Deep Generative Models III

Autoregressive Models – Part I

Machine Perception

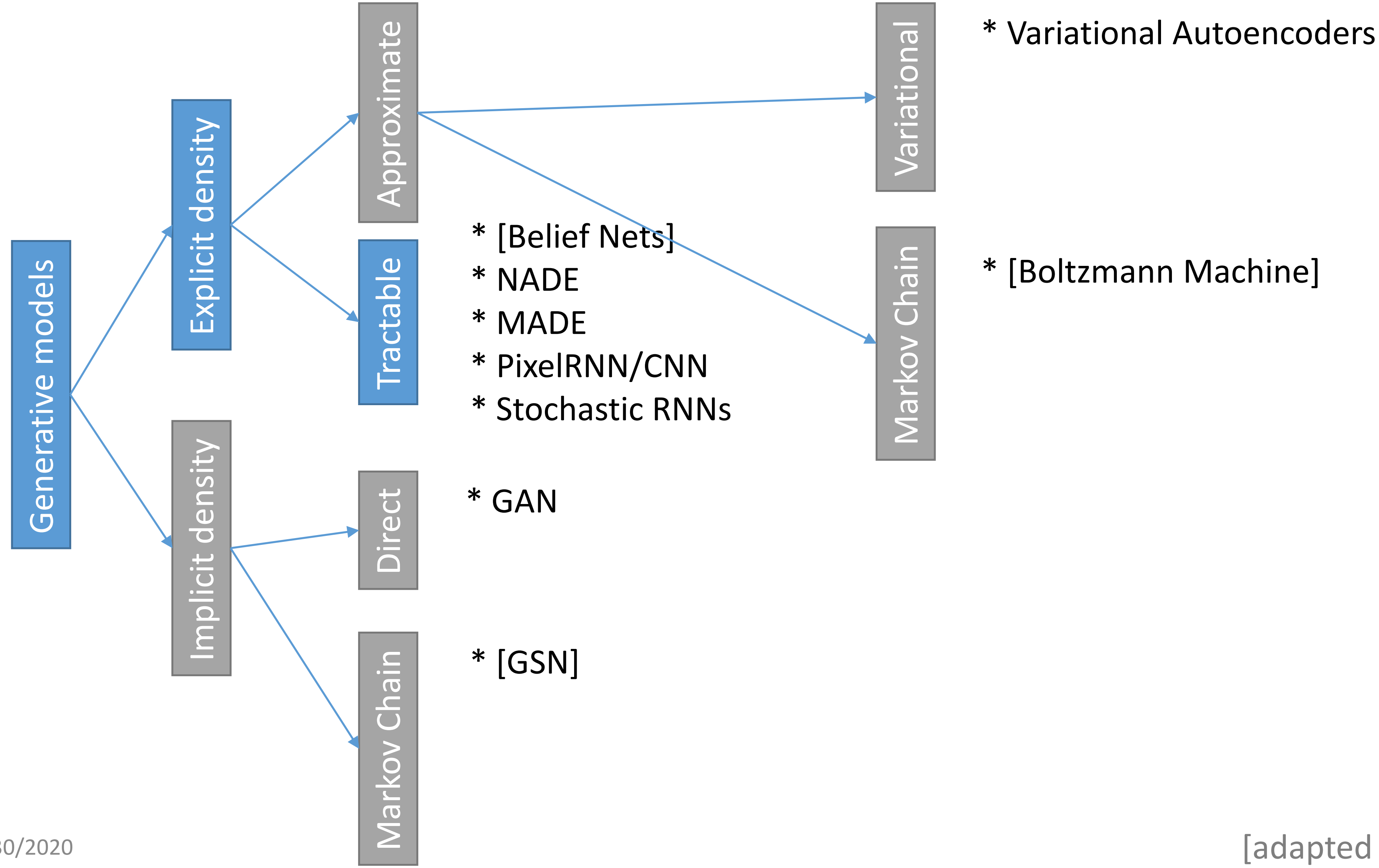
Otmar Hilliges

30 April 2020

Advanced Interactive Technologies

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Taxonomy of Generative Models



[adapted from Ian Goodfellow]

Auto-regressive property

A regression model, such as linear regression, models an output value based on a linear combination of input values. For example:

$$\hat{y} = b_0 + b_1 x_1$$

This technique can be used on time series where input variables are taken as observations at previous time steps, called lag variables.

