Deep approaches to semi-supervised learning

Anders Boesen Lindbo Larsen

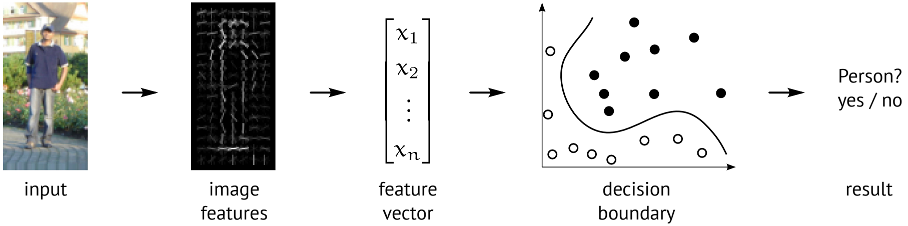
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Outline

- Deep learning introduction
- Unsupervised pretraining with autoencoders
- Ladder networks (semi-supervised autoencoders)
- Generative adversarial networks
- Scaling up autoencoders to complex data distributions (images)

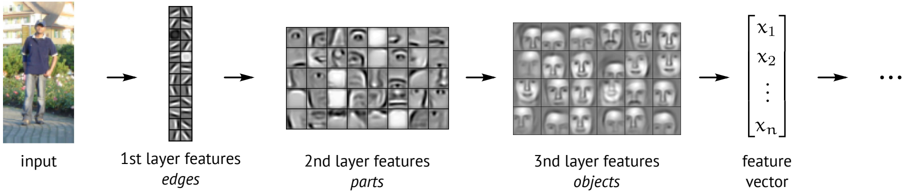
'Shallow' computer vision

Hand-engineer a *clever* representation of the input image.

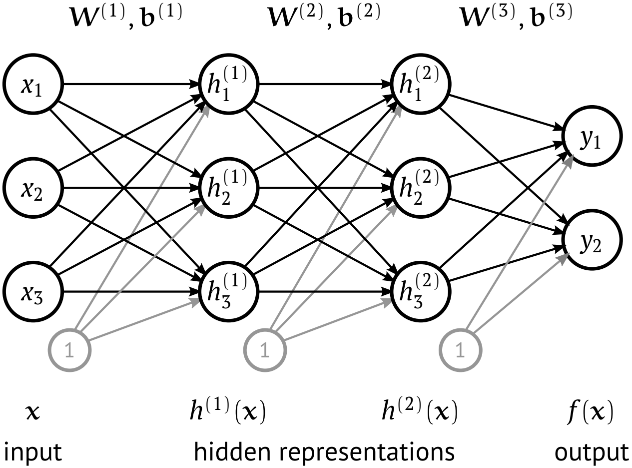


Deep feature learning

Learn a hierarchical representation of the input.



Neural networks 101



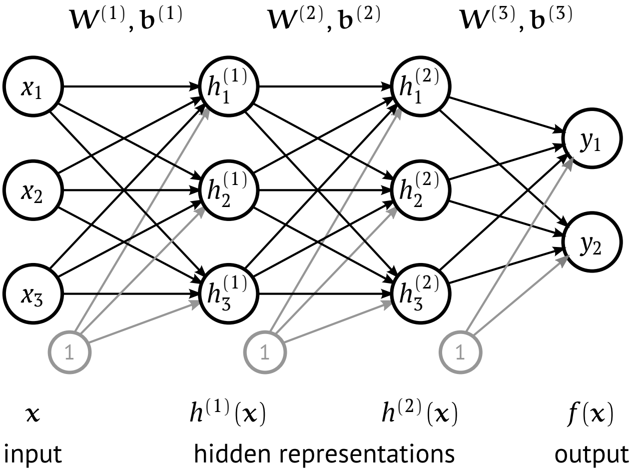
Hidden units are calculated from

$$h^{(0)}(\mathbf{x}) = \mathbf{x}$$

$$h^{(i)}(\mathbf{x}) = \sigma\left(\mathbf{W}^{(i)} h^{(i-1)}(\mathbf{x}) + \mathbf{b}^{(i)}\right)$$

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

Neural networks 101



Given a training sample $(\mathbf{x}, \mathbf{y})$, and a loss function $\mathcal{L}$, e.g.

$$\mathcal{L}(f(\mathbf{x}), \mathbf{y}) = \|f(\mathbf{x}) - \mathbf{y}\|_2^2$$

the network parameters are optimized using *back-propagation*.

Deep learning overview

Learn to solve the given task from data.

$$y = f(x; \theta)$$

- x: Input (e.g. image).
- y: Output (e.g. *bird*, *cat*, *dog*).
- f: Neural network.
- θ: Network parameters.

Deep learning overview

Learn to solve the given task from data.

$$y = f(x; \theta)$$

- x: Input (e.g. image).
- y: Output (e.g. *bird*, *cat*, *dog*).
- f: Neural network.
- θ: Network parameters.

Learn θ from from pairs x, y using gradient descent wrt. a chosen loss function.

$$\mathcal{L}\left(y, f(x; \theta)\right)$$

Deep learning overview

Learn to solve the given task from data.

$$y = f(x; \theta)$$

- x: Input (e.g. image).
- y: Output (e.g. *bird*, *cat*, *dog*).
- f: Neural network.
- θ: Network parameters.

