# Nonlinear independent component analysis: Identifiability, Self-Supervised Learning, and Likelihood

Aapo Hyvärinen

Department of Computer Science
University of Helsinki
Finland

# Abstract

- Short introduction to deep learning

# Abstract

- Short introduction to deep learning
- Importance of unsupervised learning

## Abstract

- Short introduction to deep learning
- Importance of unsupervised learning
- "Disentanglement" methods try to find independent factors

# Abstract

- Short introduction to deep learning
- Importance of unsupervised learning
- "Disentanglement" methods try to find independent factors
- In linear case, independent component analysis (ICA) successful, can we extend to a nonlinear method?

# Abstract

- Short introduction to deep learning
- Importance of unsupervised learning
- "Disentanglement" methods try to find independent factors
- In linear case, independent component analysis (ICA) successful, can we extend to a nonlinear method?
- Problem: Nonlinear ICA not identifiable

# Abstract

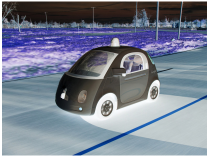
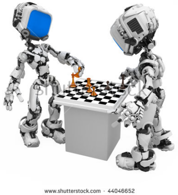
- Short introduction to deep learning
- Importance of unsupervised learning
- "Disentanglement" methods try to find independent factors
- In linear case, independent component analysis (ICA) successful, can we extend to a nonlinear method?
- Problem: Nonlinear ICA not identifiable
- Solution: use temporal structure in time series (two kinds)
  - Temporal dependencies (preferably non-Gaussian)
  - Non-stationarity
  - A more general auxiliary variable framework

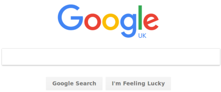
# Abstract

- Short introduction to deep learning
- Importance of unsupervised learning
- "Disentanglement" methods try to find independent factors
- In linear case, independent component analysis (ICA) successful, can we extend to a nonlinear method?
- Problem: Nonlinear ICA not identifiable
- Solution: use temporal structure in time series (two kinds)
  - Temporal dependencies (preferably non-Gaussian)
  - Non-stationarity
  - A more general auxiliary variable framework
- Estimation methods
  - Likelihood: noise-free or with noise term
  - Self-supervised

# Success of Artificial Intelligence

- Autonomous vehicles, machine translation, game playing, search engines, recommendation machine, etc.




- Most modern applications based on deep learning

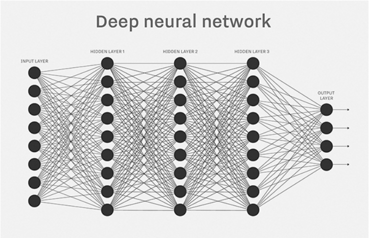
# Neural networks

- Layers of "neurons" repeating linear transformations and simple nonlinearities $f$

$$x_i(L+1) = f(\sum_j w_{ij}(L)x_j(L)), \text{ where } L \text{ is layer} \tag{1}$$

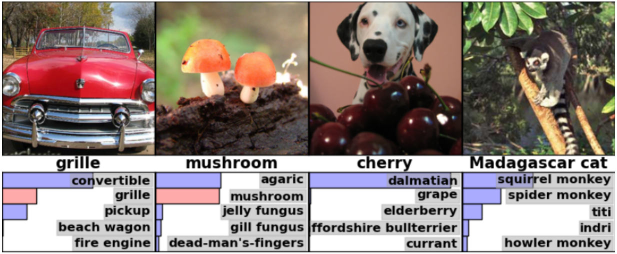
with e.g. $f(x) = \max(0, x)$

- Can approximate "any" nonlinear input-output mappings
- Learning by various statistical objectives (e.g. least-squares)



# Deep learning

- Deep Learning = learning in neural network with many layers
- With enough data, can learn any input-output relationship: image-category / past-present / friends - political views
- Present boom started by Krizhevsky, Sutskever, Hinton, 2012: Superior recognition success of objects in images



# Importance unsupervised learning

- Success stories in deep learning need category labels
  - Is it a cat or a dog? Liked or not liked?

