CS F425: Deep Learning

28 Autoencoder Architecture

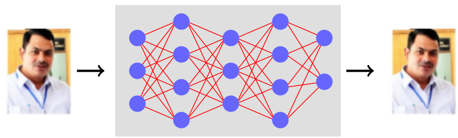
Dr. Kamlesh Tiwari
Assistant Professor, Department of CSIS,
BITS Pilani, Pilani Campus, Rajasthan-333031 INDIA

April 10, 2023 ON-CAMPUS Campus @ BITS-Pilani [Jan-May 2023]

http://ktiwari.in/dl

Autoencoder

A **neural network** that is trained to copy its input to its output (not sure how successful it would be)

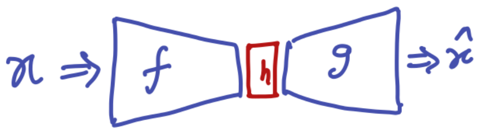


- Internally, it has a hidden layer h that describes a code used to represent the input x
- The network has two parts
  1. An **encoder** function $h = f(x)$
  2. And a decoder that produces a reconstruction $r = g(h)$
- Network is designed such that it is not easy to learn a perfect copy

Essentially, we need $g(f(x)) = x$

Undercomplete Autoencoders [1]

Interesting things happen when we force h to be of smaller then x



- Network is forced to prioritize which aspects of the input should be copied, it often learns useful properties of the data
- Learning process minimizes the loss $L(x, g(f(x)))$
- When decoder is linear and L is MSE, it learns subspace as PCA
- f and g with too much capacity may fail to learn

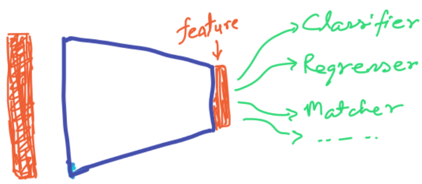
[1] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, **Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,** Journal of machine learning research, vol. 11, no. Dec, pp. 3371–3408, 2010.

Regularized Autoencoders

- In **overcomplete** cases, encoder and decoder could learn to copy input to output without learning anything useful
- Encoder/decoder architecture should be chosen based on the complexity of *distribution to be modeled*
- Loss function of **regularized autoencoders** encourages the model to have other properties like sparsity of the representation, smallness of the derivative of the representation, and robustness to noise or to missing inputs
  1. Sparse autoencoder has loss function as $L(x, g(f(x))) + \Omega(h)$ where $\Omega(h)$ correspond to some other task such as classification
  2. Denoising autoencoder minimizes $L(x, g(f(\tilde{x})))$ where $\tilde{x}$ is a copy corrupted by some noise
  3. Contractive autoencoder minimizes $L(x, g(f(x))) + \lambda \sum_i || \nabla_x h_i||^2$ that forces h not to change much when x changes

Autoencoders As Feature Extractor

Can serves as feature extractor for many applications



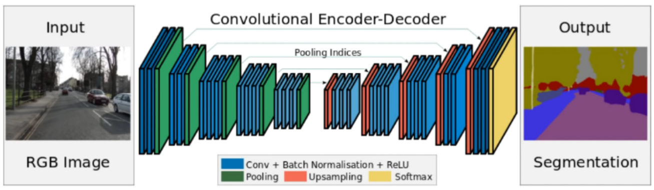
- End-to-end training could be done on extracted feature
- Train classifier, regressor, matching module, segmntation, or ...
- Feature could also be taken from inside the encoder
- With classifier, you could sift focus of the feature.

Autoencoders As Feature Extractor

- Autoencoder could be an unsupervised feature extractor
- Classifier on bottleneck converses very fast
- Decoder had never seen the original image it has only seen the code h. But, distribution is captured
- What would happen if you randomly perter h and show to decoder
- How to get new cat images?

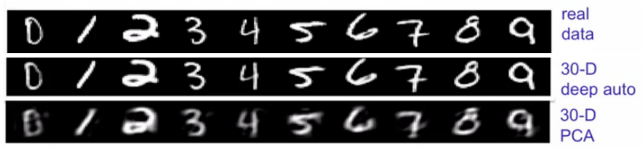
Applications of Autoencoders

- Dimensionality reduction (representation learning)
- Information retrieval/semantic hashing tasks - we can store all database entries in a hash table that maps binary code vectors to entries
- Classification
- Denoising autoencoders
- Useful for segmentation and deep-feature
- Neural inpainting



Autoencoder example-1

Compressing digit images to 30 numbers.



MNIST[2] digit images 28×28 three hidden layers (weights were transpose)

$$784 \rightarrow 1000 \rightarrow 500 \rightarrow 250 \rightarrow 30$$

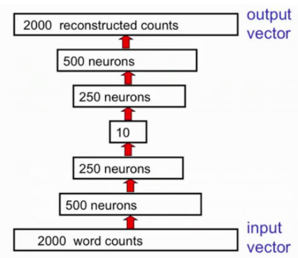
It is a supervised learning method to do unsupervised learning

[2] http://yann.lecun.com/exdb/mnist/

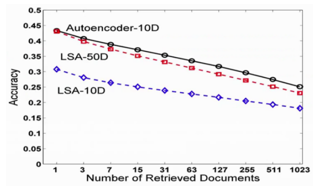
Autoencoder example-2

Compare documents for similarity

- Bag of words
- Word count is normalized for probability
- Compressed to 10 denominational
- Softmax is used at output
- 400K business documents
- Hand labeled for ground truth categories
- cosine similarity

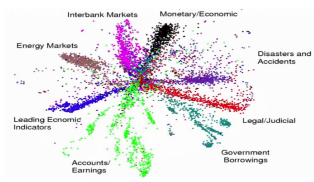


Autoencoder example-2



Reduce to 2 real numbers using PCA with $\log(1 + count)$

Deep autoencoders



Linear semantic analysis is much worse

Thank You!

Thank you very much for your attention!