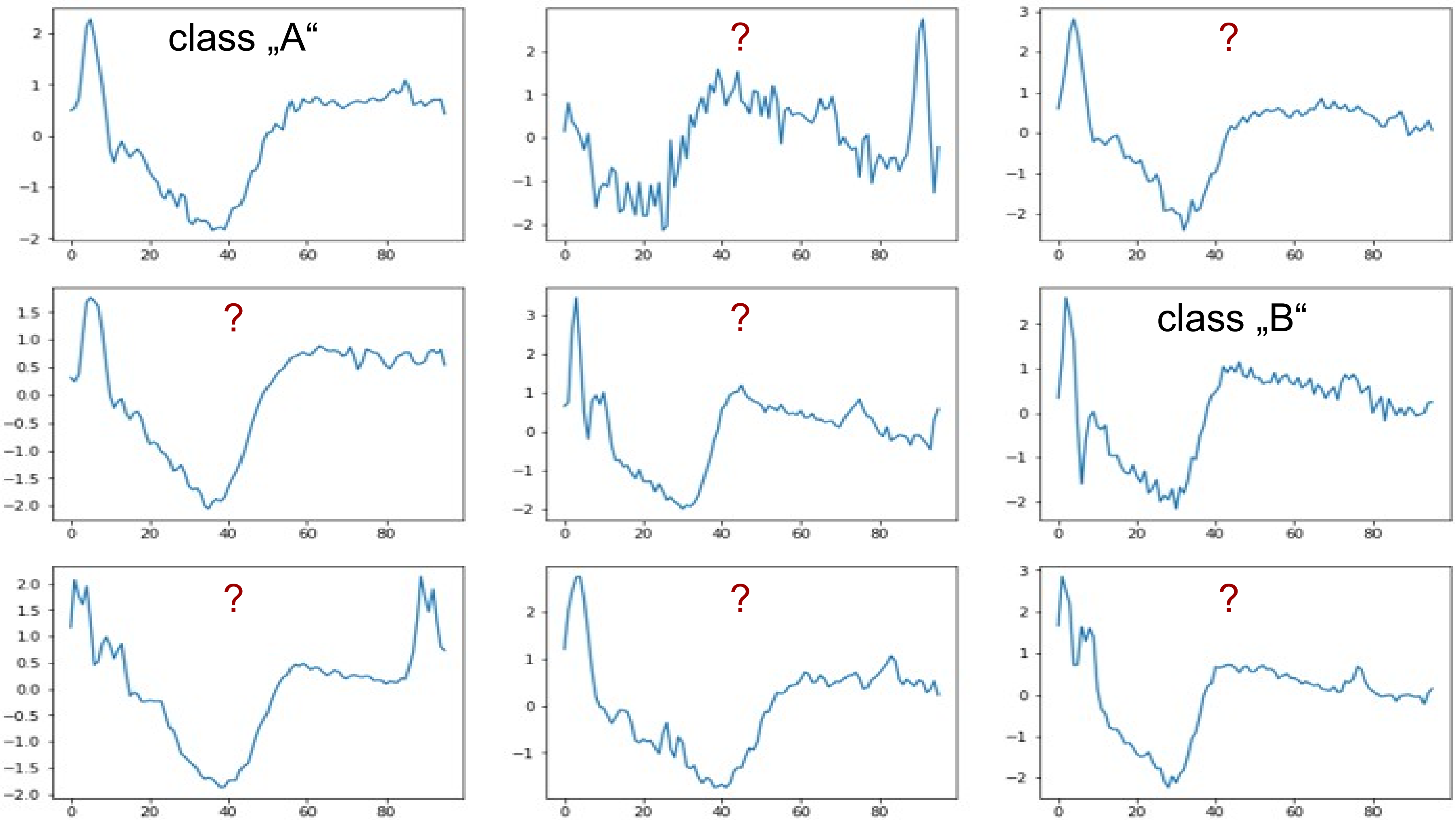
Semi-Supervised Classification



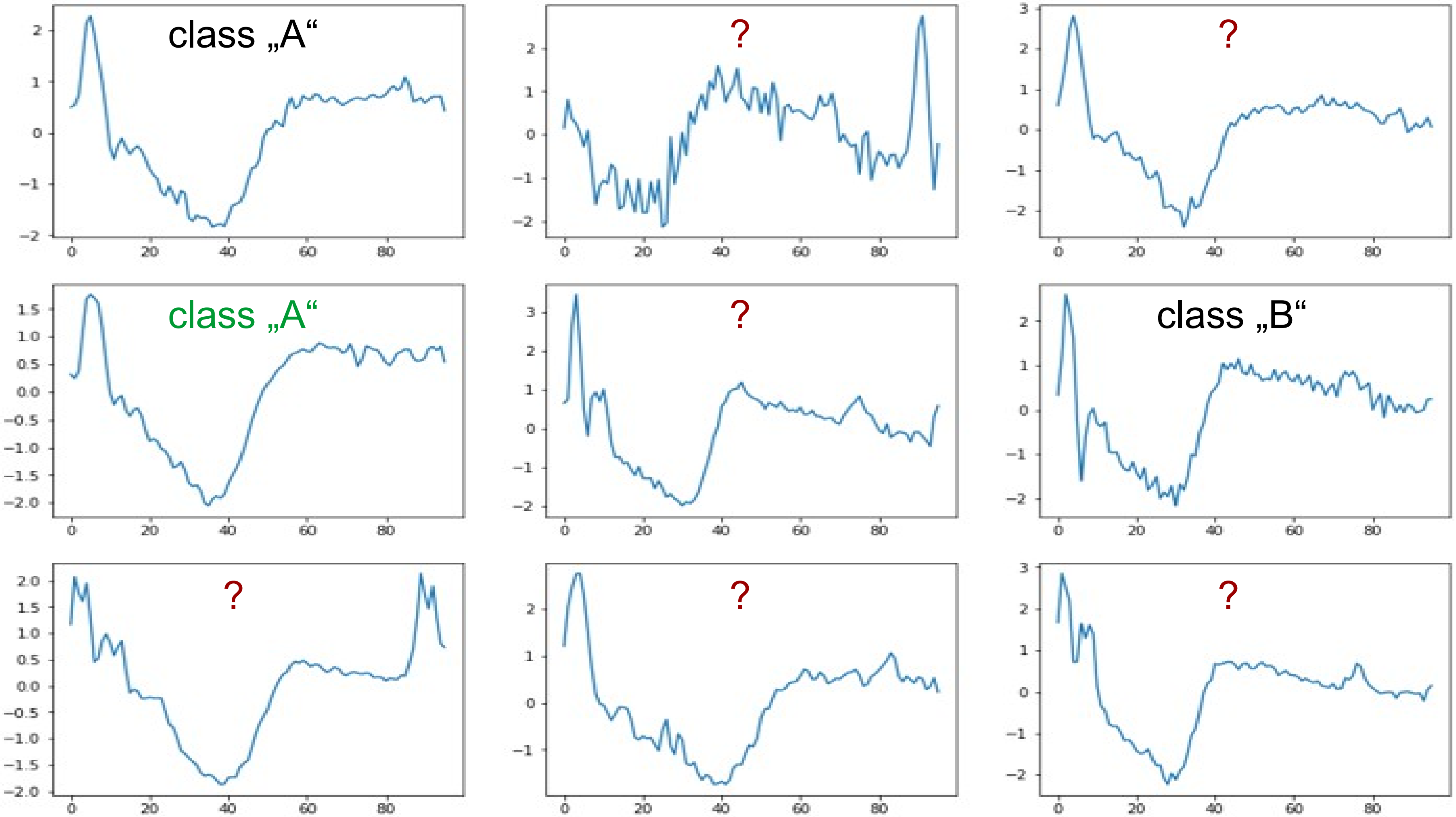
(a) The training set. (b) Decision boundary with supervised training. (c) 1st iteration of self-training. (d) 2nd iteration of self-training. (e) 3rd iteration of self-training. (f) Classification with self-training.

K. Marussy, K. Buza (2013): SUCCESS: A New Approach for Semi-Supervised Classification of Time-Series, ICAISC, LNCS Vol. 7894, pp. 437-447, Springer.

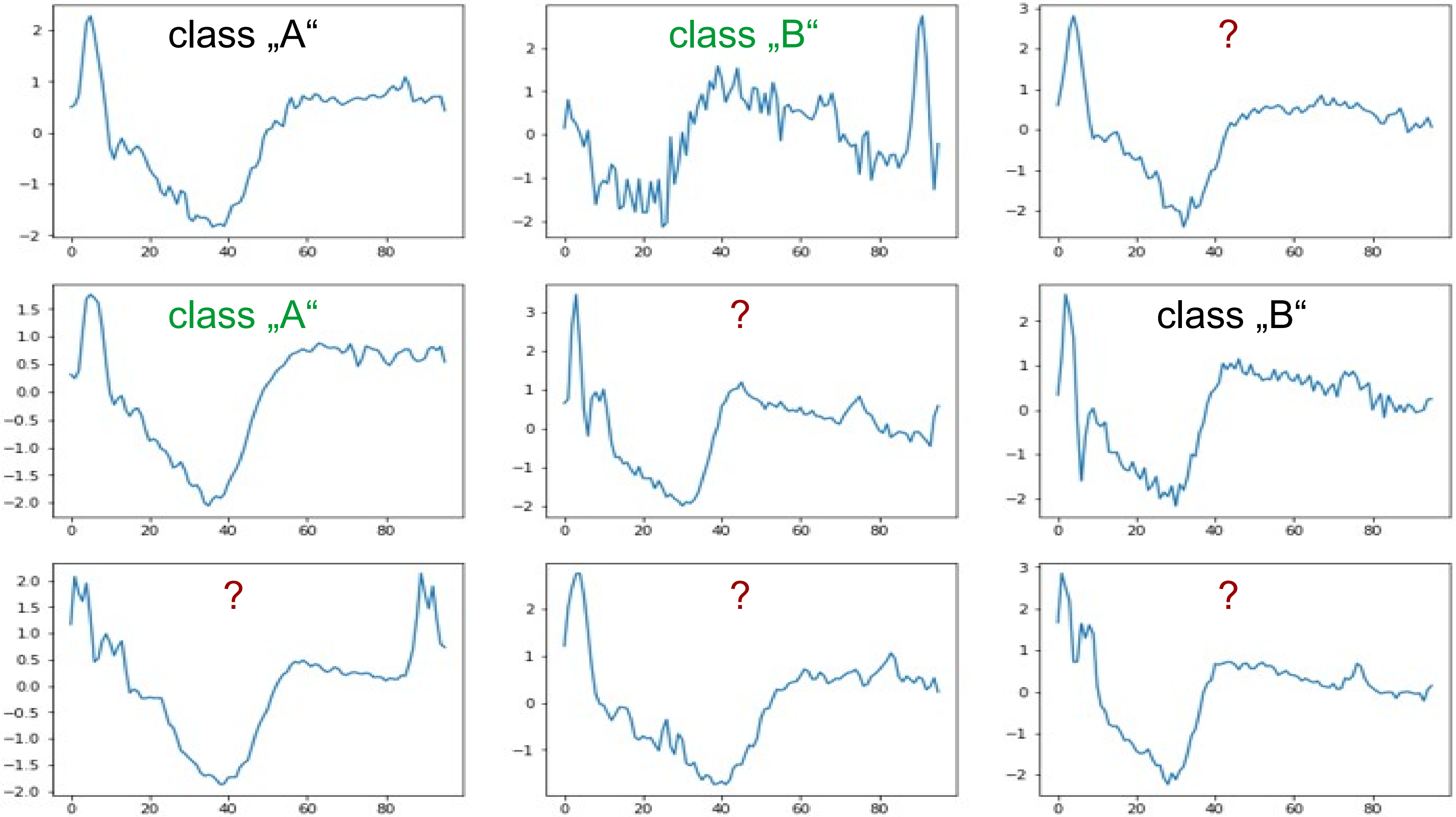
Semi-Supervised Classification of Time Series



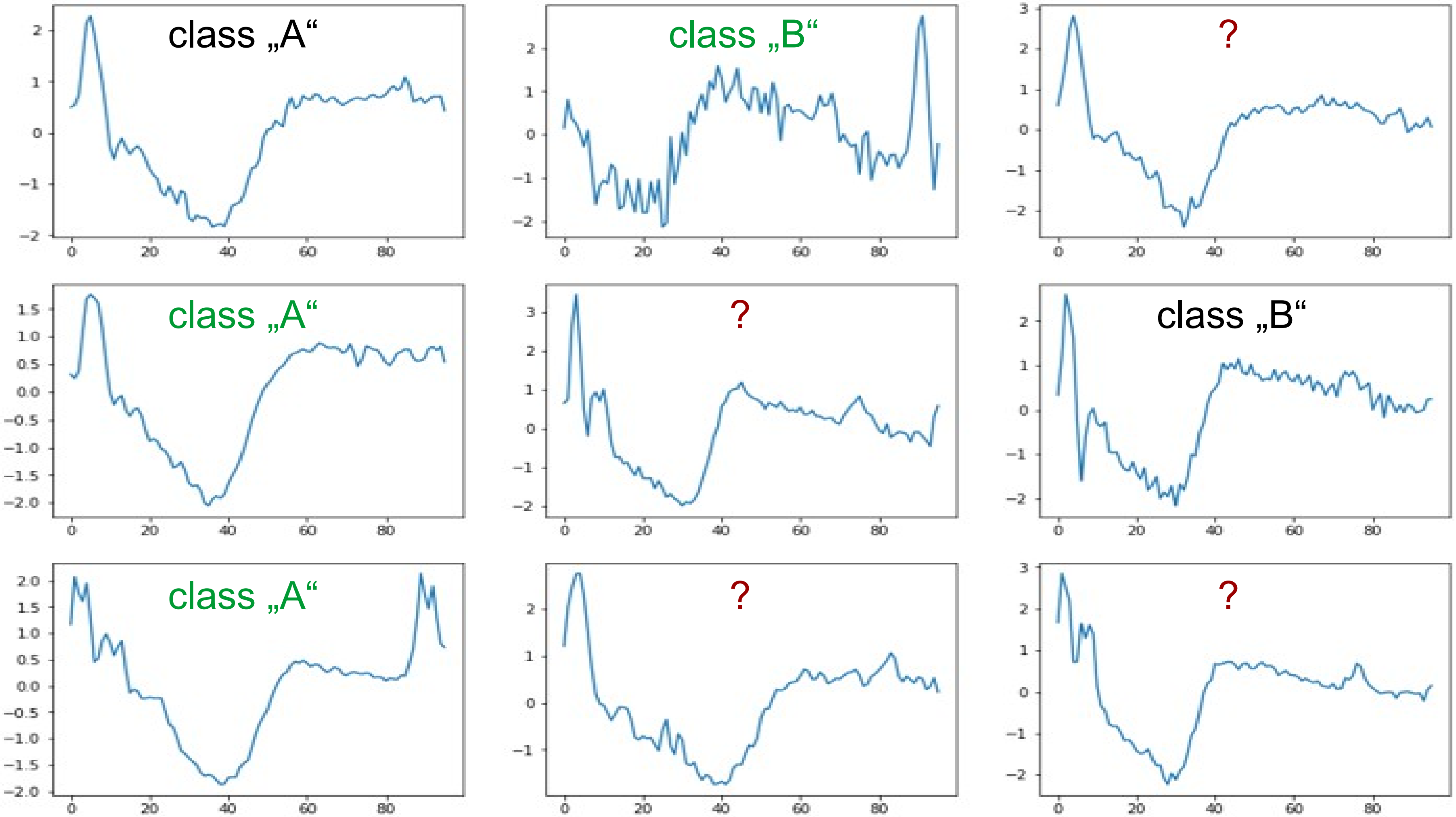
Semi-Supervised Classification of Time Series



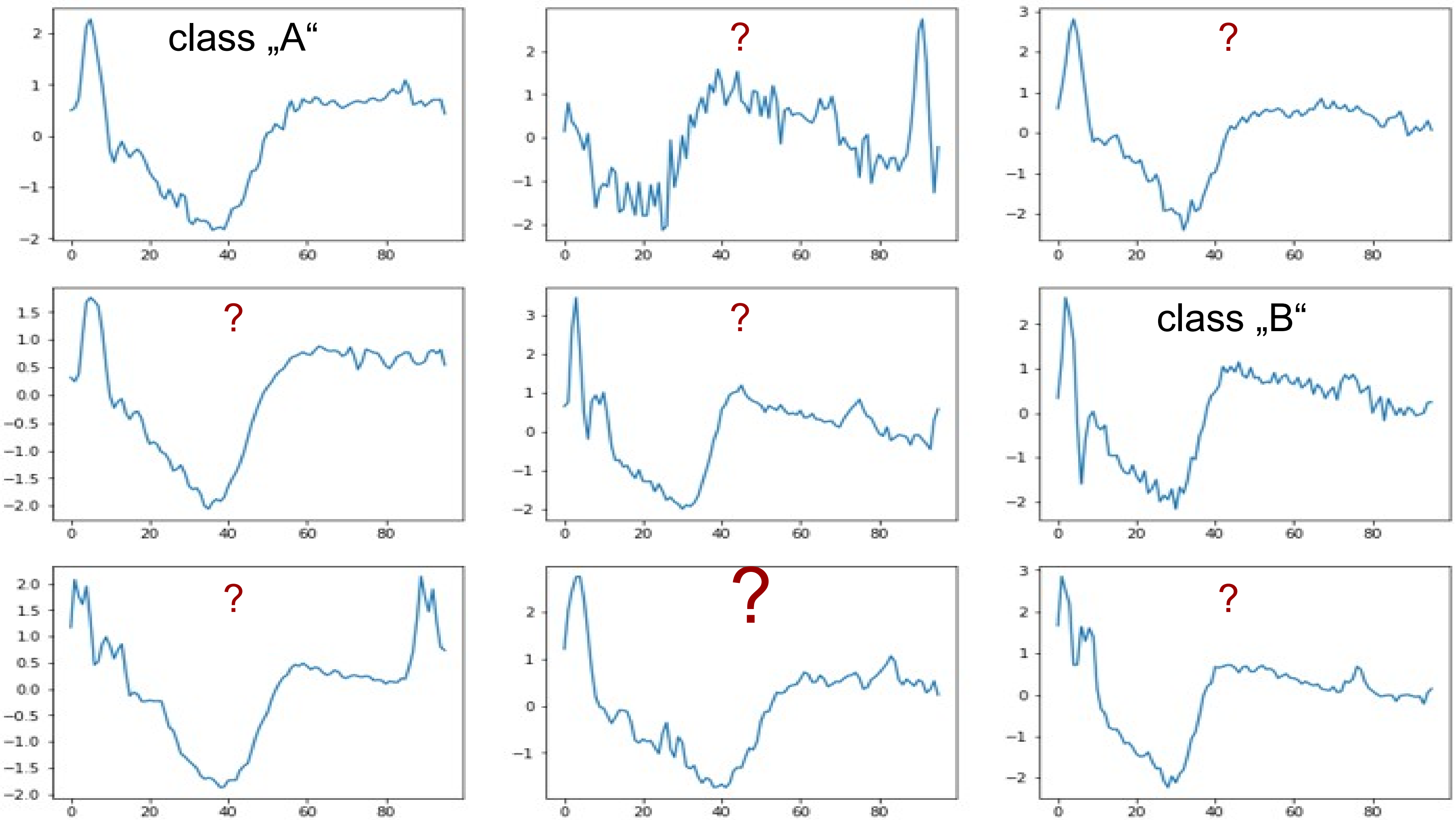
Semi-Supervised Classification of Time Series



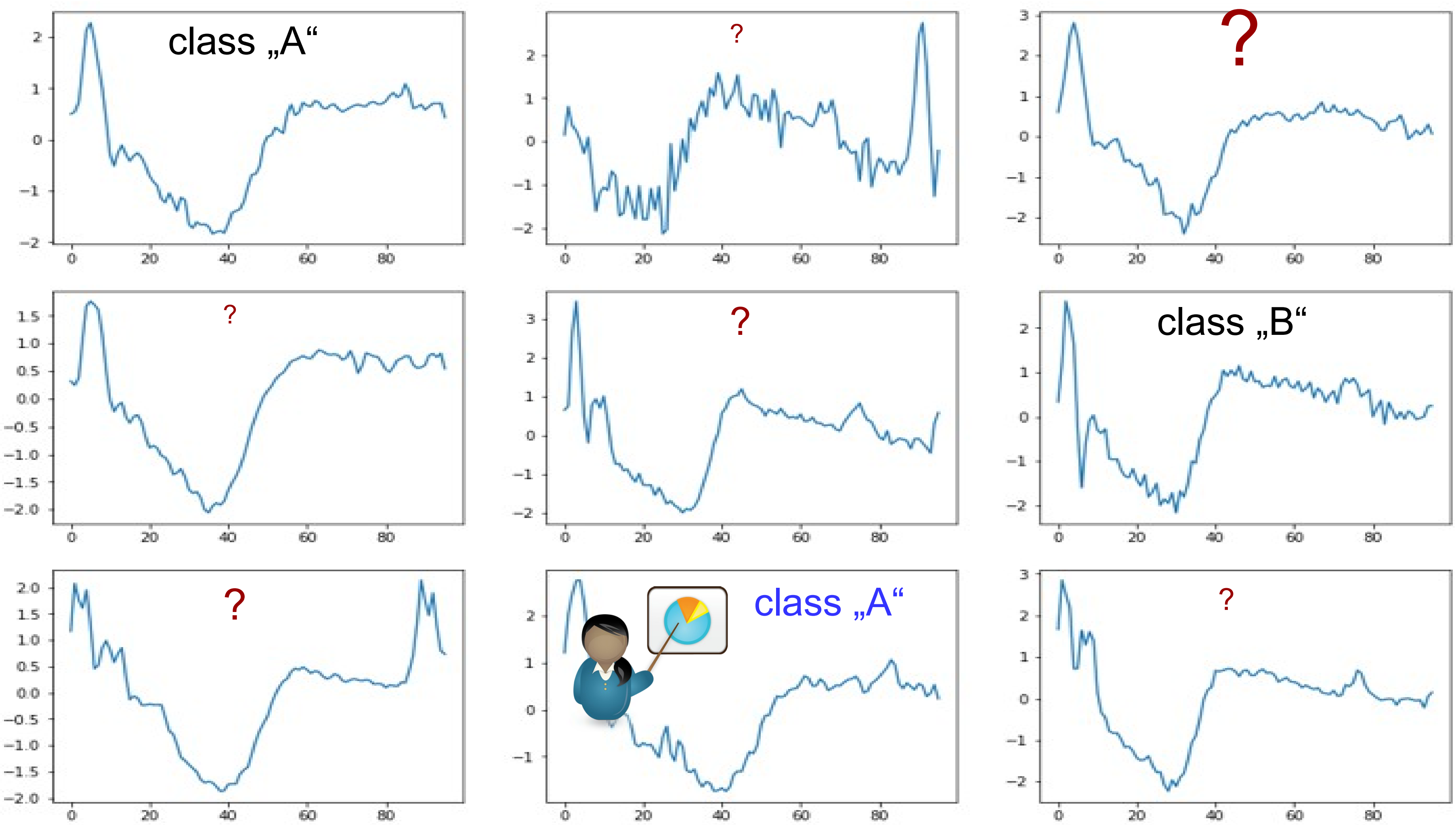
Semi-Supervised Classification of Time Series



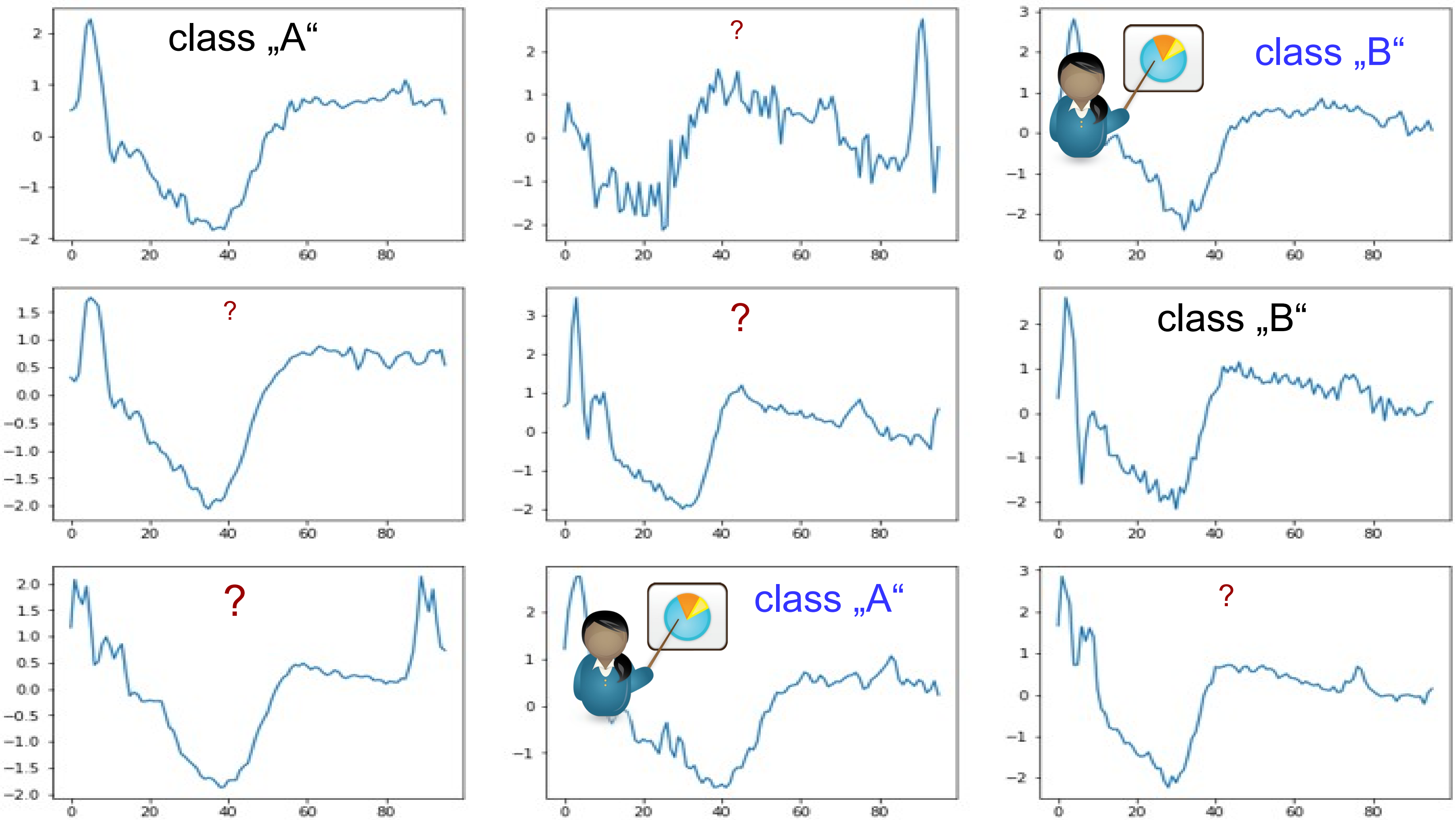
Active Learning for Time Series Classification



