Everything you wanted to know about Deep Learning for Computer Vision but were afraid to ask

Auto-Encoders

Moacir Ponti, Leonardo Ribeiro, Tiago Nazare
ICMC, Universidade de São Paulo

Tu Bui, John Collomosse
CVSSP, University of Surrey

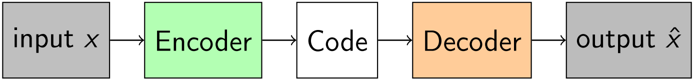
Contact: `www.icmc.usp.br/~moacir` — `moacir@icmc.usp.br`

Rio de Janeiro/Brazil – October, 2017

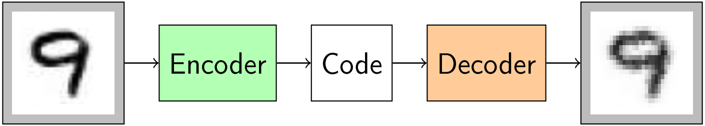
Agenda

1. Autoencoders basics
2. Undercomplete AEs
3. Overcomplete Regularized AEs
4. Concluding remarks

General architecture of a Deep Autoencoder



General architecture of a Deep Autoencoder



Autoencoders basics: encoder and decoder

Encoder

Produces Code or Latent Representation

$$\mathbf{h} = s(\mathbf{W}\mathbf{x} + \mathbf{b}) = f(\mathbf{x})$$

Autoencoders basics: encoder and decoder

Encoder

Produces Code or Latent Representation

$$\mathbf{h} = s(\mathbf{W}\mathbf{x} + \mathbf{b}) = f(\mathbf{x})$$

Decoder

Produces Reconstruction of the input

$$\hat{\mathbf{x}} = s(\mathbf{W}'\mathbf{h} + \mathbf{b}') = g(\mathbf{h})$$

Tied weights when $\mathbf{W}' = \mathbf{W}^T$

Autoencoders basics: loss function

Given the output $\hat{x} = g(f(x))$
We want to minimize some reconstruction loss:

$$\mathcal{L}(x, g(f(x)) = \hat{x})$$

Cross entropy (bits or probability vectors)

$$\mathcal{L}(x, \hat{x}) = x \log \hat{x} + (1 - x) \log(1 - \hat{x})$$

Mean squared error (continuous values)

$$\mathcal{L}(x, \hat{x}) = ||x - \hat{x}||^2$$

Autoencoders basics: flavours

Undercomplete

- Bottleneck layer produces code $\mathbf{h}$ with less dimensions then input $\mathbf{x}$

Overcomplete

- Code $\mathbf{h}$ has more dimensions then the input $\mathbf{x}$
- Different versions e.g. sparse, denoising, contractive.

Agenda

1 Autoencoders basics

2 Undercomplete AEs

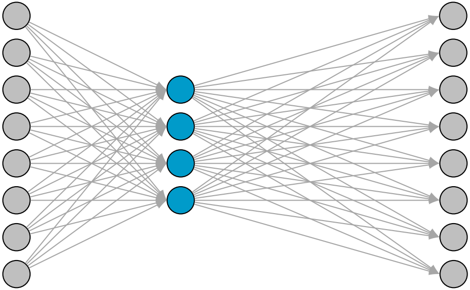
3 Overcomplete Regularized AEs

4 Concluding remarks

Undercomplete

Learns a Lossy Compression of the input data.

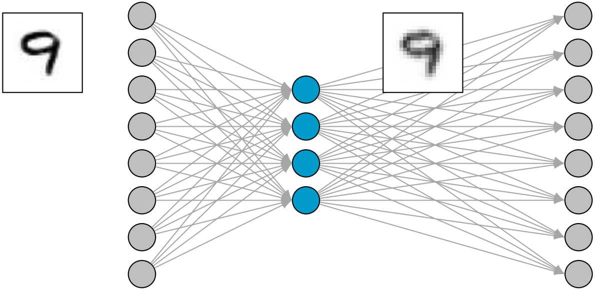
- has a “bottleneck” layer
- can be used for Dimensionality Reduction — often compared to Principal Component Analysis (PCA)
- often code is a good representation for the training data only



Undercomplete

Learns a Lossy Compression of the input data.

