Assignment 2

Learning representations

Benjamin Negrevergne, Laurent Meunier

PSL University – Paris Dauphine – Équipes *MILES*

Outline

1. What is a good representation
2. Learning representations with Deep Learning
3. Making sense of learned representations
4. Assignments: learning useful embeddings
   - Translation with word embeddings
   - Variational autoencoders
   - Normalizing flow

Reptile classification challenge

Task: classify pictures of crocodiles and alligators

■ **Option 1: Use raw features**

$\rightarrow \quad f_1 \quad \rightarrow$ croc

Reptile classification challenge

Task: classify pictures of crocodiles and alligators

Option 1: Use raw features

 $\rightarrow$ f_1 $\rightarrow$ gator

Reptile classification challenge

Task: classify pictures of crocodiles and alligators

- **Option 1: Use raw features**

$\rightarrow \quad f_1 \quad \rightarrow$ gator

- **Option 2: Use pre-processed features**

$$\begin{bmatrix} beast\_color & = & Light \\ beast\_size & = & Large \end{bmatrix} \rightarrow f_2 \rightarrow \text{croc}$$

Reptile classification challenge

Task: classify pictures of crocodiles and alligators

- **Option 1: Use raw features**

 $\longrightarrow \quad f_1 \quad \longrightarrow$ gator

- **Option 2: Use pre-processed features**

$$\begin{bmatrix} beast\_color & = & Dark \\ beast\_size & = & Small \end{bmatrix} \longrightarrow \quad f_2 \quad \longrightarrow \text{gator}$$

Reptile classification challenge

Task: classify pictures of crocodiles and alligators

Option 1: Use raw features

$\longrightarrow \quad f_1 \quad \longrightarrow \quad$ gator

Option 2: Use pre-processed features

$$\begin{bmatrix} \textit{beast\_color} & = & \textit{Dark} \\ \textit{beast\_size} & = & \textit{Small} \end{bmatrix} \longrightarrow \quad f_2 \quad \longrightarrow \quad \text{gator}$$

Which **representation** of the input is easier to deal with ? Why ?

What's the difference?

$f_1 \quad : \quad \mathbb{R}^{w \times h \times 3} \quad \rightarrow \quad \mathbb{R}$

Input space 1: $\mathbb{R}^{w \times h \times 3}$

$f_2 \quad : \quad \mathbb{R}^2 \quad \rightarrow \quad \mathbb{R}$

Input space 2: $\mathbb{R}^2$

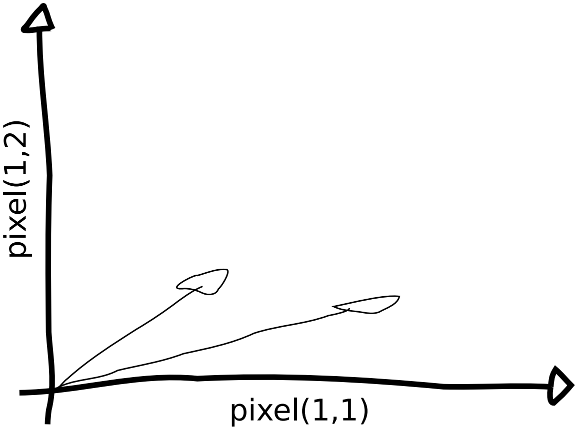
What's the difference?

$f_1 \quad : \quad \mathbb{R}^{w \times h \times 3} \quad \rightarrow \quad \mathbb{R}$

Input space 1: $\mathbb{R}^{w \times h \times 3}$

$f_2 \quad : \quad \mathbb{R}^2 \quad \rightarrow \quad \mathbb{R}$

Input space 2: $\mathbb{R}^2$



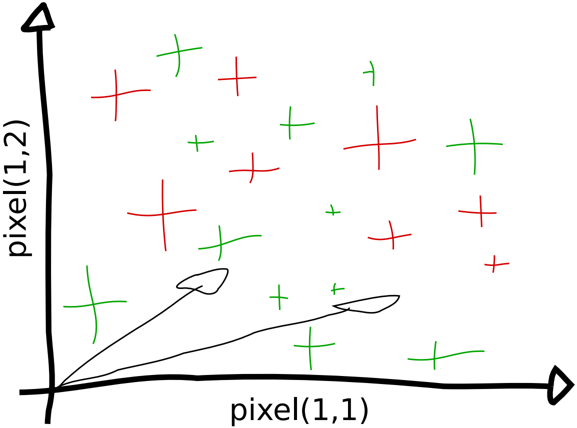
What's the difference?