# Importance unsupervised learning

- Success stories in deep learning need category labels
  - Is it a cat or a dog? Liked or not liked?
- Problems:
  - Labels may be difficult obtain
  - Human annotation may be required
  - Labels may not be very informative

# Importance unsupervised learning

- Success stories in deep learning need category labels
  - Is it a cat or a dog? Liked or not liked?
- Problems:
  - Labels may be difficult obtain
  - Human annotation may be required
  - Labels may not be very informative
- Unsupervised learning :
  - we only observe a data vector $\mathbf{x}$, no label or target $y$
  - E.g. photographs with no labels

# Importance unsupervised learning

- Success stories in deep learning need category labels
  - Is it a cat or a dog? Liked or not liked?
- Problems:
  - Labels may be difficult obtain
  - Human annotation may be required
  - Labels may not be very informative
- Unsupervised learning :
  - we only observe a data vector $\mathbf{x}$, no label or target $y$
  - E.g. photographs with no labels
- Very difficult, largely unsolved problem

# ICA as principled unsupervised learning

- Linear independent component analysis (ICA)

$$x_i(k) = \sum_{j=1}^{n} a_{ij} s_j(k) \quad \text{for all } i = 1 \dots n, k = 1 \dots K \tag{2}$$

  - $x_i(k)$ is $i$-th observed signal in sample point $k$ (possibly time)
  - $a_{ij}$ constant parameters describing "mixing"
  - Assuming independent, non-Gaussian latent "sources" $s_j$

# ICA as principled unsupervised learning

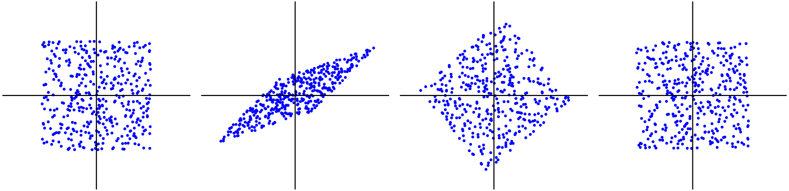
- Linear independent component analysis (ICA)

$$x_i(k) = \sum_{j=1}^{n} a_{ij} s_j(k) \quad \text{for all } i = 1 \ldots n, k = 1 \ldots K \tag{2}$$

  - $x_i(k)$ is $i$-th observed signal in sample point $k$ (possibly time)
  - $a_{ij}$ constant parameters describing "mixing"
  - Assuming independent, non-Gaussian latent "sources" $s_j$
- ICA is identifiable, i.e. well-defined: (Darmois-Skitovich ~1950; Comon, 1994)
  - Observing only $x_i$ we can recover both $a_{ij}$ and $s_j$

# Fundamental difference between ICA and PCA

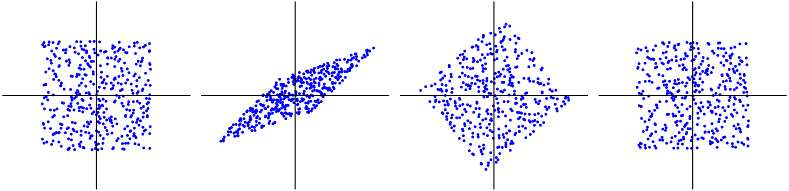
Original comps, observed mixtures, PCA, ICA



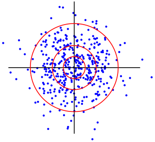
- PCA does not find original coordinates, ICA does!

# Fundamental difference between ICA and PCA

Original comps, observed mixtures, PCA, ICA

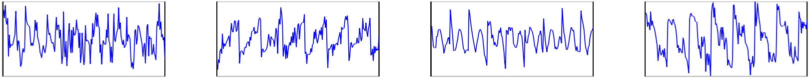


- PCA does not find original coordinates, ICA does!
- PCA, Gaussian factor analysis are *not* identifiable:
  - Any orthogonal rotation is equivalent: $\mathbf{s}' = \mathbf{U}\mathbf{s}$ has same distribution.

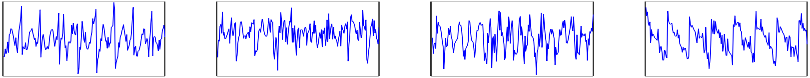


# Identifiability means ICA does blind source separation

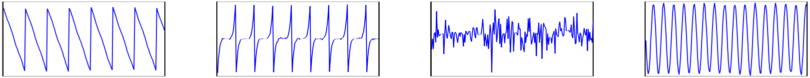
Observed signals:



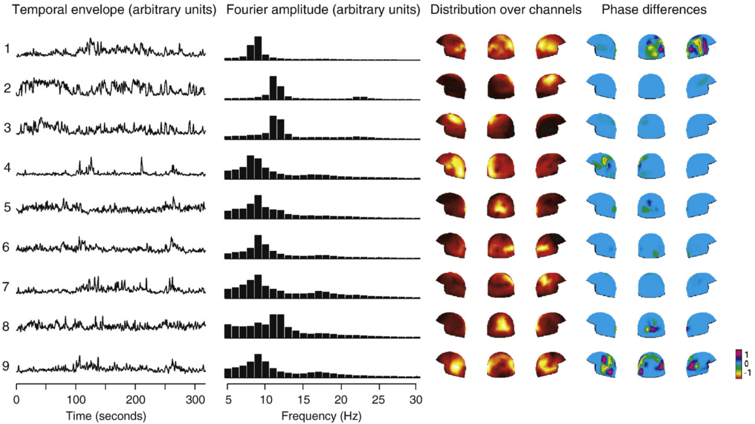
Principal components:



Independent components are original sources:



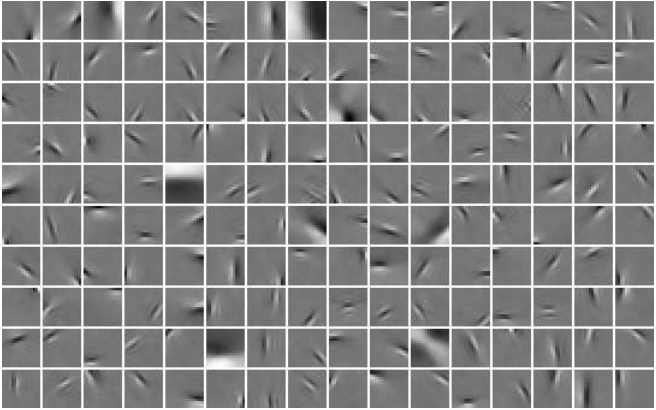
# Example of ICA: Brain source separation




(Hyvärinen, Ramkumar, Parkkonen, Hari, 2010)

# Example of ICA: Image features

(Olshausen and Field, 1996; Bell and Sejnowski, 1997)



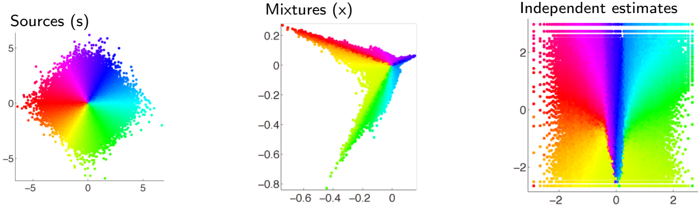
Features similar to wavelets, Gabor functions, simple cells.

# Nonlinear ICA is an unsolved problem

- Extend ICA to nonlinear case to get general disentanglement?
- Unfortunately, "basic" nonlinear ICA is not identifiable:
- If we define nonlinear ICA model for random variables $x_i$ as

$$x_i = f_i(s_1, \ldots, s_n) \quad \text{for all } i = 1 \ldots n \tag{3}$$

we cannot recover original sources (Darmois, 1952; Hyvärinen & Pajunen, 1999)



# Darmois construction

- Darmois (1952) showed impossibility of nonlinear ICA:
- For any $x_1, x_2$, can always construct $y = g(x_1, x_2)$ independent of $x_1$ as

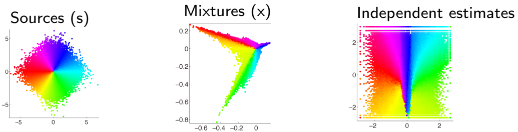
$$g(\xi_1, \xi_2) = P(x_2 < \xi_2 | x_1 = \xi_1) \tag{4}$$

# Darmois construction

- Darmois (1952) showed impossibility of nonlinear ICA:
- For any $x_1, x_2$, can always construct $y = g(x_1, x_2)$ independent of $x_1$ as

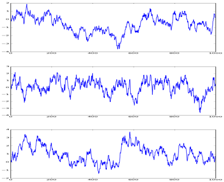
$$g(\xi_1, \xi_2) = P(x_2 < \xi_2 | x_1 = \xi_1) \tag{4}$$

- Independence alone too weak for identifiability:
  We could take $x_1$ as independent component which is absurd

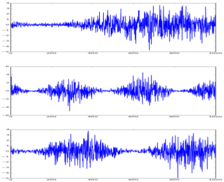


# Temporal structure helps in nonlinear ICA

- Theory above considered i.i.d. sampled random variables
- What if we have time series? with specific temporal structure?



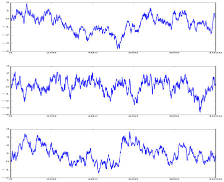
Autocorrelations
(Harmeling et al 2003)



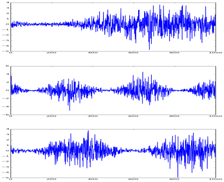
Nonstationarity
(Hyvärinen and Morioka, NIPS2016)

# Temporal structure helps in nonlinear ICA

- Theory above considered i.i.d. sampled random variables
- What if we have time series? with specific temporal structure?