Learn θ from from pairs x, y using gradient descent wrt. a chosen loss function.

$$\mathcal{L}\left(y, f(x; \theta)\right)$$

Hierarchical function decomposition allows us to learn distributed representations of our input.

$$f(x; \theta) = f^{(n)}\left(\ldots f^{(1)}\left(x; \theta^{(1)}\right)\ldots; \theta^{(n)}\right)$$

Deep learning overview

The good

- Powerful function approximation.
- Local optima not problematic with high-dimensional parameters.
- Feature disentangling.

The bad

- Computationally intensive.
- Can easily overfit.
- Require lots of data.

The ugly

- Finding a good architecture (layer types, layer ordering).
- Hyperparameter tuning (layer sizes, learning rate, weight initialization).

Beyond supervised models

- By construction, neural networks lend themselves to supervised learning.
- How do we leverage the power of neural networks in the unlabeled case?
- How do we combine network functions to form a semi-supervised model handling both unlabeled and labeled data?

Unsupervised pretraining

Vincent, P., Larochelle, H., Lajoie, I., et al. [2010]. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion". In: *Journal of Machine Learning Research* 11.Dec, pp. 3371–3408.

Hinton, G. E. and Salakhutdinov, R. R. [2006]. "Reducing the dimensionality of data with neural networks". In: *Science* 313.5786, pp. 504–507.

Idea

- Learn features in an unsupervised manner.
- Transfer learned features to a supervised model.
- Hope that the pretrained features alleviate overfitting.

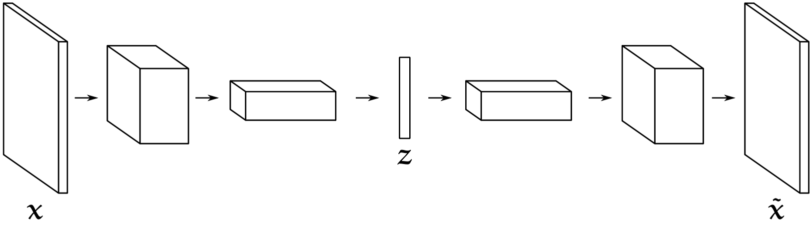
Idea

- Learn features in an unsupervised manner.
- Transfer learned features to a supervised model.
- Hope that the pretrained features alleviate overfitting.

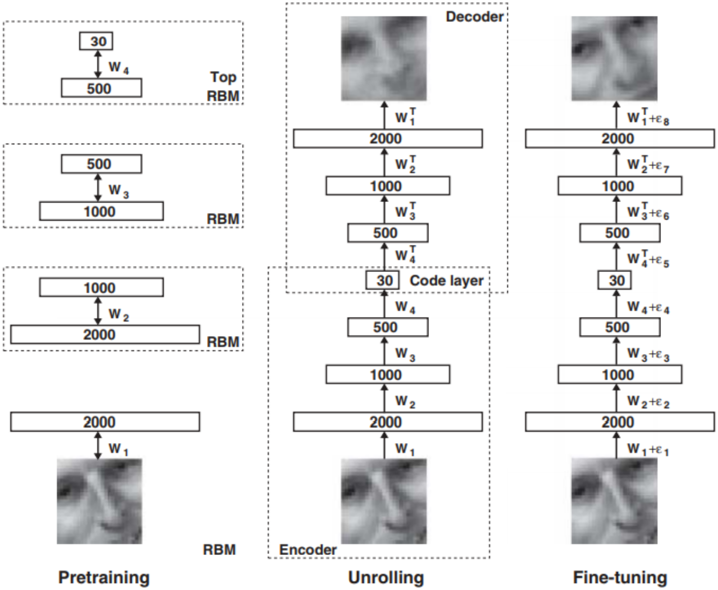
Note: Today, supervised training of neural networks has improved such that pretraining rarely is beneficial.

Autoencoders

- Learn an encoder-decoder architecture to reconstruct a dataset sample x as $\tilde{x}$.
- Train using a chosen loss function, e.g. $\mathcal{L}(x, \tilde{x}) = \|x - \tilde{x}\|^2$.
- Bottleneck representation z forces encoder to *disentangle* input.

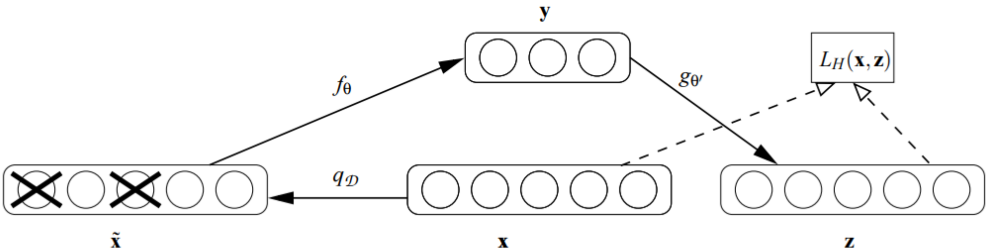


Layer-wise pretraining scheme

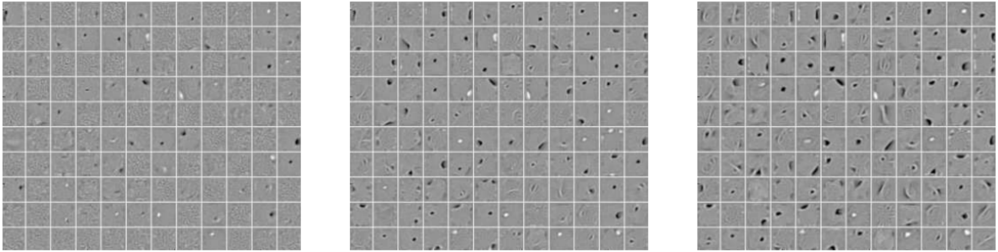


Denoising autoencoders

- Corrupt input to make higher level representations more robust.
- Very similar to dropout.
  - Prevents co-adaptation of features.
  - Effective regularizer.
  - Model averaging.



