# Morphological Multi-scale Decomposition and efficient representations with Auto-Encoders

•••

Research Internship Project - April 23$^{rd}$ 2018 - September 28$^{th}$ 2018

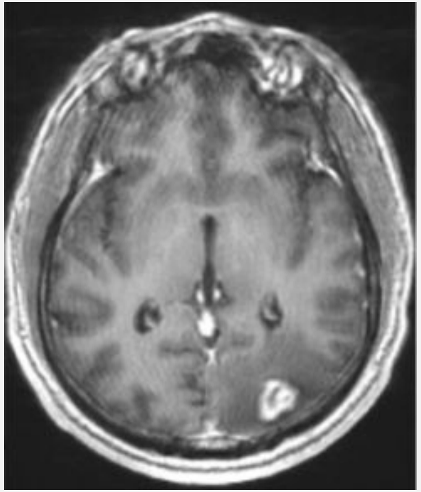
**Bastien PONCHON**

**11 Juin 2018**

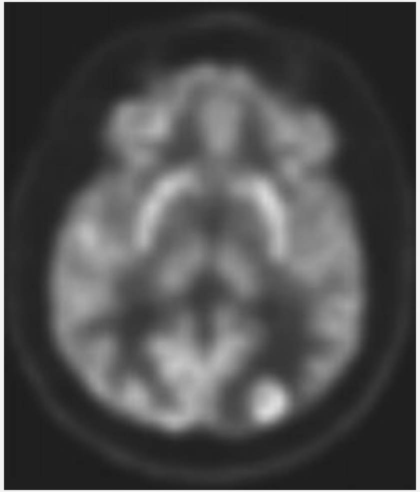
# Objectives

# Object Detection using low dimensional representation

Brain Tumor detection on multi-modal brain images

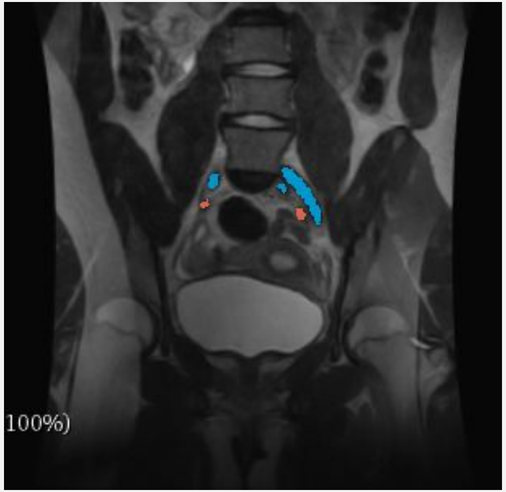


MRI image



TEP

Blood vessels detection on Pelvis images




MRI image

# Representation Learning and Dimensionality Reduction

Goals:

- Learning the underlying structure and latent feature in the input.
- Capturing the posterior distribution of the underlying explanatory factors for the observed input.

Many techniques:

- Linear: Principal Components Analysis, Non-Negative Matrix Factorization...
- Non Linear: Isomap, Kernel PCA, Multi-Dimensional Scaling...

Why using Auto-encoders:

- Able to learn a wide range of (non-linear) functions.
- **Universal approximator theorem:** *a feedforward neural network with at least one hidden layer can represent an approximation of any function (within a broad class) to an arbitrary degree of accuracy, provided that it has enough hidden units.*

# Enhancement using Mathematical Morphology

Idea:

- Training and applying the auto-encoder on multi-scale decompositions of the images rather than on the original images.
- **Input**: $x \in \mathbb{R}^{d \times d \times c \times l}$ instead of $x \in \mathbb{R}^{d \times d \times c}$, $d$ image dimension, $c$ number of channels, $l$ number of decompositions

Motivations:

- Capturing structures into redundant representations preserving level-sets.
- Orienting the representation learned by the Auto-Encoder.

Objective of the project:

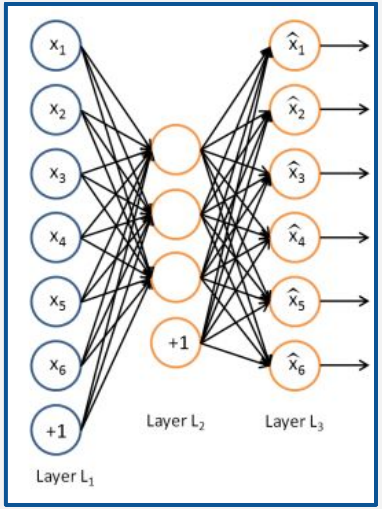
- Comparing the learned representations with and without prior Morphological Decompositions.
- Various latent representation dimensions (code size).
- Various Auto-Encoder architectures and flavours.
- Various types of decomposition.