Autocorrelations
(Harmeling et al 2003)



Nonstationarity
(Hyvärinen and Morioka, NIPS2016)

- Identifiability of nonlinear ICA can be proven (rest of this talk)
  (Sprekeler et al, 2014; Hyvärinen and Morioka, NIPS2016 & AISTATS2017):
  Can find original sources!

# Source model I: Temporal dependencies

- Assume mixing model $\mathbf{x_t} = \mathbf{f}(\mathbf{s_t})$ where
  - $\mathbf{x}_t$ observed $n$-dimensional time series
  - $\mathbf{s}_t$ latent $n$-dimensional independent time series
  - $\mathbf{f}$ invertible (bijective) mixing
- Assume $s_t^i$ temporally dependent and non-Gaussian, technically
  - "uniform dependence": pdf of $(s_t^i, s_{t-1}^i)$ not locally factorizable
  - "quasi-Gaussianity" $\approx$ not Gaussian or pointwise transformed
- E.g., non-Gaussian AR model with non-quadratic $G$:

$$\log p(s_t^i | s_{t-1}^i) = G(s_t^i - \rho s_{t-1}^i)$$

# Source model I: Temporal dependencies

- Assume mixing model $\mathbf{x_t} = \mathbf{f}(\mathbf{s_t})$ where
  - $\mathbf{x}_t$ observed $n$-dimensional time series
  - $\mathbf{s}_t$ latent $n$-dimensional independent time series
  - $\mathbf{f}$ invertible (bijective) mixing
- Assume $s_t^i$ temporally dependent and non-Gaussian, technically
  - "uniform dependence": pdf of $(s_t^i, s_{t-1}^i)$ not locally factorizable
  - "quasi-Gaussianity" $\approx$ not Gaussian or pointwise transformed
- E.g., non-Gaussian AR model with non-quadratic $G$:

$$\log p(s_t^i | s_{t-1}^i) = G(s_t^i - \rho s_{t-1}^i)$$

- We prove identifiability (Hyvärinen and Morioka, AISTATS2017) see also (Oberhauser and Schell, Arxiv 2021)
- Why would this work? Impose independence over time lags $\rightarrow$ more constraints $\rightarrow$ unique solution

# Source model II: Non-stationarity

- Assume mixing model $\mathbf{x_t} = \mathbf{f}(\mathbf{s_t})$ as above
- Assume piece-wise stationary source model based on exponential family and time segments $\tau$:
  $\log p_\tau(s_t^i) = q_{i,0}(s_t^i) + \sum_{v=1}^{V} \lambda_{i,v}(\tau) q_{i,v}(s_t^i) - \log Z$
  (assumed 1st-order from now on)
- Assume sufficient non-stationarity: Matrix $\mathbf{L}$ with
  $[\mathbf{L}]_{\tau,i} = \lambda_{i,1}(\tau) - \lambda_{i,1}(1)$ has full column rank $n$.

# Source model II: Non-stationarity

- Assume mixing model $\mathbf{x_t} = \mathbf{f}(\mathbf{s_t})$ as above
- Assume piece-wise stationary source model based on exponential family and time segments $\tau$:
  $\log p_\tau(s_t^i) = q_{i,0}(s_t^i) + \sum_{v=1}^{V} \lambda_{i,v}(\tau) q_{i,v}(s_t^i) - \log Z$
  (assumed 1st-order from now on)
- Assume sufficient non-stationarity: Matrix $\mathbf{L}$ with
  $[\mathbf{L}]_{\tau,i} = \lambda_{i,1}(\tau) - \lambda_{i,1}(1)$ has full column rank $n$.
- We prove (partial) identifiability : identifiable up to pointwise + linear transforms (Hyvärinen and Morioka NIPS2016)

$$[q_1(\hat{s}_t^1), \ldots, q_n(\hat{s}_t^n)]^T = \mathbf{A}[q_1(s_t^1), \ldots, q_n(s_t^n)]^T \qquad (5)$$

  for some unknown matrix $\mathbf{A}$ and *pointwise* nonlinearities $q_i$
- Why would this work? Impose independence at every segment $\rightarrow$ more constraints $\rightarrow$ unique solution

# Noise-free likelihood I: Formulation

- Noise-free likelihood for invertible mixing

$$\mathbf{x_t} = \mathbf{f}(\mathbf{s_t}), \tag{6}$$

  where again
  - $\mathbf{x}_t$ observed $n$-dimensional time series
  - $\mathbf{s}_t$ latent $n$-dimensional "independent components"
  - $\mathbf{f}$ invertible (bijective) mixing
- Log-likelihood $\log L(\mathbf{x}_1, \ldots, \mathbf{x}_\mathrm{T})$ easy to formulate with $\mathbf{g} = \mathbf{f}^{-1}$ and $\mathbf{Jg}$ its Jacobian:

$$\log L = \sum_i \log p_i(g_i(\mathbf{x}_1), \ldots, g_i(\mathbf{x}_T)) + \sum_t \log |\det \mathbf{Jg}(\mathbf{x}_t)|$$

- Preceding slides give possible $p_i$: Just your time series model
- Computationally, can be very difficult: Jacobian of neural net?