$$f_1 \quad : \quad \mathbb{R}^{w \times h \times 3} \quad \rightarrow \quad \mathbb{R}$$

Input space 1: $\mathbb{R}^{w \times h \times 3}$

$$f_2 \quad : \quad \mathbb{R}^{2} \quad \rightarrow \quad \mathbb{R}$$

Input space 2: $\mathbb{R}^2$



f_1 inputs **are not linearly separable** :(

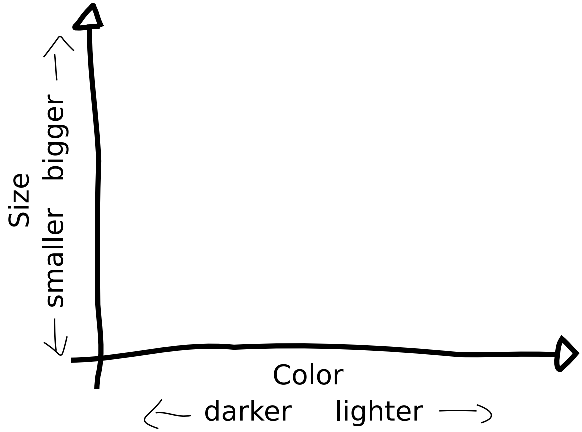
What's the difference?

$f_1 \quad : \quad \mathbb{R}^{w \times h \times 3} \quad \rightarrow \quad \mathbb{R}$

Input space 1: $\mathbb{R}^{w \times h \times 3}$

$f_2 \quad : \quad \mathbb{R}^2 \quad \rightarrow \quad \mathbb{R}$

Input space 2: $\mathbb{R}^2$



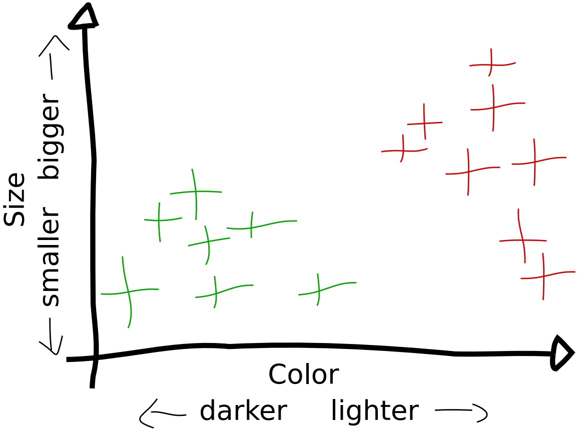
What's the difference?

$f_1 \quad : \quad \mathbb{R}^{w \times h \times 3} \quad \to \quad \mathbb{R}$

Input space 1: $\mathbb{R}^{w \times h \times 3}$

$f_2 \quad : \quad \mathbb{R}^2 \quad \to \quad \mathbb{R}$

Input space 2: $\mathbb{R}^2$



f_2 Inputs **are linearly separable** in $\mathbb{R}^2$:)

Pro/cons

- **Representation 1: raw pixel data**
  - \+ Contains all the information available
  - − Input data points are not (nearly) linearly separable
    Features are individually non-discriminant
  - − Difficult to process with simple models

- **Representation 2: high-level features**
  - \+ Input data points are (almost) linearly separable
    Individual features are highly discriminant
  - \+ Can be processed with simple (linear) models
  - − Requires expertise and manual labor to be built

Pro/cons

- **Representation 1: raw pixel data**
  - \+ Contains all the information available
  - – Input data points are not (nearly) linearly separable
    Features are individually non-discriminant
  - – Difficult to process with simple models

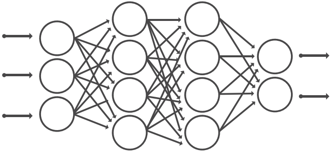
- **Representation 2: high-level features**
  - \+ Input data points are (almost) linearly separable
    Individual features are highly discriminant
  - \+ Can be processed with simple (linear) models
  - – Requires expertise and manual labor to be built

Can we build high level representations automatically ?