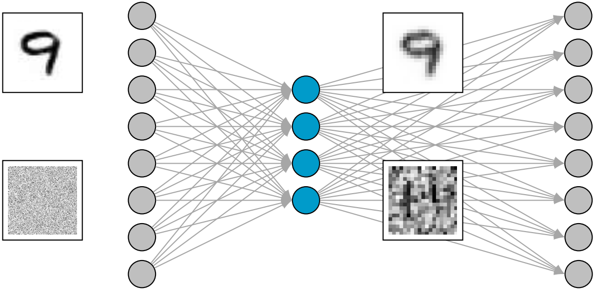
- has a “bottleneck” layer
- can be used for Dimensionality Reduction — often compared to Principal Component Analysis (PCA)
- often code is a good representation for the training data only



Undercomplete

Learns a Lossy Compression of the input data.

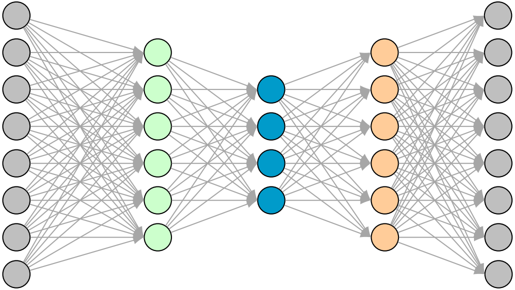
- has a “bottleneck” layer
- can be used for Dimensionality Reduction — often compared to Principal Component Analysis (PCA)
- often code is a good representation for the training data only



Undercomplete

Increasing the number of layers adds capacity to the AE.

- Encoder and Decoder layers can also be convolutional layers



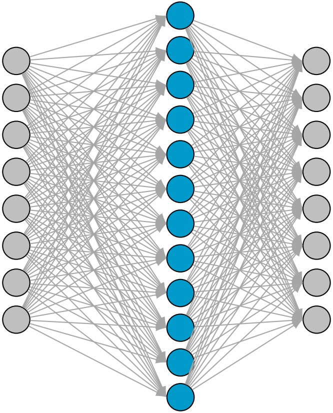
In principle with a sufficiently large capacity it may map every input to a single neuron on bottleneck layer.

Agenda

1. Autoencoders basics
2. Undercomplete AEs
3. Overcomplete Regularized AEs
4. Concluding remarks

Overcomplete AEs

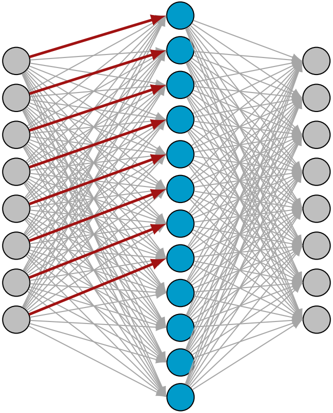
High-dimensional intermediate layer



Overcomplete AEs

High-dimensional intermediate layer

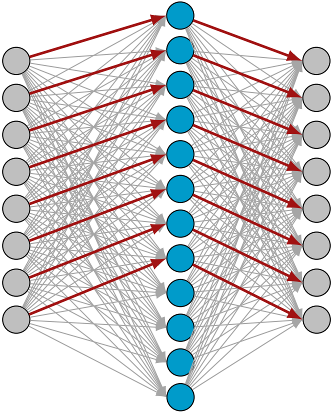
- a naive implementation would allow a copy so that $\mathbf{x} = \hat{\mathbf{x}}$



Overcomplete AEs

High-dimensional intermediate layer

- a naive implementation would allow a copy so that $\mathbf{x} = \hat{\mathbf{x}}$