# Auto-Encoders

# Introduction to Auto-Encoders

## An Auto-Encoder is a neural network made of two elements:

- An Encoder: $f : \mathbf{x} \longmapsto f(\mathbf{x}) = \mathbf{h} \in \mathbb{R}^k$, $k$ is the code size
- A Decoder: $g : \mathbf{h} \longmapsto g(\mathbf{h}) = \hat{\mathbf{x}} \in \mathbb{R}^N$ with $N = d \times d \times c$ or $N = d \times d \times c \times l$
- $g \circ f$ is trained to approximate identity
  - Reconstruction error: $L(\mathbf{x}, \hat{\mathbf{x}})$ where $\hat{\mathbf{x}} = g(f(\mathbf{x}))$ is the reconstruction of input sample $\mathbf{x}$
  - L can be of various type: *mean squared error, binary cross-entropy, etc.*
- The capacity of the AE increases with the code size and the neural network depth.




## Under-complete Auto-Encoders:

- Learning a code of size $k < N$ : dimensionality reduction.
- Similar to PCA when linear Encoder and Decoder and squared reconstruction error.

## Regularized Auto-Encoders:

- Limiting the model capacity by enforcing some properties by additional loss functions
  - Sparsity of the representation, smallness of the derivatives of the representation, robustness to noise, etc.

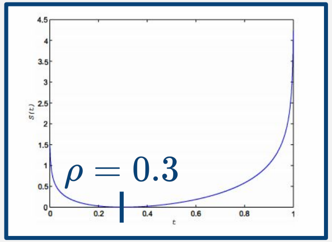
# Adding Sparsity Constraints

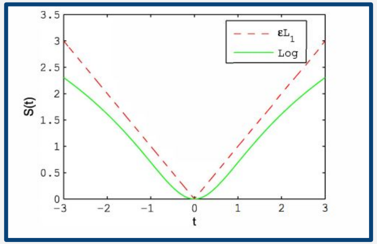
## Regularization of the representation learned by the Auto-Encoders:

- Enforcing most code coefficients $\mathbf{h}_i, i \in [1, k]$ to be close to 0 (to be inactive)
- Capturing a more robust representation of the manifold structure.

## Common implementation:

- Adding a sparsity regularizer loss to the AE loss function: $L_{AE} = \frac{1}{m}\sum_{j=1}^{m} L(\mathbf{x}^{(j)}, \hat{\mathbf{x}}^{(j)}) + \underbrace{\boxed{\frac{\lambda}{2}\sum_{l=1}^{L} ||W_l||_2^2}}_{\text{Weight decay}} + \underbrace{\boxed{\beta \sum_{i=1}^{k} S(t_i)}}_{\text{Sparsity constraint}}$
- With: $t_i = \frac{1}{m}\sum_{j=1}^{m} h_i^{(j)}$, the mean of the activation of the latent code coefficient $i$ on a batch of $m$ input samples
- Various sparsity regularizers $S$: $S(t) = KL(\rho, t) = \rho \log \frac{\rho}{t} + (1-\rho)\log\frac{1-\rho}{1-t}$ $\quad S(t) = \sqrt{t^2 + \epsilon}$ **or** $S(t) = \log(1 + t^2)$




**Other methods exist:** *Intrinsic Plasticity, Weights pruning, etc.*

# Adding Non-Negativity Constraints

## Another type of Auto-Encoder regularization

- Enforcing the positivity of the parameters and states of the network
- Motivation: part-based representation of the inputs (non negative) as composition of images

## Common implementation:

- Replacing the weight decay term $\frac{\lambda}{2}\sum_{s=1}^{S}||W_s||_2^2$ in the AE cost function by an asymmetric decay term $\frac{\lambda}{2}\sum_{s=1}^{S}\sum_{i,j}f(W_{s,i,j})$ with:

$$f(w_{i,j}^{(s)}) = \begin{cases} \alpha(w_{i,j}^{(s)})^2 \; if \, w_{i,j}^{(s)} < 0 \\ \gamma(w_{i,j}^{(s)})^2 \; if \, w_{i,j}^{(s)} \geq 0 \end{cases} \qquad \textit{and} \qquad 0 \leq \gamma < \alpha \leq 1$$

# A new popular approach : Variational Auto-Encoders

Using the Auto-Encoder as a generative model:

- A stochastic AE:

Input $\mathbf{x}$ marginal $p_\theta(\mathbf{x})$ → **Encoder** $p_{\theta_E}(\mathbf{h}|\mathbf{x})$ → Latent representation $\mathbf{h}$ prior $p_\theta(\mathbf{h})$ → **Decoder** $p_{\theta_D}(\mathbf{x}|\mathbf{h})$ → Input $\mathbf{x}$ marginal $p_\theta(\mathbf{x})$

In practice :