Outline

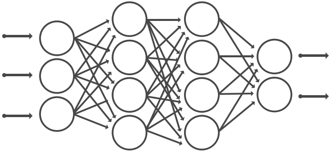
1. What is a good representation
2. Learning representations with Deep Learning
3. Making sense of learned representations
4. Assignments: learning useful embeddings
   - Translation with word embeddings
   - Variational autoencoders
   - Normalizing flow

Deep learning



$$f = f_1 \circ f_2 \circ f_3$$

Deep learning



$$f = f_1 \circ f_2 \circ \ldots \circ f_{n-1} \circ f_n$$

$$\begin{array}{llll} f_1 & : & \mathcal{X} & \rightarrow & \mathcal{Z}_1 \\ f_i & : & \mathcal{Z}_i & \rightarrow & \mathcal{Z}_{i+1} \\ f_n & : & \mathcal{Z}_{n-1} & \rightarrow & \mathcal{Y} \end{array}$$

- $\mathcal{X}$: input space
- $\mathcal{Z}_1, \ldots, \mathcal{Z}_{n-1}$: latent spaces
- $\mathcal{Y}$: output space

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

What is h?

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

What is h?

- A linear classifier: $\mathcal{Z}_{n-1} \rightarrow \mathcal{Y}$

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

What is h?

- A linear classifier: $\mathcal{Z}_{n-1} \to \mathcal{Y}$

What is g?

Remarks:

- Inputs x **are not** linearly separable in $\mathcal{X}$

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

What is h?

- A linear classifier: $\mathcal{Z}_{n-1} \rightarrow \mathcal{Y}$

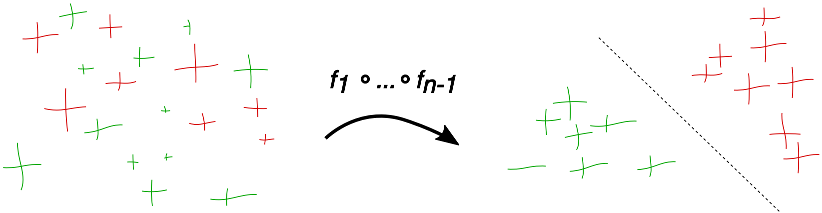
What is g?

Remarks:

- Inputs x **are not** linearly separable in $\mathcal{X}$
- Outputs $g(x)$ **are** linearly separable in $\mathcal{Z}_{n-1}$ (because h is a linear classifier)

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$



Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

What is h?

- A linear classifier: $\mathcal{Z}_{n-1} \to \mathcal{Y}$

What is g?

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

What is h?

- A linear classifier: $\mathcal{Z}_{n-1} \rightarrow \mathcal{Y}$

What is g?

- a function that maps variables from the input space $\mathcal{X}$ into a latent space $\mathcal{Z}_{n-1}$
- such that images $g(x)$ are linearly separable in $\mathcal{Z}_{n-1}$

Deep learning

$$f = \underbrace{f_1 \circ f_2 \circ \ldots \circ f_{n-1}}_{g} \circ \underbrace{f_n}_{h}$$

What is h?

- A linear classifier: $\mathcal{Z}_{n-1} \to \mathcal{Y}$

What is g?

- a function that maps variables from the input space $\mathcal{X}$ into a latent space $\mathcal{Z}_{n-1}$
- such that images $g(x)$ are linearly separable in $\mathcal{Z}_{n-1}$

Can we call $g(x)$ a *high level representation* of x?

Outline

1. What is a good representation
2. Learning representations with Deep Learning
3. Making sense of learned representations
4. Assignments: learning useful embeddings
   - Translation with word embeddings
   - Variational autoencoders
   - Normalizing flow

Meaningful distances

- D: a database with 80 000 pictures.
- $f = g \circ h$: a classifier trained on object recognition
  g non-linear function, h linear classifier
- x a random picture from the internet

$$x_1 = \arg\min_{x' \in D} ||g(x) - g(x')||$$

$$x_2 = \arg\min_{x' \in D \setminus \{x_1\}} ||g(x) - g(x')||$$

Meaningful distances

- D: a database with 80 000 pictures.
- $f = g \circ h$: a classifier trained on object recognition
  g non-linear function, h linear classifier
- x a random picture from the internet

$$x_1 = \arg\min_{x' \in D} ||g(x) - g(x')||$$

$$x_2 = \arg\min_{x' \in D \setminus \{x_1\}} ||g(x) - g(x')||$$

x

Meaningful distances

- D: a database with 80 000 pictures.
- $f = g \circ h$: a classifier trained on object recognition
  g non-linear function, h linear classifier
- x a random picture from the internet

$$x_1 = \underset{x' \in D}{\arg\min} ||g(x) - g(x')||$$

$$x_2 = \underset{x' \in D \setminus \{x_1\}}{\arg\min} ||g(x) - g(x')||$$

x

x_1

x_2

Query by image (2)

x

Query by image (2)