Denoising autoencoders, filters



(a) No corruption (b) 25% corruption (c) 50% corruption

(d) Neuron A (0%, 10%, 20%, 50% corruption) (e) Neuron B (0%, 10%, 20%, 50% corruption)

Denoising autoencoders, results

Data Set	SVM$_{rbf}$	DBN-1	SAE-3	DBN-3	SDAE-3 (ν)
MNIST	**1.40**±0.23	**1.21**±0.21	**1.40**±0.23	**1.24**±0.22	**1.28**±0.22 (25%)
basic	**3.03**±0.15	3.94±0.17	3.46±0.16	3.11±0.15	**2.84**±0.15 (10%)
rot	11.11±0.28	14.69±0.31	10.30±0.27	10.30±0.27	**9.53**±0.26 (25%)
bg-rand	14.58±0.31	9.80±0.26	11.28±0.28	**6.73**±0.22	10.30±0.27 (40%)
bg-img	22.61±0.37	**16.15**±0.32	23.00±0.37	**16.31**±0.32	**16.68**±0.33 (25%)
bg-img-rot	55.18±0.44	52.21±0.44	51.93±0.44	47.39±0.44	**43.76**±0.43 (25%)
rect	**2.15**±0.13	4.71±0.19	2.41±0.13	2.60±0.14	**1.99**±0.12 (10%)
rect-img	24.04±0.37	23.69±0.37	24.05±0.37	22.50±0.37	**21.59**±0.36 (25%)
convex	19.13±0.34	19.92±0.35	**18.41**±0.34	**18.63**±0.34	**19.06**±0.34 (10%)
tzanetakis	**14.41**±2.18	18.07±1.31	**16.15**±1.95	18.38±1.64	**16.02**±1.04(0.05)

Semi-supervised learning with Ladder networks

Rasmus, A., Berglund, M., Honkala, M., et al. [2015]. “Semi-supervised Learning with Ladder Networks”. In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes, N. D. Lawrence, D. D. Lee, et al. Curran Associates, Inc., pp. 3546–3554.

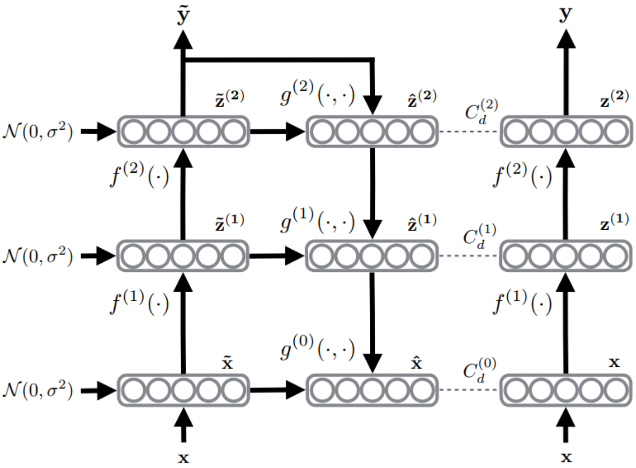
Mohammad, P., Linxi, F., Philemon, B., et al. [2016]. “Deconstructing the Ladder Network Architecture”. In: *Proceedings of the 33rd International Conference on Machine Learning (ICML)*.

Idea

- Combine a discriminative network with the encoder network of an autoencoder.
- Perform layerwise denoising (lateral connection).
- Ingenious architecture engineering.
  - Gaussian noise after batch normalization.
  - Squared-error denoising criterion after batch normalization.

Ladder architecture

- Encode x both with and without noise.
- Decode by combining lateral and downward signal.
- Layer-wise reconstruction error with clean encoding as target.
- Cross-entropy error for labeled examples.



Results, MNIST

Table 1: A collection of previously reported MNIST test errors in the permutation invariant setting followed by the results with the Ladder network. * = SVM. Standard deviation in parentheses.

Test error % with # of used labels	100	1000	All
Semi-sup. Embedding (Weston *et al.*, 2012)	16.86	5.73	1.5
Transductive SVM (from Weston *et al.*, 2012)	16.81	5.38	1.40*
MTC (Rifai *et al.*, 2011)	12.03	3.64	0.81
Pseudo-label (Lee, 2013)	10.49	3.46	
AtlasRBF (Pitelis *et al.*, 2014)	8.10 (± 0.95)	3.68 (± 0.12)	1.31
DGN (Kingma *et al.*, 2014)	3.33 (± 0.14)	2.40 (± 0.02)	0.96
DBM, Dropout (Srivastava *et al.*, 2014)			0.79
Adversarial (Goodfellow *et al.*, 2015)			0.78
Virtual Adversarial (Miyato *et al.*, 2015)	2.12	1.32	0.64 (± 0.03)
Baseline: MLP, BN, Gaussian noise	21.74 (± 1.77)	5.70 (± 0.20)	0.80 (± 0.03)
Γ-model (Ladder with only top-level cost)	3.06 (± 1.44)	1.53 (± 0.10)	0.78 (± 0.03)
Ladder, only bottom-level cost	1.09 (±0.32)	0.90 (± 0.05)	0.59 (± 0.03)
Ladder, full	**1.06** (± 0.37)	**0.84** (± 0.08)	**0.57** (± 0.02)

Variational autoencoders

Later this week!