Overcomplete regularized AEs

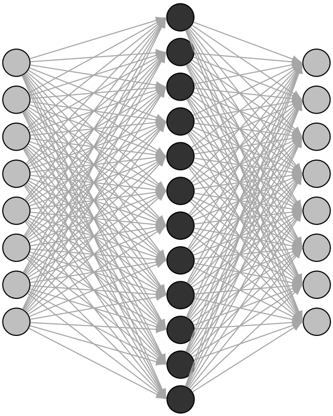
Regularization with sparsity constraint

$$\mathcal{L}(x, g(f(x))) + \Omega(f(x))$$
$$\mathcal{L}(x, g(f(x))) + \lambda \sum_i |h_i|,$$

- loss function tries to keep a low number of activation neurons per training input

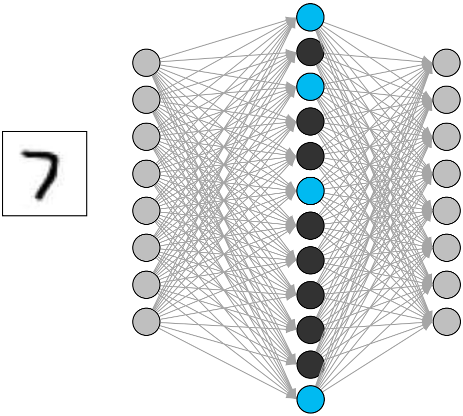
Overcomplete regularized AEs

Regularization with sparsity constraint



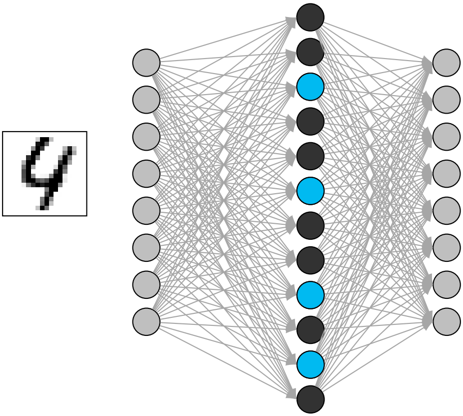
Overcomplete regularized AEs

Regularization with sparsity constraint



Overcomplete regularized AEs

Regularization with sparsity constraint



Denoising AEs (DAEs)

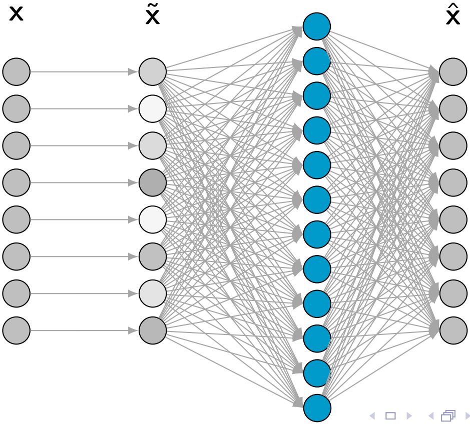
Regularization achieved by adding noise to $\mathbf{x}$

- the loss is computed using the noiseless input $\mathbf{x}$
- AE has to reconstruct $\mathbf{x}$ using a noisy input $\tilde{\mathbf{x}}$, so representation must be robust to noise
- this prevents the overcomplete AE to simply copy the data

Denoising AEs (DAEs)

Regularization achieved by adding noise to $\mathbf{x}$

- DAEs aim to learn a good internal representation as a side effect of learning to denoise the input



Denoising AEs (DAEs)