- The Encoder maps each input to the parameters of some distribution
  - Usually the mean and variance of a multi-dimensional Gaussian distribution
- The decoder map a sample from this distribution to a reconstruction of the input
- Loss of the Variational Auto-Encoder:

$$L_{VAE} = \underbrace{\boxed{-KL(p_{\theta_E}(\mathbf{h}|\mathbf{x}^{(i)})|p_\theta(\mathbf{h}))}}_{\sim \text{ Regularization}} + \underbrace{\boxed{\log(p_{\theta_D}(\hat{\mathbf{x}}^{(i)}|\mathbf{h}^{(i)}))}}_{\sim \text{ Reconstruction Loss}}$$

# Multi-Scale Morphological Decomposition

# Additive Morphological Decomposition

One of the considered Morphological Decomposition:

- From *Classification of hyperspectral images by tensor modeling and additive morphological decomposition, S. Velasco-Forero, J. Angulo*
- Given a set of *m* anti-extensive operators $\underline{\Phi}^i$, and extensive operators $\overline{\Phi}^i$ such that:

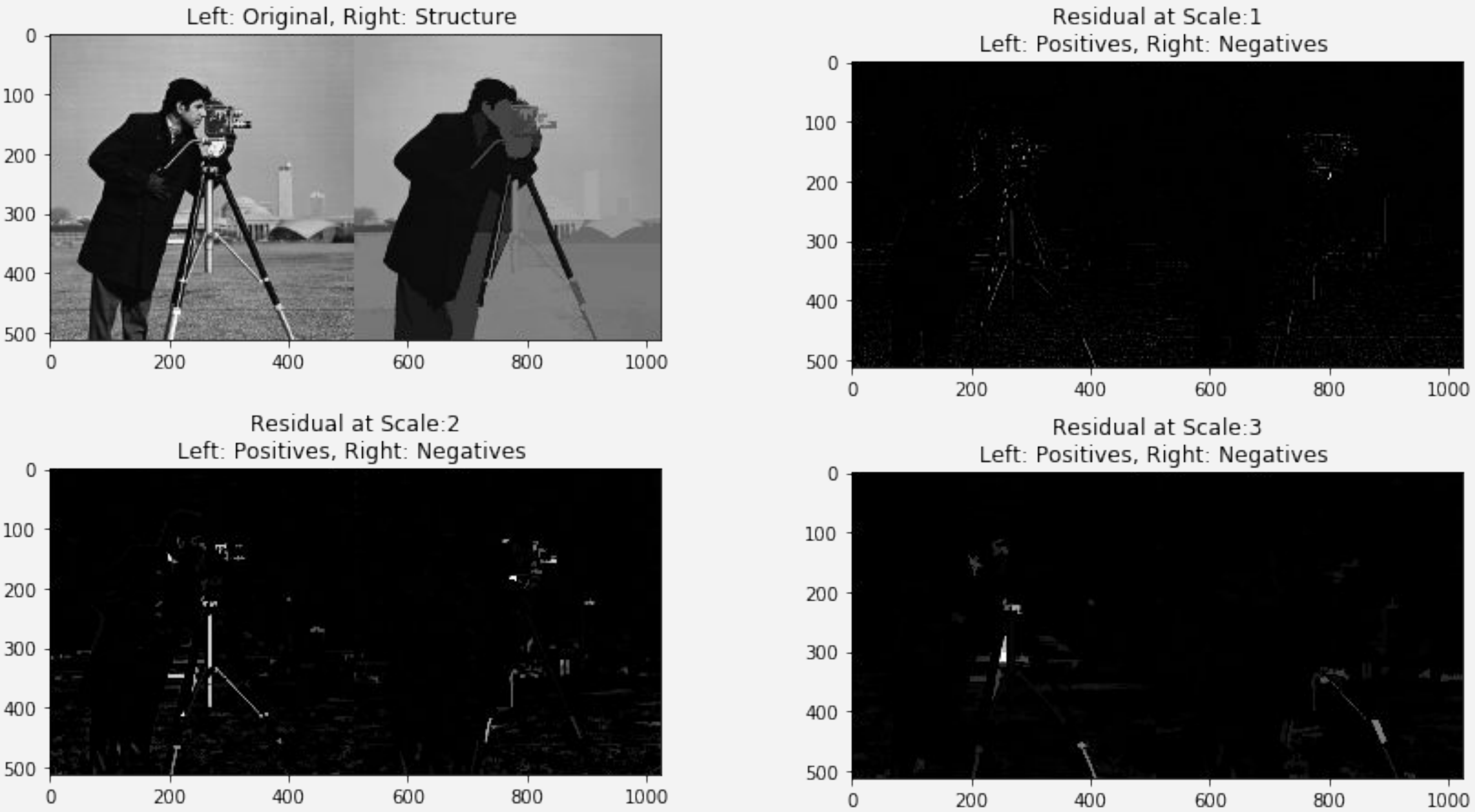
$$\underline{\Phi}^m(\underline{\Phi}^{m-1}(\mathbf{I})) \leq \ldots \leq \underline{\Phi}^2(\underline{\Phi}^1(\mathbf{I})) \leq \underline{\Phi}^1(\mathbf{I}) \leq \mathbf{I}$$
$$\overline{\Phi}^m(\overline{\Phi}^{m-1}(\mathbf{I})) \geq \ldots \geq \overline{\Phi}^2(\overline{\Phi}^1(\mathbf{I})) \geq \overline{\Phi}^1(\mathbf{I}) \geq \mathbf{I}$$