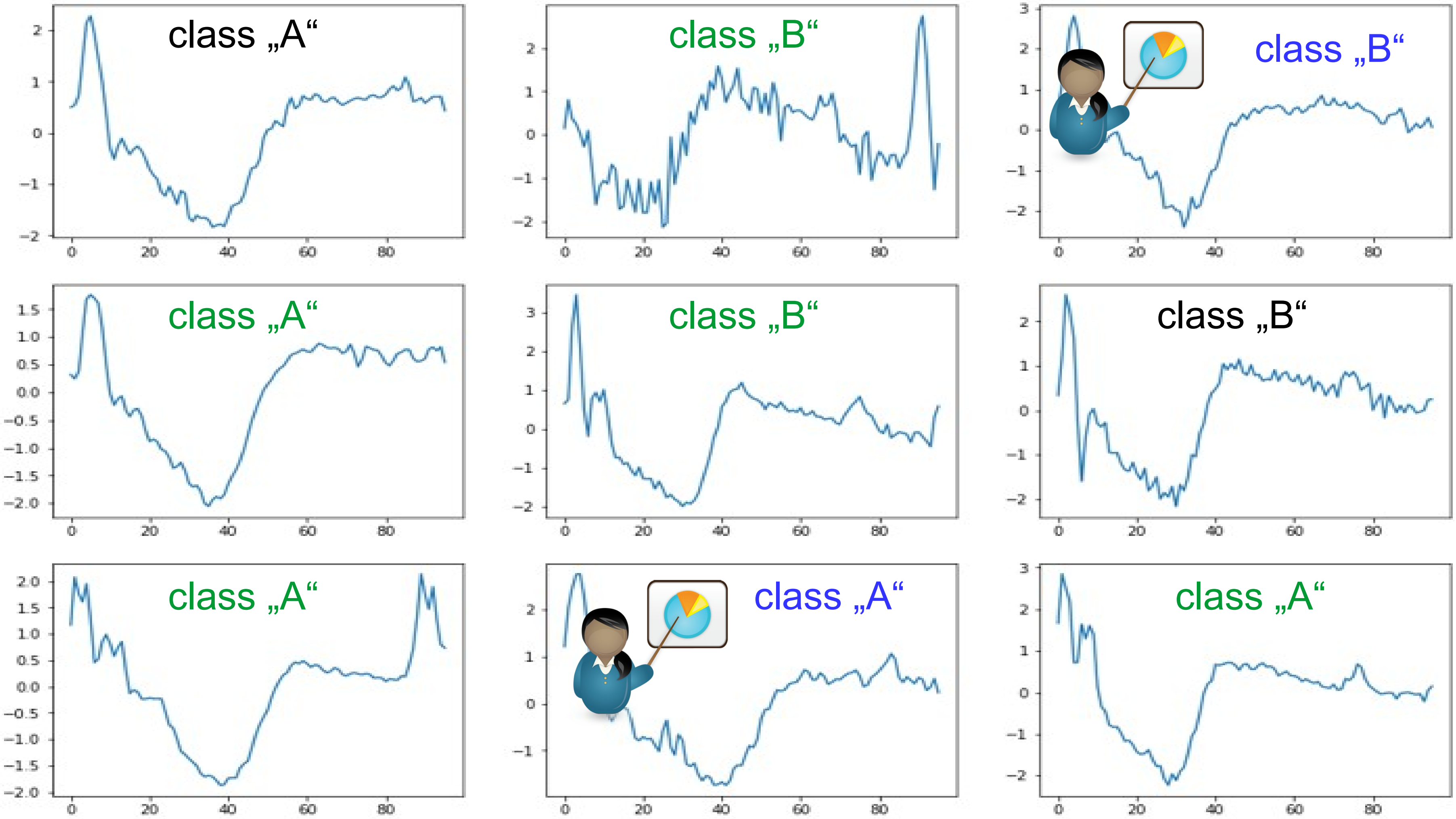
Active Learning for Time Series Classification



Active Learning for Time Series Classification

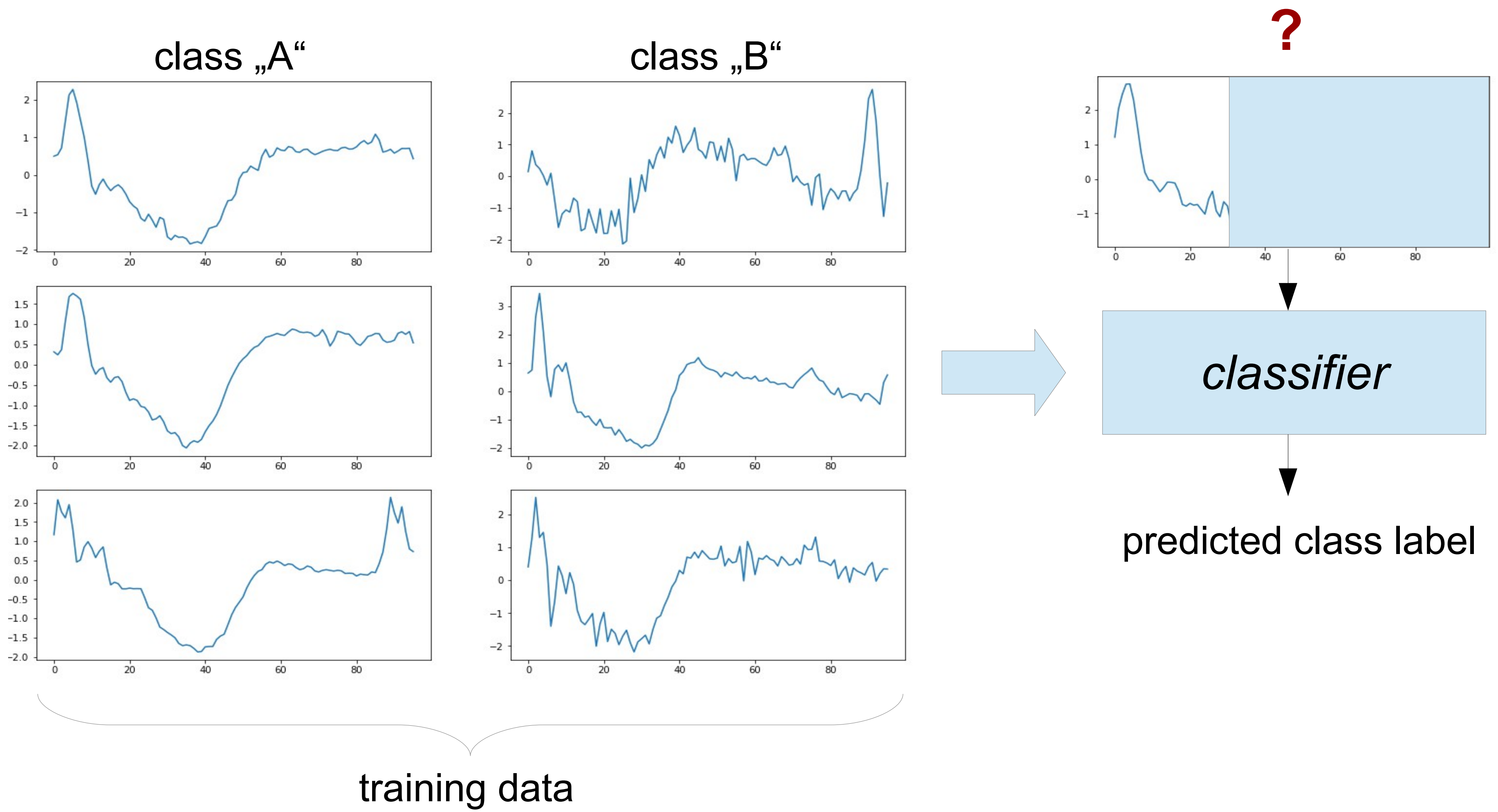


Active Learning for Time Series Classification



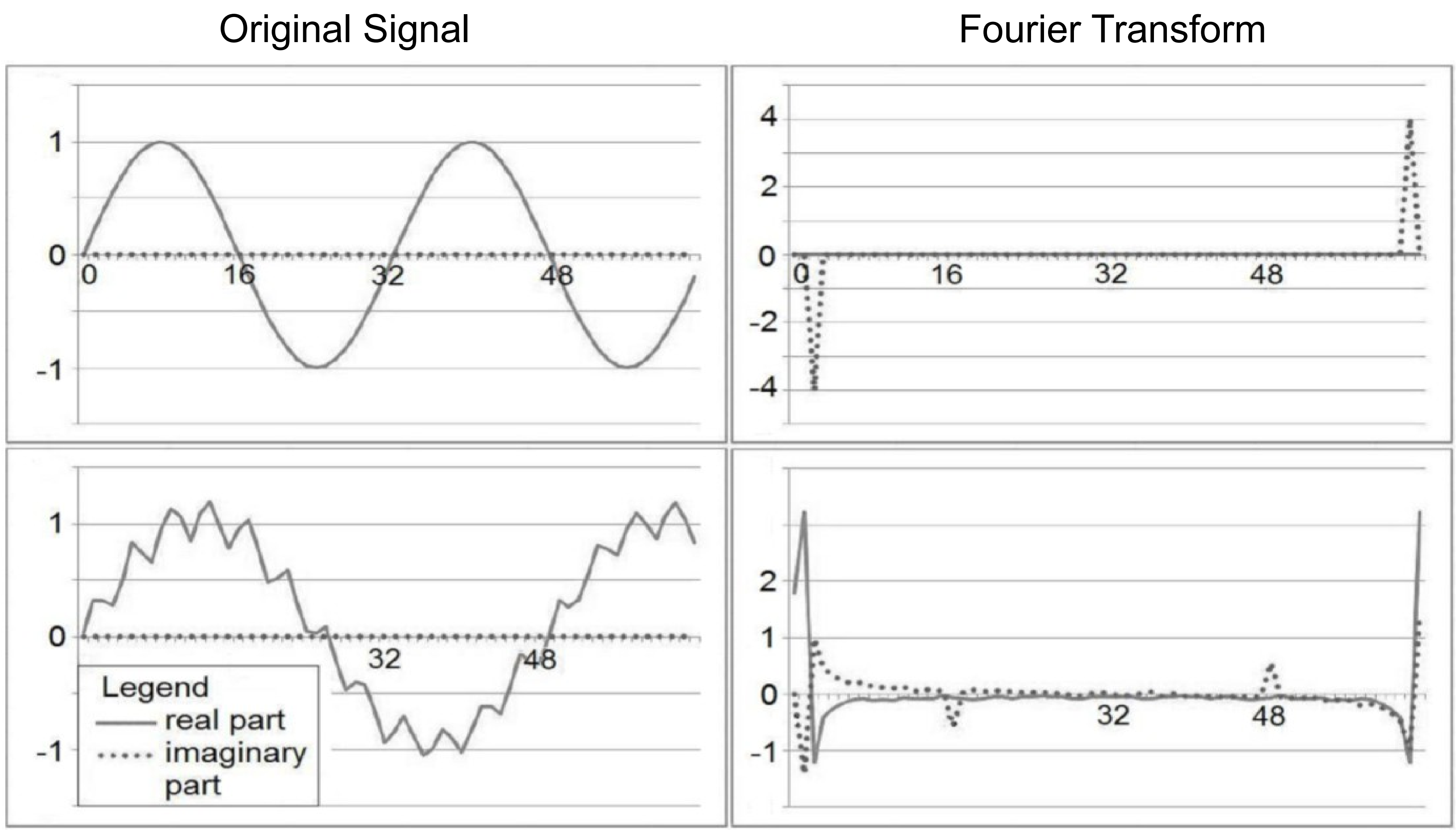
Early Classification of Time Series

- Can we build a model that recognizes the class before the entire time series is observed?
- Trade-off between accuracy and earliness of classification



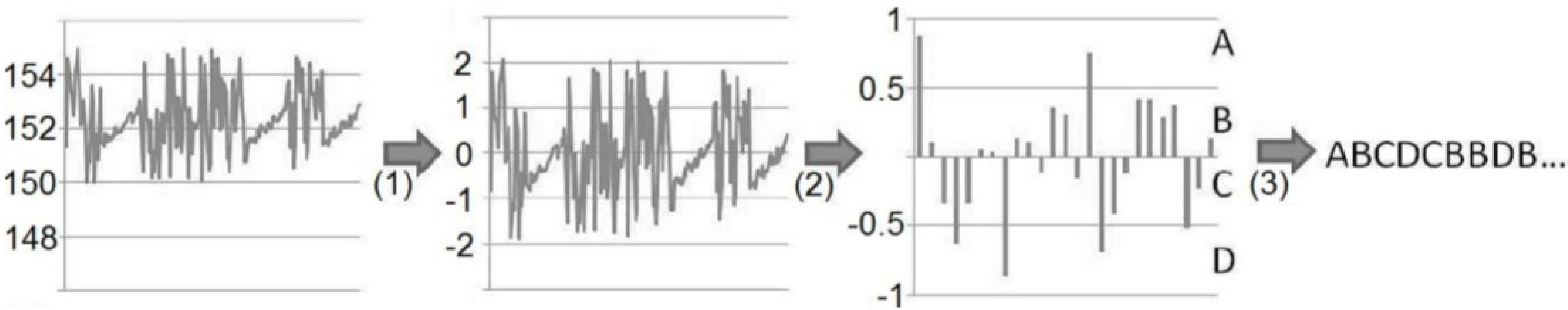
(Some of the) Preprocessing Techniques

Transformation into Frequency Domain



SAX: Symbolic Aggregate Approximation

- Normalisation (1)
- PAA: Piecewise Aggregate Approximation (2)
- Mapping to discrete symbols (3)



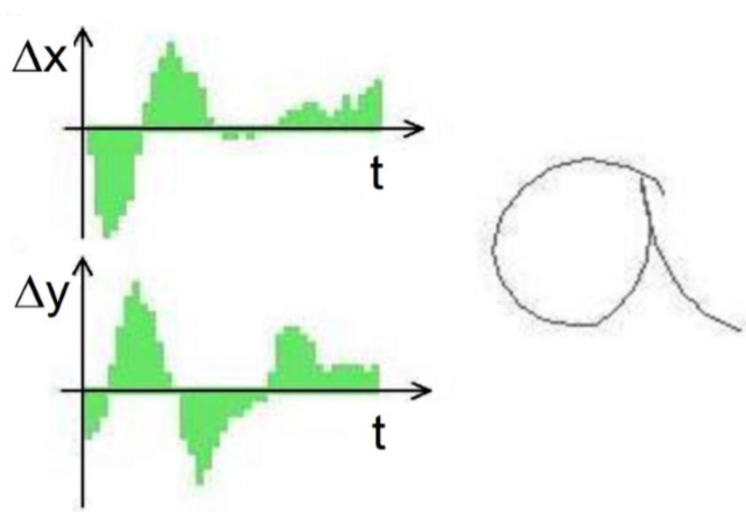
raw time series normalized time series PAA SAX-representation

Lin, Jessica, et al (2003): A symbolic representation of time series, with implications for streaming algorithms, Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery.

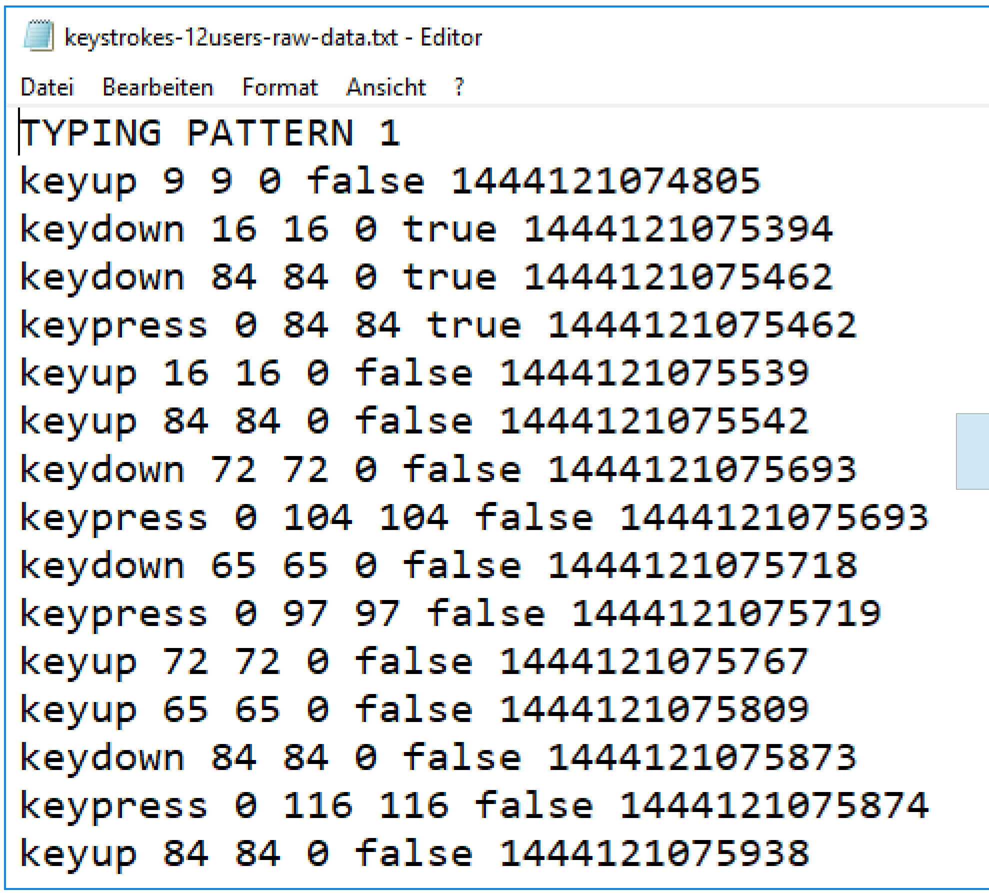
Change Instead of Absolute Values

$$T = \big((x_1, y_1), \ldots, (x_n, y_n)\big)$$

$$T' = \big((x_2 - x_1, y_2 - y_1), \ldots, (x_n - x_{n-1}, y_n - y_{n-1})\big)$$



Domain-specific Preprocessing – Example

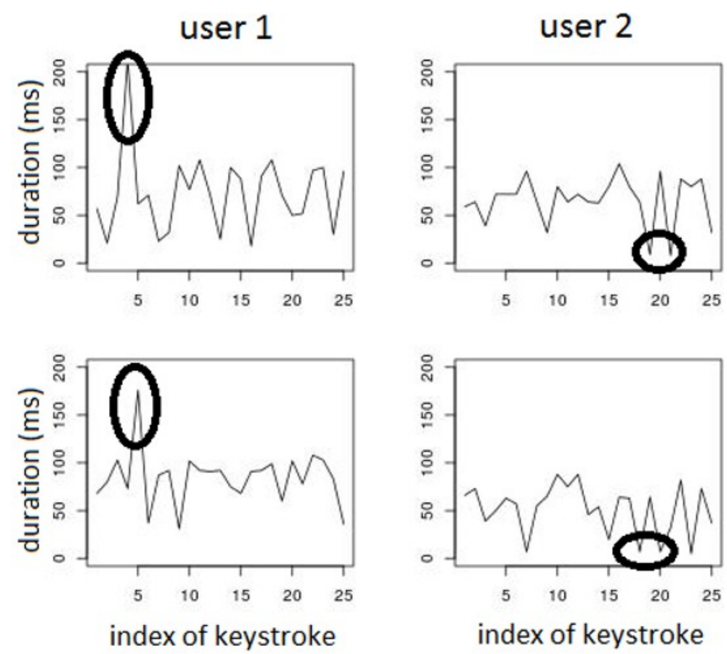


keystrokes-12users-raw-data.txt - Editor

Datei Bearbeiten Format Ansicht ?

```
TYPING PATTERN 1
keyup 9 9 0 false 1444121074805
keydown 16 16 0 true 1444121075394
keydown 84 84 0 true 1444121075462
keypress 0 84 84 true 1444121075462
keyup 16 16 0 false 1444121075539
keyup 84 84 0 false 1444121075542
keydown 72 72 0 false 1444121075693
keypress 0 104 104 false 1444121075693
keydown 65 65 0 false 1444121075718
keypress 0 97 97 false 1444121075719
keyup 72 72 0 false 1444121075767
keyup 65 65 0 false 1444121075809
keydown 84 84 0 false 1444121075873
keypress 0 116 116 false 1444121075874
keyup 84 84 0 false 1444121075938
```

raw data (keystroke dynamics)



time series

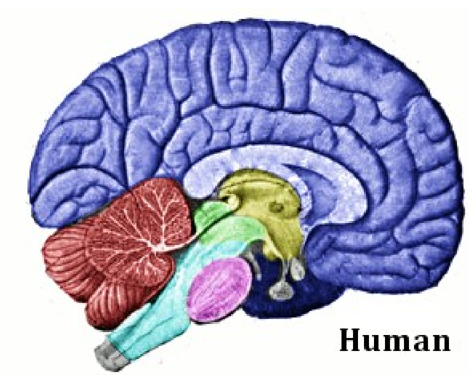
Time Series Classification Techniques

Time Series Classification Techniques – Overview

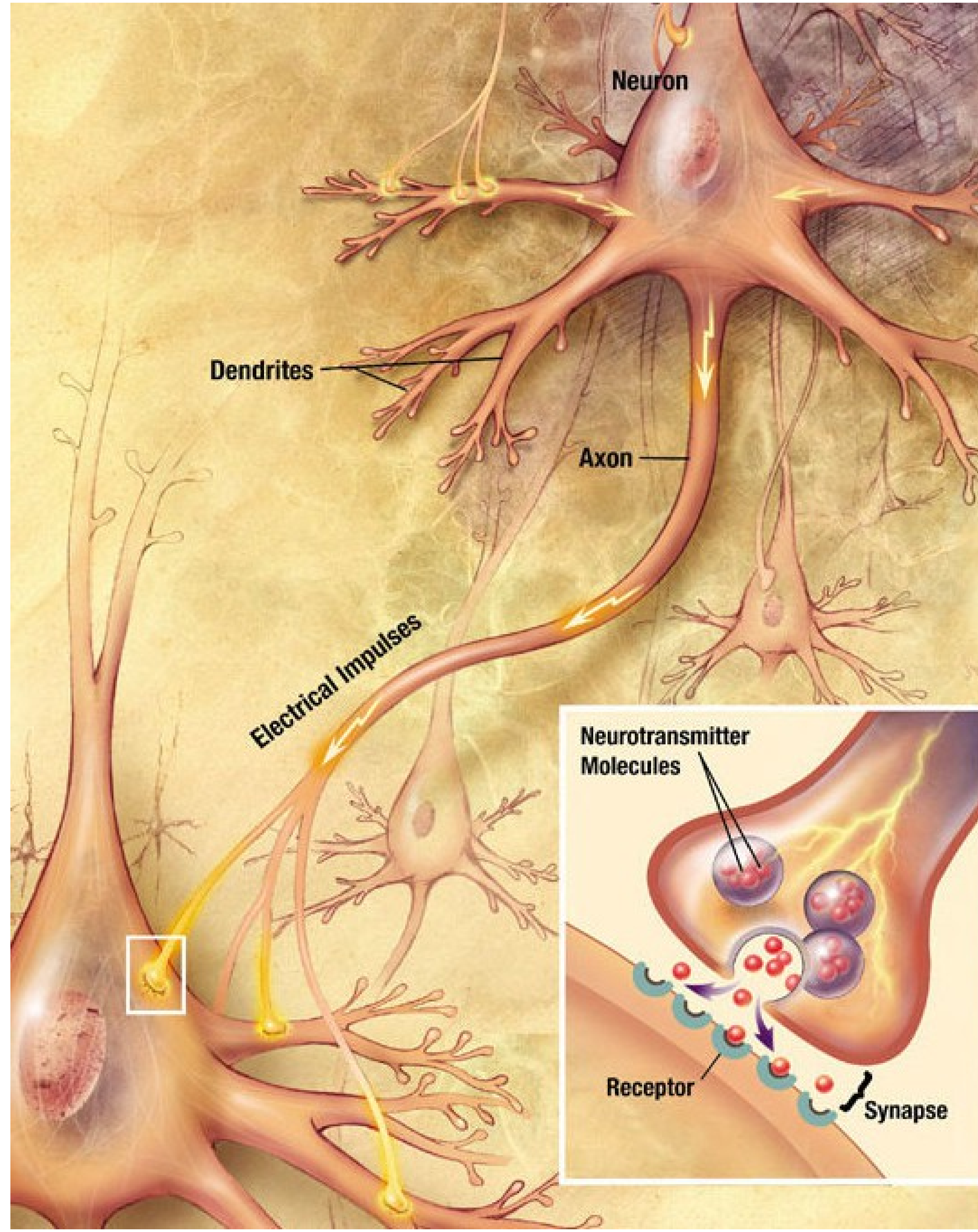
- Feature-based classification
  - feature extraction + a standard classifier
    (such as SVM, Naive Bayes, decision tree...)
  - Possilbe features:
    - min, max, avg, std, number of local optima, number of sign changes,...
    - distances from other time series
- Classification based on characteristic local patterns
  (motif-based, shapelet-based, convolutional neural networks)
- Similarity-based classification
  (nearest neighbor and its extensions, such as hubness-aware classifiers)
- Hidden Markov Models
- Deep Learning
  - Convolutional neural networks