# Noise-free likelihood I: Optimization

- Modelling $\mathbf{f}^{-1} = \mathbf{g}$ with a neural network $\mathbf{g}_\theta$, how to optimize

$$\log |\det \mathbf{J}\mathbf{g}_\theta(\mathbf{x})| \tag{7}$$

for a single observation $\mathbf{x}$ ?

- The difficulty comes from need for inversion:

$$\nabla_{\mathbf{W}} \log |\det \mathbf{W}| = (\mathbf{W}^{-1})^T \tag{8}$$

as well as combining this with backpropagation (chain rule).

# Noise-free likelihood I: Optimization

- Modelling $\mathbf{f}^{-1} = \mathbf{g}$ with a neural network $\mathbf{g}_{\boldsymbol{\theta}}$, how to optimize

$$\log |\det \mathbf{J}\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{x})| \tag{7}$$

  for a single observation $\mathbf{x}$ ?
- The difficulty comes from need for inversion:

$$\nabla_{\mathbf{W}} \log |\det \mathbf{W}| = (\mathbf{W}^{-1})^T \tag{8}$$

  as well as combining this with backpropagation (chain rule).
- Solution: relative gradient (Gresele et al, NeurIPS2020)
- Consider *multiplicative* perturbation with any objective $h$:

$$h((\mathbf{I}+\epsilon)\mathbf{W}) - h(\mathbf{W}) = \langle \nabla h(\mathbf{W})\mathbf{W}^\top, \epsilon \rangle + o(\mathbf{W}) \tag{9}$$

- Steepest descent method: $\mathbf{W} \leftarrow \mathbf{W} + \mu \nabla h(\mathbf{W})\mathbf{W}^\top\mathbf{W}$
- Inverses disappear: $(\mathbf{W}^{-1})^T\mathbf{W}^\top\mathbf{W} = \mathbf{W} \rightarrow$ Easy to compute!

# Noisy Likelihood I: Background

- Deep Latent Variable Models: Widely-used, general framework with observed data vector $\mathbf{x}$ and latent $\mathbf{s}$:

$$p(\mathbf{x}, \mathbf{s}) = p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})p(\mathbf{s}), \quad p(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{s})d\mathbf{s}$$

where $\boldsymbol{\theta}$ is a vector of parameters, e.g. in a neural network

# Noisy Likelihood I: Background

- Deep Latent Variable Models: Widely-used, general framework with observed data vector $\mathbf{x}$ and latent $\mathbf{s}$:

$$p(\mathbf{x}, \mathbf{s}) = p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})p(\mathbf{s}), \quad p(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{s})d\mathbf{s}$$

  where $\boldsymbol{\theta}$ is a vector of parameters, e.g. in a neural network
- In variational autoencoders (VAE) :
  - Define prior $p(\mathbf{s})$ so that $\mathbf{s}$ white Gaussian (thus $s_i$ all independent)
  - Define posterior $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})$ as $\mathbf{x} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{s}) + \mathbf{n}$, with $\mathbf{n}$ Gaussian noise

# Noisy Likelihood I: Background

- Deep Latent Variable Models: Widely-used, general framework with observed data vector $\mathbf{x}$ and latent $\mathbf{s}$:

$$p(\mathbf{x},\mathbf{s}) = p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})p(\mathbf{s}), \quad p(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{s})d\mathbf{s}$$

  where $\boldsymbol{\theta}$ is a vector of parameters, e.g. in a neural network
- In variational autoencoders (VAE) :
  - Define prior $p(\mathbf{s})$ so that $\mathbf{s}$ white Gaussian (thus $s_i$ all independent)
  - Define posterior $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})$ as $\mathbf{x} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{s}) + \mathbf{n}$, with $\mathbf{n}$ Gaussian noise
- VAE implements "black-box" variational inference: approximate maximum likelihood

# Noisy likelihood II: Identifiable Variational Autoencoder

- But VAE gives noisy version of Nonlinear ICA!