- The Additive Morphological Decomposition decomposes the image $\mathbf{I}$ as: $\mathbf{I} = S + \sum_{i=1}^{m} R_i$ with the $m$ residuals:

$$R_i = \frac{R_i^- - R_i^+}{2}$$

$$R_i^- = \underline{\Phi}^{i-1}(\underline{\Phi}^{i-2}(\mathbf{I})) - \underline{\Phi}^i(\underline{\Phi}^{i-1}(\mathbf{I}))$$
$$R_i^+ = \overline{\Phi}^i(\overline{\Phi}^{i-1}(\mathbf{I})) - \overline{\Phi}^{i-1}(\overline{\Phi}^{i-2}(\mathbf{I}))$$

- The input of our representation learning algorithm is then of size $N = d \times d \times c \times (m+1)$
- Example of morphological operators: *operator by reconstruction indexed by scale of structuring element.*


Left: Original, Right: Structure
Residual at Scale:1
Left: Positives, Right: Negatives
Residual at Scale:2
Left: Positives, Right: Negatives
Residual at Scale:3
Left: Positives, Right: Negatives
0 100 200 300 400 500
0 200 400 600 800 1000

# Implementation

# Fashion MNIST Data Set

Zalando Research proposal as a remplacement to the original MNIST data set:

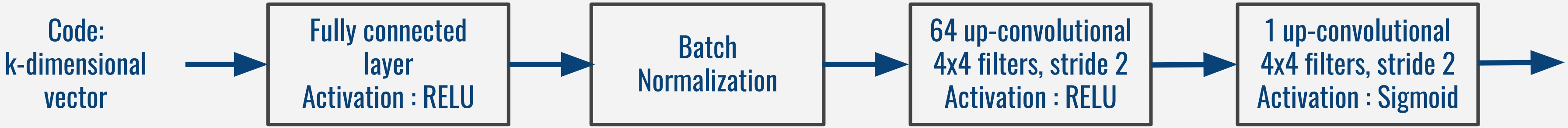
# Choice of Architecture: InfoGAN

Inspired from *InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets, Xi Chen et al.*

- Designed for representation learning
- State-of-the art deep-learning tricks: convolutional layers, Batch Normalization, lRELU activations, etc.

**Encoder:**


**Decoder:**




- Limits:
  - Network high capacity and expensive training.

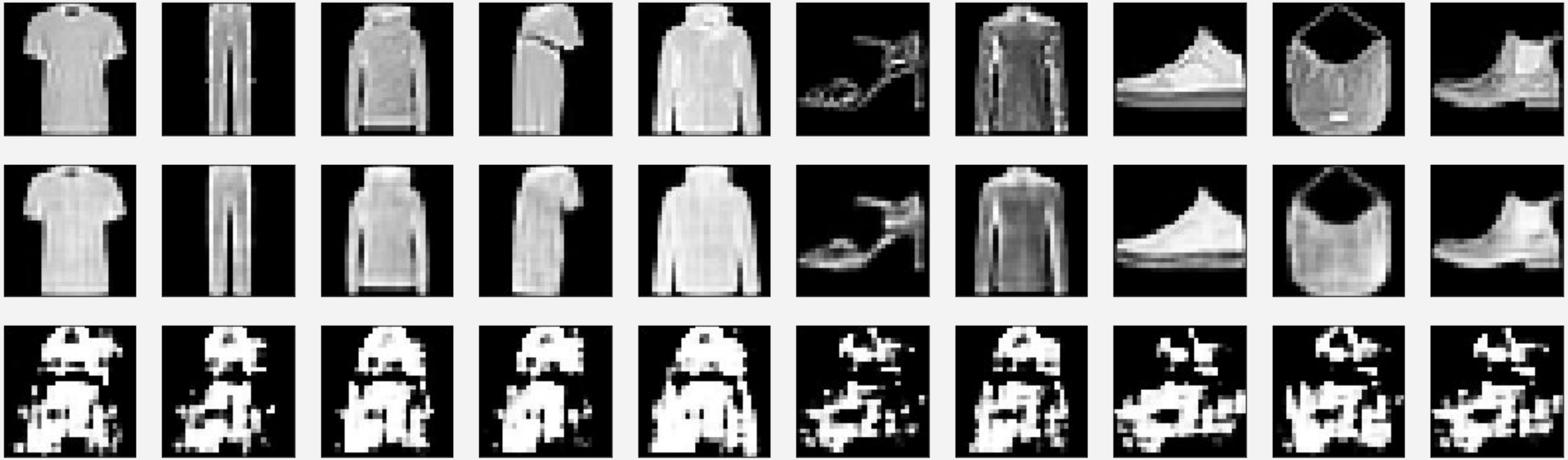
# Adding Sparsity Constraints

Implemented sparsity constraint:

- L1-regularization of activity of the last layer of the encoder

Noticed effects:

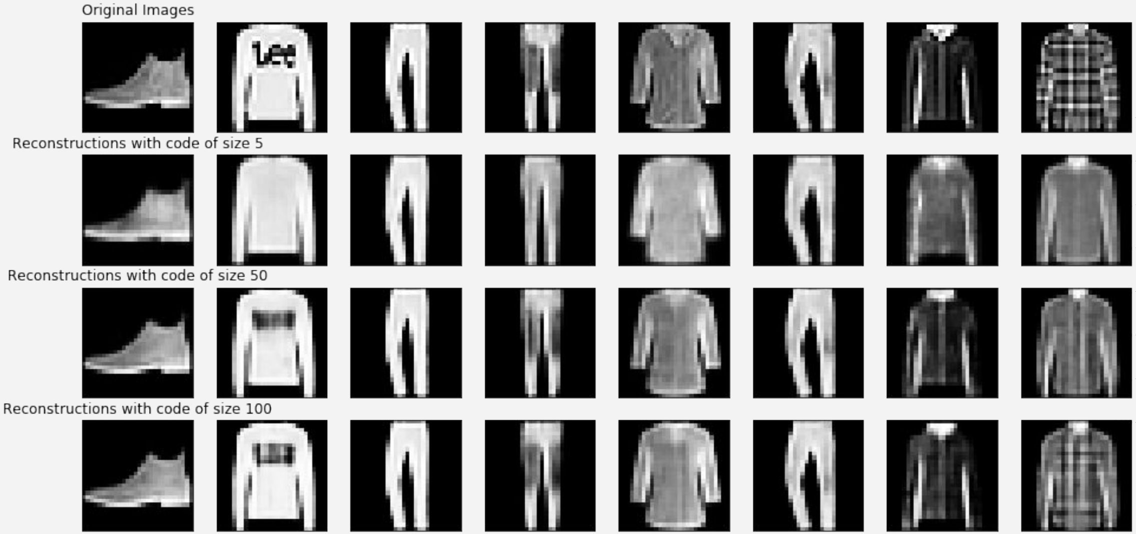
- Poor reconstruction ability.
- Collapsing of the code coefficients with very low standard deviation.
- However a t-SNE on the learned representation still show some structure in the latent space.