(Deep) Neural Networks for Time Series Classification

Neural Networks

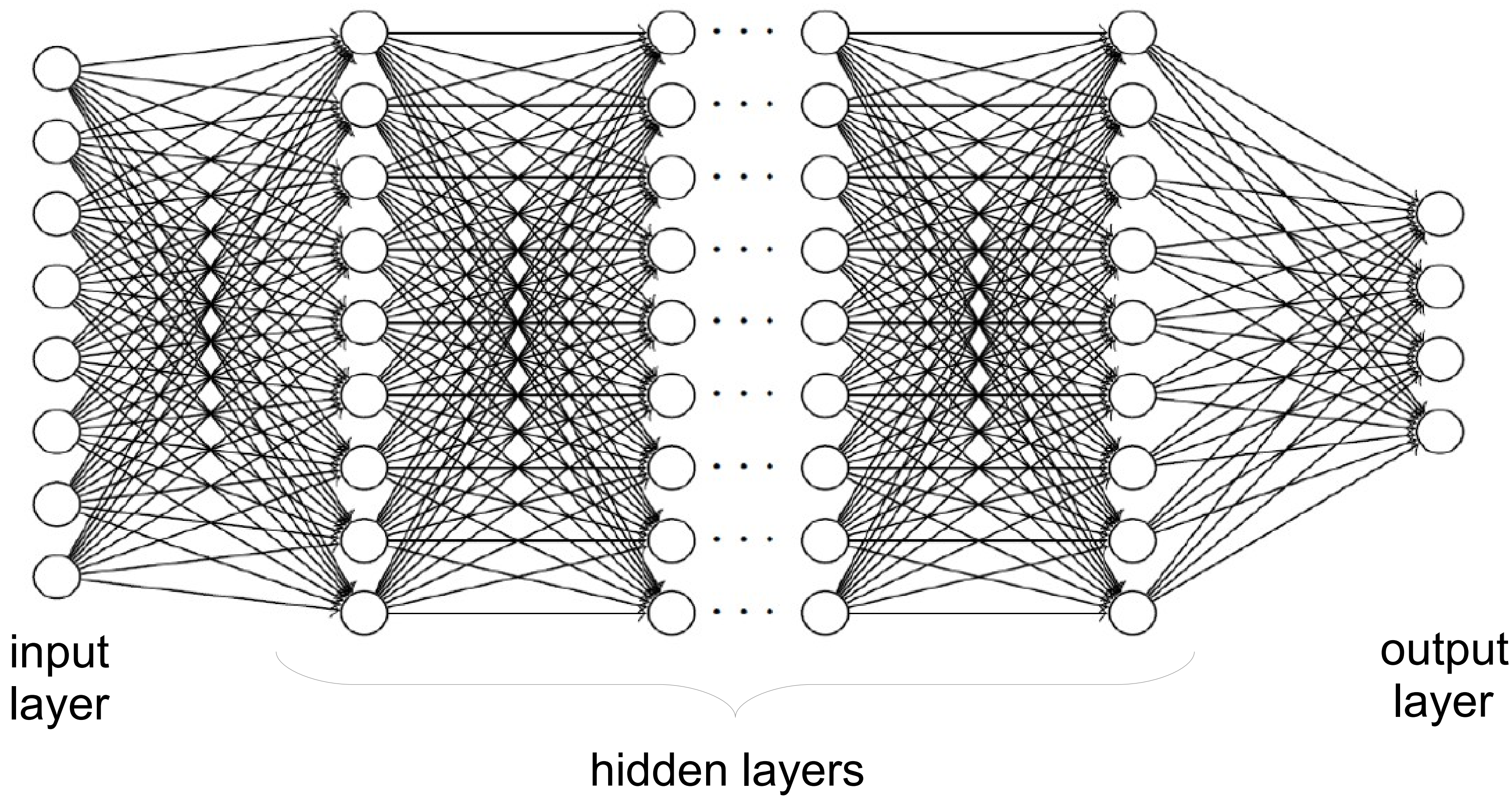


By Vertebrate-brain-regions.png: Looie496derivative work: Looie496 (Vertebrate-brain-regions.png) [Public domain], via Wikimedia Commons

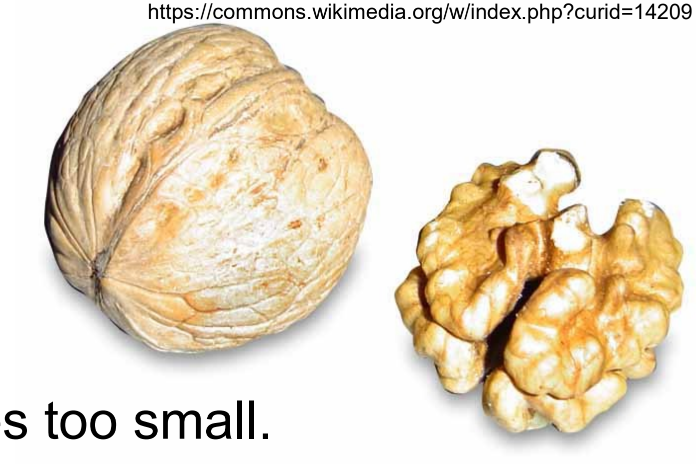


By user:Looie496 created file, US National Institutes of Health, National Institute on Aging created original [Public domain], via Wikimedia Commons

Deep Feed-Forward Neural Networks



Deep Learning in a Nutshell



- What was wrong with backpropagation in 1986?
  (Geoff Hinton, „Deep Learning“, May 22, 2015)
  - Our labeled datasets were thousands of times too small.
  - Our computers were millions of times too slow.
  - We initialized the weights in a stupid way.
  - We used the wrong type of non-linearity.
- From “conventional” neural networks to deep learning
  - Size and structure of the network: few layers → many layers
  - Activation function: sigmoid → rectified linear unit (ReLU)
  - Loss function: quadratic loss → cross-entropy
  - Initialization of weights: random → (unsupervised) pre-training
  - Size of training data, much more memory, distributed computation, GPUs…
  - New regularization techniques:
    “sparsity-enforcing” regularisation terms, drop-out, early stop

Convolution*

Input of the convolution (time series):

-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution*

Input of the convolution (time series):

-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6

Filter (i.e., a set of weights)

1	0	-1	0	1

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution*

Input of the convolution (time series):

-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6

Filter (i.e., a set of weights)

1	0	-1	0	1

Output („convolved“ time series)

0

$$(-0.8) \times 1 \; + \; (-0.5) \times 0 \; + \; (-0.2) \times (-1) \; + \; 0.2 \times 0 \; + \; 0.6 \times 1 = 0$$

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution*

Input of the convolution (time series):

-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6

Filter (i.e., a set of weights)

1	0	-1	0	1

Output („convolved“ time series)

0	0.1

$$(-0.5) \times 1 \; + \; (-0.2) \times 0 \; + \; 0.2 \times (-1) \; + \; 0.6 \times 0 \; + \; 0.8 \times 1 = -0.1$$

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution*

Input of the convolution (time series):

-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6

Filter (i.e., a set of weights)

1	0	-1	0	1

Output („convolved“ time series)

0	0.1	0.1	0.4	0.6	0.5	0.2	0	-0.2	0.8	1.6	0.5

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution*

Input of the convolution (time series):

0	0	-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6	0	0

Filter (i.e., a set of weights)

1	0	-1	0	1

Output („convolved“ time series)

0.6	0.7	0	0.1	0.1	0.4	0.6	0.5	0.2	0	-0.2	0.8	1.6	0.5	-1.4	-0.8

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution*

Input of the convolution (time series):

0	0	-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6	0	0
0	0	0.9	0.3	0.1	-0.2	0.5	0.3	0.1	0	0.2	-0.1	-0.2	0.4	0.5	0.5	0.6	0.3	0	0

Filter (i.e., a set of weights)

1	0	-1	0	1
0.5	0.3	0.1	-0.2	-0.3

Output („convolved“ time series)

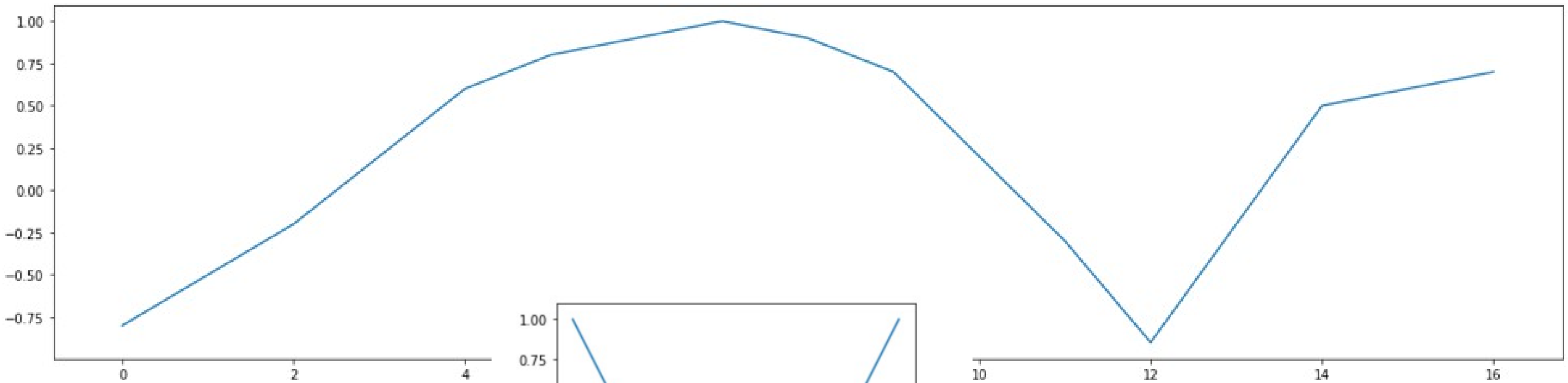
0.6	1.0	0.4	0.1	0.1	0.5	0.9	0.7	0.4	0	-0.4	0.5	1.4	0.7	-1.0	-0.3

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution*

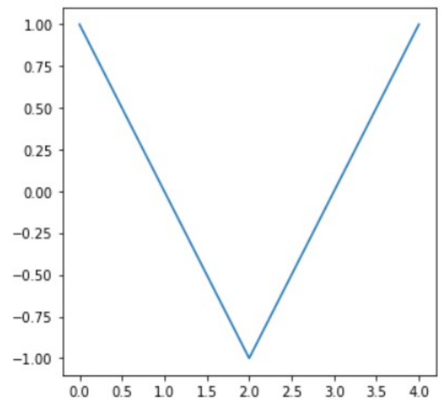
In

0	0	0	0



Filter (i.e., a set of weigh

1	0	-1	0	1



Output („convolved“ time

0.6	0.7	0	0.1	0.1	0.4	0.6	0.5	0.2	0	-0.2	0.8	1.6	0.5	-1.4	-0.8

* Remark: while being conceptually the same as traditional convolution in mathematics, the convolution used in neural networks is slightly different in terms of technical details.

Convolution and Max Pooling*

Input of the convolution (time series):

0	0	-0.8	-0.5	-0.2	0.2	0.6	0.8	0.9	1.0	0.9	0.7	0.2	-0.3	-0.9	-0.2	0.5	0.6	0	0

Filter (i.e., a set of weights)

1	0	-1	0	1

Output („convolved“ time series)

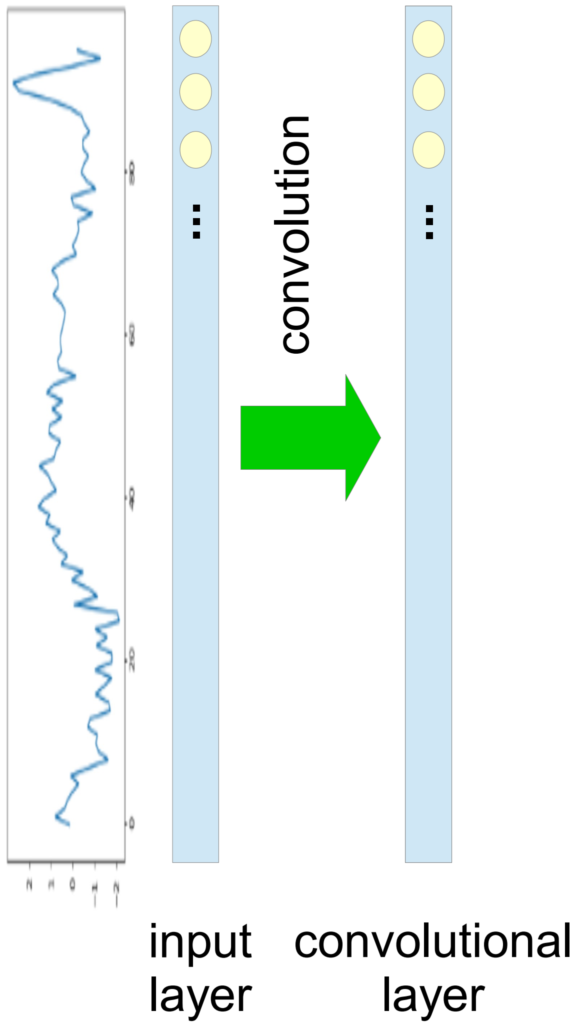
0.6	0.7	0	0.1	0.1	0.4	0.6	0.5	0.2	0	-0.2	0.8	1.6	0.5	-0.7	-0.8

Max pooling

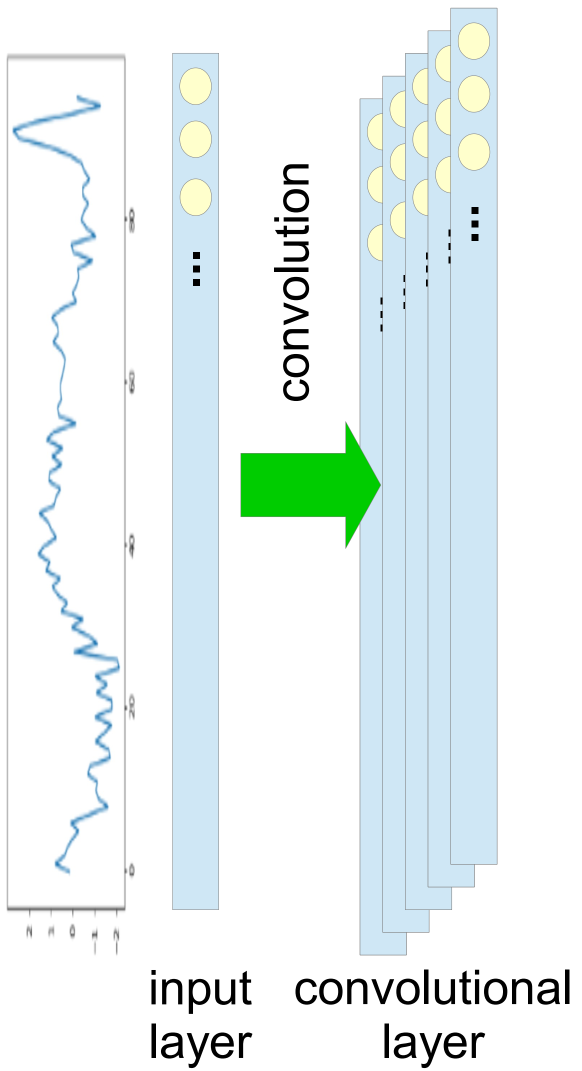
0.7	0.6	0.8	1.6

* Strictly speaking, max pooling has nothing to do with convolution, however, in convolutional neural networks (CNNs), the convolutional layer is often followed by a max pooling layer.