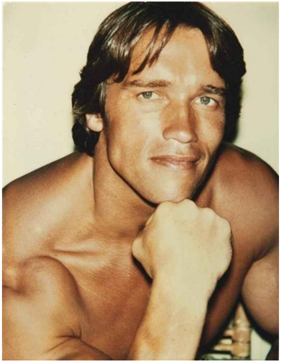
x

x_1

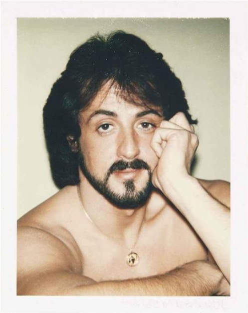
x_2

Query by image (2)

x



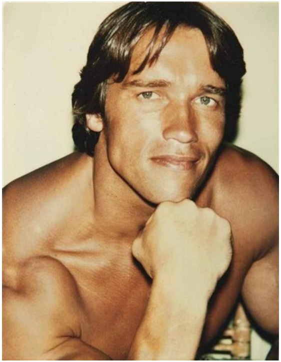
x_1



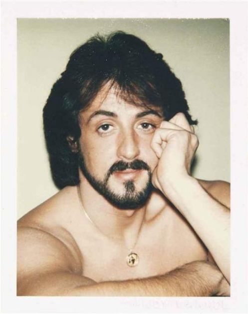
x_2

Query by image (2)

x



x_1



x_2

Result: The distance in the latent space seems to be meaningful!

More meaningful distances

Desired properties for a meaningful distance d:

1. $d(x_1, x_2)$ is small if x_1 and x_2 are from the same class
2. $d(x_1, x_3)$ is large if x_1 and x_3 are from different classes

More meaningful distances

Desired properties for a meaningful distance d:

1. $d(x_1, x_2)$ is small if x_1 and x_2 are from the same class
2. $d(x_1, x_3)$ is large if x_1 and x_3 are from different classes

In **query by image**

- $d(x_i, x_j) = \|g(x_i) - g(x_j)\|_2$

Counter example

More meaningful distances

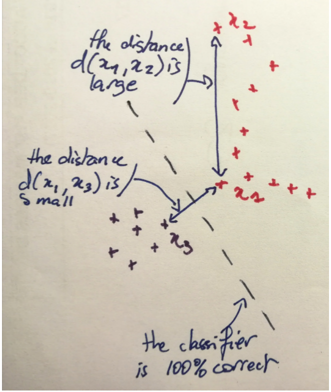
Desired properties for a meaningful distance d:

1. $d(x_1, x_2)$ is small if x_1 and x_2 are from the same class
2. $d(x_1, x_3)$ is large if x_1 and x_3 are from different classes

In **query by image**

- $d(x_i, x_j) = \|g(x_i) - g(x_j)\|_2$

▶ Neither 1 nor 2 are guaranteed if g is trained as part of a classifier



Counter example

Meaningful features

Dimensionality reduction by learning an invariant mapping.
Hadsell, R., Chopra, S., LeCun, Y. CVPR (2006)

Learns a mapping g that maps inputs with **few controlled variations** to a **low dimensional space**

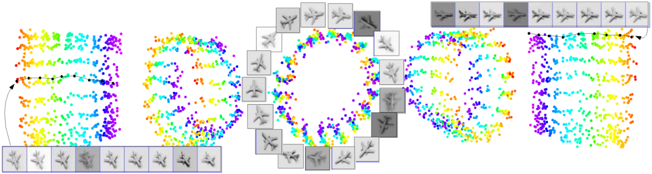
- all pictures are pictures of planes with different poses
- 9 different elevations and 18 different azimuth (orientation).
- input pictures are projected into a low 3-dimensional space

Meaningful features

Dimensionality reduction by learning an invariant mapping.
Hadsell, R., Chopra, S., LeCun, Y. CVPR (2006)