Noise processes

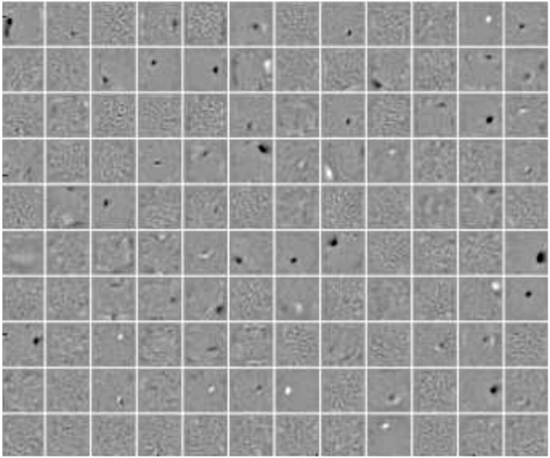
- Additive Gaussian Noise with $\mu = 0$, and some σ;
- Set a percentage of the input data to zero with some probability *p*.

Interpretation

- Learns to project data around some manifold to the distribution of the original (noiseless) data
- If some input is to far from the original distribution, it produces a high reconstruction error

Denoising AEs (DAEs): example

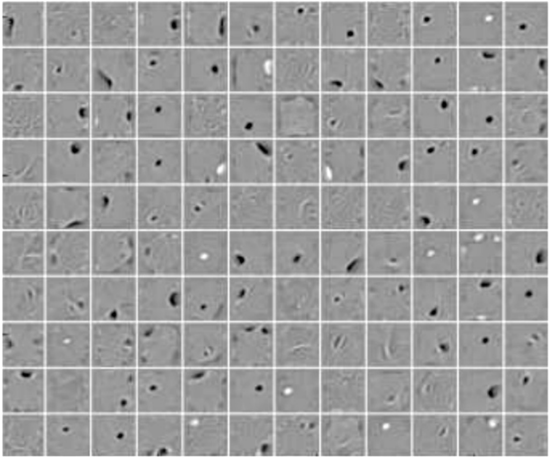
Using MNIST dataset, without noise



Vincent, Pascal, et al. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." Journal of Machine Learning Research. 2010;

Denoising AEs (DAEs): example

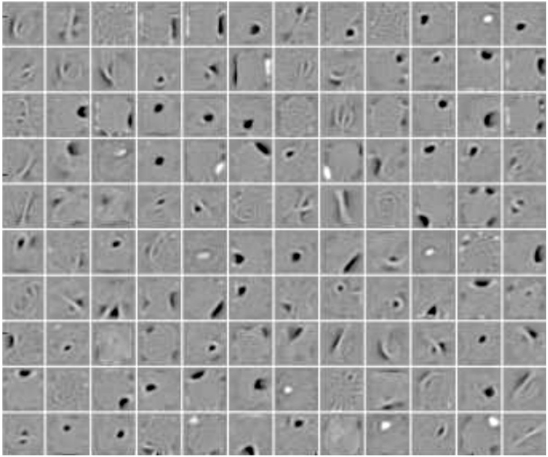
Using MNIST dataset, zero input variable with 25% probability



Vincent, Pascal, et al. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." Journal of Machine Learning Research. 2010;

Denoising AEs (DAEs): example

Using MNIST dataset, zero input variable with 50% probability



Vincent, Pascal, et al. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." Journal of Machine Learning Research, 2010;

Contractive AEs (CAEs)

Regularization based on the gradient of code $f(\mathbf{x}) = \mathbf{h}$ with respect to $\mathbf{x}$

- adds a term to the Loss function
- it is referred to as the Frobenius norm of the Jacobian of the Encoder

$$\ell(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda ||\nabla_{\mathbf{x}_i} f(\mathbf{x}_i)||_F^2$$

$$\ell(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda \sum_j \sum_k \left(\frac{\partial f(\mathbf{x}_i)_j}{\partial x_i^{(k)}} \right)^2$$

j – index for the code (intermediate layer unit)
k – index for the input vector

The Jacobian is a matrix of the derivatives of all elements of the code with respect to all elements of the input

Contractive AEs (CAEs)

Regularization based on the gradient of code $f(\mathbf{x}) = \mathbf{h}$ with respect to $\mathbf{x}$