Generative adversarial networks

Goodfellow, I., Pouget-Abadie, J., Mirza, M., et al. [2014]. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, et al. Curran Associates, Inc., pp. 2672–2680.

Salimans, T., Goodfellow, I. J., Zaremba, W., et al. [2016]. “Improved Techniques for Training GANs”. In: *CoRR* abs/1606.03498.

Radford, A., Metz, L., and Chintala, S. [2016]. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *Proceedings of the International Conference on Learning Representations*.

Idea

- Learn to generate samples that imitate real data samples.
- Discriminator network: learn to tell generated samples from real dataset samples (binary classification).
- Generator network: learn to fool the discriminator.

Setup

$\boldsymbol{x} \sim p_{\text{data}()}$, Dataset sample

$\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, Noisy variable

$\text{Dis}(\cdot)$, Discriminator network

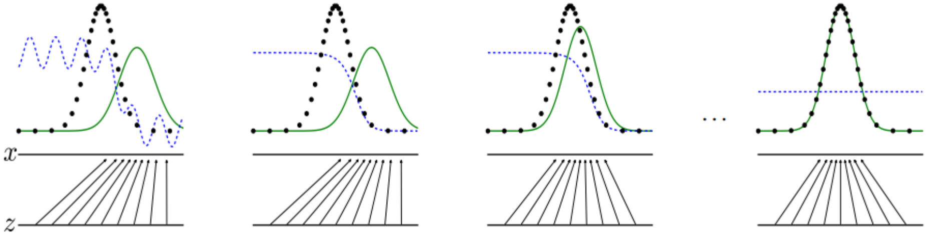
$\text{Gen}(\cdot)$, Generator network

Training objective:

$$\min_{\text{Gen}} \max_{\text{Dis}} \; \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})} \left[\log \text{Dis}(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})} \left[\log(1 - \text{Dis}(\text{Gen}(\boldsymbol{z})))\right]$$

GAN example

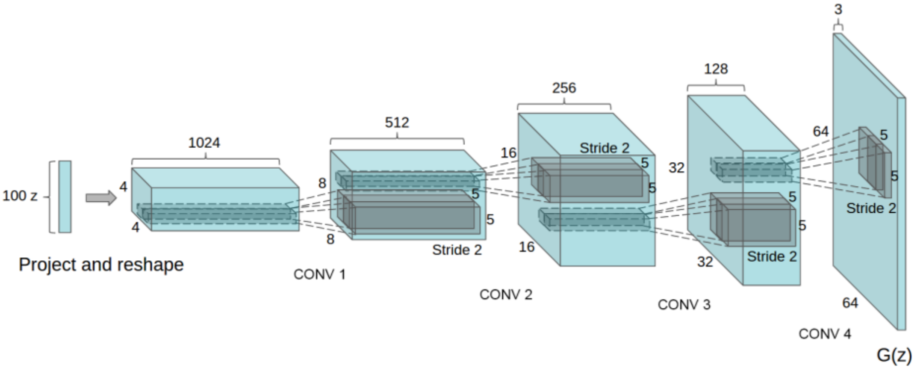
Near-convergence behavior on 1D data.



- Black dotted line: Data distribution, $p_{\text{data}}(x) \sim \mathcal{N}(\cdot)$
- Green line: Generative distribution, $\text{Gen}(x)$
- Blue dashed line: Discriminative distribution
- x, black line: data space
- z, black line: z-space, $p_z(z) \sim \text{Uniform}(\cdot)$

Convolutional decoder architecture

When generating images, the generator network dilutes high-dimensionsional features in exchange for increased resolution.



Semi-supervised GAN discriminator

For classification: Discriminator predicts $K+1$ classes where the extra class represents the generated sample.

$$
\begin{aligned}
L &= -\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}(\boldsymbol{x},y)}[\log p_{\text{model}}(y|\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim G}[\log p_{\text{model}}(y=K+1|\boldsymbol{x})] \\
&= L_{\text{supervised}} + L_{\text{unsupervised}}, \text{ where} \\
L_{\text{supervised}} &= -\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}(\boldsymbol{x},y)} \log p_{\text{model}}(y|\boldsymbol{x}, y<K+1) \\
L_{\text{unsupervised}} &= -\{\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})} \log[1-p_{\text{model}}(y=K+1|\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim G}\log[p_{\text{model}}(y=K+1|\boldsymbol{x})]\},
\end{aligned}
$$

Semi-supervised GAN discriminator

For classification: Discriminator predicts $K+1$ classes where the extra class represents the generated sample.

$$
\begin{aligned}
L &= -\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}(\boldsymbol{x},y)}[\log p_{\text{model}}(y|\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim G}[\log p_{\text{model}}(y=K+1|\boldsymbol{x})] \\
&= L_{\text{supervised}} + L_{\text{unsupervised}}, \text{ where} \\
L_{\text{supervised}} &= -\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}(\boldsymbol{x},y)} \log p_{\text{model}}(y|\boldsymbol{x}, y<K+1) \\
L_{\text{unsupervised}} &= -\{\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})} \log[1-p_{\text{model}}(y=K+1|\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim G} \log[p_{\text{model}}(y=K+1|\boldsymbol{x})]\},
\end{aligned}
$$