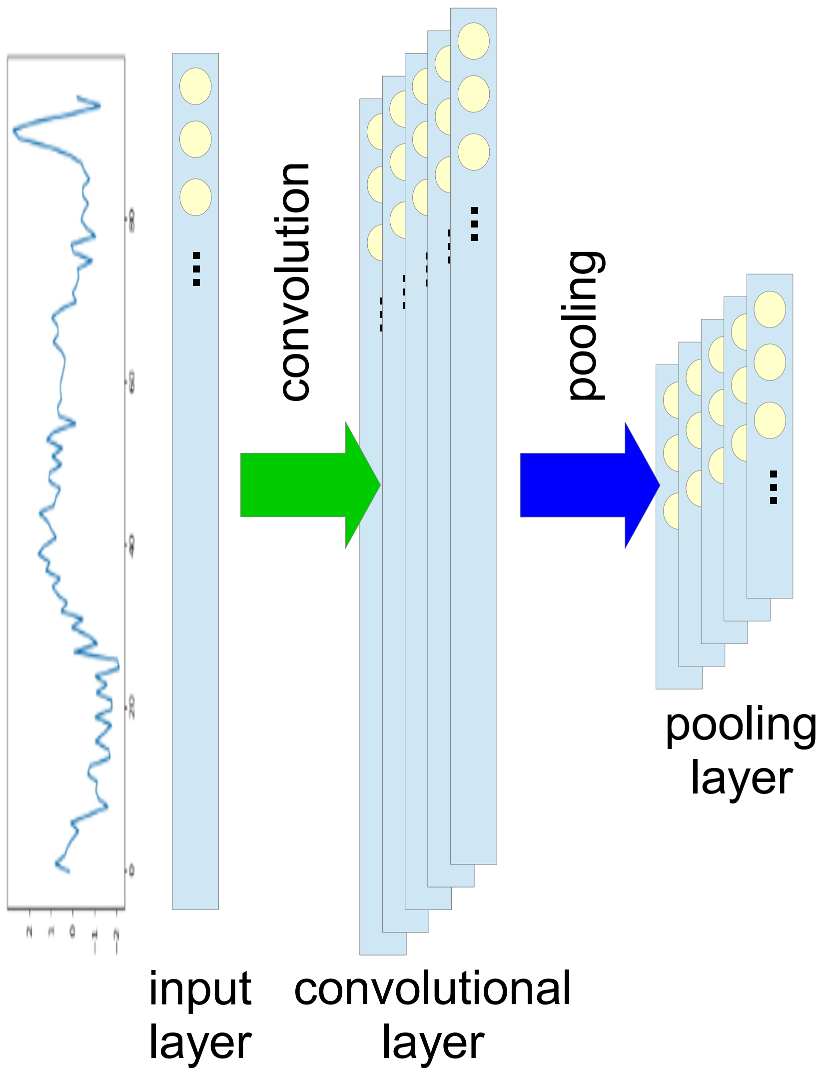
Convolutional Neural Networks



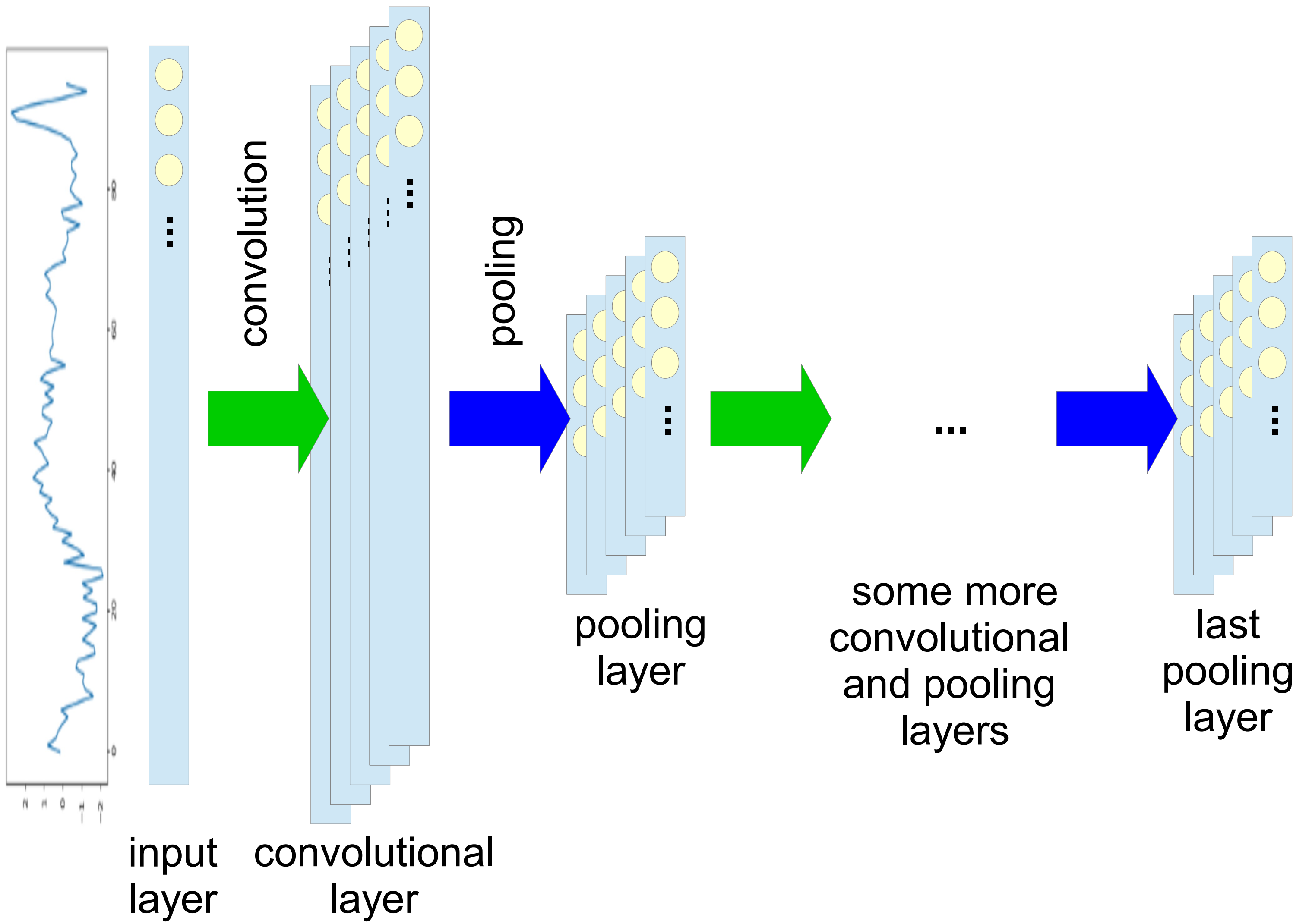
Convolutional Neural Networks



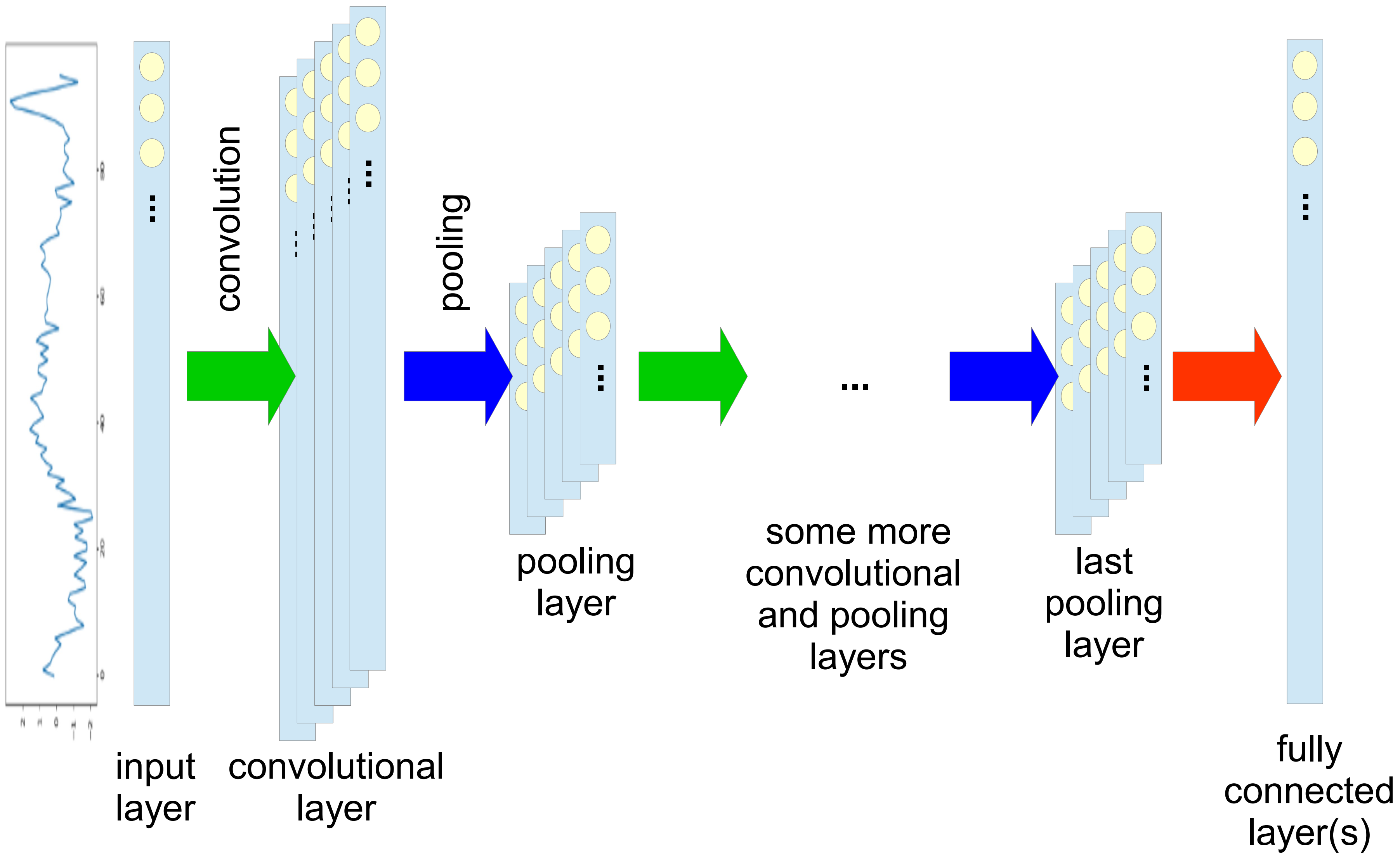
Convolutional Neural Networks



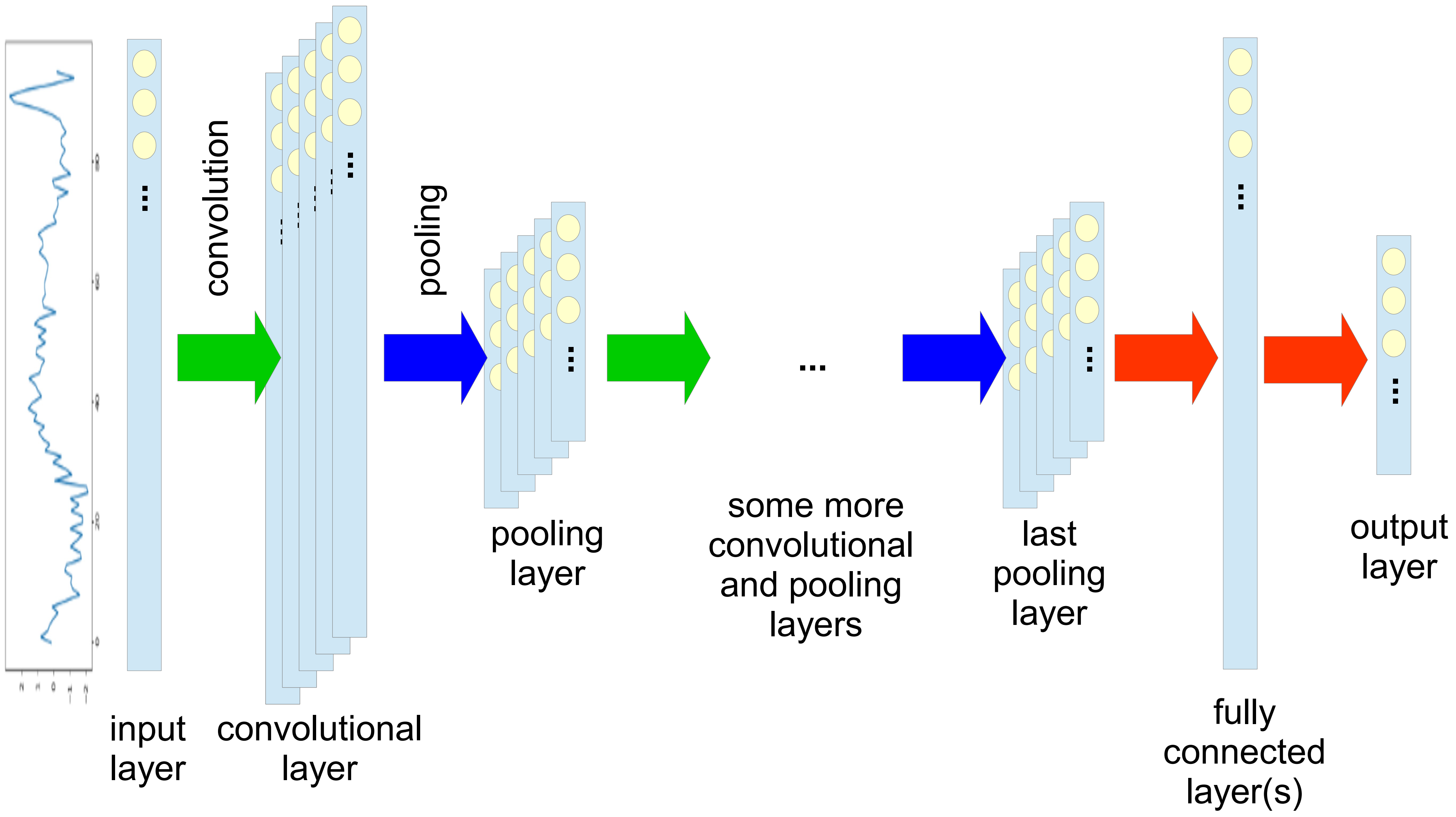
Convolutional Neural Networks



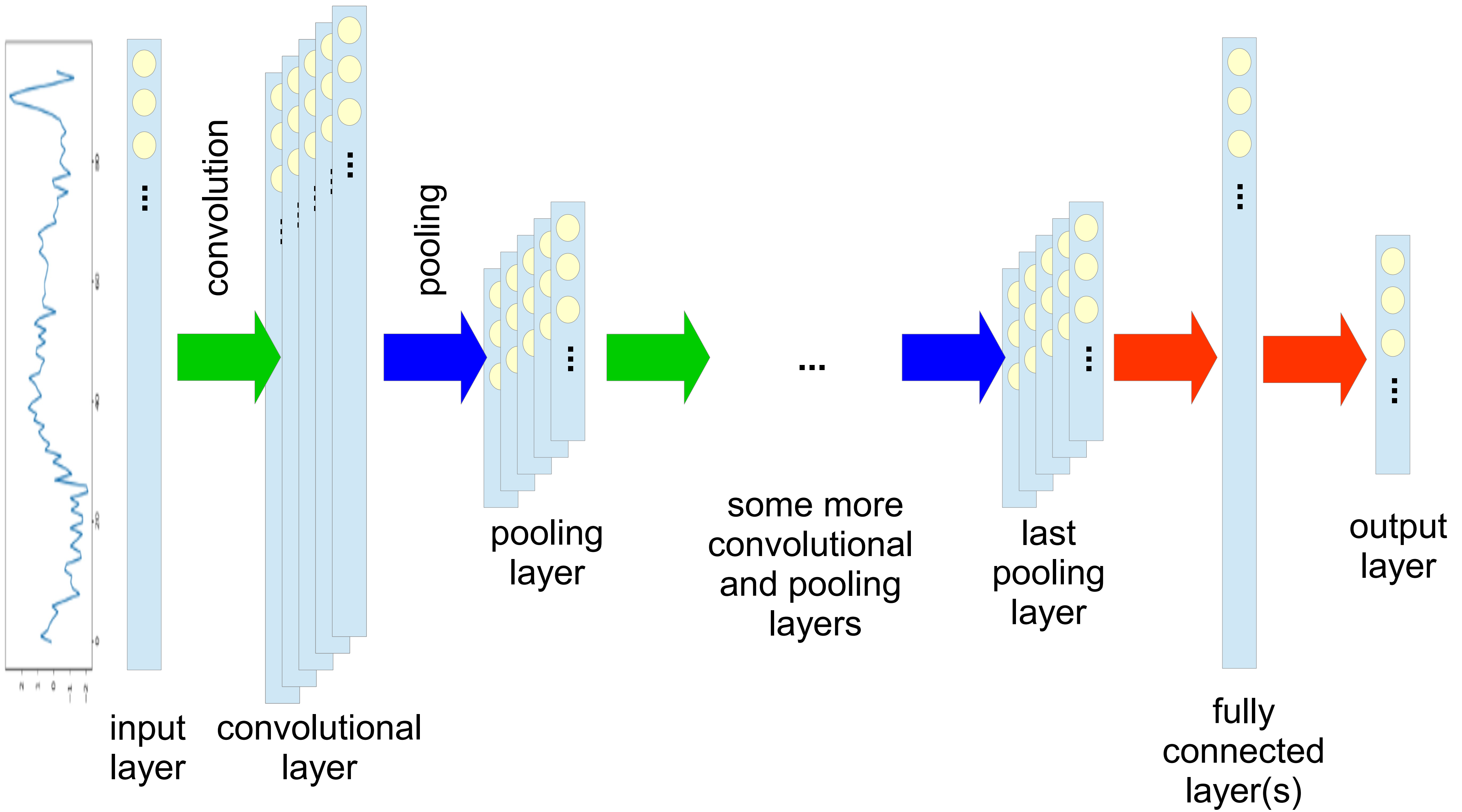
Convolutional Neural Networks



Convolutional Neural Networks



Convolutional Neural Networks



Classification based on Local Patterns

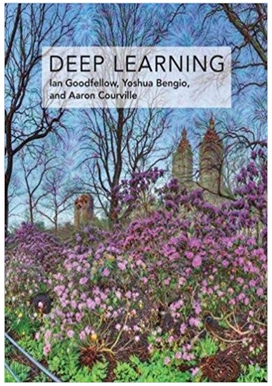
- Motif-based classification

  Buza, Schmidt-Thieme (2009): Motif-based classification of time series with Bayesian networks and SVMs, Advances in Data Analysis, Data Handling and Business Intelligence. Springer, Berlin, Heidelberg, pp. 105-114

- Shapelet-based classification

  Hills et al. (2014): Classification of time series by shapelet transformation, Data Mining and Knowledge Discovery, 28(4), pp. 851-881

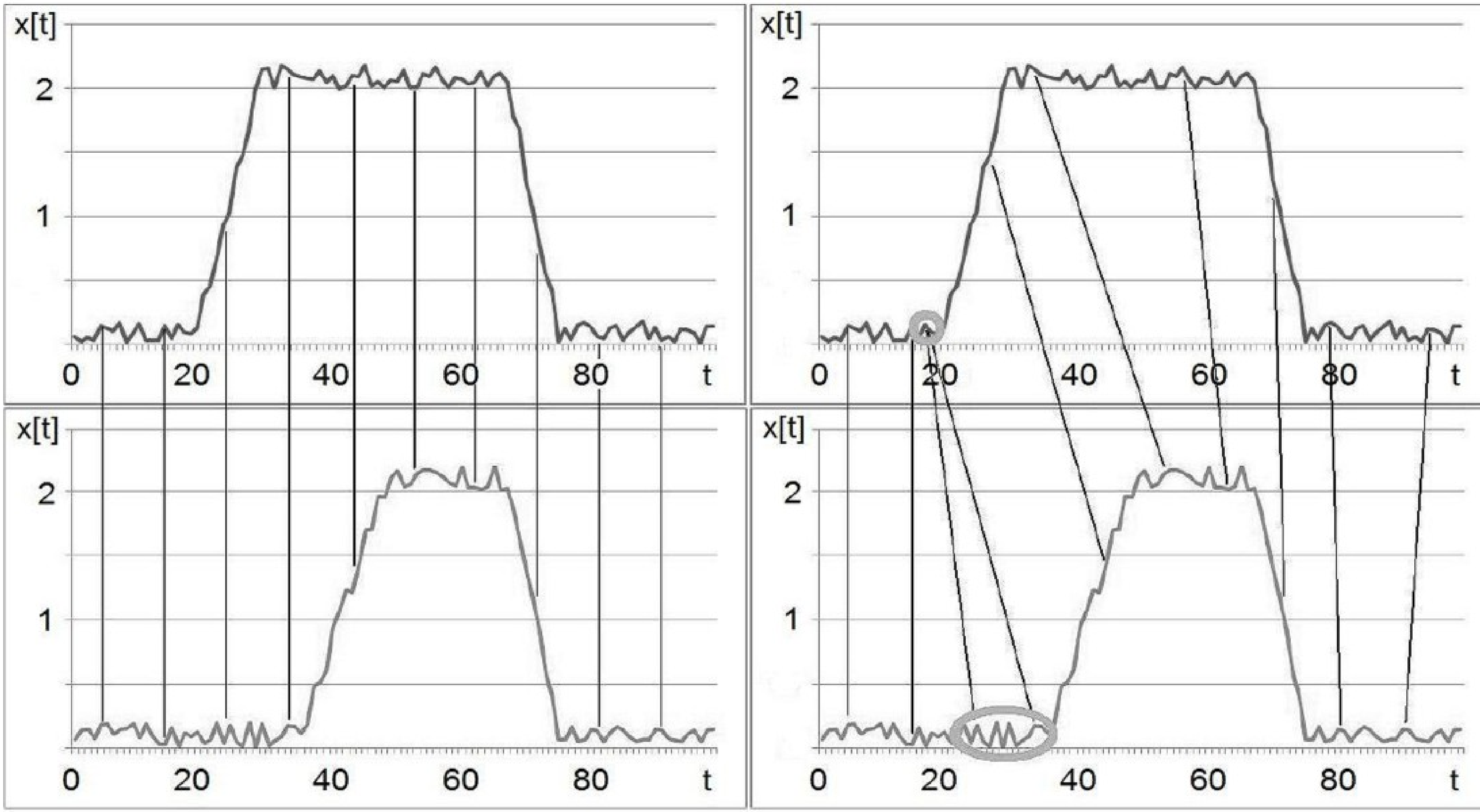
- Convolutional Networks



Ian Goodfellow, Yoshua Bengio, Aaron Courville (2016): Deep Learning, http://www.deeplearningbook.org

Dynamic Time Warping

Comparison of Time Series

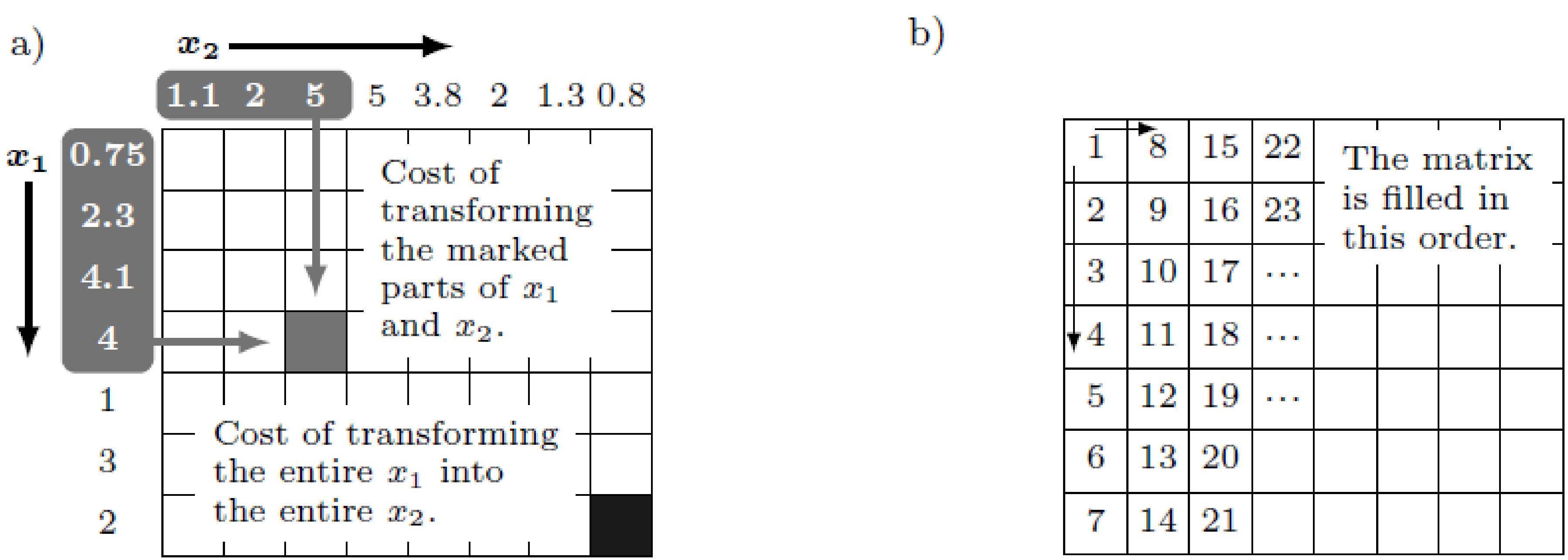


Similarity Measures vs. Distance Measures

- Similarity measure
  - High value → two time series are similar
  - Low value → two time series are different
- Distance measure
  - High value → two time series are different (dissimilar)
  - Low value → two time series are similar
- Dynamic Time Warping (DTW, next slides) is a <u>distance measure</u>

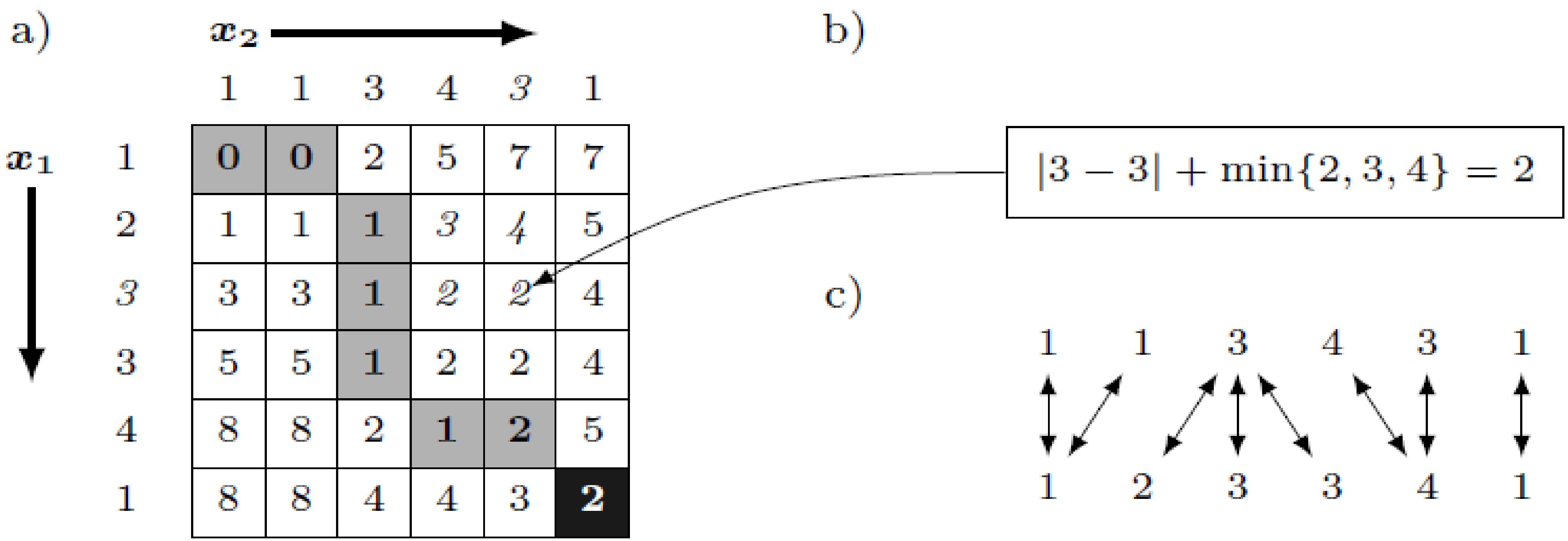
Dynamic Time Warping

Levenshtein distance (text mining),
Smith-Waterman distance (bioinformatics)



Sakoe, Chiba (1978): Dynamic programming algorithm optimization for spoken word recognition, IEEE transactions on acoustics, speech, and signal processing, 26(1), pp. 43-49.

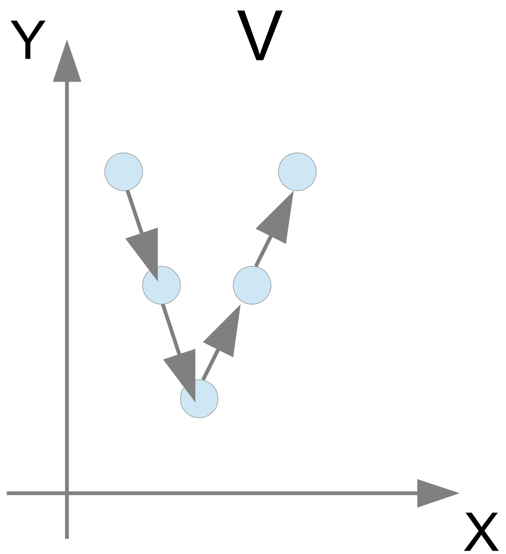
Dynamic Time Warping

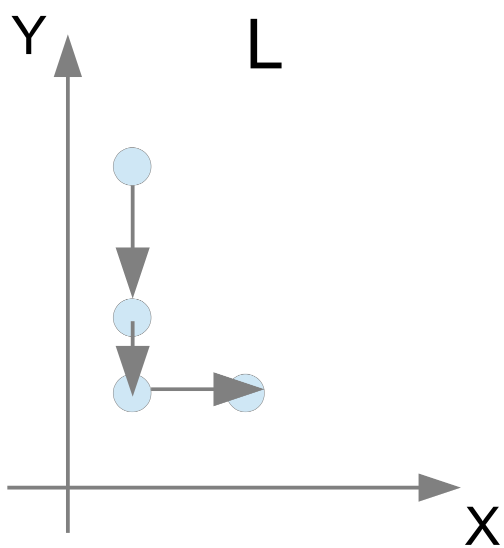


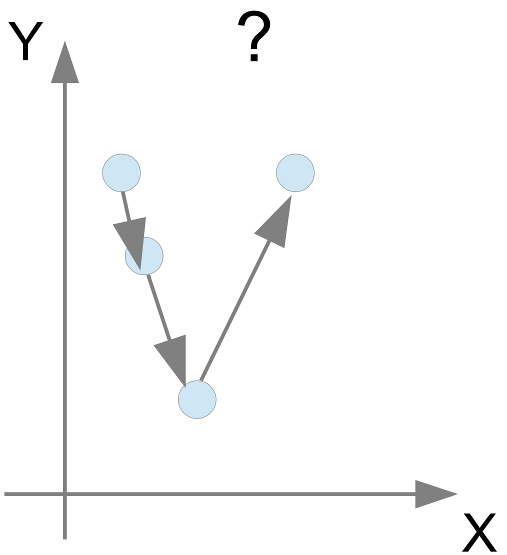
Notes:

- DTW has many variants:
  - additional elongation cost, various internal distances, etc.
- DTW is not a metric (does not fulfil metric axioms).

Multivariate Time Series:
Recognition of Handwriting on a Touchscreen







Time series (deltaX, deltaY):

(1,-2), (1, -2), (1, 2), (1, 2)   (0,-3), (0, -1), (3, 0)   (0.5,-1), (1.5, -3), (2, 4)

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	(1.5, -3),	(2, 4)
(1, -2)	**1.118**		
(1, -2)			
(1, 2)			
(1, 2)			

$$\sqrt{(1 - 0.5)^2 + ((-2) - (-1))^2}$$

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	(1.5, -3),	(2, 4)
(1, -2)	**1.118**		
(1, -2)	**2.236**		
(1, 2)			
(1, 2)			

$$1.118 + \sqrt{(1 - 0.5)^2 + ((-2) - (-1))^2}$$

Dynamic Time Warping for Multivariate Time Series

	$\boxed{(0.5, -1),}$	(1.5, -3),	(2, 4)
(1, -2)	**1.118**		
(1, -2)	**2.236**		
$\boxed{(1, 2)}$	**5.277**		
(1, 2)			

$$2.236 + \sqrt{(1 - 0.5)^2 + (2 - (-1))^2}$$

Dynamic Time Warping for Multivariate Time Series

	$\boxed{(0.5, -1),}$	(1.5, -3),	(2, 4)
(1, -2)	**1.118**		
(1, -2)	**2.236**		
(1, 2)	**5.277**		
$\boxed{(1, 2)}$	**8.318**		

$$5.277 + \sqrt{(1 - 0.5)^2 + (2 - (-1))^2}$$

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	$\boxed{(1.5, -3),}$	(2, 4)
$\boxed{(1, -2)}$	**1.118**	**2.236**	
(1, -2)	**2.236**		
(1, 2)	**5.277**		
(1, 2)	**8.318**		

$$1.118 + \sqrt{(1 - 1.5)^2 + ((-2) - (-3))^2}$$

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	(1.5, -3),	(2, 4)
(1, -2)	**1.118**	**2.236**	
(1, -2)	**2.236**	**2.236**	
(1, 2)	**5.277**		
(1, 2)	**8.318**		

$$1.118 + \sqrt{(1 - 1.5)^2 + ((-2) - (-3))^2}$$

Min {1.118, 2.236, 2.236}

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	$\boxed{(1.5, -3),}$	(2, 4)
(1, -2)	**1.118**	**2.236**	
(1, -2)	**2.236**	**2.236**	
$\boxed{(1, 2)}$	**5.277**	**7.261**	
(1, 2)	**8.318**		

$$\underbrace{2.236}_{\text{Min } \{5.277,\ 2.236,\ 2.236\}} + \sqrt{(1 - 1.5)^2 + (2 - (-3))^2}$$

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	(1.5, -3),	(2, 4)
(1, -2)	**1.118**	**2.236**	
(1, -2)	**2.236**	**2.236**	
(1, 2)	**5.277**	**7.261**	
(1, 2)	**8.318**	**12.286**	

$$5.277 + \sqrt{(1 - 1.5)^2 + (2 - (-3))^2}$$

Min {5.277, 7.261, 8.318}

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	(1.5, -3),	(2, 4)
(1, -2)	**1.118**	**2.236**	**8.319**
(1, -2)	**2.236**	**2.236**	**8.319**
(1, 2)	**5.277**	**7.261**	**4.472**
(1, 2)	**8.318**	**12.286**	$\boxed{\textbf{6.708}}$

Dynamic Time Warping for Multivariate Time Series

	(0.5, -1),	(1.5, -3),	(2, 4)
(1, -2)	**1.118**	**2.236**	
(1, -2)	**2.236**	**2.236**	
(1, 2)	**5.277**	**7.261**	
(1, 2)	**8.318**	**12.286**	

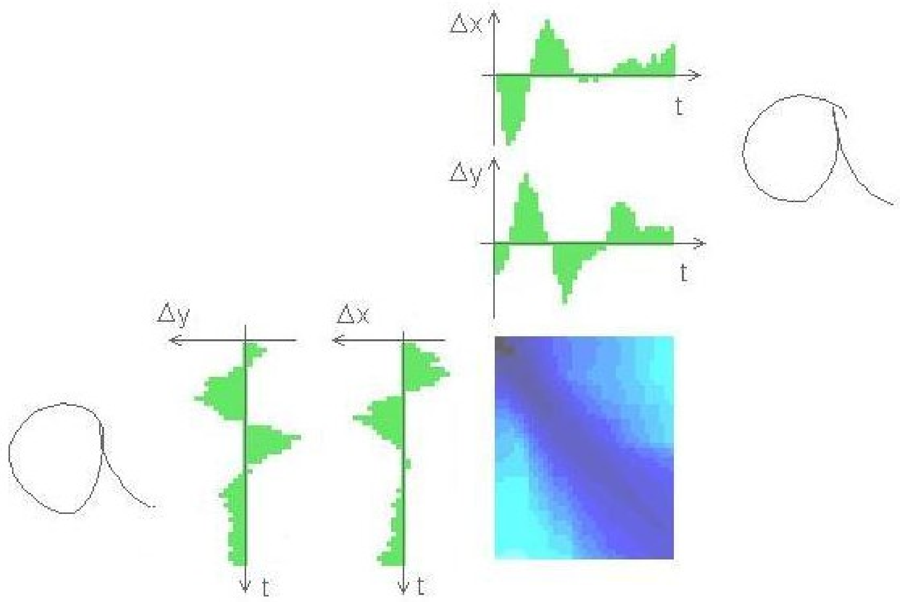
$$5.277 + \sqrt{(1 - 1.5)^2 + (2 - (-3))^2}$$

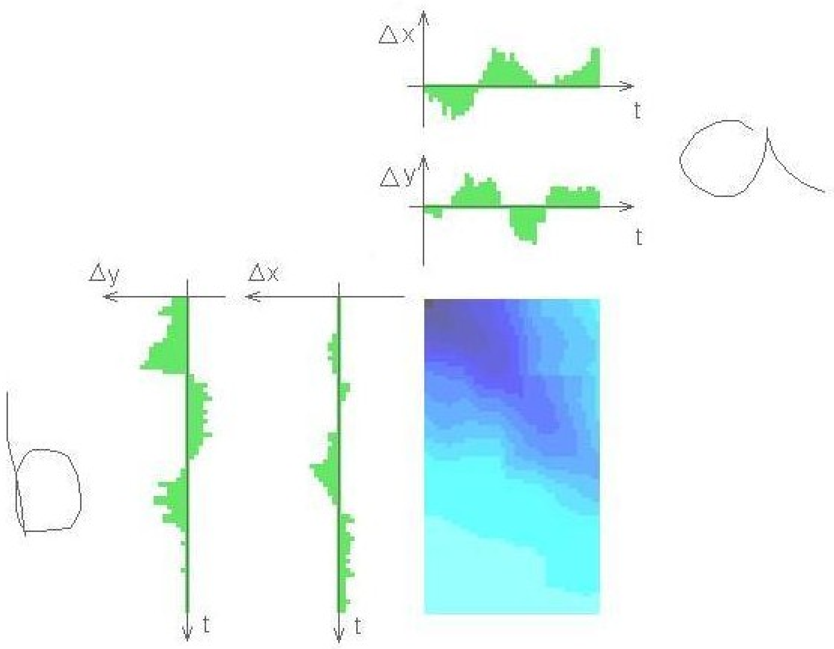
Instead of the Euclidean distance, we could calculate other distances, such as cosine distance.

Min {5.277, 7.261, 8.318}

Nearest Neighbor Classification

Example: Handwriting Recognition





„1NN-DTW is an exceptionally competitive classifier...“

- „... in spite of massive research effort on time series classification problems. We arrived at this conclusion after an extensive literature search“
- „In Rodriguez & Alonso (2004), the authors use a DTW based decision tree to classify time series. On the Two Patterns dataset, they report an error rate of 4.9%, but our experiments on the same dataset using 1NN give an error rate of 1.04% for Euclidean distance and 0.0% for DTW.“
- „In Rodriguez & Alonso et al. (2000), the authors use first order logic rules with boosting (...), they report an error rate of 3.6%, but our experiments on the same dataset using 1NN-DTW give an error rate of 0.33%.“
- „In Nanopoulos & Alcock et al. (2001), the authors use a multi-layer perceptron neural network (...) to achieve their best performance of 1.9% error rate. Using 1NN-DTW on the same dataset gives 0.33% error rate.“
- „In Wu & Chang (2004), the authors use a “super-kernel fusion scheme” to achieve an error rate of 0.79% (...) 1NN-DTW (…) gives an error rate of 0.33%.“
- „In Kim & Smyth et al. (2004), the authors use hidden Markov Models to achieve 98% accuracy on the PCV-ECG classification problem, but both DTW and Euclidean distance achieves a perfect accuracy on the same problem.“
- „The above list is truncated for brevity.“

Xi et al. (2006): Fast Time Series Classification Using Numerosity Reduction, ICML

„1NN-DTW is an exceptionally competitive classifier...“

- „There are dozens of similar examples in the literature. In addition to the above, there are a handful of papers in the literature that do explicitly claim to have a distance measure that beats DTW.“
- „Lei & Govindaraju (2004) claim that DTW gets 96.5% accuracy on the Gun-Point problem whereas their approach gets 98.0%. However, DTW actually gets 99.0% on that problem.“
- „1NN-DTW is very hard to beat.“

Xi et al. (2006): Fast Time Series Classification Using Numerosity Reduction, ICML

Improvements of Nearest Neighbor Classification ...

speeding-up classification and

… aim at

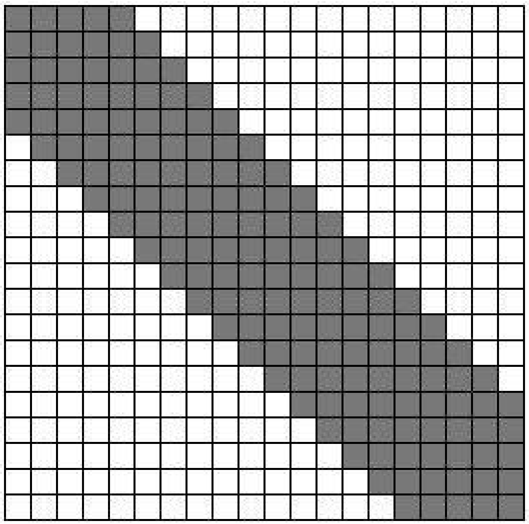
making classification more accurate.