For example, we can predict the value for the next time step (t+1) given the observations at the last two time steps (t-1 and t-2). As a regression model, this would look as follows:

$$x_t = b_0 + b_1 x_{t-1} + b_2 x_{t-2}$$

Because the regression model uses data from the same input variable at previous time steps, it is referred to as an autoregression (regression of self).

Sequence models are generative models

Discriminative models model the conditional distribution $P(Y|X)$

A generative model models the joint distribution $P(X, Y)$ of the observation X and the target Y or $P(X)$ in the unsupervised setting (density estimation)

A sequence model deals with sequential data:

- mapping sequences to scalars (e.g. language models)
- mapping seq. to seq. (e.g. machine translation models)

Sequence models are generative:

- Given seed $x_1, x_2, \dots, x_k$ predict $x_k + 1$.
- Then use $x_2, x_3, \dots, x_{k+1}$ to predict x_{k+2}
- rinse & repeat

Learning the distribution of natural data

$$p(\boldsymbol{x}) = \prod_{k} \prod_{j} \prod_{i} p(\boldsymbol{x}_{i,j,k} | \underline{\boldsymbol{x}_{<}})$$

$\{x_1, \ldots, x_{i-1}\}$

PixelRNN/PixelCNN (Images) [van den Oord, Kalchbrenner, Kavukcuoglu, 2016]

[van den Oord, Kalchbrenner, Vinyals, et al. 2016]

Video Pixel Nets (Videos) [Kalchbrenner, van den Oord, Simonyan, et al. 2016]

ByteNet (Language/Seq2Seq) [Kalchbrenner, Espeholt, Simonyan, et al. 2016]

WaveNet (Audio) [van den Oord, Dielemann, Zen, et al. 2016]

Attempt 1: Autoregressive models – tabular approach

Via the chain rule of probabilities we can factorize the joint distribution over the n-dimensions:

$$p(x) = \prod_{1}^{n} p(x_i | x_1, \ldots, x_{i-1}) = \prod_{1}^{n} p(x_i | \underline{x_{<i}})$$

$1, \ldots, i-1$

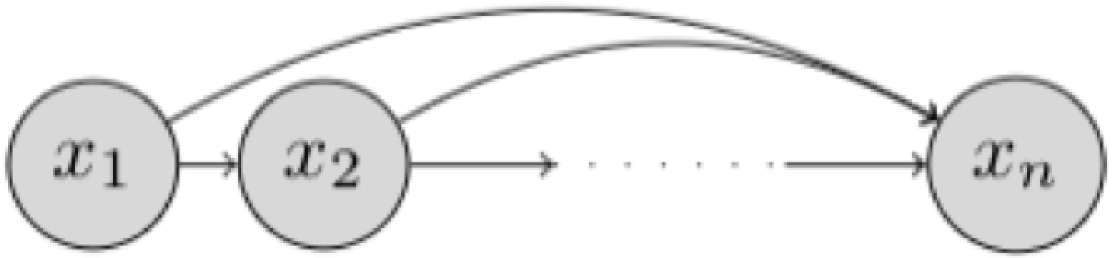
given some ordering of D

\# param:

$2^{n-1} - 1$

↑

binary variables



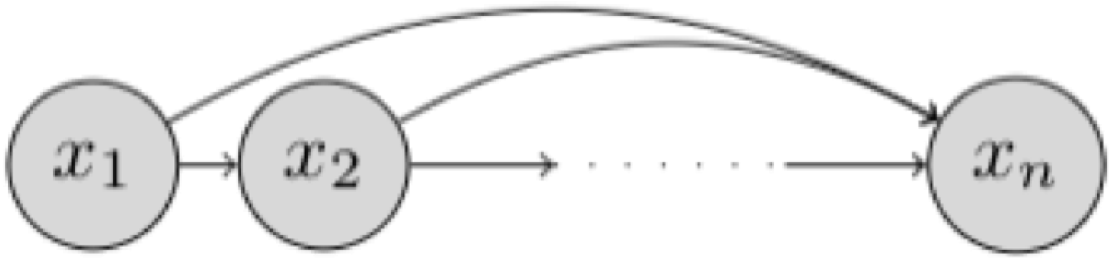
Attempt 2: Autoregressive generative models