Model	Test error rate for a given number of labeled samples			
	1000	2000	4000	8000
Ladder network [24]			20.40±0.47	
CatGAN [14]			19.58±0.46	
Our model	21.83±2.01	19.61±2.09	18.63±2.32	17.72±1.82
Ensemble of 10 of our models	19.22±0.54	17.25±0.66	15.59±0.47	14.87±0.89

Semi-supervised GAN discriminator

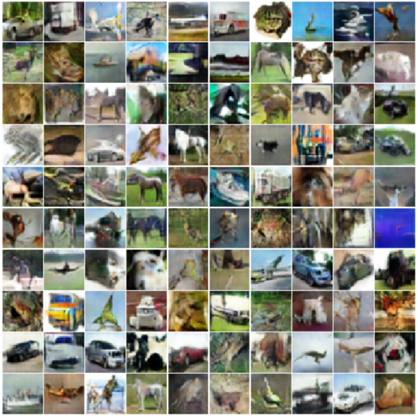
For classification: Discriminator predicts $K+1$ classes where the extra class represents the generated sample.

$$L = -\mathbb{E}_{\boldsymbol{x},y\sim p} \quad \text{odel}(y = K+1|\boldsymbol{x})]$$

$$= L_{\text{supervised}}$$

$$L_{\text{supervised}} = -\mathbb{E}_{\boldsymbol{x},y\sim p}$$

$$L_{\text{unsupervised}} = -\{\mathbb{E}_{\boldsymbol{x}\sim p} \quad \mathbb{E}_{\boldsymbol{x}\sim G}\log[p_{\text{model}}(y = K+1|\boldsymbol{x})]\},$$



Model			r rate for f labeled samples 4000	8000
Ladder network [24]			20.40±0.47	
CatGAN [14]			19.58±0.46	
Our model			18.63±2.32	17.72±1.82
Ensemble of 10 of our models	19.22±0.54	17.25±0.66	15.59±0.47	14.87±0.89

Autoencoding beyond pixels using a learned similarity measure

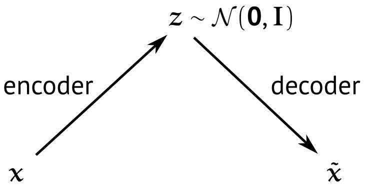
Larsen, A. B. L., Sønderby, S. K., Larochelle, H., et al. [2016]. "Autoencoding beyond pixels using a learned similarity metric". In: *Proceedings of the 33rd International Conference on Machine Learning (ICML)*.

Idea

- Element-wise loss function $\mathcal{L}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|^2$ is unsuitable for natural images.
- Why not try convnet features as a basis for measuring image similarity?
- Let's use features from a *generative adversarial network* to remain unsupervised.

Our building blocks

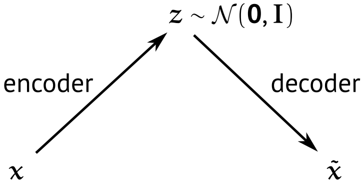
Variational autoencoder



- Good for MNIST-style data.
- Doesn't scale to natural images because of element-wise similarity measures.

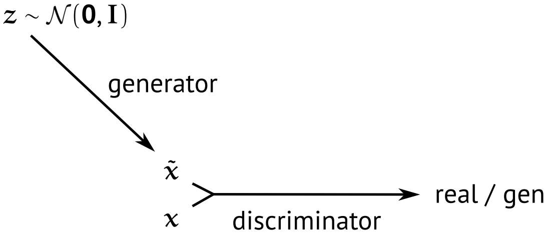
Our building blocks

Variational autoencoder



- Good for MNIST-style data.
- Doesn’t scale to natural images because of element-wise similarity measures.

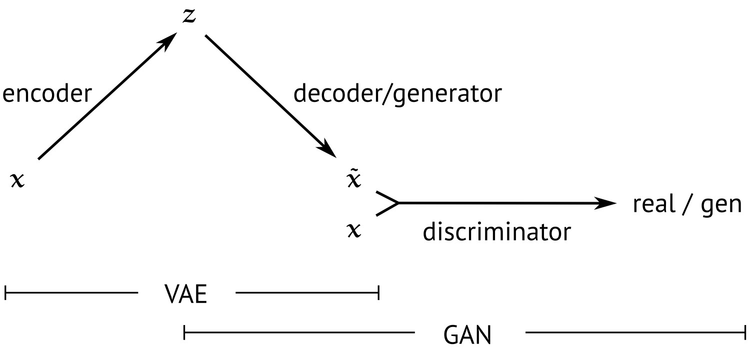
Generative adversarial network



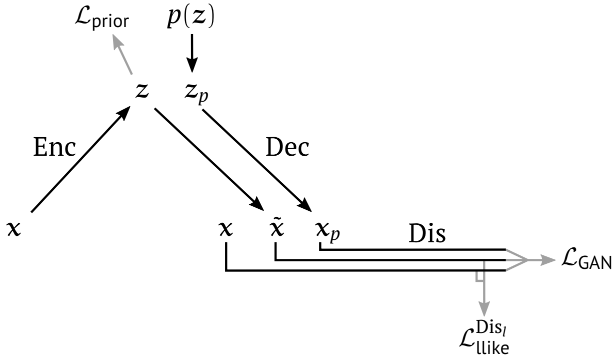
- Capable of generating natural looking images.
- No inference network.