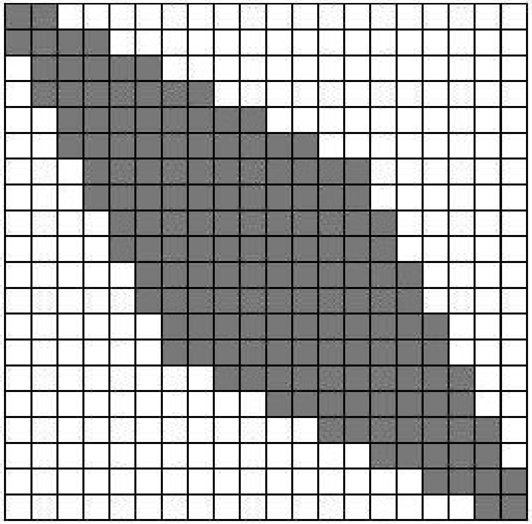
Speed-up techniques

Speed-up Techniques for Nearest Neighbor Classifiation of Time Series

- Efficient computation of the similarity / distance of time series
- Avoiding the computation of all the distances
  (lower bounding, early stopping of DTW-computation)
- Preprocessing techniques (e.g. SAX)
- Numerosity reduction / instance selection

Constrained DTW

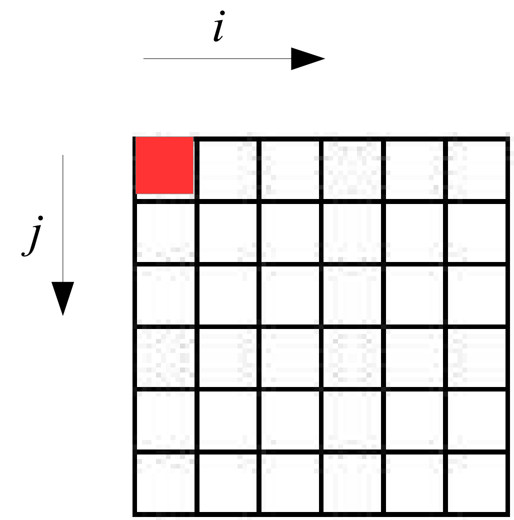




- Calculate only the marked entries of the DTW-matrix, i.e., the ones that are „close“ to the diagonal of the matrix
  - Sakoe-Chiba band (top)
  - Itakura parallelogram (bottom)
  - Beam search
  - Extreme variant of beam search: Lucky Time Warping (Spiegel, 2014)

Spiegel, Stephan, Brijnesh-Johannes Jain, Sahin Albayrak (2014): Fast Time Series Classification under Lucky Time Warping Distance, 29th Annual ACM Symposium on Applied Computing

Lucky Time Warping (LTW)



Algorithm 1 LTW Distance Measure

Input: $Q, C \ldots$ time series; $w \ldots$ warping window
Output: $d \ldots$ lucky distance

$i, j \leftarrow 1$
$d \leftarrow (q_i - c_j)^2$ $\{q_i, c_j$ equals $Q(i), C(j)\}$
$n \leftarrow$ length of Q
$m \leftarrow$ length of C
while $(i \leq n)$ and $(j \leq m)$ **do**
  if $(i+1 \leq n)$ and $(j+1 \leq m)$ **then**
    $d_{dia} \leftarrow (q_{i+1} - c_{j+1})^2$
  end if
  if $(i+1 \leq n)$ and $(|i+1-j| \leq w)$ **then**
    $d_{up} \leftarrow (q_{i+1} - c_j)^2$
  end if
  if $(j+1 \leq m)$ and $(|j+1-i| \leq w)$ **then**
    $d_{right} \leftarrow (q_i - c_{j+1})^2$
  end if
  $d_{\min} = \min(d_{dia}, d_{up}, d_{right})$
  $d \leftarrow d + d_{\min}$
  $i, j \leftarrow \text{index}(d_{\min})$ {update position}
end while

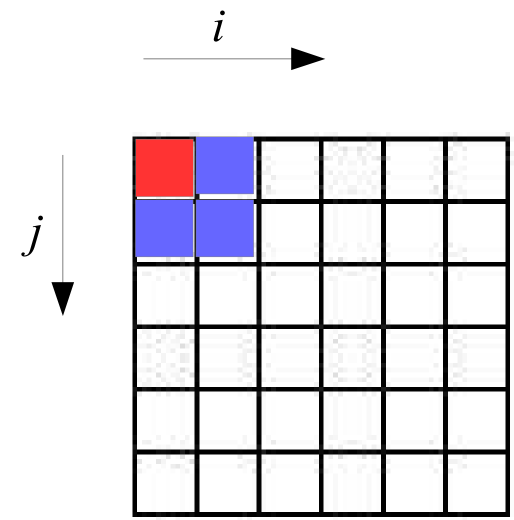
Spiegel, Stephan, Brijnesh-Johannes Jain, Sahin Albayrak (2014): Fast Time Series Classification under Lucky Time Warping Distance, 29th Annual ACM Symposium on Applied Computing

Lucky Time Warping (LTW)

Algorithm 1 LTW Distance Measure

Input: $Q, C \ldots$ time series; $w \ldots$ warping window
Output: $d \ldots$ lucky distance

$i, j \leftarrow 1$
$d \leftarrow (q_i - c_j)^2$ $\{q_i, c_j$ equals $Q(i), C(j)\}$
$n \leftarrow$ length of Q
$m \leftarrow$ length of C
while $(i \leq n)$ and $(j \leq m)$ **do**
  if $(i + 1 \leq n)$ and $(j + 1 \leq m)$ **then**
    $d_{dia} \leftarrow (q_{i+1} - c_{j+1})^2$
  end if
  if $(i + 1 \leq n)$ and $(|i + 1 - j| \leq w)$ **then**
    $d_{up} \leftarrow (q_{i+1} - c_j)^2$
  end if
  if $(j + 1 \leq m)$ and $(|j + 1 - i| \leq w)$ **then**
    $d_{right} \leftarrow (q_i - c_{j+1})^2$
  end if
  $d_{\min} = \min(d_{dia}, d_{up}, d_{right})$
  $d \leftarrow d + d_{\min}$
  $i, j \leftarrow \text{index}(d_{\min})$ {update position}
end while



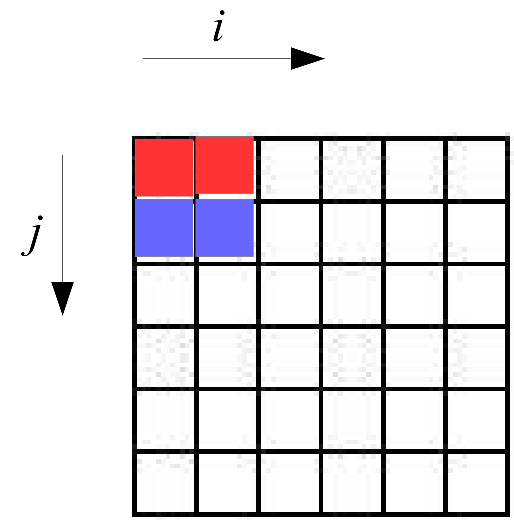
Spiegel, Stephan, Brijnesh-Johannes Jain, Sahin Albayrak (2014): Fast Time Series Classification under Lucky Time Warping Distance, 29th Annual ACM Symposium on Applied Computing

Lucky Time Warping (LTW)

Algorithm 1 LTW Distance Measure

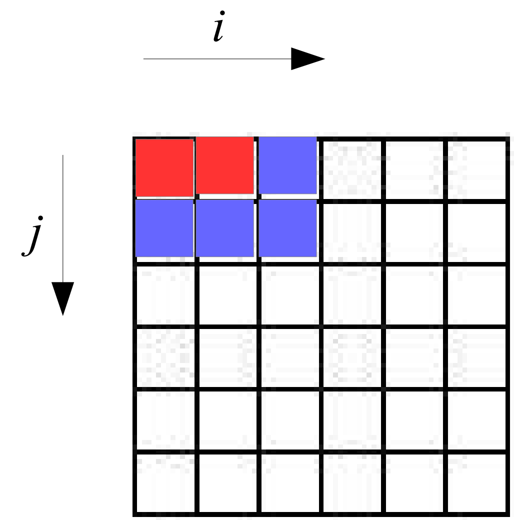
Input: $Q, C \ldots$ time series; $w \ldots$ warping window
Output: $d \ldots$ lucky distance

$i, j \leftarrow 1$
$d \leftarrow (q_i - c_j)^2$ $\{q_i, c_j$ equals $Q(i), C(j)\}$
$n \leftarrow$ length of Q
$m \leftarrow$ length of C
while $(i \leq n)$ and $(j \leq m)$ **do**
    if $(i+1 \leq n)$ and $(j+1 \leq m)$ **then**
        $d_{dia} \leftarrow (q_{i+1} - c_{j+1})^2$
    end if
    if $(i+1 \leq n)$ and $(|i+1-j| \leq w)$ **then**
        $d_{up} \leftarrow (q_{i+1} - c_j)^2$
    end if
    if $(j+1 \leq m)$ and $(|j+1-i| \leq w)$ **then**
        $d_{right} \leftarrow (q_i - c_{j+1})^2$
    end if
    $d_{\min} = \min(d_{dia}, d_{up}, d_{right})$
    $d \leftarrow d + d_{\min}$
    $i, j \leftarrow \text{index}(d_{\min})$ $\{$update position$\}$
end while



Spiegel, Stephan, Brijnesh-Johannes Jain, Sahin Albayrak (2014): Fast Time Series Classification under Lucky Time Warping Distance, 29th Annual ACM Symposium on Applied Computing

Lucky Time Warping (LTW)



Algorithm 1 LTW Distance Measure

Input: $Q, C \ldots$ time series; $w \ldots$ warping window
Output: $d \ldots$ lucky distance

$i, j \leftarrow 1$
$d \leftarrow (q_i - c_j)^2$ $\{q_i, c_j$ equals $Q(i), C(j)\}$
$n \leftarrow$ length of Q
$m \leftarrow$ length of C
while $(i \leq n)$ and $(j \leq m)$ **do**
  if $(i + 1 \leq n)$ and $(j + 1 \leq m)$ **then**
    $d_{dia} \leftarrow (q_{i+1} - c_{j+1})^2$
  end if
  if $(i + 1 \leq n)$ and $(|i + 1 - j| \leq w)$ **then**
    $d_{up} \leftarrow (q_{i+1} - c_j)^2$
  end if
  if $(j + 1 \leq m)$ and $(|j + 1 - i| \leq w)$ **then**
    $d_{right} \leftarrow (q_i - c_{j+1})^2$
  end if
  $d_{\min} = \min(d_{dia}, d_{up}, d_{right})$
  $d \leftarrow d + d_{\min}$
  $i, j \leftarrow \mathrm{index}(d_{\min})$ {update position}
end while

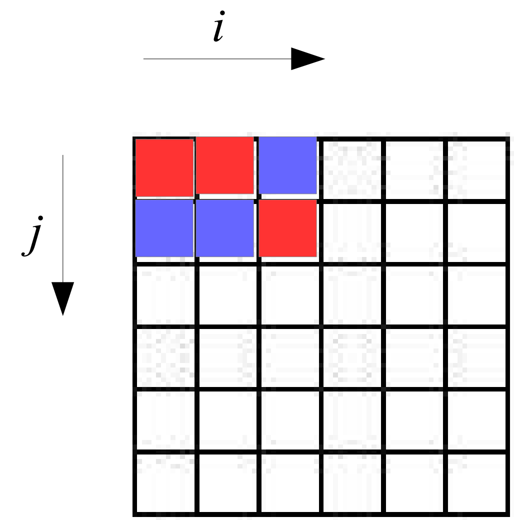
Spiegel, Stephan, Brijnesh-Johannes Jain, Sahin Albayrak (2014): Fast Time Series Classification under Lucky Time Warping Distance, 29th Annual ACM Symposium on Applied Computing

Lucky Time Warping (LTW)

Algorithm 1 LTW Distance Measure

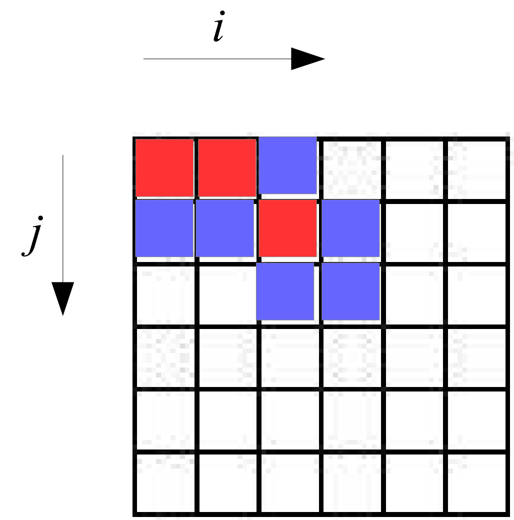
Input: $Q, C \ldots$ time series; $w \ldots$ warping window
Output: $d \ldots$ lucky distance

$i, j \leftarrow 1$
$d \leftarrow (q_i - c_j)^2$ $\{q_i, c_j$ equals $Q(i), C(j)\}$
$n \leftarrow$ length of Q
$m \leftarrow$ length of C
while $(i \leq n)$ and $(j \leq m)$ **do**
  if $(i + 1 \leq n)$ and $(j + 1 \leq m)$ **then**
    $d_{dia} \leftarrow (q_{i+1} - c_{j+1})^2$
  end if
  if $(i + 1 \leq n)$ and $(|i + 1 - j| \leq w)$ **then**
    $d_{up} \leftarrow (q_{i+1} - c_j)^2$
  end if
  if $(j + 1 \leq m)$ and $(|j + 1 - i| \leq w)$ **then**
    $d_{right} \leftarrow (q_i - c_{j+1})^2$
  end if
  $d_{\min} = \min(d_{dia}, d_{up}, d_{right})$
  $d \leftarrow d + d_{\min}$
  $i, j \leftarrow \text{index}(d_{\min})$ {update position}
end while



Spiegel, Stephan, Brijnesh-Johannes Jain, Sahin Albayrak (2014): Fast Time Series Classification under Lucky Time Warping Distance, 29th Annual ACM Symposium on Applied Computing

Lucky Time Warping (LTW)



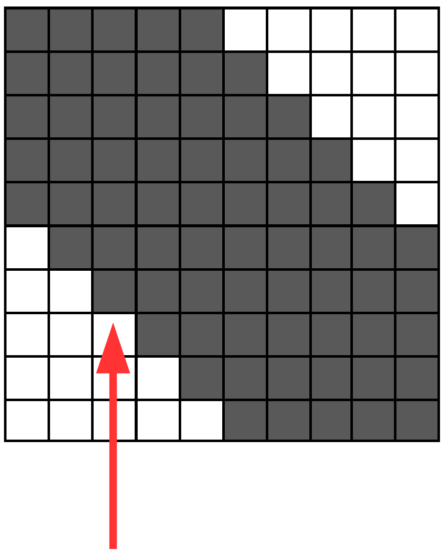
Algorithm 1 LTW Distance Measure

Input: $Q, C \ldots$ time series; $w \ldots$ warping window
Output: $d \ldots$ lucky distance

$i, j \leftarrow 1$
$d \leftarrow (q_i - c_j)^2$ $\{q_i, c_j$ equals $Q(i), C(j)\}$
$n \leftarrow$ length of Q
$m \leftarrow$ length of C
while $(i \leq n)$ and $(j \leq m)$ **do**
  if $(i + 1 \leq n)$ and $(j + 1 \leq m)$ **then**
    $d_{dia} \leftarrow (q_{i+1} - c_{j+1})^2$
  end if
  if $(i + 1 \leq n)$ and $(|i + 1 - j| \leq w)$ **then**
    $d_{up} \leftarrow (q_{i+1} - c_j)^2$
  end if
  if $(j + 1 \leq m)$ and $(|j + 1 - i| \leq w)$ **then**
    $d_{right} \leftarrow (q_i - c_{j+1})^2$
  end if
  $d_{\min} = \min(d_{dia}, d_{up}, d_{right})$
  $d \leftarrow d + d_{\min}$
  $i, j \leftarrow \text{index}(d_{\min})$ {update position}
end while

Spiegel, Stephan, Brijnesh-Johannes Jain, Sahin Albayrak (2014): Fast Time Series Classification under Lucky Time Warping Distance, 29th Annual ACM Symposium on Applied Computing

Early Stop



This column was just calculated. If all the entries in this column are larger than d', we do not need to calculate the rest of the matrix.

- We want to determine the nearest neighbours of the time series T^*
- We are in an intermediate step, i.e., we already calculated the distance between T^* and some of the time series of the training data $\rightarrow$ we know that the distance between T^* and another time series T' is d'
- Currently, we are calculating the distance between T^* and the time series T.
- If the DTW matrix has only entries being greater than d' in the column that was calculated last $\rightarrow$ stop and consider the next time series (in this case, T can not be the nearest neighbour of T^* because the distance between T^* and T' is lower than the distance between T^* and T).
- If the distance between T and T^* turns out to be less than d' $\rightarrow$ update d' and T'

Nearest Neighbor with Lower Bounding

T^* – Time series to be classified

d^* – distance of the currently found closest time series

```
d* ← infinity
for each time series T of the training data
      d ← estimate_distance(T*, T)
      if d > d*
            continue
      d′ ← DTW(T*, T)
      if d′ < d*
            d* ← d′
            nearest_neighbor ← t
```

d is a lower bound, i.e., the estimation is done in a way that the true distance is greater than or equal to d

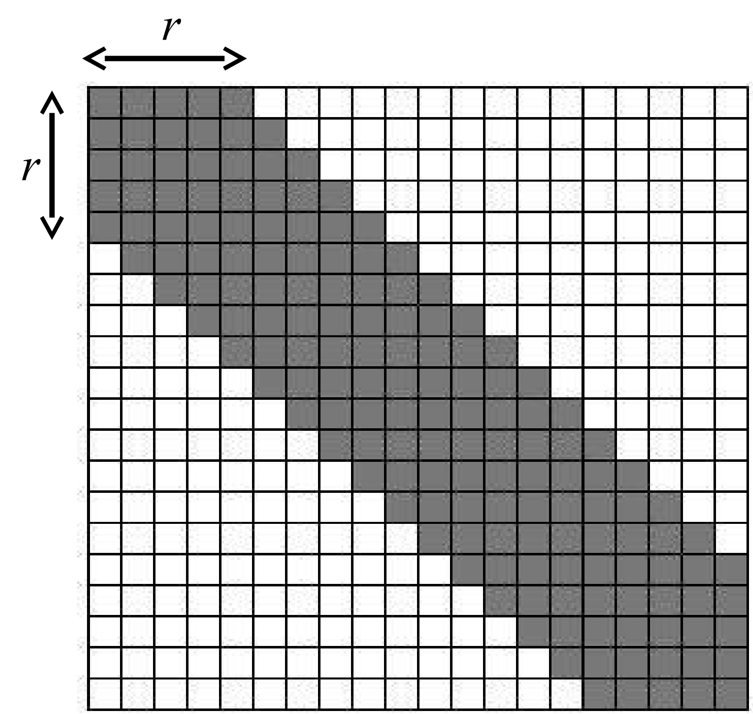
Lower Bound for Constrained DTW

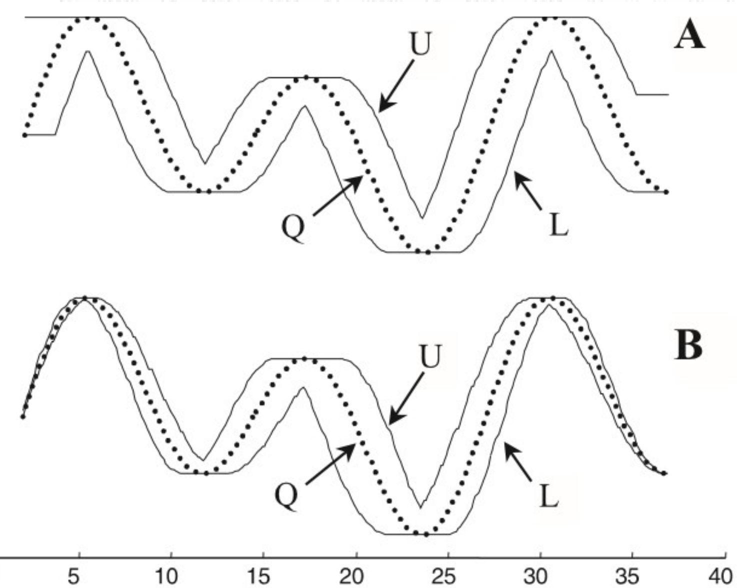
- Compare time series T_1: $q_1,\ldots,q_n$ and T_2: $c_1,\ldots,c_m$
- Sakoe-Chiba band, r = warping window size
- Define upper and lower time series:

$$\mathrm{U}_i = \max(q_{i-r} : q_{i+r})$$
$$\mathrm{L}_i = \min(q_{i-r} : q_{i+r})$$

- A lower bound (i.e., a possible implementatior of the `estimate_distance` function) is:

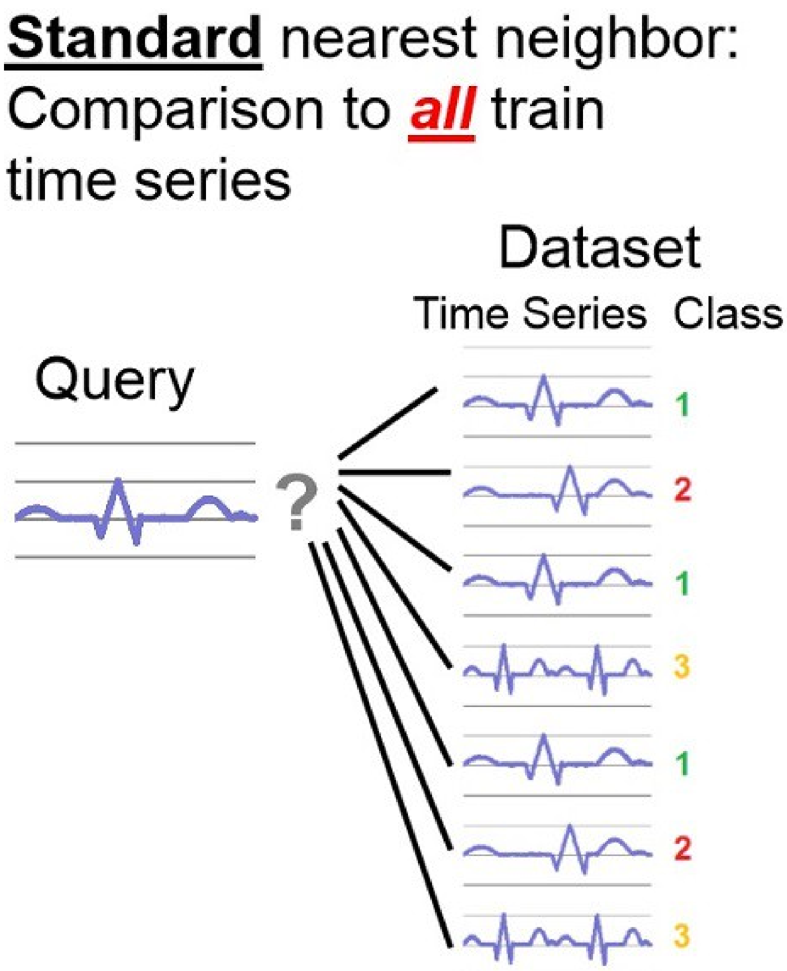
$$\sum_{i=1}^{n} \begin{cases} |c_i - \mathrm{U}_i| & \text{if } c_i > \mathrm{U}_i \\ |c_i - \mathrm{L}_i| & \text{if } c_i < \mathrm{L}_i \\ 0 & \text{otherwise} \end{cases}$$





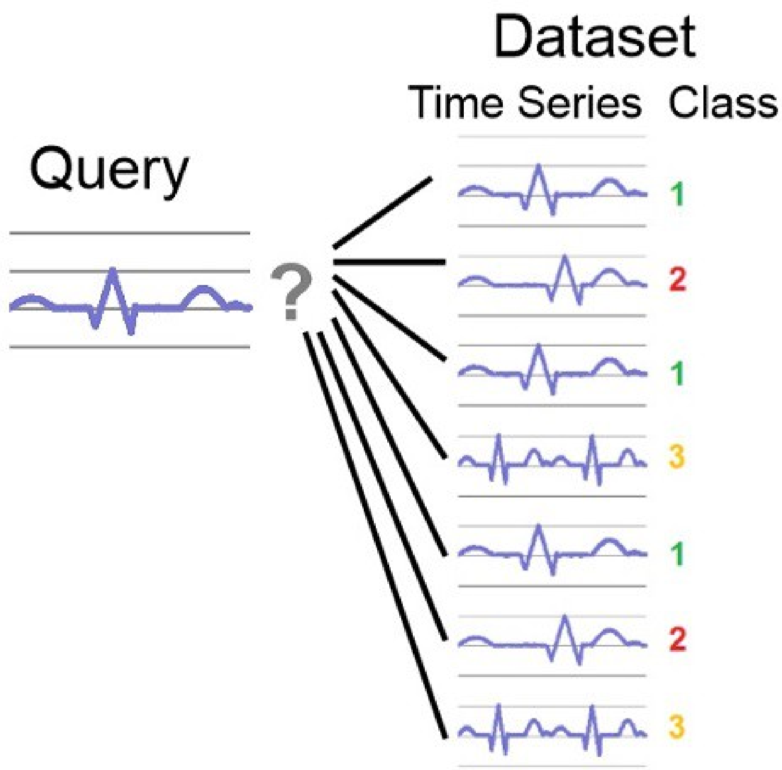
Keogh, Ratanamahatana (2005): Exact indexing of dynamic time warping, Knowledge and Information Systems 7.3, pp. 358.
Rath, Manmatha (2003): Lower-bounding of dynamic time warping distances for multivariate time series
Note: notations have been adapted.