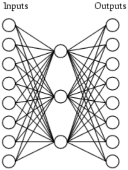
Learns a mapping g that maps inputs with **few controlled variations** to a **low dimensional space**

- all pictures are pictures of planes with different poses
- 9 different elevations and 18 different azimuth (orientation).
- input pictures are projected into a low 3-dimensional space



Result: There is a clear relation between learned features (spacial coordinates) and desired features (elevation, azimuth)

Autoencoders



A target function:

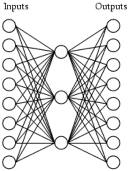
Input		Output
10000000	$\rightarrow$	10000000
01000000	$\rightarrow$	01000000
00100000	$\rightarrow$	00100000
00010000	$\rightarrow$	00010000
00001000	$\rightarrow$	00001000
00000100	$\rightarrow$	00000100
00000010	$\rightarrow$	00000010
00000001	$\rightarrow$	00000001

Can this be learned??

slides Tom Mitchell

Autoencoders

A network:



Learned hidden layer representation:

Input		Hidden Values				Output
10000000	→	.89	.04	.08	→	10000000
01000000	→	.01	.11	.88	→	01000000
00100000	→	.01	.97	.27	→	00100000
00010000	→	.99	.97	.71	→	00010000
00001000	→	.03	.05	.02	→	00001000
00000100	→	.22	.99	.99	→	00000100
00000010	→	.80	.01	.98	→	00000010
00000001	→	.60	.94	.01	→	00000001

slides Tom Mitchell

Deep representations: take aways

- DNN learn how to projet inputs into a latent space
- The input projected in the latent space is a new representation of the input that is more adequate for the task at hand.
- Understanding the features of the latent representation remains difficult (can be done in some cases)
- But comparing the distance between objects in the latent space is meaningful.
- The function g is an **embedding** of the images into a vector space because it preserves the structure (similar images are mapped to vectors close to one another).

Outline

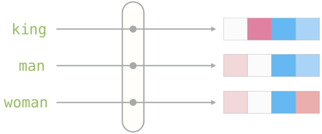
1. What is a good representation
2. Learning representations with Deep Learning
3. Making sense of learned representations
4. Assignments: learning useful embeddings
   - Translation with word embeddings
   - Variational autoencoders
   - Normalizing flow

Goal of these assignments

- Learn and manipulate high level representations (a.k.a. *embeddings*) that are useful to solve a variety of tasks.
- Become familiar with the geometric nature of deep neural networks.

Text manipulation with word embeddings

Words can be embedded in a vector space using neural networks.

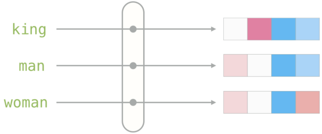


The embeddings space is geometric:

$$v(king) - v(man) + v(woman) = \ ?$$

Text manipulation with word embeddings

Words can be embedded in a vector space using neural networks.



The embeddings space is geometric:

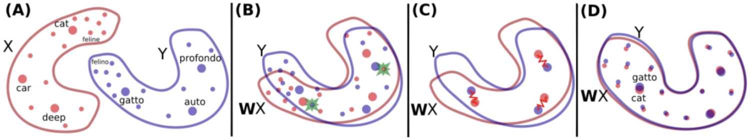
$v(king) - v(man) + v(woman) = \ ?$

Pretrained versions of word embeddings:

- GloVe
- word2vec.

click here for more info!

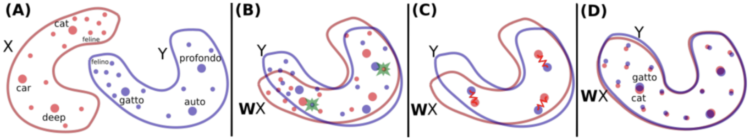
Translation with word embeddings



Exploit linear transformations and rotations to translate a word.

$$Y = \mathbf{W}X$$

Translation with word embeddings



Exploit linear transformations and rotations to translate a word.

$$Y = \mathbf{W}X$$

Learn **W**

- with a parallel corpus (e.g. supervised dataset FR-EN)
- without a parallel corpus using a GAN

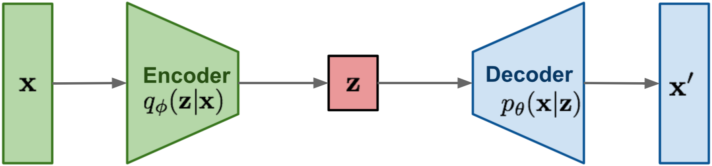
Goals of the project

- Build an efficient supervised word translator for English to French and French to English task.
- Build an unsupervised word translator.
- Optional: Compare with other language translations.
- Optional: Make a study when words come from different kinds of vocabularies.