- adds a term to the Loss function
- it is referred to as the Frobenius norm of the Jacobian of the Encoder

$$\ell(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda ||\nabla_{\mathbf{x}_i} f(\mathbf{x}_i)||_F^2$$

$$\ell(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda \sum_j \sum_k \left(\frac{\partial f(\mathbf{x}_i)_j}{\partial x_i^{(k)}} \right)^2$$

j – index for the code (intermediate layer unit)
k – index for the input vector

The Jacobian is a matrix of the derivatives of all elements of the code with respect to all elements of the input

Contractive AEs (CAEs)

Effects of terms on the encoder:

- $\ell(\mathbf{x}_i, g(f(\mathbf{x}_i)))$: relies on keeping relevant information;
- $\lambda||\nabla_{\mathbf{x}_i} f(\mathbf{x}_i)||_F^2$: throws away changes in code with respect to input.

Interpretations:

- rate of change of the code must follow the rate of change of the input;
- if noise is added to input, the code should not be affected (compare to Denoising AEs!);
- a good balance between terms will result in keeping only the relevant information.

Contractive AEs (CAEs)

Jacobian matrix can be seen as a linear approximation of a nonlinear encoder.

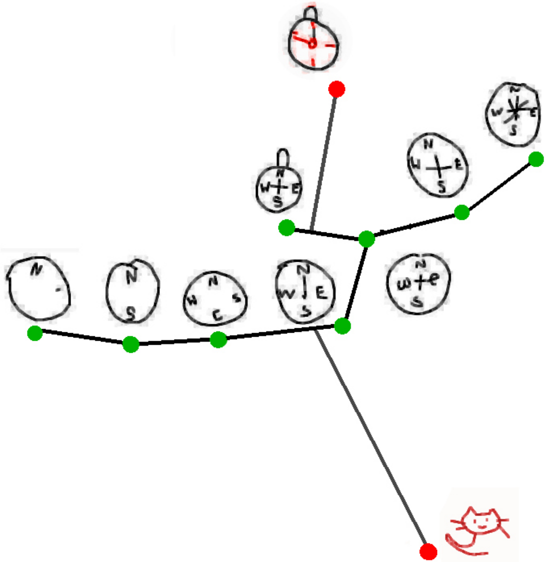
- A linear operator is said to be contractive if the norm of $\mathbf{J_x}$ is kept less than or equal to 1 for all unit-norm of x, i.e. if it shrinks the unit sphere around each point;
- CAE encourages each of the local linear operators to become a contraction;
- only a few directions of the manifold of the data approaches zero, likely the directions approximating the tangent planes of the manifold.

Contractive AEs (CAEs): interpretation for images

CAE learns to reconstruct data that is:

- tangent to the manifold or within some sphere;
- those are likely to represent real variations of the data
- in images that would be related to rotation, style change, etc.

Contractive AEs (CAEs): a sketch manifold illustration



Concluding remarks

- AEs can be a good choice with unsupervised data;
- Deep autoencoders can be useful to many applications, via manifold learning;
- The potential for manifold learning can be used for instance on Generative tasks (Generative and Variational Autoencoders).
- Those can also be plugged in supervised architectures.

References

- **Ponti, M.; Ribeiro, L.; Nazare, T.; Bui, T.; Collomosse, J. Everything you wanted to know about Deep Learning for Computer Vision but were afraid to ask. In: SIBGRAPI – Conference on Graphics, Patterns and Images, 2017.** http://sibgrapi.sid.inpe.br/rep/sid.inpe.br/sibgrapi/2017/09.05.22.09
- Rifai, Salah, et al. "Higher order contractive auto-encoder." Machine Learning and Knowledge Discovery in Databases (2011): 645-660.
- Vincent, Pascal, et al. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." Journal of Machine Learning Research, 2010: 3371-3408.
- Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, 2016.