Instance Selection (a.k.a. numerosity reduction)

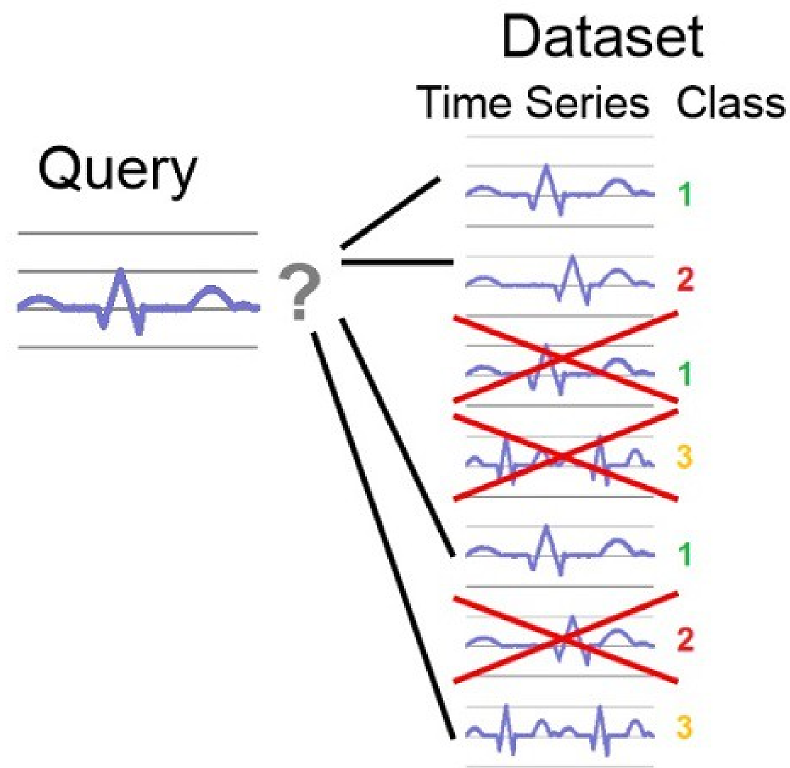


Instance Selection (a.k.a. numerosity reduction)

Standard nearest neighbor:
Comparison to ***all*** train time series



With instance selection:
Comparison to the ***selected*** train time series

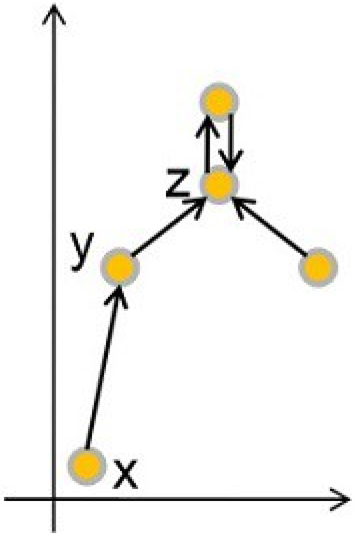


Hubness

- Instance *y* is a good (bad) *k*-nearest neighbor of the instance *x* if
  (i) *y* is one of the *k*-nearest neighbors of *x*, and
  (ii) both have the same (different) class labels.

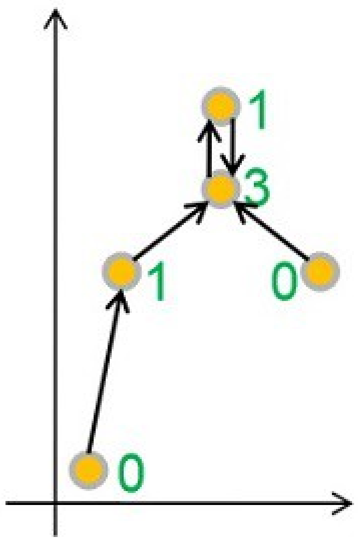
Hubness

- Instance *y* is a good (bad) *k*-nearest neighbor of the instance *x* if
  - (i) *y* is one of the *k*-nearest neighbors of *x*, and
  - (ii) both have the same (different) class labels.



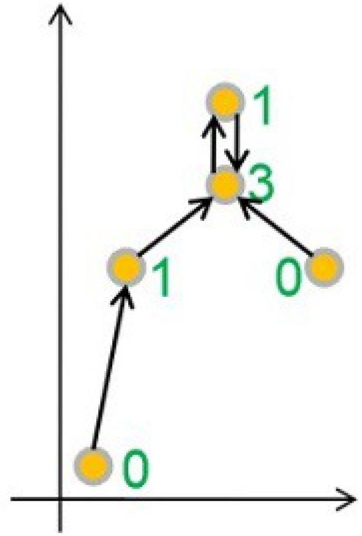
Hubness

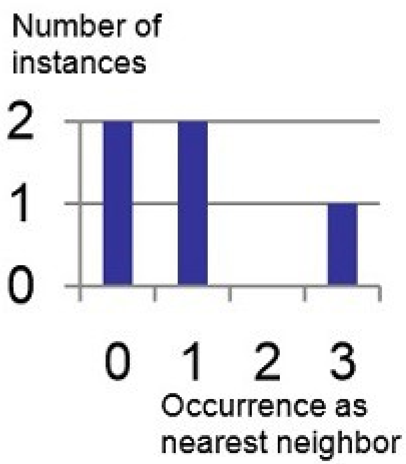
- Instance *y* is a good (bad) *k*-nearest neighbor of the instance *x* if
  - (i) *y* is one of the *k*-nearest neighbors of *x*, and
  - (ii) both have the same (different) class labels.



Hubness

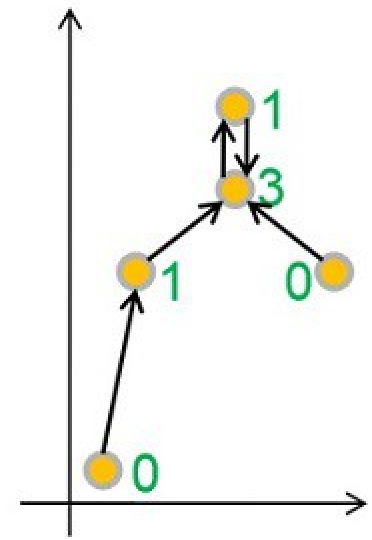
- Instance *y* is a good (bad) *k*-nearest neighbor of the instance *x* if
  (i) *y* is one of the *k*-nearest neighbors of *x*, and
  (ii) both have the same (different) class labels.

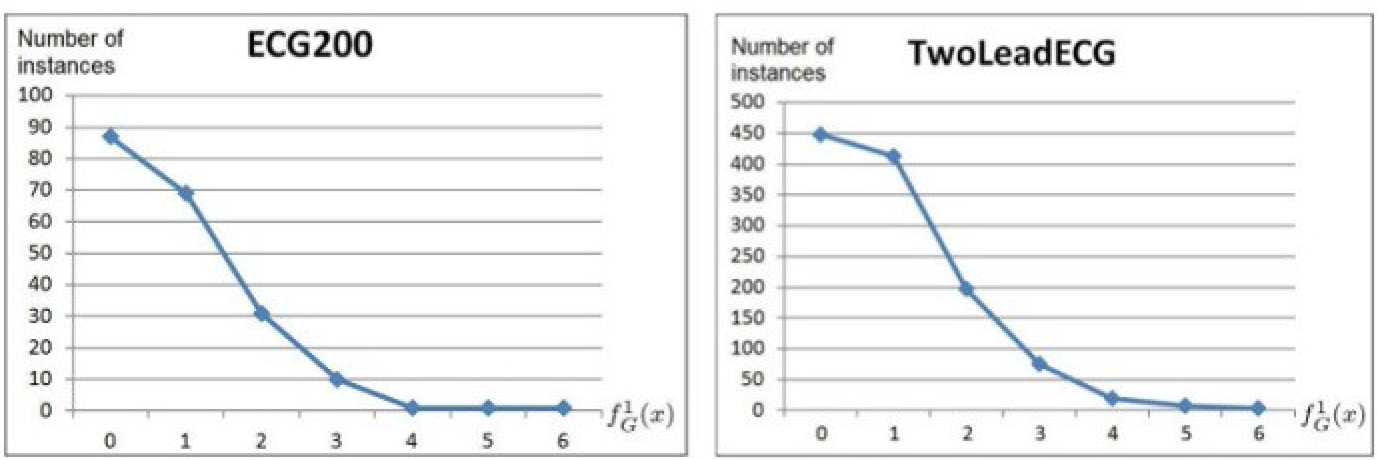


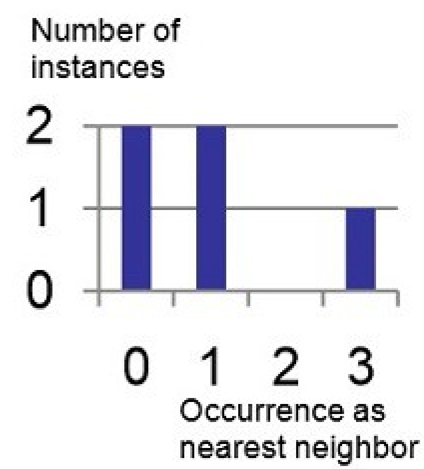


Hubness

- Instance *y* is a good (bad) *k*-nearest neighbor of the instance *x* if
  - (i) *y* is one of the *k*-nearest neighbors of *x*, and
  - (ii) both have the same (different) class labels.
- The distribution of good (bad) nearest neighbors is substantially **skewed** → **good (bad) hubs**







Distribution of good 1-nearest neighbors for some ECG datasets

Instance Selection based on Hubness

- Good (bad) occurrence of an instance *x* is the number of other instances that have *x* as one of their **good** (bad) *k*-nearest neighbors, denoted as $f_G^k(x)$ and $f_B^k(x)$.

- Good 1-occurrence score: $f_G(x) = f_G^1(x)$

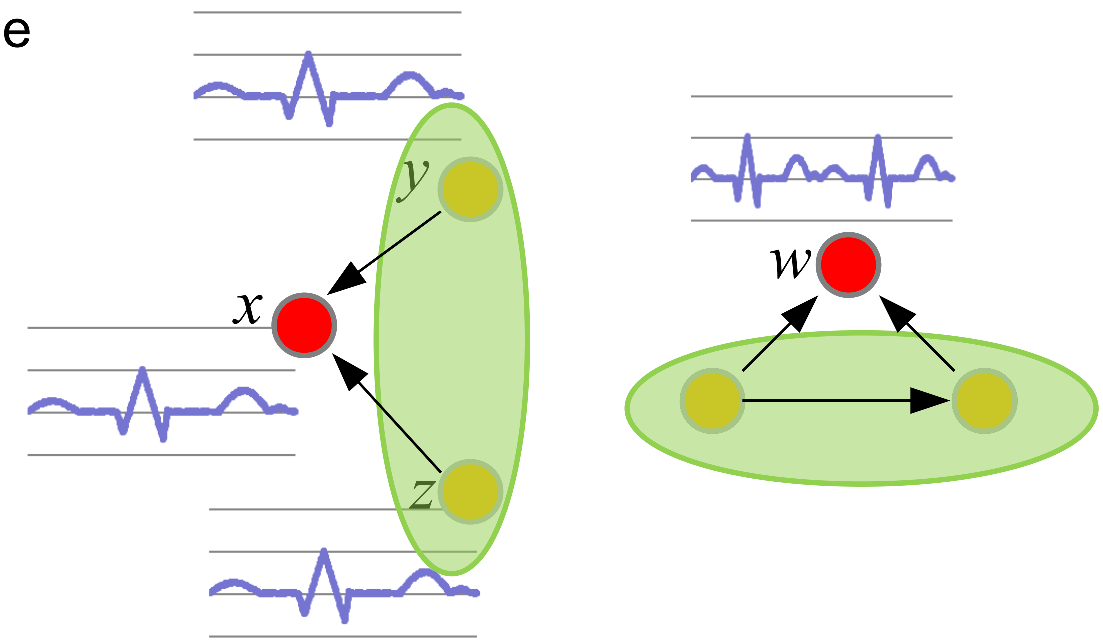
- Relative score: $f_R(x) = \dfrac{f_G^1(x)}{f_N^1(x) + 1}$ where $f_N^k(x) = f_G^k(x) + f_B^k(x)$

- Xi's score: $f_{Xi}(x) = f_G^1(x) - 2f_B^1(x)$
- A simple instance selection approach ("INSIGHT"):
  - rank instances based on one of these scores, and select the top-ranked instances

K. Buza, A. Nanopoulos, L. Schmidt-Thieme (2011): INSIGHT: Efficient and Effective Instance Selection for Time-Series Classification, 15th Pacific-Asia Conference on Knowledge Discovery and Data Mining

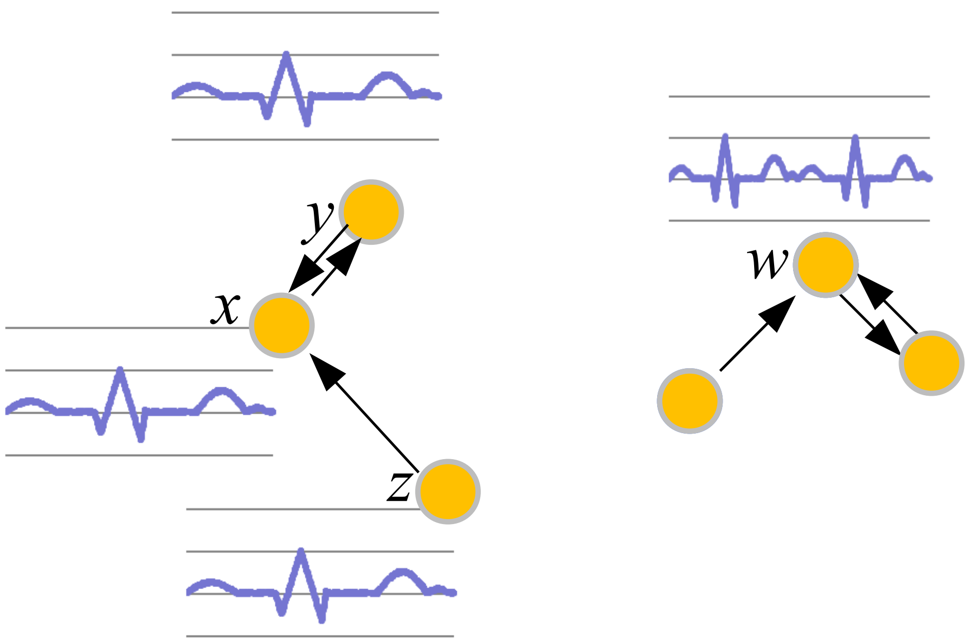
Coverage Graphs

- Each vertex corresponds to a time series
  - x covers y if x contributes to the correct classification of y
  - edge: $y \rightarrow x$
- Examples:
  - x cover both y and z
  - x and w together cover all coverable vertices
- Instance Selection Problem (ISP)
  - Find a set of vertices with minimal size that cover all coverable vertices
  - ISP is NP-complete
    - ISP is equivalent to the Set-covering problem



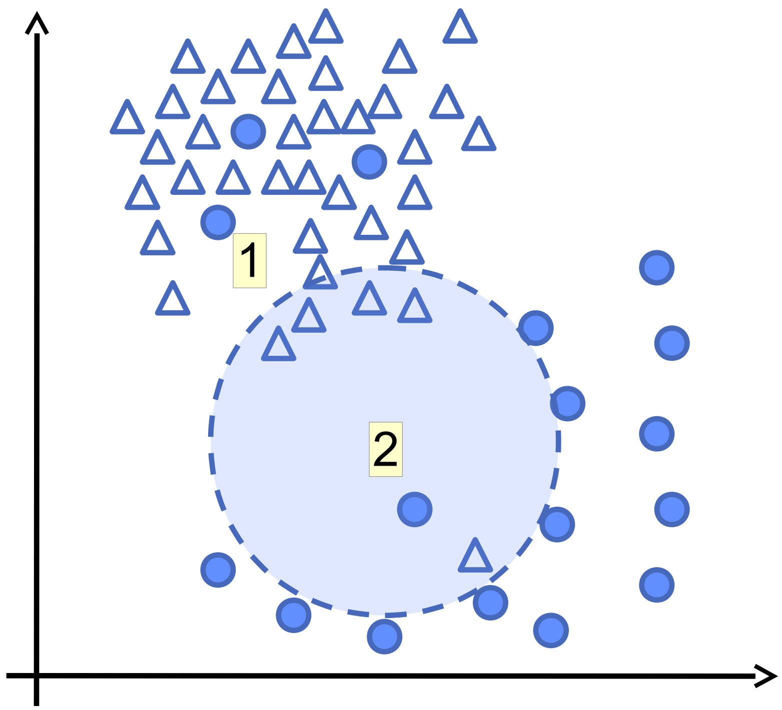
1-Nearest Neighbor Coverage Graphs

- Vertices are connected with their first nearest neighbor if it is a good neighbour
- m-limited Instance Selection Problem (m-ISP)
  - select m vertices that maximize coverage
- For 1-NN coverage graphs:
  - INSIGHT with good 1-occurrence score maximizes coverage



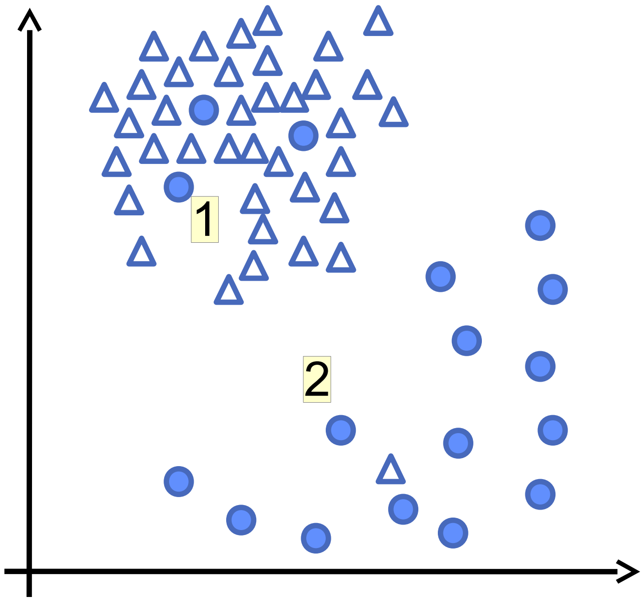
Improving the Accuracy

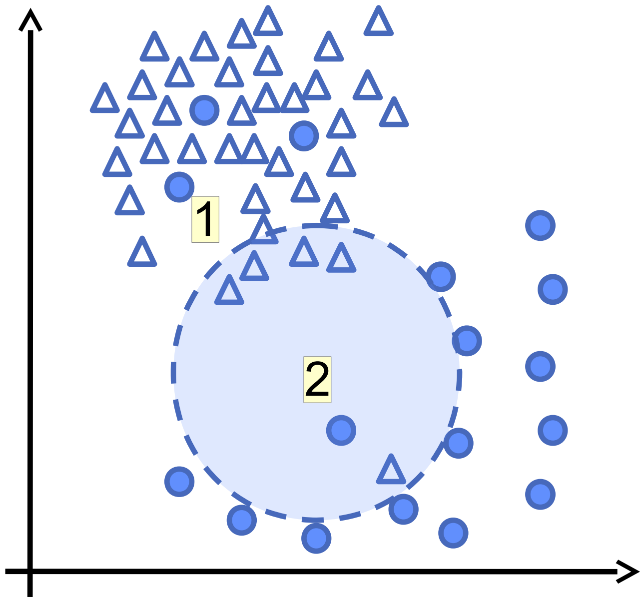
What is the appropriate number of nearest neighbors? (Motivating Example)



- Ground truth
  - “1” is triangle
  - “2” is circle
- 1-NN classifier
  - “1” is circle → mistake
  - “2” is circle → correct
- 6-NN classifier
  - “1” is triangle → correct
  - “2” is triangle → mistake
- Different k may be necessary in different regions

What is the appropriate number of nearest neighbors? (Motivating Example)





	1-NN
1	Circle
2	Circle
	Meta model for 1-NN
1	Incorrect
2	Correct

	6-NN
1	triangle
2	triangle
	Meta model for 6-NN
1	Correct
2	Incorrect

Individualized Quality Estimation

- In contrast to the previous (simple) example, meta models do not output a binary decision, but the likelihood of correct classification, i.e., the estimated quality of the primary model.

	Meta model for 1-NN
1	Mistake
2	Correct

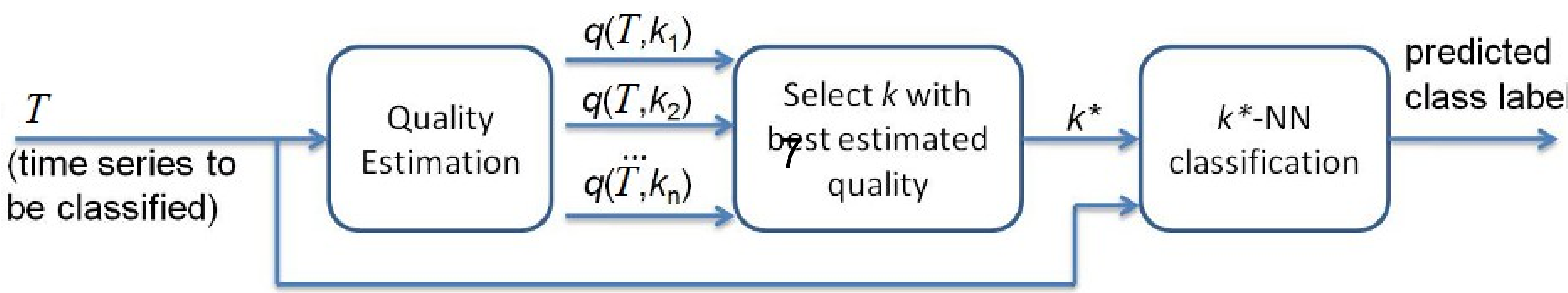
	Meta model for 1-NN
1	0.05
2	0.91

	Meta model for 6-NN
1	Correct
2	Mistake

	Meta model for 6-NN
1	0.82
2	0.07

Individual Quality Estimation

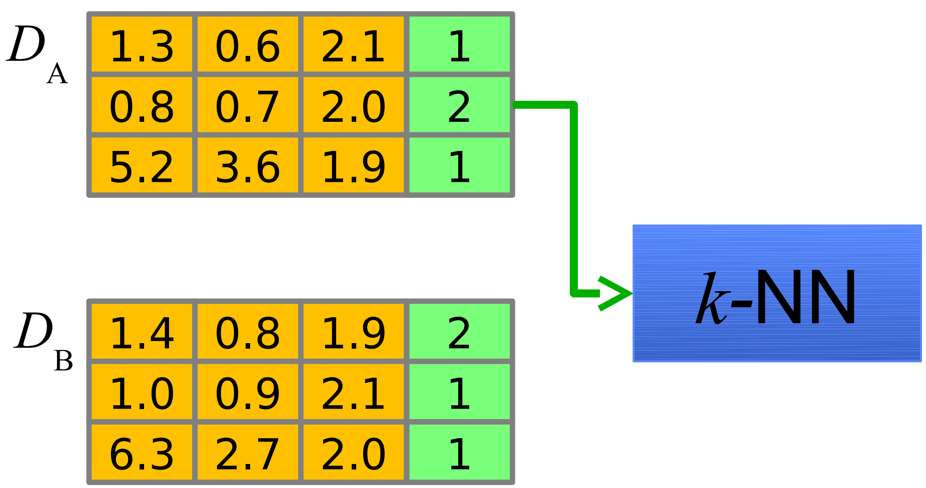
- Primary models (time series classifiers): k-NN classifiers with DTW
- Meta models (for error estimation): k'-NN regression with DTW (k' = 5)
- For each time series T to be classified:
  select k with maximal estimated quality
  - alternatively: weighted voting according to estimated qualities



K. Buza (2011): Fusion Methods for Time Series Classification
Peter Lang Verlag, http://www.biointelligence.hu/books.html

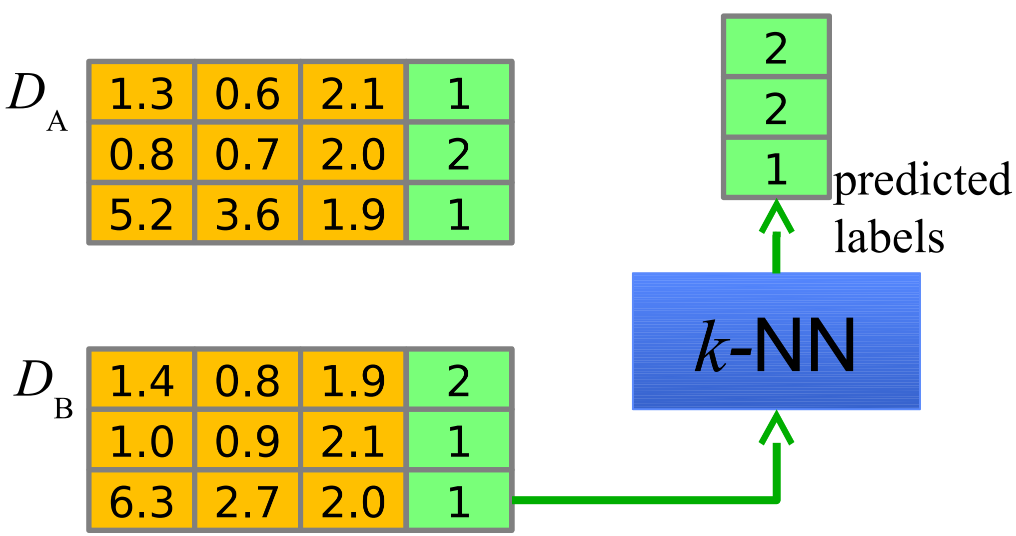
Training Meta Models

- Split labeled training data into D_A and D_B
- Train the primary model (k-NN) on D_A
- Let the primary model predict the labels of D_B
- Calculate quality of the predicted labels
- Train meta model M^* on D_B using the calculated quality scores as labels



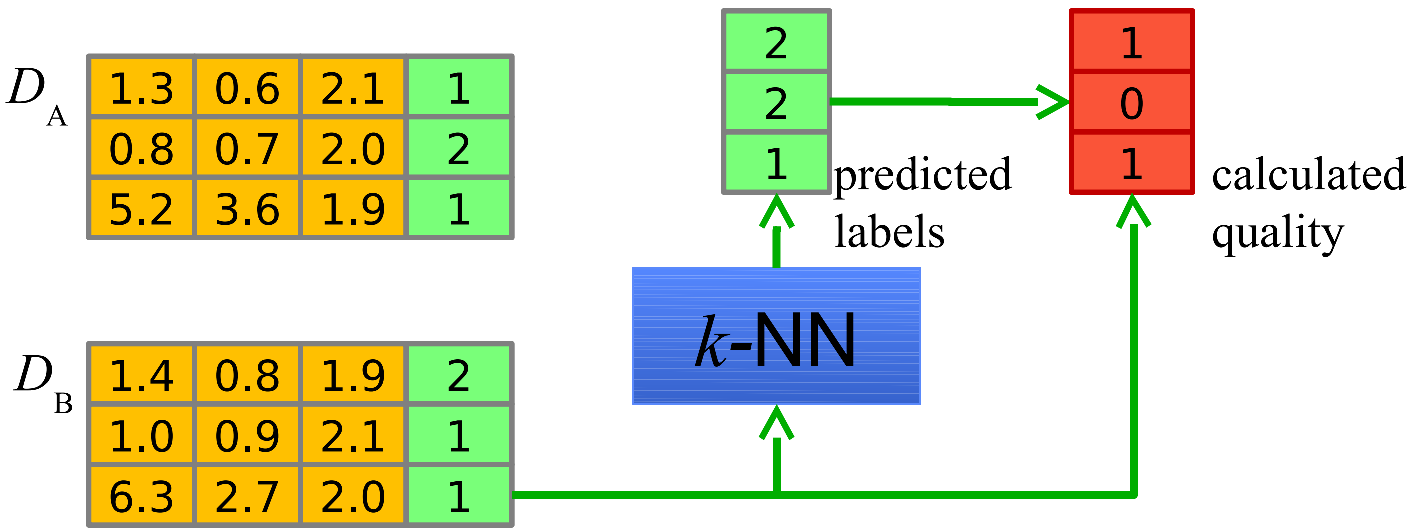
Training Meta Models

- Split labeled training data into D_A and D_B
- Train the primary model (k-NN) on D_A
- Let the primary model predict the labels of D_B
- Calculate quality of the predicted labels
- Train meta model M^* on D_B using the calculated quality scores as labels