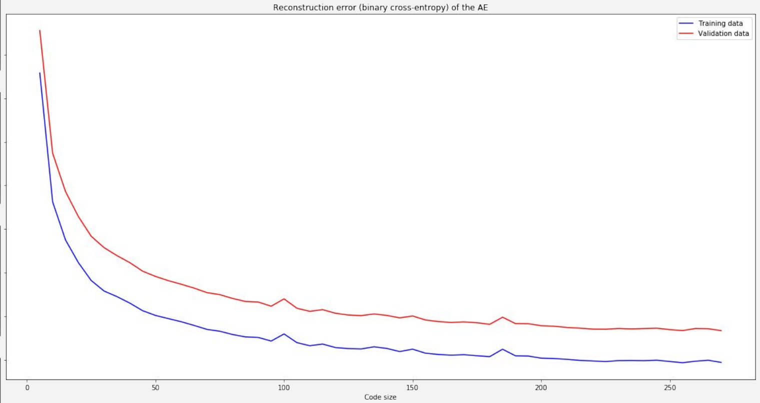


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

# How to compare encodings ?

# Reconstruction Errors



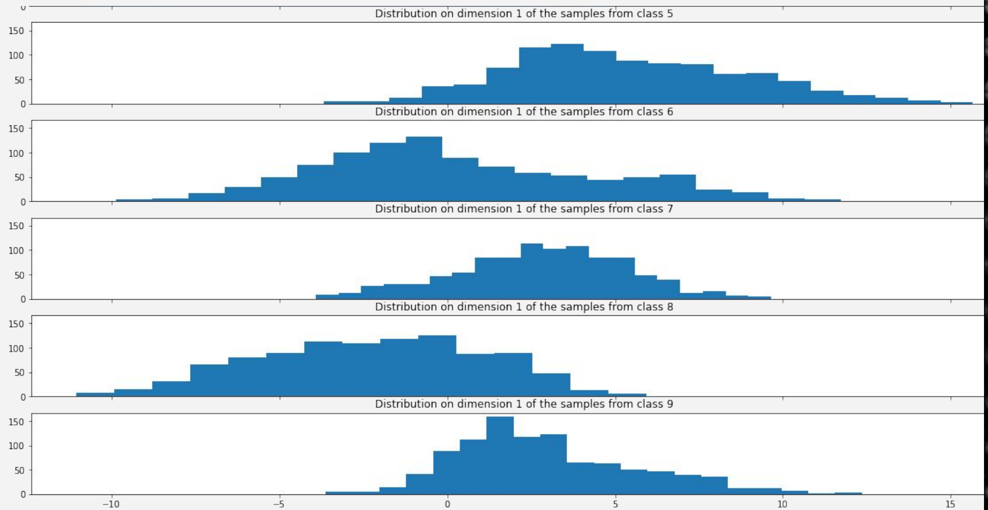



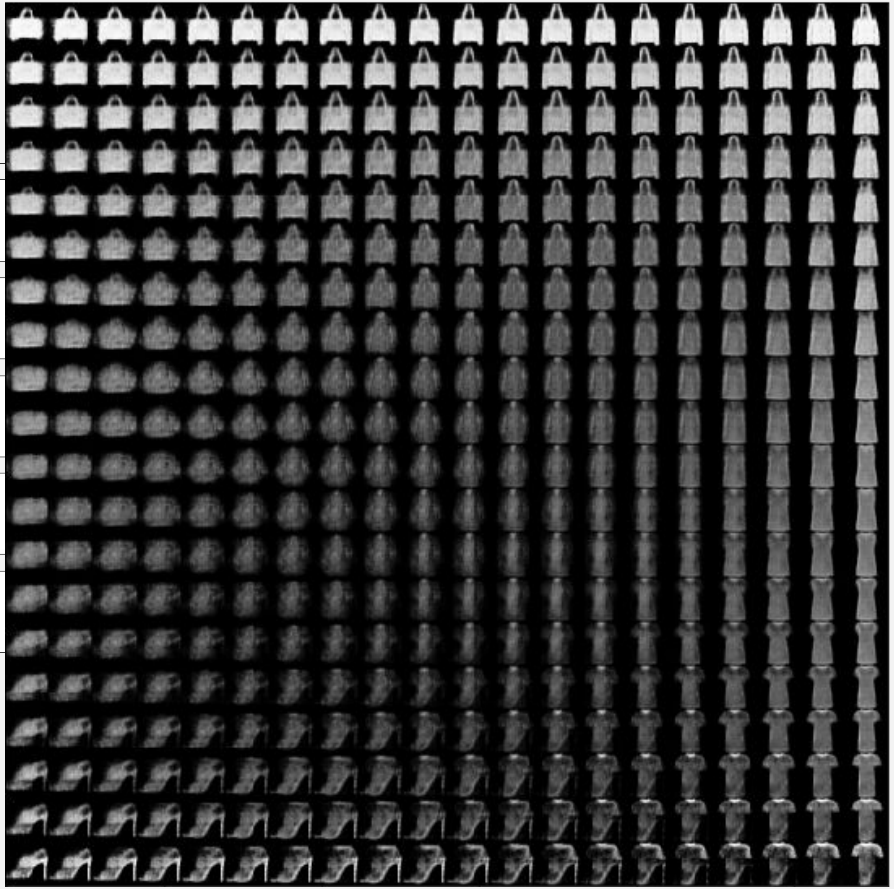

Reconstruction error is not a good metric of the efficiency of the learned representation:

- A high enough capacity can get an arbitrary good reconstruction, without learning the underlying structure in the images

## How to Compare Encodings ?

# Code Visualization



# t-SNE

**Visualizing the embedding of our data as a two or three dimensional space.**

- Iteratively minimizes the KL divergence between two probability distributions modeling the local similarity of points in both the high dimensional space and the 2-dimensional space