Specify conditionals with fixed number of parameters:

Assume $p(x_i|x_{<i})$ to correspond to Bernoulli random variable and learn to map previous inputs to the mean: 期望值 p

$$p_{\theta_i}(x_i|x_{<i}) = Ber(f(x_1, .. x_{i-1}))$$

$$\sum_{i=1}^{n} |\theta_i| = O(n^2)$$



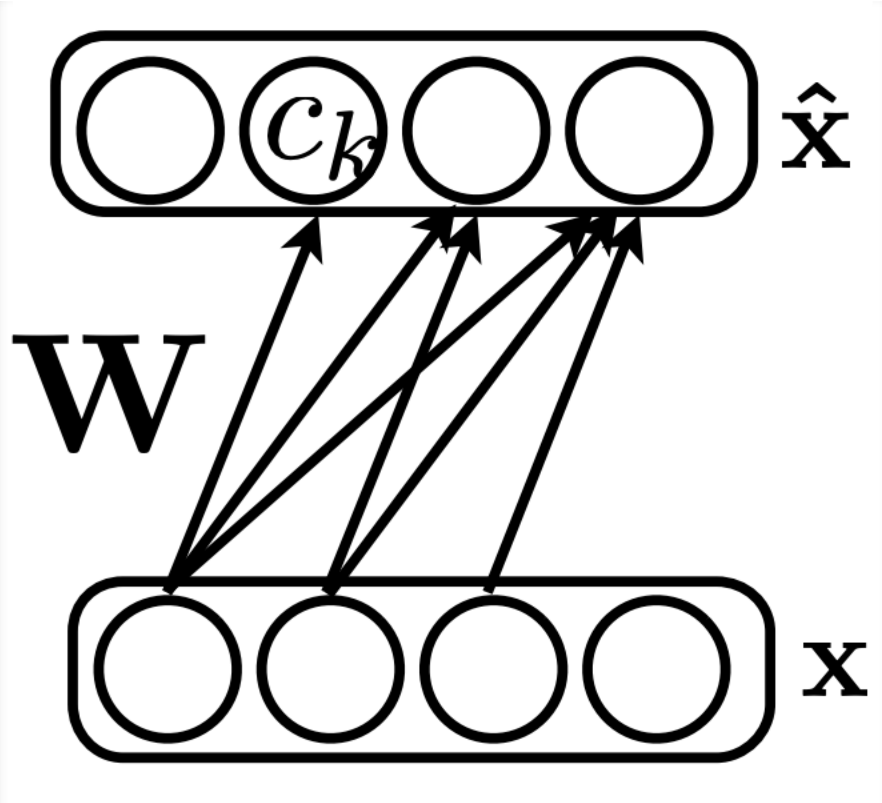
Fully Visible Belief Networks

$\hat{x}_i = p(x_i = 1 | x_{<i})$

Modelled via logistic regression:

$f_i(x_1, x_2, \ldots, x_{i-1}) =$
$\sigma\left(\alpha_0^i + \alpha_1^i x_1 + \cdots + \alpha_i^i - 1 x_{i-1}\right)$

$\# \sum_i^n |\alpha_i| = O(n^2)$



Neural Autoregressive Density Estimator (NADE)

autoencoder-like neural network to learn
$p(x_i = 1|x_{<i})$: ← binary data

first i-1 col

$$\boldsymbol{h}_i = \sigma(\boldsymbol{b} + \boldsymbol{W}_{\boxed{\cdot,<i}}\,\boldsymbol{x}_{<i})$$