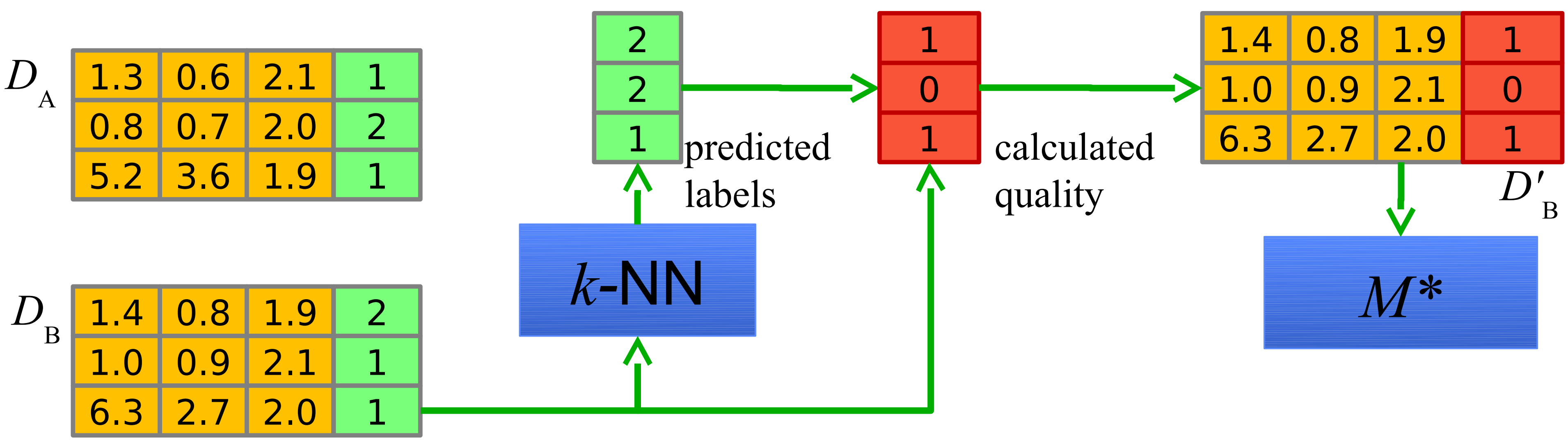
Training Meta Models

- Split labeled training data into D_A and D_B
- Train the primary model (k-NN) on D_A
- Let the primary model predict the labels of D_B
- Calculate quality of the predicted labels
- Train meta model M^* on D_B using the calculated quality scores as labels

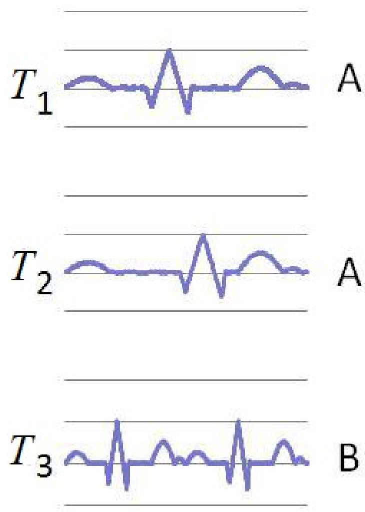


Training Meta Models

- Split labeled training data into D_A and D_B
- Train the primary model (k-NN) on D_A
- Let the primary model predict the labels of D_B
- Calculate quality of the predicted labels
- Train meta model M^* on D_B using the calculated quality scores as labels



Distance Learning

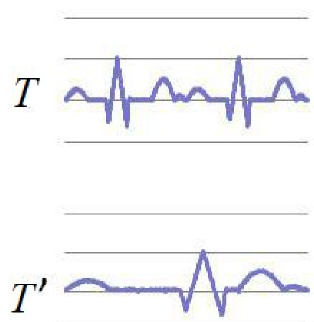


Train time series and their class labels (A or B)

	d_{EU}	d_{DTW}	d^F_{EU}	...	I
T_1 vs. T_1	0	0	0	...	0
T_1 vs. T_2	2.3	0.6	1.2	...	0
T_1 vs. T_3	8.6	6.5	9.5	...	1
	...	...	...	...	

Distances of time series pairs and their respective indicators

Train



Test time series

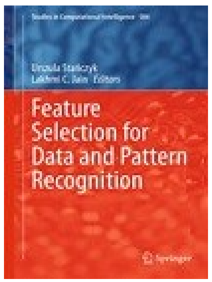
d_{EU}	d_{DTW}	d^F_{EU}	...
1.8	7.3	6.2	...

M

I
0.81

Hubness-aware Classifiers for Time Series Classification

- hwKNN, hFNN, NHBNN, HIKNN



Tomasev et al. (2015): Hubness-aware Classification, Instance Selection and Feature Construction: Survey and Extensions to Time-Series,
In: U. Stanczyk, L. Jain (eds.), Feature selection for data and pattern recognition, Springer-Verlag.
http://www.biointelligence.hu/books.html
http://www.biointelligence.hu/course.html

Radovanović et al. (2010): Time-series classification in many intrinsic dimensions, Proceedings of the 2010 SIAM International Conference on Data Mining, pp. 677-688

Evaluation of Time Series Classifiers

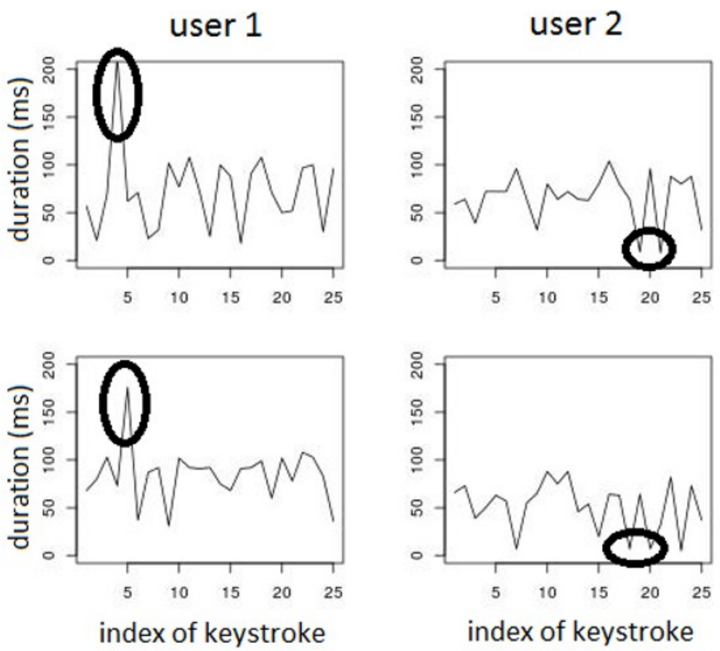
Evaluation of Time Series Classifiers

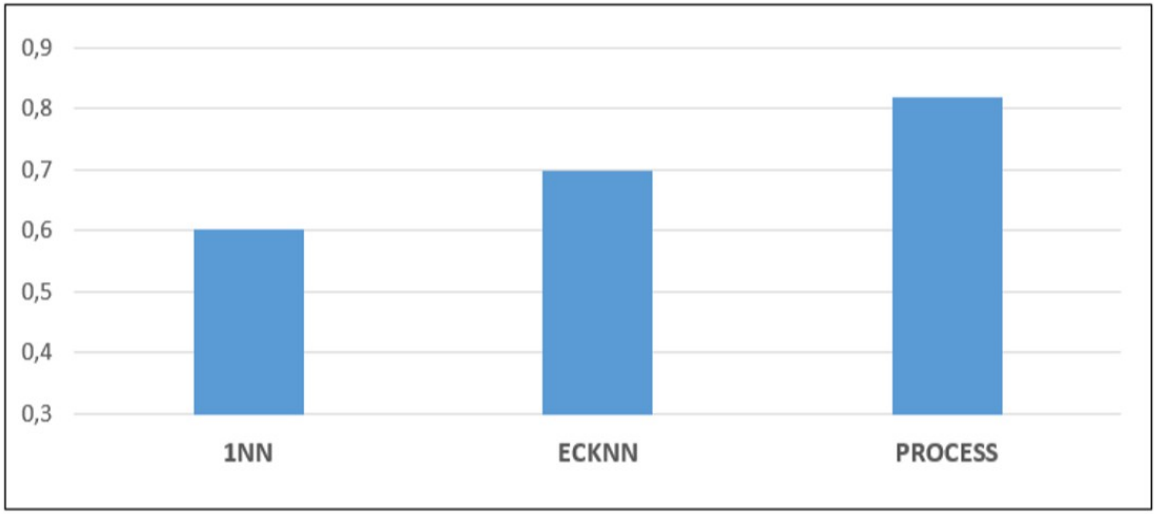
- Evaluation protocol
  - Test set must be independent
    (be careful with trying different hyperparameters!)
  - Goal: simulate an application – make realistic assumptions
    - Availability of training data (e.g. rare diseases)
    - Split data carefully (temporal splits, patient-based splits...)
  - Cross-validation
- Evaluation metrics
  - Accuracy, AUC, precision, recall, F-measure, AUPR
    (be careful when classifying imbalanced data)
  - Standard deviation, statistical significance tests

Selected Applications

Person Identification based on Keystroke Dynamics

- Duration of a keystroke = the time between pressing and releasing a key
- Mapping into a 60-dimensional vector space

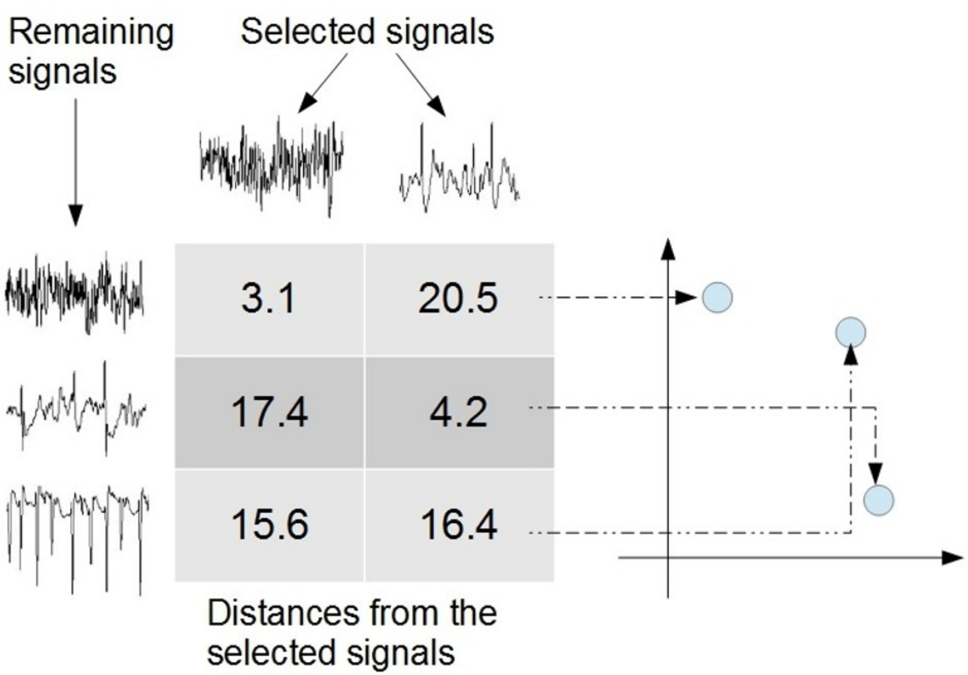


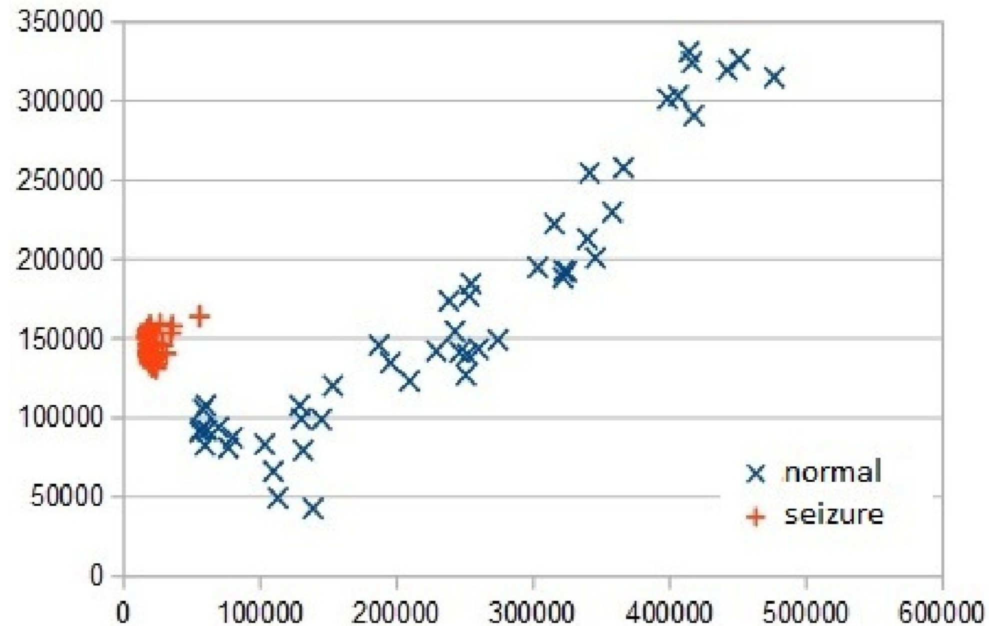


D. Neubrandt, K. Buza (2017): Projection-based Person Identification,
Proceedings of the 10th International Conference on Computer Recognition Systems (CORES), Springer.

Classification of Brain Activity Data

- Electroencephalograph (EEG) data
- Logistic regression using DTW-distance from randomly selected time series as features

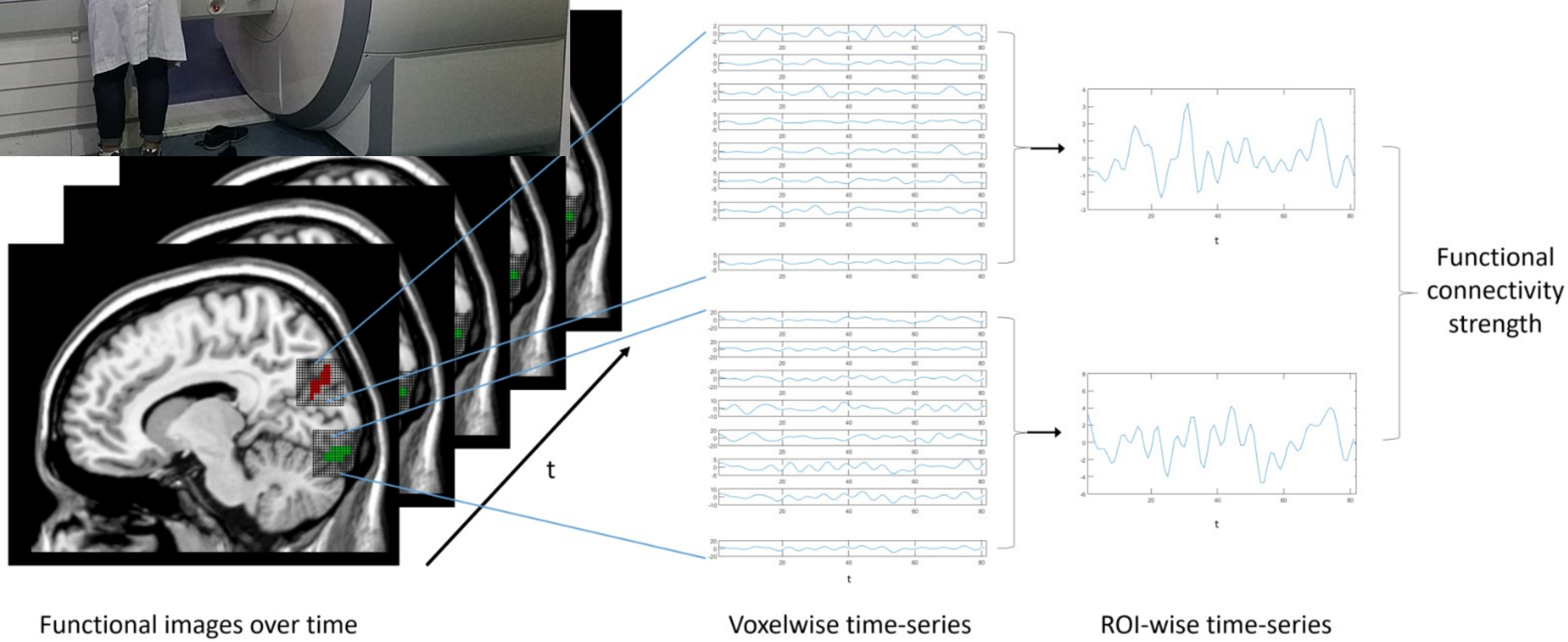




K. Buza, J. Koller, K. Marussy (2015): PROCESS: Projection-Based Classification of Electroencephalograph Signals, ICAISC, LNCS Vol. 9120, pp. 91-100, Springer.

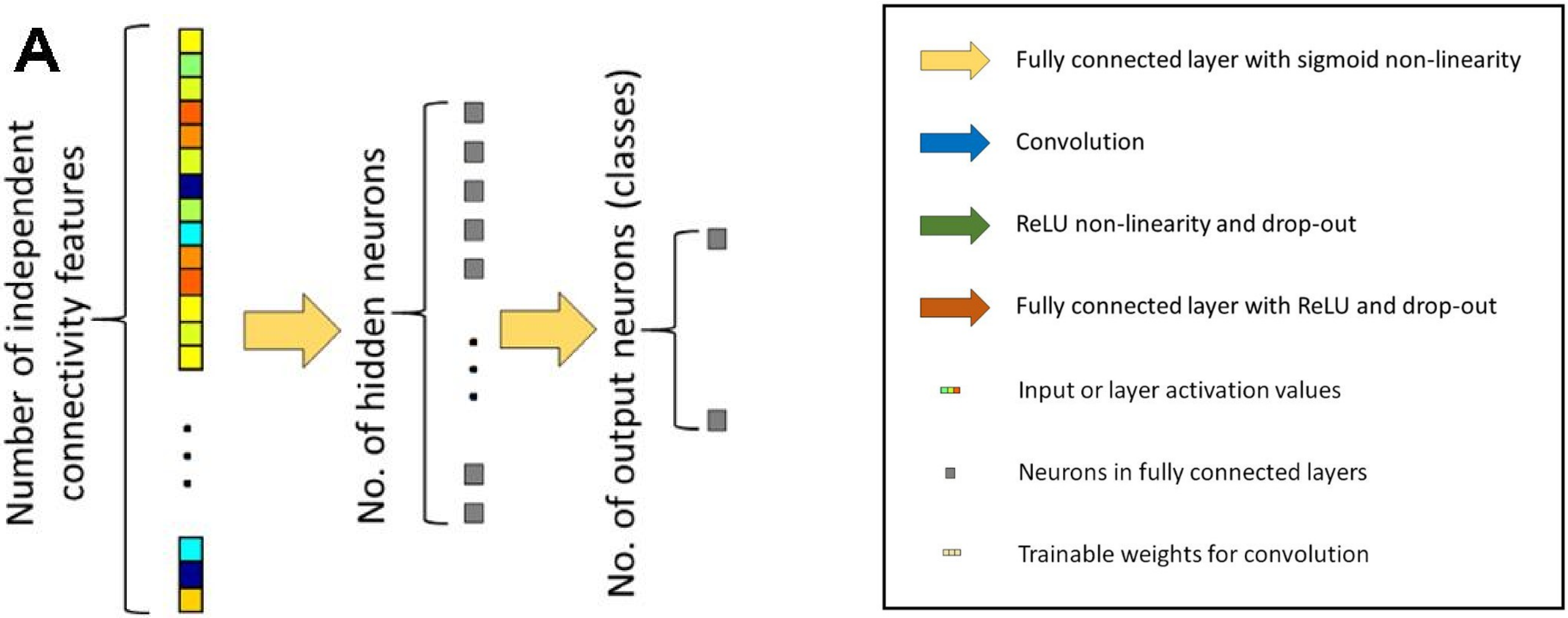
Classification of Brain Imaging Data

by Ptrump16 (Own work) [CC BY-SA 4.0 (https://creativecommons.org/licenses/by-sa/4.0)], via Wikimedia Commons

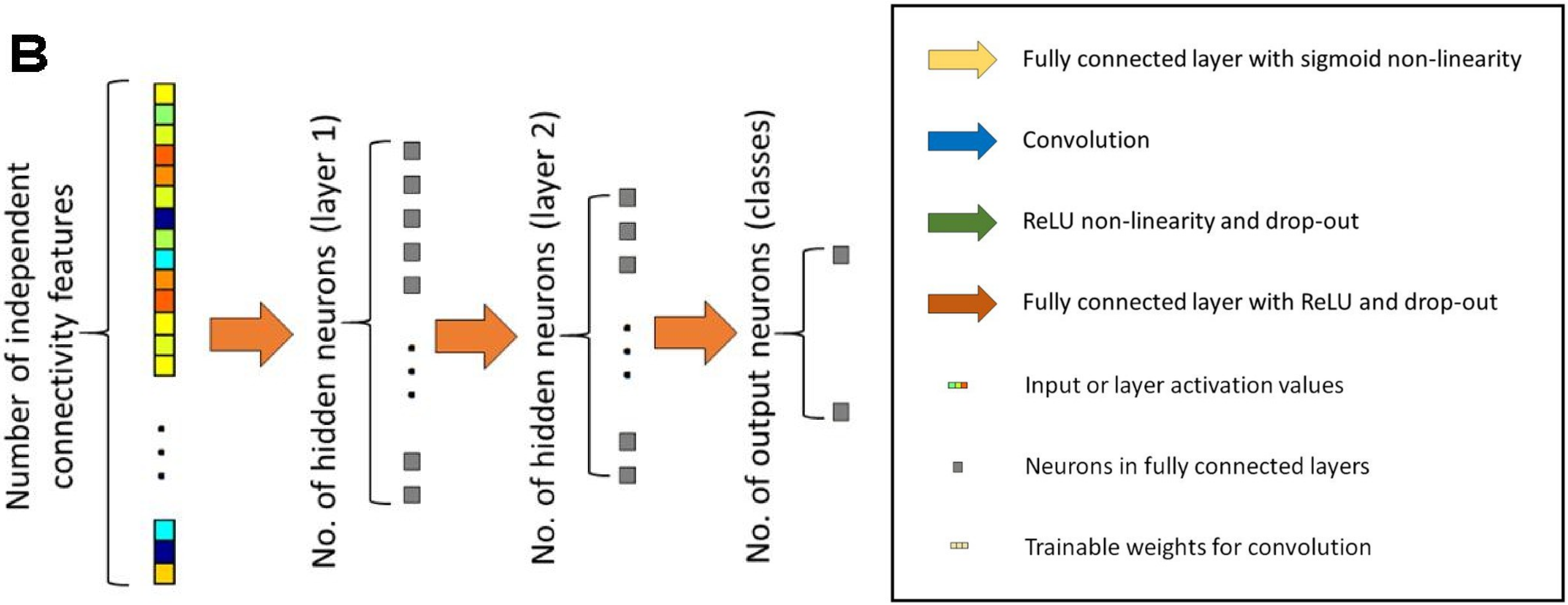


Regina J. Meszlényi, Krisztian Buza, Zoltán Vidnyánszky (2017): Resting State fMRI Functional Connectivity-Based Classification Using a Convolutional Neural Network Architecture, Frontiers in Neuroinformatics, Vol. 11