$$\mathbf{x} = \mathbf{f}_\theta(\mathbf{s}) + \mathbf{n} \tag{10}$$

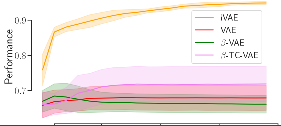
  with Gaussian, independent $s_i$
- Not identifiable: $\mathbf{x}$ observed i.i.d. and even Gaussian
- Intuitively, original VAE is more like PCA (instead of ICA)

# Noisy likelihood II: Identifiable Variational Autoencoder

- But VAE gives noisy version of Nonlinear ICA!

$$\mathbf{x} = \mathbf{f}_\theta(\mathbf{s}) + \mathbf{n} \tag{10}$$

  with Gaussian, independent $s_i$
- Not identifiable: $\mathbf{x}$ observed i.i.d. and even Gaussian
- Intuitively, original VAE is more like PCA (instead of ICA)
- We propose identifiable version: iVAE (Khemakhem et al, AISTATS2020)
- We generalize theory of preceding slides
  - Assume there is some "auxiliary" observed variable $\mathbf{u}$
  - $\mathbf{u}$ can be audio for video, time index, history, etc.
  - Assume: $s_i$ conditionally independent given $\mathbf{u}$
  - Temporal structure as special case



# Heuristic approach: "Self-supervised" learning

- Supervised learning: we have
  - "input" $\mathbf{x}$, "output" $\mathbf{y}$

# Heuristic approach: "Self-supervised" learning

- Supervised learning: we have
  - "input" $\mathbf{x}$, "output" $\mathbf{y}$
- **Un**supervised learning: we have
  - only "input" $\mathbf{x}$

# Heuristic approach: "Self-supervised" learning

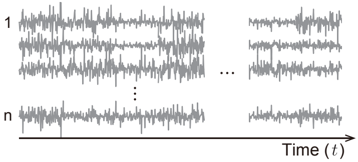
- Supervised learning: we have
  - "input" $\mathbf{x}$, "output" $\mathbf{y}$
- **Un**supervised learning: we have
  - only "input" $\mathbf{x}$
- **Self-supervised** learning: we have
  - only "input" $\mathbf{x}$ to begin with
  - *but we invent* $\mathbf{y}$ somehow, e.g. by creating corrupted data, and use supervised algorithms
- E.g. *Noise-contrastive estimation*: Train a neural network to discriminate between $\mathbf{x}$ and artificially generated noise (Gutmann and Hyvärinen, 2010)

# Heuristic approach: "Self-supervised" learning

- Supervised learning: we have
  - "input" $\mathbf{x}$, "output" $\mathbf{y}$
- **Un**supervised learning: we have
  - only "input" $\mathbf{x}$
- **Self-supervised** learning: we have
  - only "input" $\mathbf{x}$ to begin with
  - *but we invent* $\mathbf{y}$ somehow, e.g. by creating corrupted data, and use supervised algorithms
- E.g. *Noise-contrastive estimation*: Train a neural network to discriminate between $\mathbf{x}$ and artificially generated noise (Gutmann and Hyvärinen, 2010)
- Our original approach to nonlinear ICA
- Easy to implement, since may use well-known algorithms

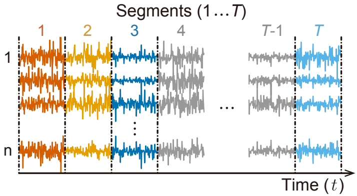
# Time-contrastive learning: (Hyvärinen and Morioka 2016)

- Observe $n$-dim time series $\mathbf{x}(t)$



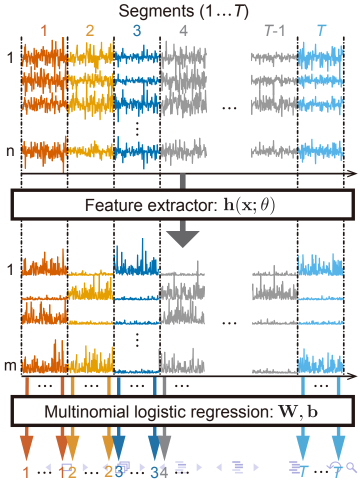
# Time-contrastive learning: (Hyvärinen and Morioka 2016)

- Observe $n$-dim time series $\mathbf{x}(t)$
- Divide $\mathbf{x}(t)$ into $T$ segments (e.g. bins with equal sizes)



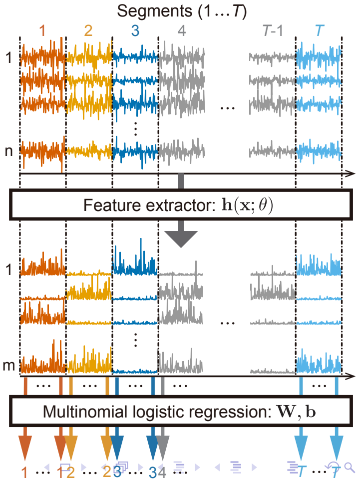
# Time-contrastive learning: (Hyvärinen and Morioka 2016)