Combining VAEs with GANs

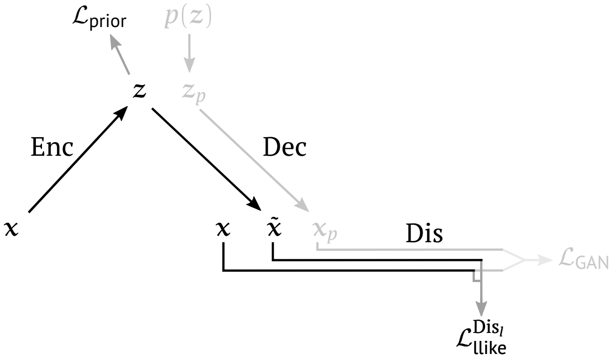
- Collapse the VAE decoder and the GAN generator into one network.
- Move the reconstruction error up in the discriminator network.
- Train both VAE and GAN simultaneously from scratch.



Implementation details



Implementation details



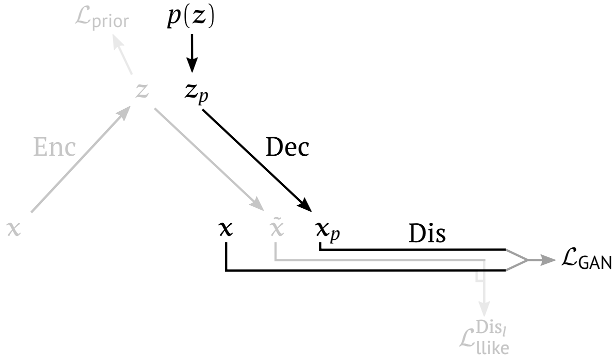
Pixel-based observation model:

$$p(x \mid z) = \mathcal{N}(x \mid \tilde{x}, \mathbf{I})$$

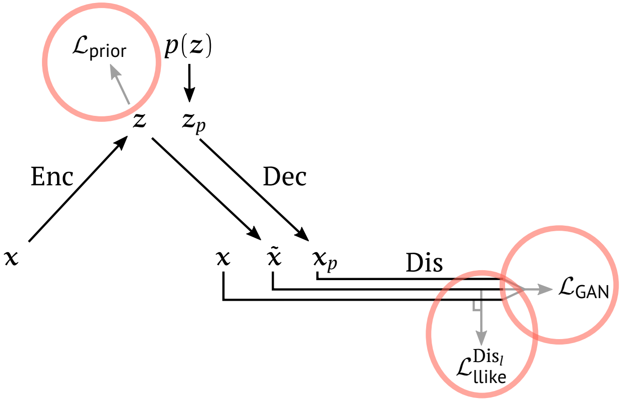
Feature-based observation model:

$$p(\text{Dis}_l(x) \mid z) =$$

Implementation details



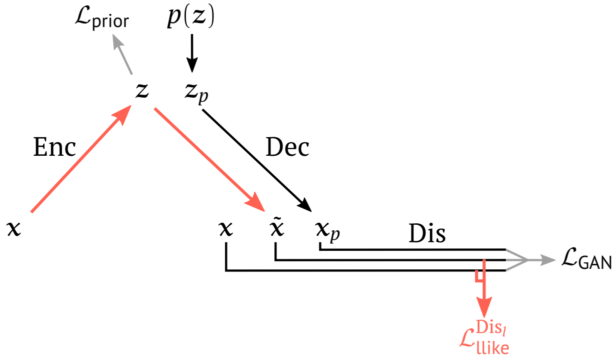
Implementation details



Training objective:

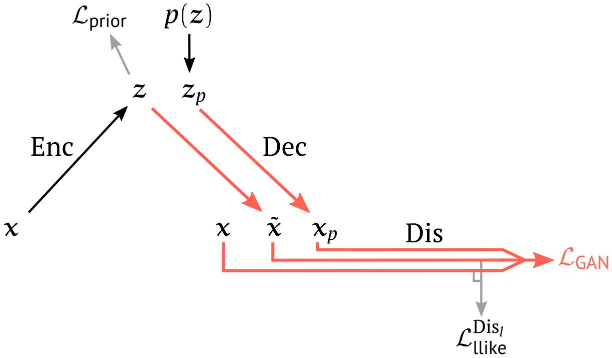
$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{llike}}^{\text{Dis}_l}}_{\mathcal{L}_{\text{VAE}}} + \mathcal{L}_{\text{GAN}}$$

Implementation details



Update only encoder/decoder networks wrt. reconstruction error.

Implementation details



Update only decoder/discriminator networks wrt. adversarial loss.

Experiments

Mainly using 64×64 images from the CelebA dataset. [Liu et al. 2015]

Models

VAE	Plain variational autoencoder with pixel-wise image similarity.
$\text{VAE}_{\text{Dis}_l}$	VAE with feature-wise similarities from a pretrained GAN.
VAE/GAN	Our hybrid method.
GAN	Plain generative adversarial network.