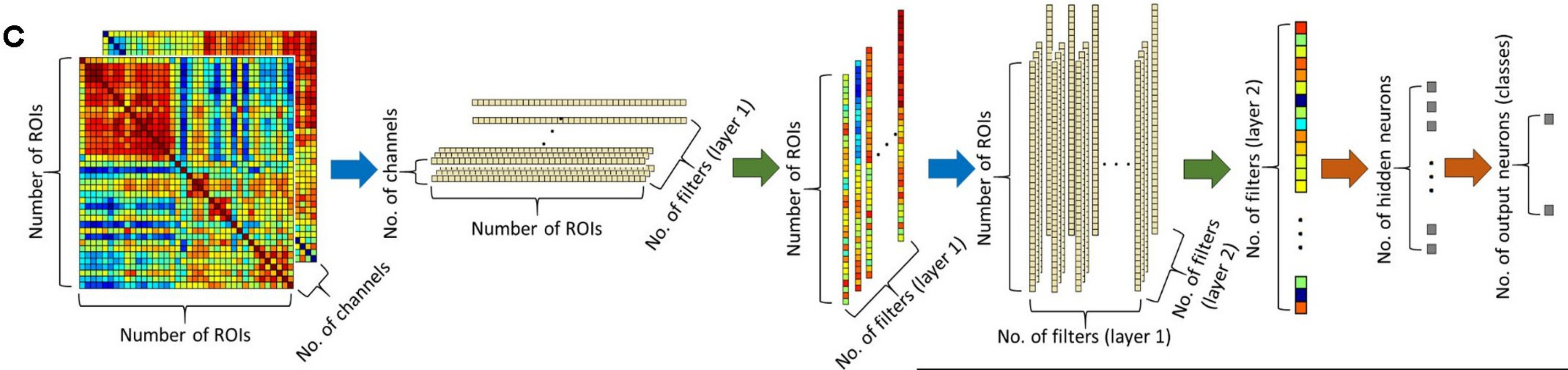
Simple Neural Network Classifier

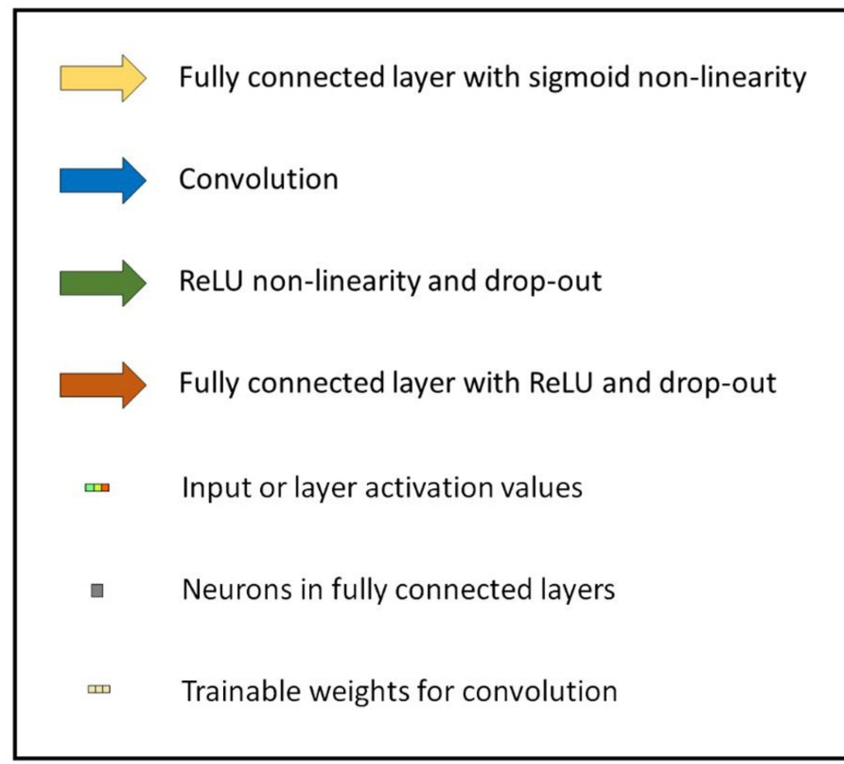


Deep Neural Network Classifier



Connectome-Convolutional Neural Network Classifier





Classification Results

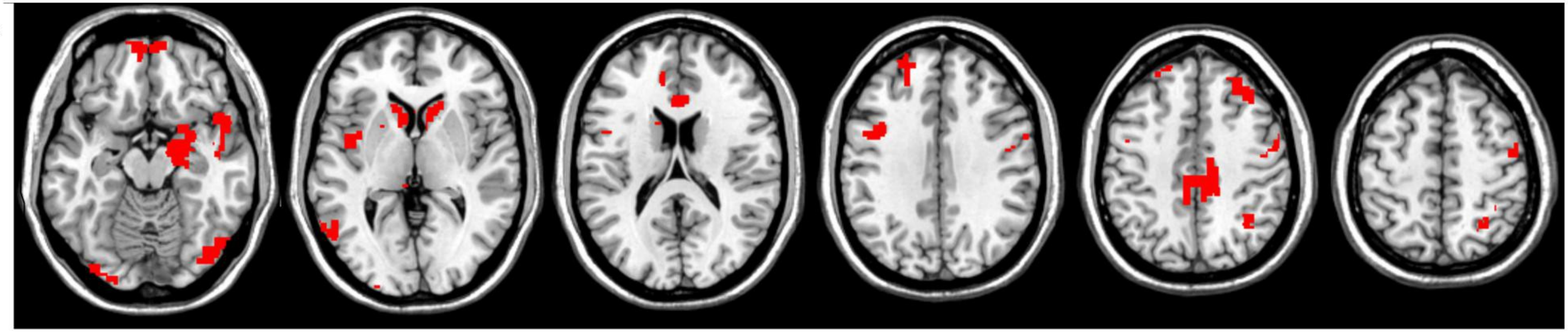
	CORR	DTW	Path length	DTW+Path length
SVM				
Accuracy (%)	54.1	67.1	64.4	66.4
AUC	0.541	0.672	0.644	0.664
LASSO				
Accuracy (%)	60.3	59.6	69.9	69.9
AUC	0.602	0.595	0.699	0.699
Simple net				
Accuracy (%)	50	52.1	57.3	56.2
AUC	0.515	0.505	0.59	0.588
Deep net				
Accuracy (%)	50.7	61.6	62.3	61.0
AUC	0.533	0.634	0.635	0.611
CCNN				
Accuracy (%)	53.4	65.1	64.4	71.9
AUC	0.521	0.684	0.672	0.746

Classification Results

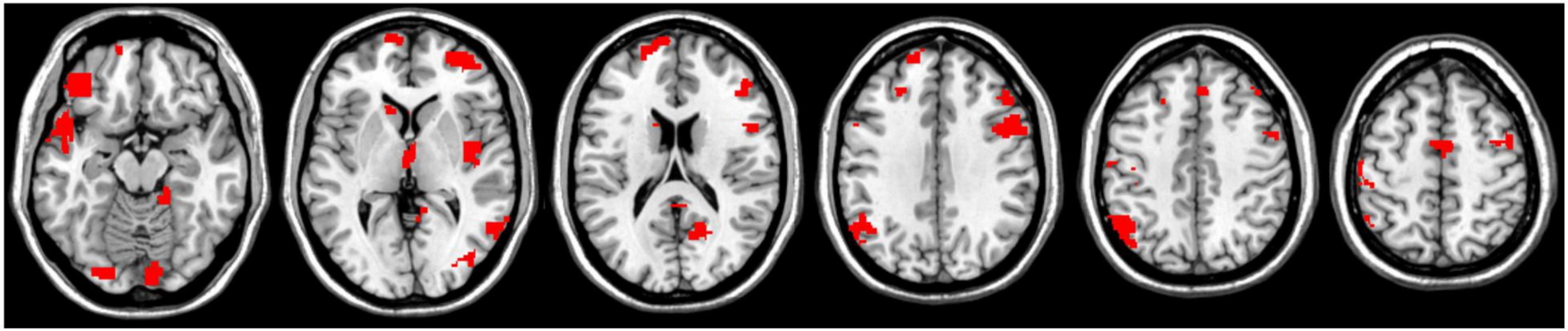
	CORR	DTW	Path length	DTW+Path length
SVM				
Accuracy (%)	54.1	67.1	64.4	66.4
AUC	0.541	0.672	0.644	0.664
LASSO				
Accuracy (%)	60.3	59.6	69.9	69.9
AUC	0.602	0.595	0.699	0.699
Simple net				
Accuracy (%)	50	52.1	57.3	56.2
AUC	0.515	0.505	0.59	0.588
Deep net				
Accuracy (%)	50.7	61.6	62.3	61.0
AUC	0.533	0.634	0.635	0.611
CCNN				
Accuracy (%)	53.4	65.1	64.4	71.9
AUC	0.521	0.684	0.672	0.746

Most Influential ROIs

A



B



Most influential ROIs based on the first convolutional layer's weights for MCI classification with CCNN.
(A) Important ROIs based on DTW distance features.
(B) Important ROIs based on warping path length features.

Conclusion

Conclusions

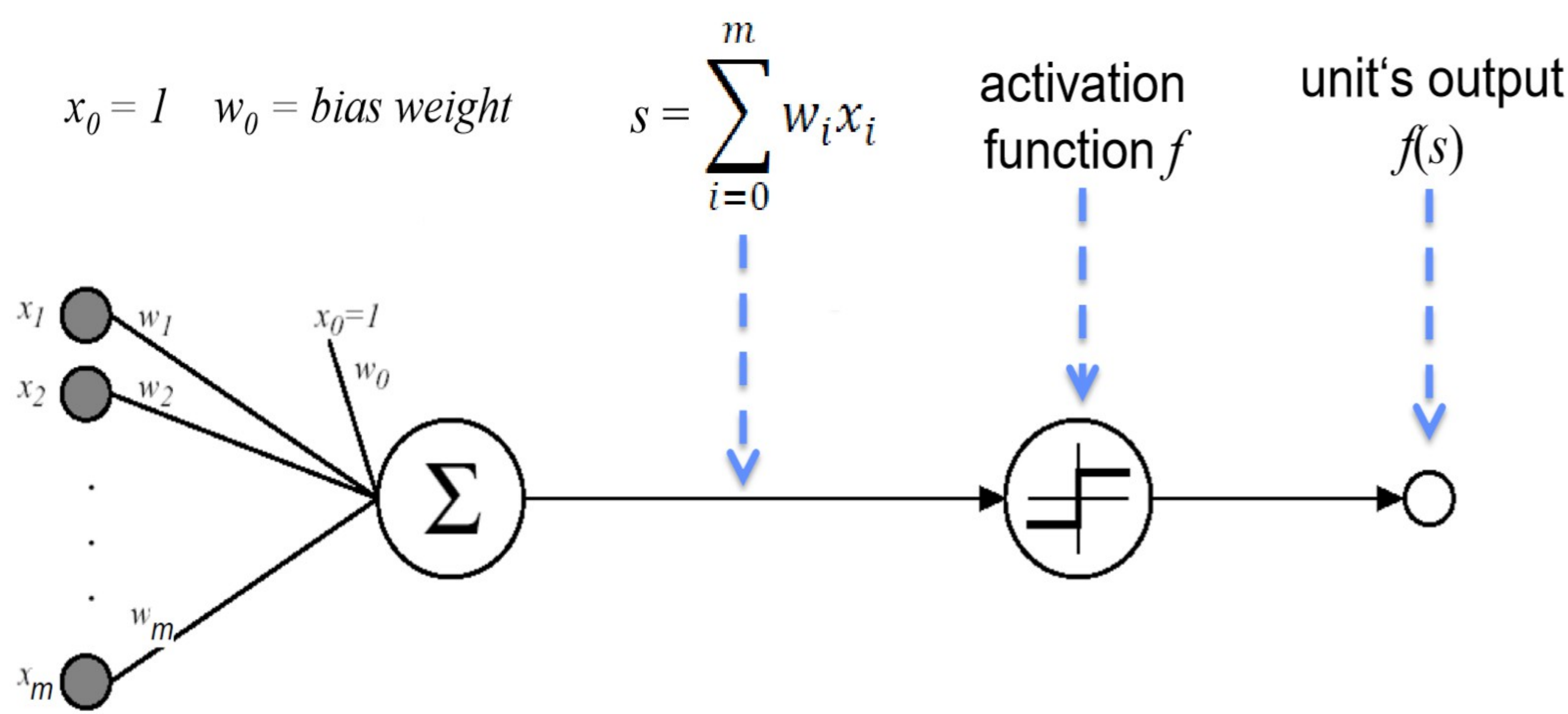
https://commons.wikimedia.org/wiki/File:DonauknieVisegrad.jpg#/media/File:DonauknieVisegrad.jpg

- “No man ever steps in the same river twice, for it’s not the same river and he’s not the same man.” (Heraclitus)
- Exciting development in sensor technology turns almost everything into time series
- This may lead to radically new applications

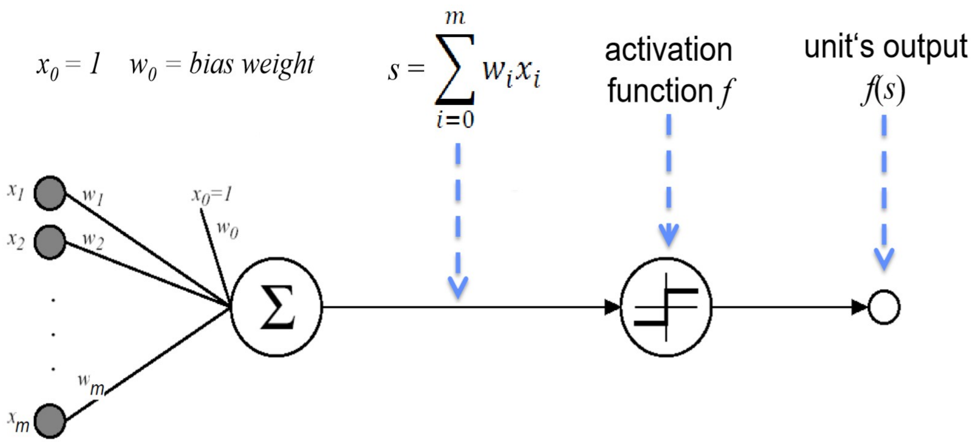
Bonus:
Some More Slides about Deep Learning

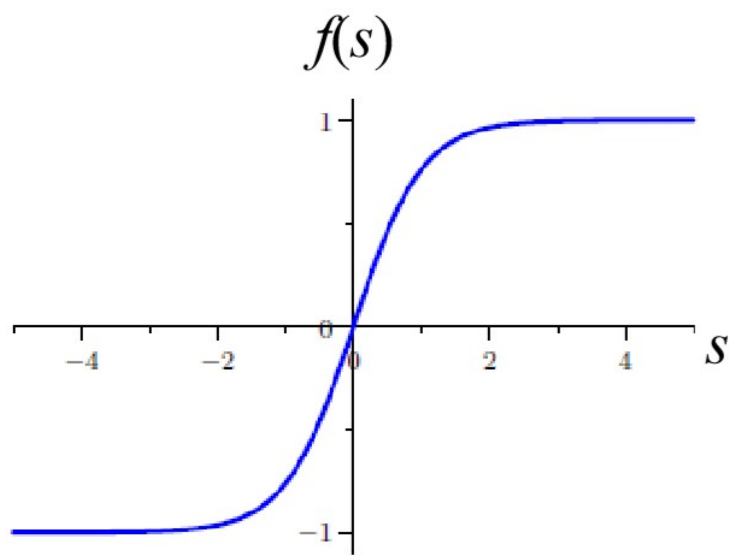
Neural Units

- synaptic summation of inputs, subsequently: activation function f
- $x_1, x_2, ..., x_m$ = inputs of a unit (usually outputs of some other units)
- $w_1, w_2, ..., w_m$ = weights of $x_1, x_2, ..., x_m$

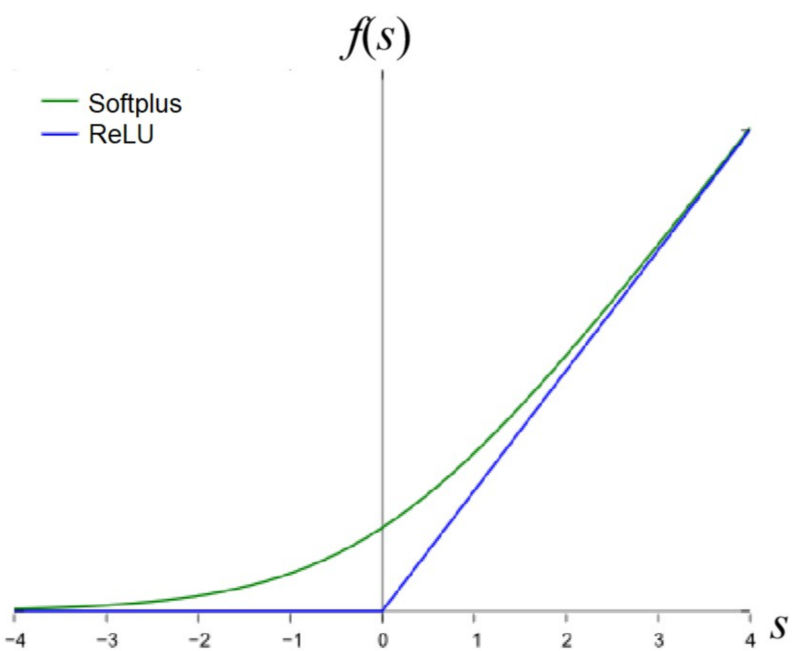


Activation Functions



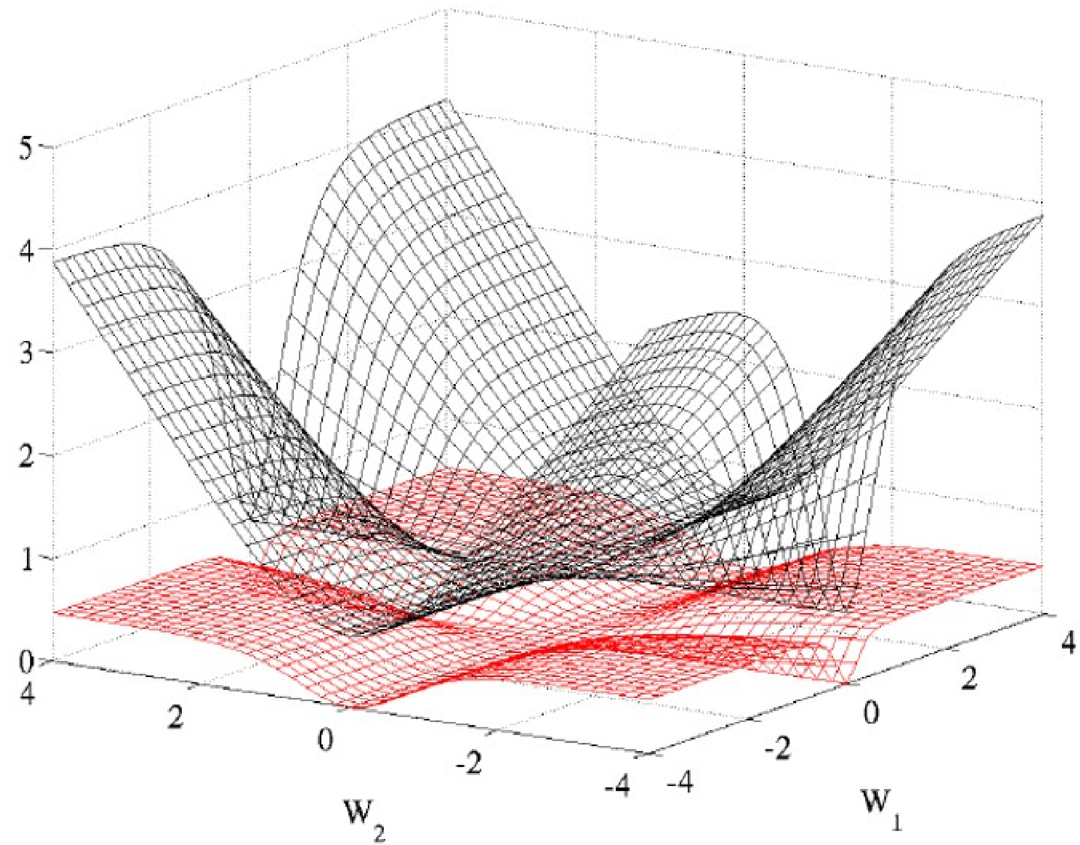


Activation Functions	
Linear	$f(s) = s$
Sigmoid	$f(s) = (1+e^{-s})^{-1}$
Hyperbolic tangent	$f(s) = \tanh(s)$
Softsign	$f(s) = s((1+\lvert s\rvert)^{-1})$
Rectifier Linear Unit (ReLU)	$f(s) = \max(0, s)$
Softplus	$f(s) = \ln(1+e^{s})$
...	...



Loss Function: Quadratic vs. Cross-Entropy

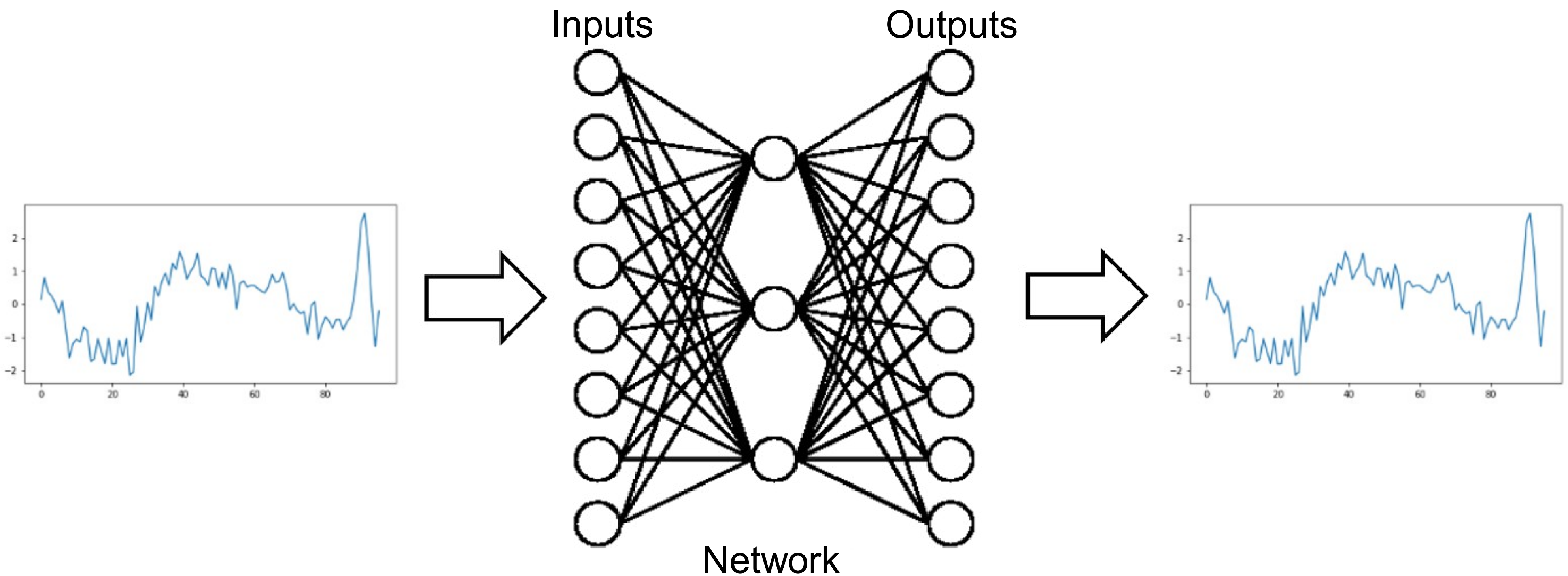
- Cross-entropy: "average length of communicating an event from one distribution with the optimal code for another distribution"
  http://colah.github.io/posts/2015-09-Visual-Information/
- "Cross-entropy (…) allows us to describe how bad it is to believe the predictions of the neural network, given what is actually true."
  https://www.tensorflow.org/tutorials/mnist/tf/
- Black: cross-entropy
  (a.k.a. Conditional log-likelihood, logistic regression cost function)
- Red: quadratic loss



X. Glorot, Y. Bengio: Understanding the difficulty of training deep feedforward neural networks

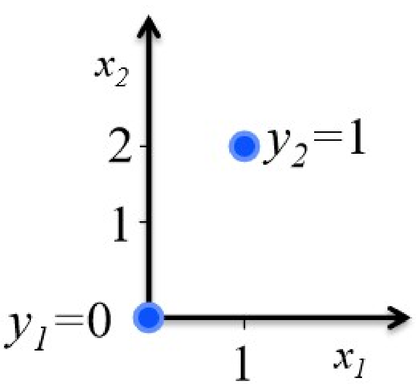
Initialisation of the Weights

- Unsupervised pre-training: autoencoders
- Supervised pre-training:
  - Train a network for a different (but somehow related...) task
  - Re-use some of the weights
    (e.g. weights of the first few convolutional layers)



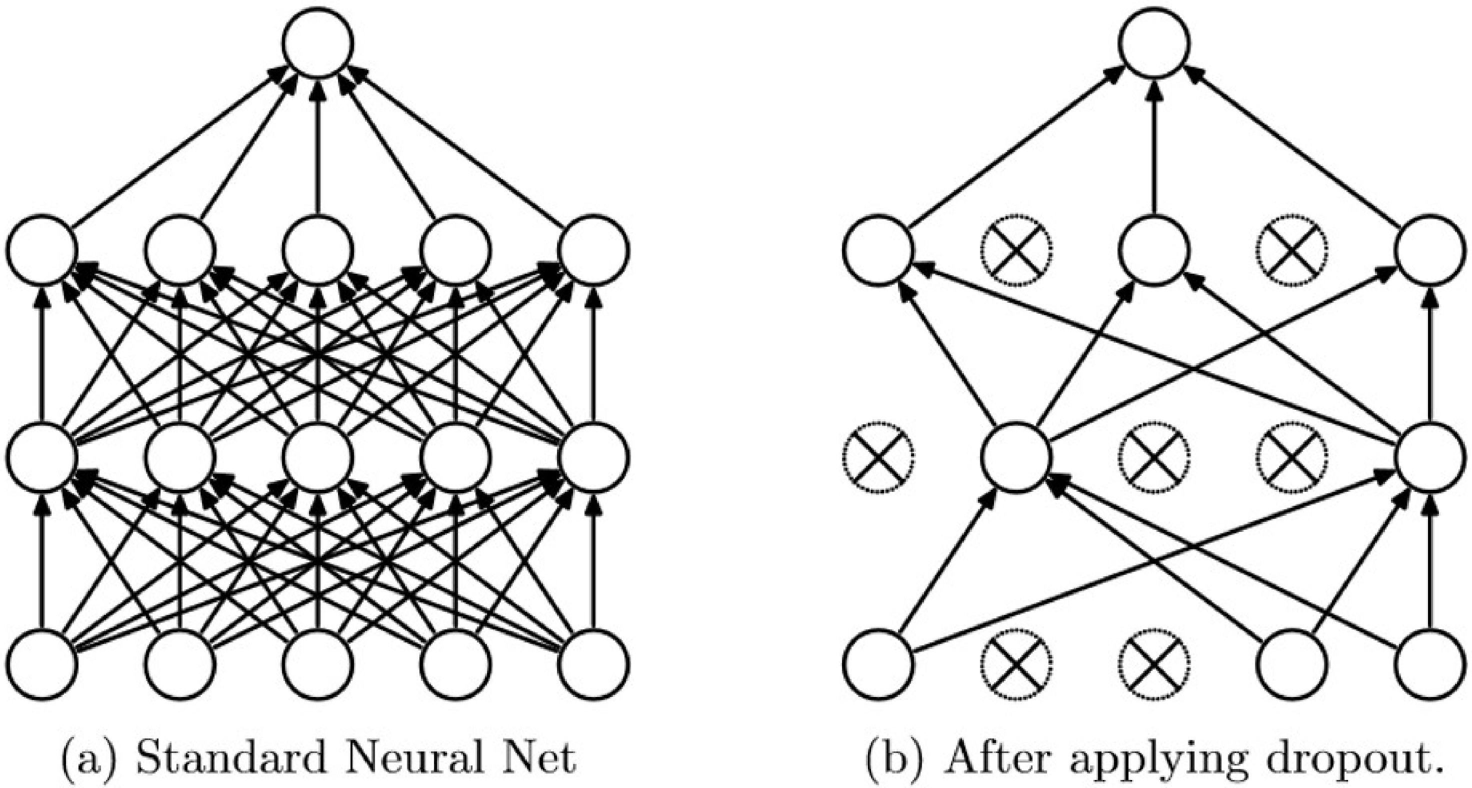
„Sparsity-enforcing“ („sparsity-encouraging“) Regularisation

- In the example below, all the three models below have the same prediction performance (on training data)
- „Traditional“ regularisation: $\sum_{i=0}^{m} w_i^2$
- „Sparsity-enforcing“ regularisation: $\sum_{i=0}^{m} |w_i|$



	$\sum_{i=0}^{m} w_i^2$	$\sum_{i=0}^{m} \lvert w_i \rvert$
$f(x) = 1\,x_1 + 0\,x_2$	$1^2+0^2=1$	$1+0=1$
$f(x) = 0\,x_1 + 0.5\,x_2$	$0^2+0.5^2=0.25$	$0+0.5=0.5$
$f(x) = 0.33\,x_1 + 0.33\,x_2$	$0.33^2+0.33^2=0.22$	$0.33+0.33=0.66$

Dropout



Srivastava et al. (2014): Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research