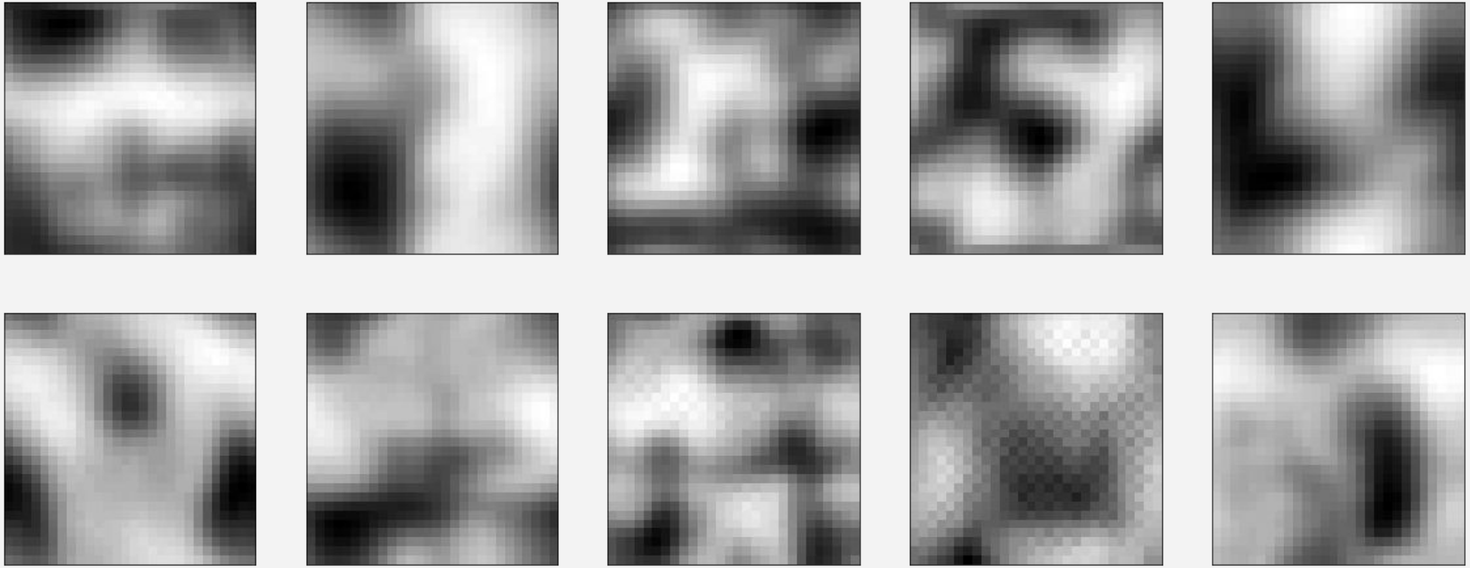
Limits:

- The algorithm converges to a local minimum
- t-SNE is sensitive to the value of a perplexity defining a notion of ‘neighborhood’ in the high dimensional space.
- Cluster sizes and distances in the visualization are meaningless.

# Activation Maximization

Generating an input image that maximize one of the components of the learned representation: $\arg\max_{\mathbf{x}} \mathbf{h}_i(\mathbf{x})$

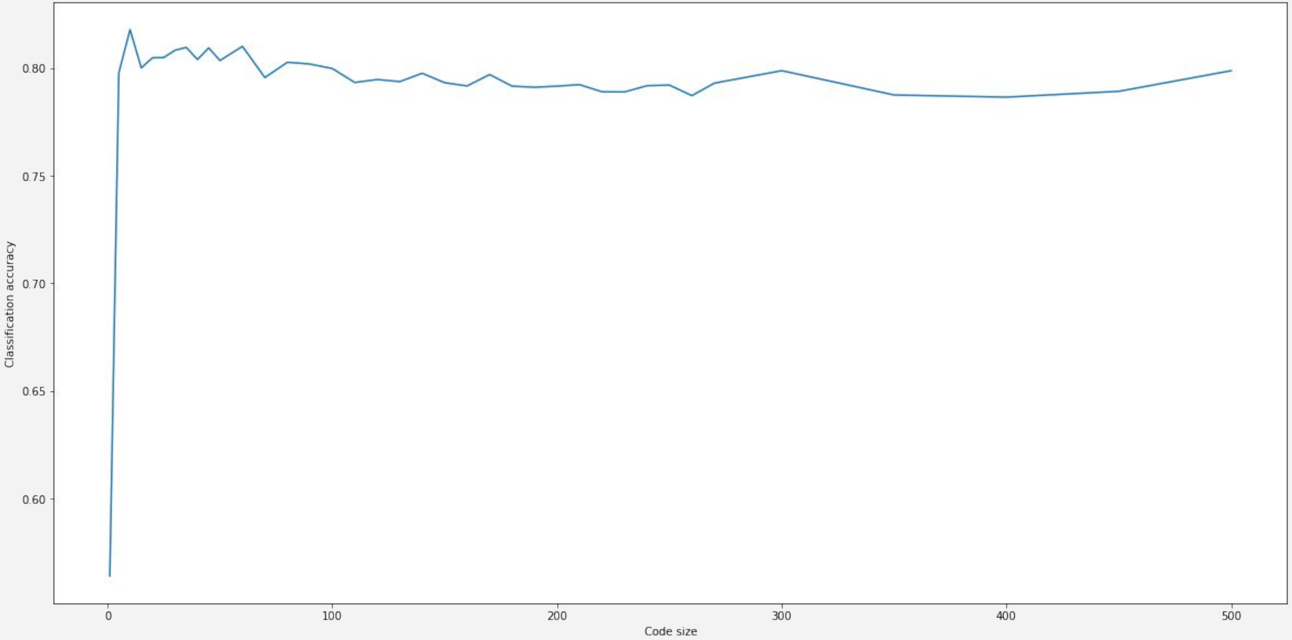
- Trying to visualize the learned structure in the images.