- Observe $n$-dim time series $\mathbf{x}(t)$
- Divide $\mathbf{x}(t)$ into $T$ segments (e.g. bins with equal sizes)
- Train MLP to tell which segment *a single data point* comes from
  - Number of classes is $T$, labels given by index of segment
  - Multinomial logistic regression



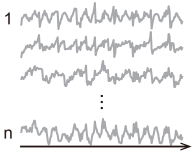
# Time-contrastive learning: (Hyvärinen and Morioka 2016)

- Observe $n$-dim time series $\mathbf{x}(t)$
- Divide $\mathbf{x}(t)$ into $T$ segments (e.g. bins with equal sizes)
- Train MLP to tell which segment *a single data point* comes from
  - Number of classes is $T$, labels given by index of segment
  - Multinomial logistic regression
- In hidden layer $\mathbf{h}$, NN should learn to represent nonstationarity (= differences between segments)
- Nonlinear ICA for nonstationary data!



# Permutation-contrastive learning (Hyvärinen and Morioka 2017)

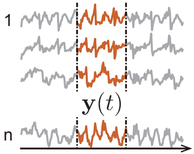
- How about stationary time series?



# Permutation-contrastive learning (Hyvärinen and Morioka 2017)

- How about stationary time series?
- Take short time windows as new data

$$\mathbf{y}(t) = \big(\mathbf{x}(t), \mathbf{x}(t-1)\big)$$



# Permutation-contrastive learning (Hyvärinen and Morioka 2017)

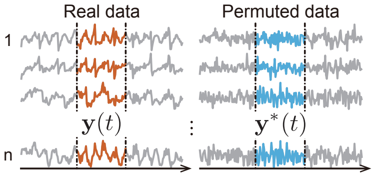
- How about stationary time series?
- Take short time windows as new data

$$\mathbf{y}(t) = (\mathbf{x}(t), \mathbf{x}(t-1))$$

- Create randomly time-permuted data

$$\mathbf{y}^*(t) = (\mathbf{x}(t), \mathbf{x}(t^*))$$

with $t^*$ a random time point.



# Permutation-contrastive learning (Hyvärinen and Morioka 2017)

- How about stationary time series?
- Take short time windows as new data

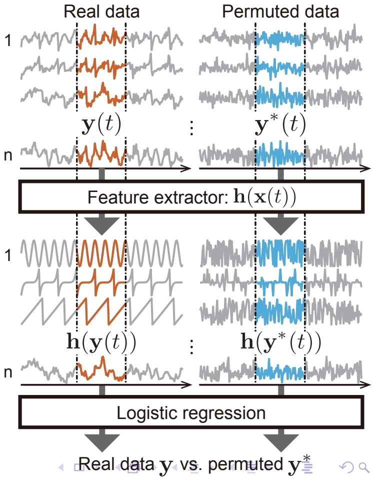
$$\mathbf{y}(t) = (\mathbf{x}(t), \mathbf{x}(t-1))$$

- Create randomly time-permuted data

$$\mathbf{y}^*(t) = (\mathbf{x}(t), \mathbf{x}(t^*))$$

with $t^*$ a random time point.

- Train NN to discriminate $\mathbf{y}$ from $\mathbf{y}^*$
- Performs Nonlinear ICA for temporally dependent components!



# Connection between self-supervised learning and likelihood

- Above, we solved classification problem by logistic regression
- Then, regression function will converge towards

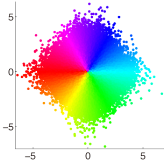
$$r(\mathbf{x}) = \log p_1(\mathbf{x}) - \log p_2(\mathbf{x}) \tag{11}$$

  where $p_1, p_2$ are the pdf's in the two classes
- E.g. Noise-contrastive estimation: Set $p_1$ observed data, $p_2$ Gaussian noise $\rightarrow$ $r$ will estimate data log-pdf up to a known additive function (Gutmann and Hyvärinen, 2010)
- Not that different from likelihood!?
- Finite-sample properties certainly different

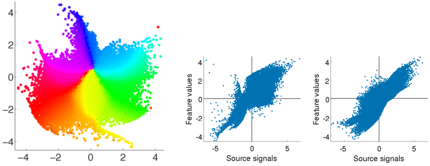
# Illustration of demixing capability by PCL