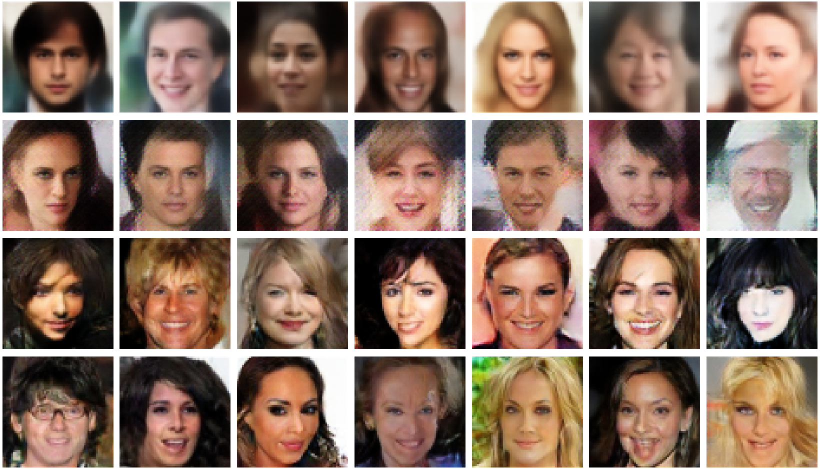
Samples

VAE

$\text{VAE}_{\text{Dis}_l}$

VAE/GAN

GAN



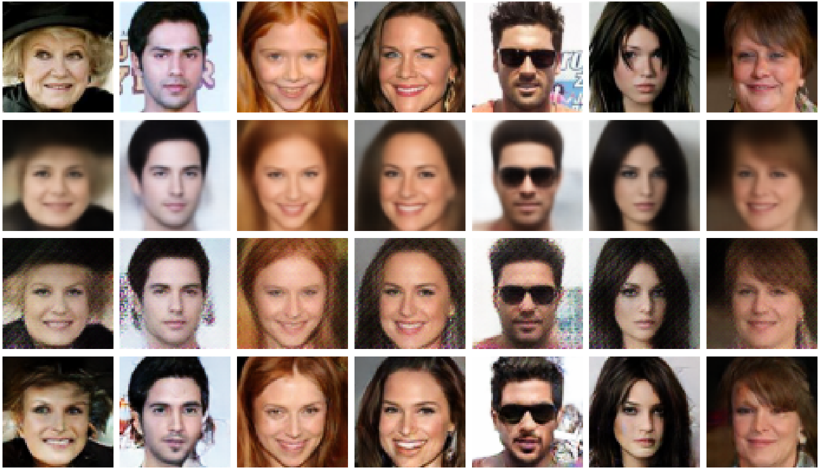
Reconstructions

Input

VAE

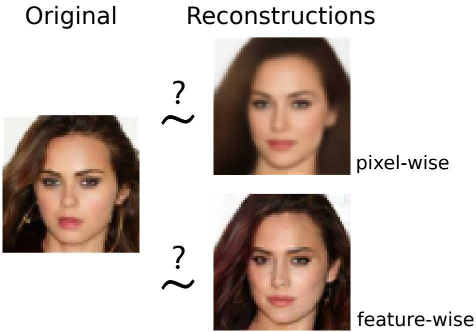
VAE$_{\mathrm{Dis}_l}$

VAE/GAN



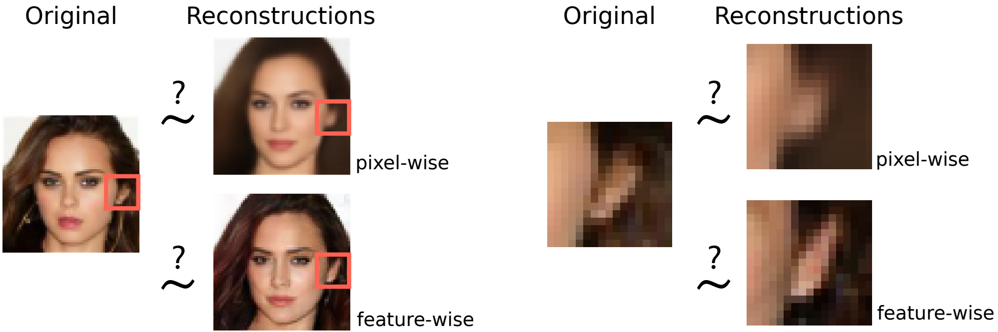
What is a good image reconstruction?

- Human visual perception is not pixel-perfect.
- Should we require perfect pixels from our model?



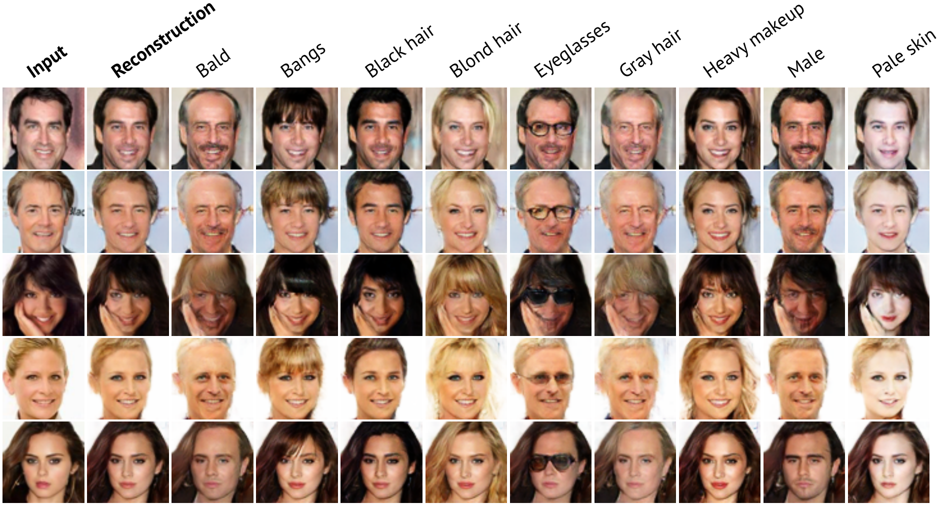
What is a good image reconstruction?