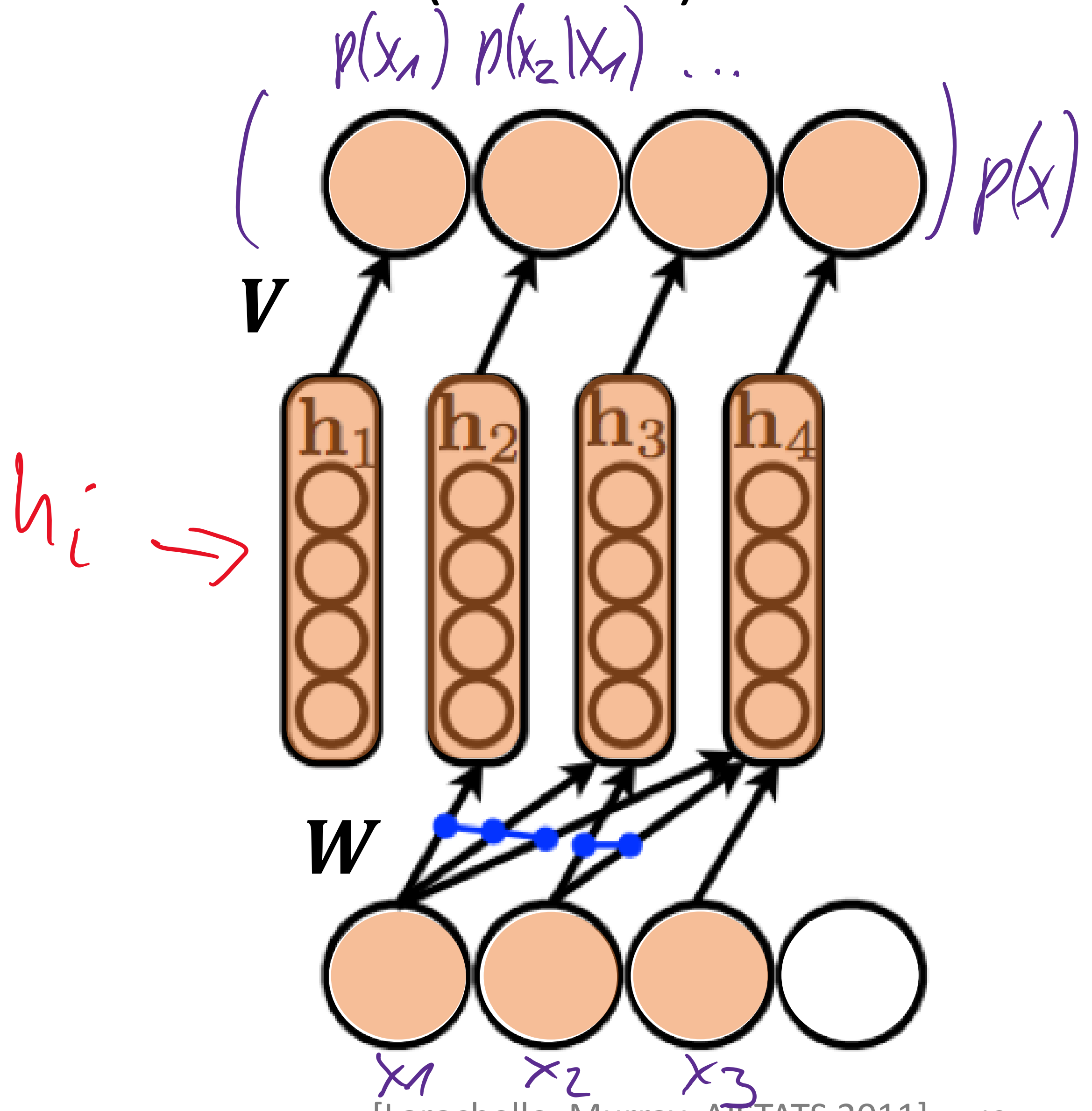
$$\hat{x}_i = \sigma\left(c_i + \boldsymbol{V}_{\boxed{i,\cdot}}\,\mathbf{h}^{(i)}\right)$$

