Autoencoders

Fredrik Bagge Carlson

Introduction

General idea

Auto: Greek auto- "self, one's own"

Encode: from en- "make, put in" + code: a system of words, letters, figures, or symbols used to represent others

Find a useful encoding, $h = f(x)$, of data x in an unsupervised manner.

Trained using an encoder $h = f(x)$ and a decoder $\hat{x} = r(h)$

Loss is some measure comparing input to reconstruction, $L(x, \hat{x})$

References

- Ch. 14 in the Deep learning book
- `https://www.youtube.com/watch?v=s96mYcicbpE`
- `https://www.youtube.com/watch?v=FzS3tMl4Nsc` (7 videos in total)

Introduction
Uses

- Feature learning
- Unsupervised pretraining
- Dimensionality reduction (think of nonlinear PCA/SVD)
- Manifold learning

Variants

Different techniques of preventing the autoencoder from learning the identity function $x = r(h) = r(f(x)) = I(x)$

- Undercomplete autoencoder
- Denoising autoencoder
- Regularized autoencoder
- Contractive autoencoder

Variants

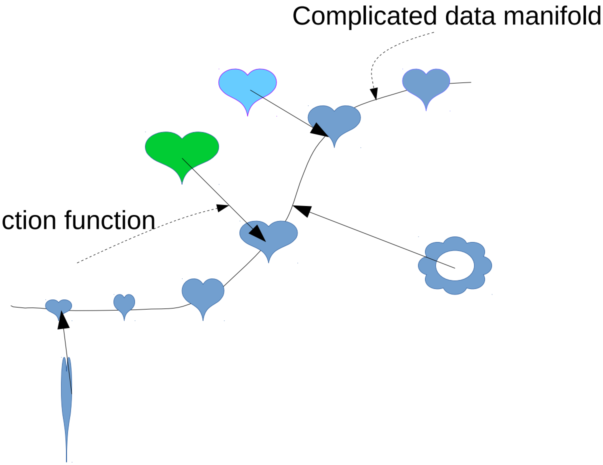
Undercomplete autoencoder

- Constrain h to have smaller dimension than x.
- Hope that h will be a useful representation of x.
- Used for, e.g., nonlinear dimensionality reduction.

Variants

Denoising autoencoder

Corrupt the data or the activations h with random (possibly structured) noise.



Variants

Regularized autoencoder

Add regularizing term to the cost function, e.g.,

- $\left\|h\right\|_1$ for a sparse activation vector.

Variants

Contractive autoencoder

Add regularizing term to the cost function, e.g.,

- $\left\|\nabla_x h\right\|$ for a locally flat output, i.e. code h becomes invariant to small perturbations in x.

Applications

- Nonlinear dimensionality reduction[1]

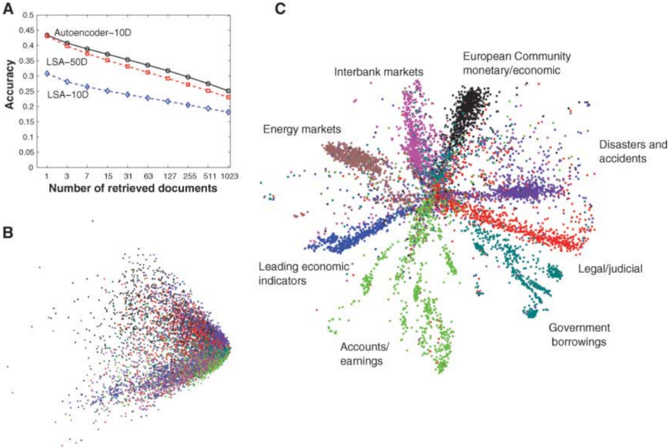


Fig. 4. (**A**) The fraction of retrieved documents in the same class as the query when a query document from the test set is used to retrieve other test set documents, averaged over all 402,207 possible queries. (**B**) The codes produced by two-dimensional LSA. (**C**) The codes produced by a 2000-500-250-125-2 autoencoder.

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

Homework

Experiment with an AE-implementation, for example this one

`https://github.com/alrojo/tensorflow-tutorial/tree/master/lab5_AE`

- Train on your data of choice (tutorial uses MNIST).
- Experiment with a low dimensional code, h, for visualization.

Homework

Details

Try at least a few of the following points

1. Start with a simple AE
   - Experiment with the number of layers and non-linearities in order to improve the reconstructions.
   - What happens with the network when we change the non-linearities in the latent layer (e.g. sigmoid)?
2. The default implementation optimizes mean squared error.
   - Find another error function that could fit this problem better.
   - Test different optimization algorithms and decide whether you should use regularizers.

Homework

Practical tips

- I used the Ipython notebook, Python 2
- Dependencies are easily installed using pip, e.g., matplotlib, sklearn
  `sudo pip install matplotlib`
- I can provide an almost working implementation in Julia-tensorflow
- I can also provide a working implementation in `MXNet.jl`, but accessing h seems tricky.