- Non-Gaussian AR model for sources
  $\log p(s(t)|s(t-1)) = -|s(t) - \rho s(t-1)|$

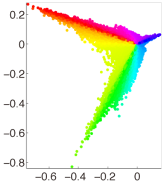
Sources (s)



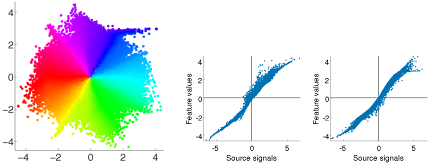
Estimates by kTDSEP (Harmeling et al 2003)




Mixtures (x)



Estimates by our PCL



# Recent extensions

- Above, one model for nonstationary data (TCL), one for temporal dependencies (PCL): How to combine?
  $\rightarrow$ Nonstationary innovations (Morioka et al, AISTATS2021)

# Recent extensions

- Above, one model for nonstationary data (TCL), one for temporal dependencies (PCL): How to combine?
  $\rightarrow$ Nonstationary innovations (Morioka et al, AISTATS2021)
- Above, predefined (manual) segmentation in TCL:
  Can segmentation be learned?
  $\rightarrow$ Combine TCL with HMM (Hälvä & Hyvärinen, UAI2020)

# Recent extensions

- Above, one model for nonstationary data (TCL), one for temporal dependencies (PCL): How to combine?
  → Nonstationary innovations (Morioka et al, AISTATS2021)
- Above, predefined (manual) segmentation in TCL:
  Can segmentation be learned?
  → Combine TCL with HMM (Hälvä & Hyvärinen, UAI2020)
- Independence can be seen as a restriction (at least by some)
  → Generalize to dependent components using energy-based modelling (Khemakhem et al, NeurIPS2020)

# Recent extensions

- Above, one model for nonstationary data (TCL), one for temporal dependencies (PCL): How to combine?
  $\rightarrow$ Nonstationary innovations (Morioka et al, AISTATS2021)
- Above, predefined (manual) segmentation in TCL:
  Can segmentation be learned?
  $\rightarrow$ Combine TCL with HMM (Hälvä & Hyvärinen, UAI2020)
- Independence can be seen as a restriction (at least by some)
  $\rightarrow$ Generalize to dependent components using energy-based modelling (Khemakhem et al, NeurIPS2020)
- (Mentioned earlier: Instead of time structure, some other conditioning variable)

# Conclusion

- Typical deep learning needs class labels, or some targets

# Conclusion

- Typical deep learning needs class labels, or some targets
- If no class labels: **un**supervised learning

# Conclusion

- Typical deep learning needs class labels, or some targets
- If no class labels: **un**supervised learning
- Independent component analysis is a principled approach
  - can be made nonlinear

# Conclusion

- Typical deep learning needs class labels, or some targets
- If no class labels: **un**supervised learning
- Independent component analysis is a principled approach
  - can be made nonlinear
- Identifiable : Can recover components that actually created the data (unlike PCA, VAE etc)

# Conclusion

- Typical deep learning needs class labels, or some targets
- If no class labels: **un**supervised learning
- Independent component analysis is a principled approach
  - can be made nonlinear
- Identifiable : Can recover components that actually created the data (unlike PCA, VAE etc)
- Special assumptions needed for identifiability, one of:
  - Nonstationarity ("time-contrastive learning")
  - Temporal dependencies ("permutation-contrastive learning")
  - Existence of auxiliary (conditioning) variable (e.g. "iVAE")

# Conclusion

- Typical deep learning needs class labels, or some targets
- If no class labels: **un**supervised learning
- Independent component analysis is a principled approach
  - can be made nonlinear
- Identifiable : Can recover components that actually created the data (unlike PCA, VAE etc)
- Special assumptions needed for identifiability, one of:
  - Nonstationarity ("time-contrastive learning")
  - Temporal dependencies ("permutation-contrastive learning")
  - Existence of auxiliary (conditioning) variable (e.g. "iVAE")
- Maximum likelihood estimation possible: noisy or noise-free
- Self-supervised methods are easy to implement

# Conclusion

- Typical deep learning needs class labels, or some targets
- If no class labels: **un**supervised learning
- Independent component analysis is a principled approach
  - can be made nonlinear
- Identifiable : Can recover components that actually created the data (unlike PCA, VAE etc)
- Special assumptions needed for identifiability, one of:
  - Nonstationarity ("time-contrastive learning")
  - Temporal dependencies ("permutation-contrastive learning")
  - Existence of auxiliary (conditioning) variable (e.g. "iVAE")
- Maximum likelihood estimation possible: noisy or noise-free
- Self-supervised methods are easy to implement
- Principled framework for "disentanglement"