ith row

each conditional is modeled by the same neural network

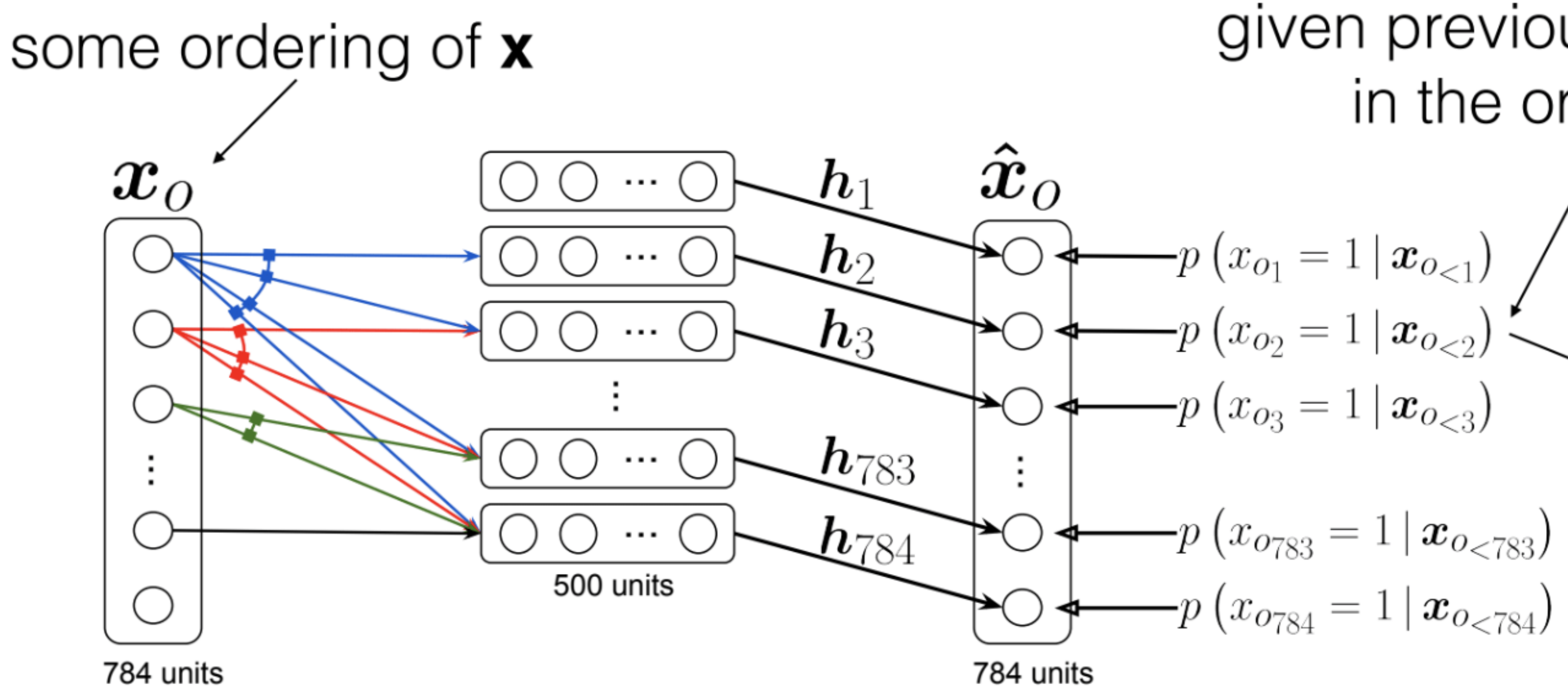
We can leverage the fact that:

$$(\boldsymbol{b} + \boldsymbol{W}_{\cdot,<i+1}\,\boldsymbol{x}_{<i+1}) - (\boldsymbol{b} + \boldsymbol{W}_{\cdot,<i}\,\boldsymbol{x}_{<i}) = \boldsymbol{W}_{\cdot,i+1}\,\boldsymbol{x}_{i+1}$$



[Larochelle, Murray. AISTATS 2011]

NADE schematically



NADE is for binary data

$$p(x_{o_d} = 1 \mid x_{o_{<d}}) = \sigma(V_{o_d,:}\mathbf{h}_d + b_{o_d})$$

$$\mathbf{h}_d = \sigma(W_{:,o_{<d}}\mathbf{x}_{o_{<d}} + \mathbf{c})$$

[image source: A. Courville. UdM]

variants of NADE could work on non-binary data

NADE – Training

training by maximizing the average log-likelihood:

$$\frac{1}{T}\sum_{t=1}^{T} \log\left(p(x^t)\right) = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{D} log\, p\left(x_i^{(t)} \middle| x_{<i}^{(t)}\right)$$

Advantages

- efficient: computations are in O(TD)
- could make use of second-order optimizers
- easily extendable to other types of observations (reals, multinomials)

Paper explores different orderings of inputs

- random order works fine!

Next Week

Wrap-up deep generative modelling

Reinforcement Learning Bootcamp