- Human visual perception is not pixel-perfect.
- Should we require perfect pixels from our model?
- Arguably, semantic concepts are more important than pixels.



Visual attribute vectors

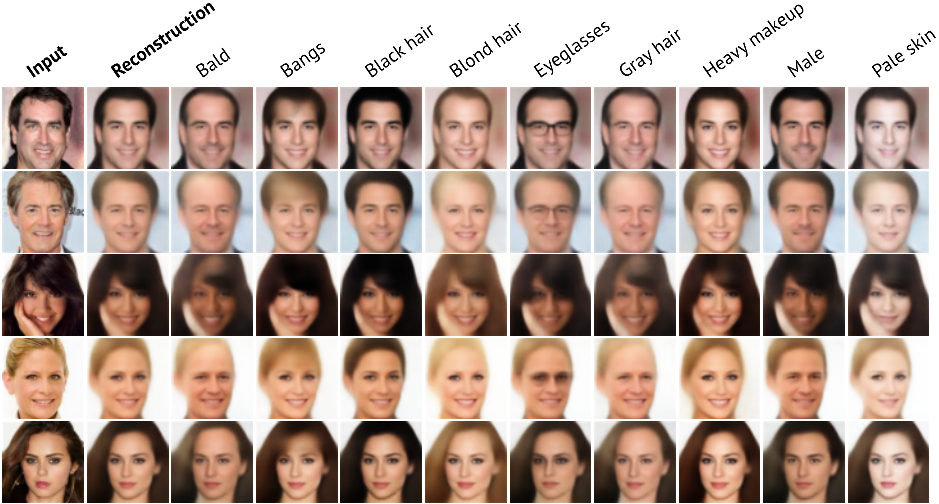
After unsupervised training, we calculate the mean z difference for images with/without an attribute. Difference vectors capture visual attributes:



VAE/GAN

Visual attribute vectors

After unsupervised training, we calculate the mean z difference for images with/without an attribute. Difference vectors capture visual attributes:



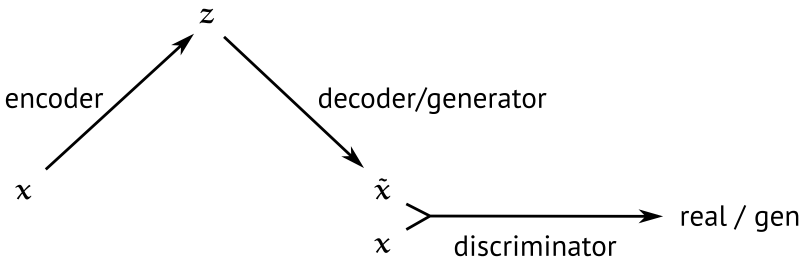
VAE

Interpolations in latent space

I sent my adviser into latent space and sampled a few visual attribute vectors:

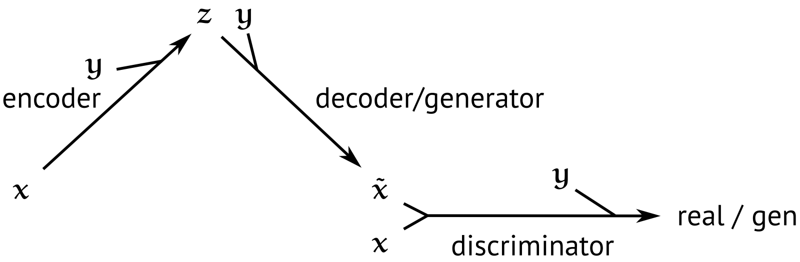
Quantitative results: Recognizability of generated attributes

Idea: Learn a conditional model $p(x \mid y, z)$ where y are visual attributes. Generate images from sampled z and attribute queries y.



Quantitative results: Recognizability of generated attributes

Idea: Learn a conditional model $p(x \mid y, z)$ where y are visual attributes. Generate images from sampled z and attribute queries y.



Quantitative results: Recognizability of generated attributes

Idea: Learn a conditional model $p(x \mid y, z)$ where y are visual attributes. Generate images from sampled z and attribute queries y.

Query

Prominent attributes: White, Mouth Closed, Male, Curly Hair, Eyes Open, Pale Skin, Frowning, Pointy Nose, Teeth Not Visible

VAE

GAN

VAE/GAN

Query

Prominent attributes: White, Male, Curly Hair, Frowning, Pointy Nose, Eyeglasses, Narrow Eyes, Teeth Not Visible, Senior

VAE

GAN

VAE/GAN

Quantitative results: Recognizability of generated attributes

Idea: Learn a conditional model $p(x \mid y, z)$ where y are visual attributes. Generate images from sampled z and attribute queries y.

Evaluation: Attribute prediction error from a separately trained regressor convnet.

Model	**Cosine similarity (best of 10)**	**Mean squared error**
LFW test set	0.9193	14.1987
VAE	0.9030	27.59 ± 1.42
GAN	0.8892	27.89 ± 3.07
VAE/GAN	**0.9114**	$\mathbf{22.39 \pm 1.16}$

Final words

Take-home messages

- You can learn useful structures from fragile error signals.
- Good disentangled representations can make the discriminative task easier.

Thanks