References:

- Exploiting Similarities among Languages for Machine Translation https://arxiv.org/pdf/1309.4168.pdf
- Word Translation Without Parallel Data https://arxiv.org/pdf/1710.04087.pdf
- Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation https://aclanthology.org/N15-1104/

Data generation with autoencoders

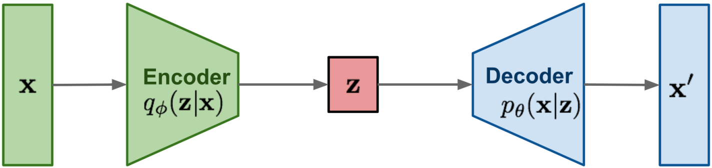


VAE, image from https://lilianweng.github.io

Autoencoder (AE):

- To generate, we can sample and decode z
- Problem?

Data generation with autoencoders

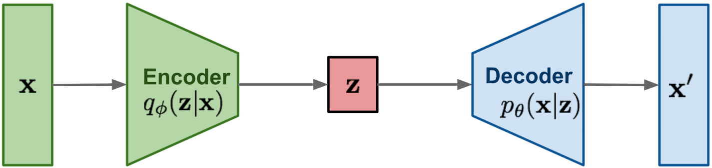


VAE, image from https://lilianweng.github.io

Autoencoder (AE):

- To generate, we can sample and decode z
- Problem? How to sample z?

Data generation with autoencoders



VAE, image from https://lilianweng.github.io

Autoencoder (AE):

- To generate, we can sample and decode z
- Problem? How to sample z?

Variational Autoencoders (VAE):

- Use a probabilistic encoder
- Regularize the latent space be as close as possible to a gaussian distribution
- To generate, sample z from a gaussian, and decode z

Goals of the project

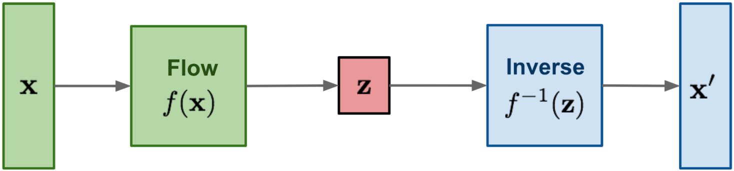
- Implement AE & VAE to generate images similar to MNIST/CIFAR
- Compare the performance of AE vs. VAE
- Add label information: Conditional (V)AEs compare both approaches
- Provide an understanding of latent spaces in VAEs.
- Compare the approach with GANs, experimentally or theoretically.
- Comparison with Wasserstein AEs

References:

- An Introduction to Variational Autoencoders
  https://arxiv.org/abs/1906.02691

Normalizing flows

With VAE: no simple way to compute (or estimate) $p(z|x)$



NF, image from https://lilianweng.github.io

Idea:

- Search for an invertible transform $f : \mathcal{X} \rightarrow \mathcal{Z}$ such that $p_Z(f(x)) \sim \mathcal{N}(0, I)$

To generate a data sample x':

- draw $z \sim \mathcal{N}(0, I)$
- decode z into x' by applying $x' = f^{-1}(z)$

Normalizing flows (2)

Benefits:

- Can perform density estimates (change of variable Theorem)

$$p_X(x) = |\det D_f(x)| p_Z(f^{-1}(x))$$

- Can be trained with max likelyhood using GD

References:

- ECCV Tutorial: *Introduction to normalizing flows*
  https://www.youtube.com/watch?v=u3vVyFVU$_I I$
- Glow: Generative Flow
  https://arxiv.org/pdf/1807.03039.pdf
- Invertible Residual Networks
  https://arxiv.org/pdf/1811.00995.pdf

Normalizing flows (3)

Goal of the project

- Use different flows GLOW with iResnet to generate image from MNIST/CIFAR
- Compare stability during training with GAN/VAEs
- Discuss the limitation of invertible networks