# Classification Results

A common way of determining the efficiency of a learned representation

- Training a classifier on the learned representations.



# Approximation of the learned encoding function

Approximating the function learned by the Encoder by other known function.

- Approximation with a linear function: comparing the input and the reconstruction with $\tilde{\mathbf{x}} = \tilde{\mathbf{W}}\mathbf{h}$ where $\tilde{\mathbf{W}} = \mathbf{X}\mathbf{H}^{\dagger}$

Example:

- infoGAN architecture with code of size 5:

$$||\tilde{\mathbf{X}}_{\mathbf{test}} - \mathbf{X}_{\mathbf{test}}|| = 1029.5$$

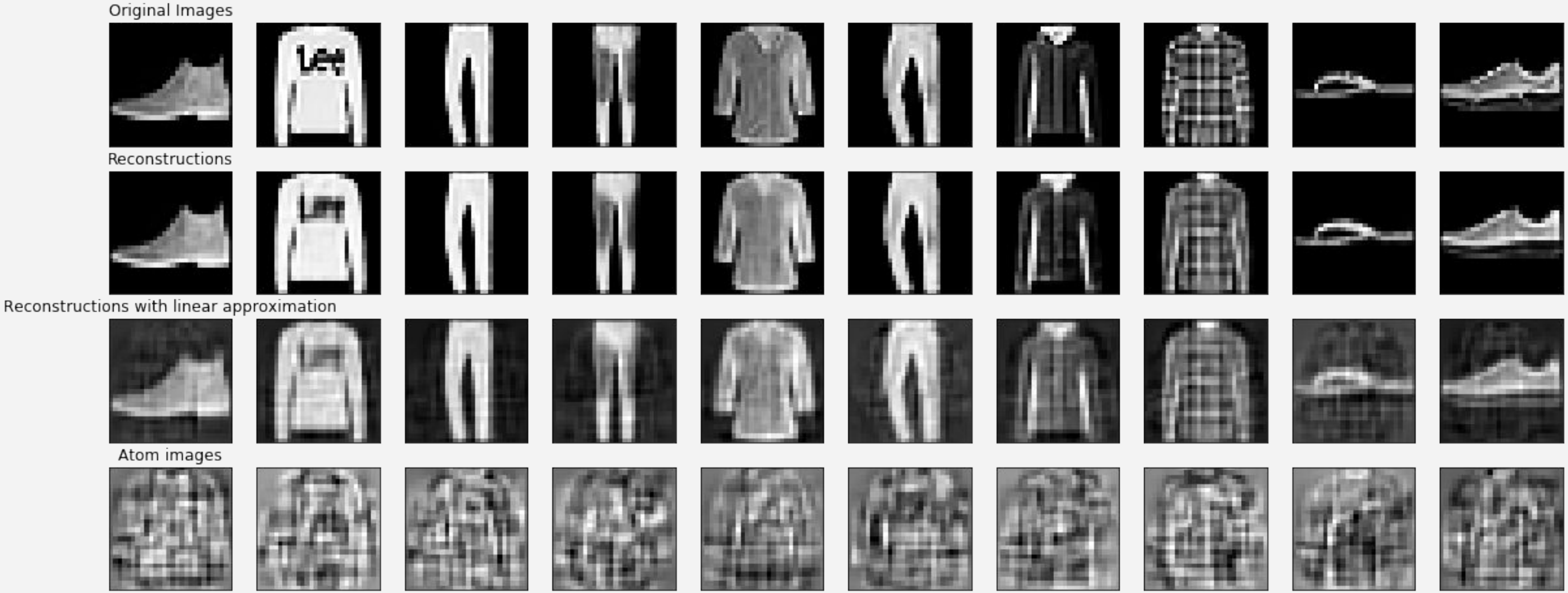
$$||\tilde{\mathbf{X}}_{\mathbf{test}} - \hat{\mathbf{X}}_{\mathbf{test}}|| = 955.4$$

Decreases with code size

- Linear and shallow Auto-Encoder with code of size 5:

$$||\tilde{\mathbf{X}}_{\mathbf{test}} - \mathbf{X}_{\mathbf{test}}|| = 543.2$$

$$||\tilde{\mathbf{X}}_{\mathbf{test}} - \hat{\mathbf{X}}_{\mathbf{test}}|| = 496.2$$


Original Images
Reconstructions
Reconstructions with linear approximation
Atom images

# Conclusion

# To Do List

- Coming back to more shallow network architecture:
  - Dictionary images visualization abilities.
  - A decomposition closer to linear.
- Implementation and testing of several sparsity regularizers.
- Implementation and testing of several morphological decomposition.