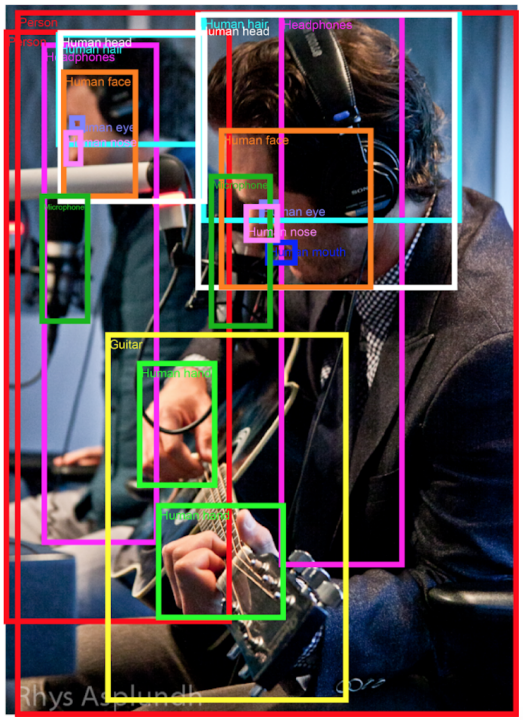
11-695: AI Engineering
Self-Supervision Introduction

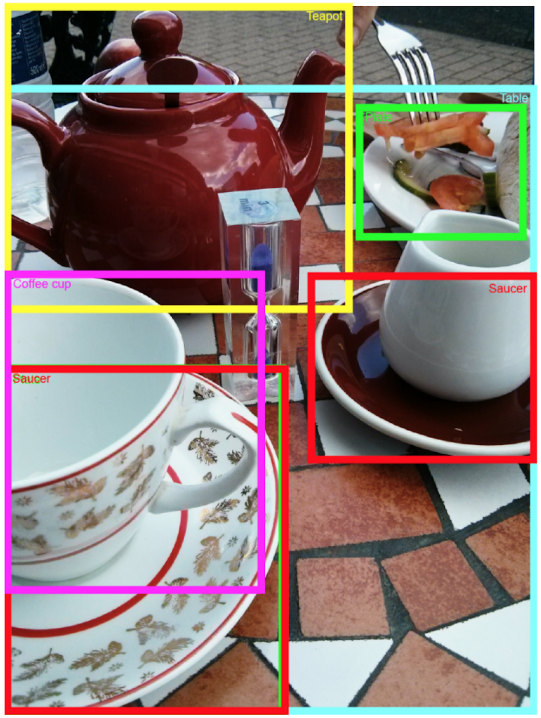
LTI/SCS

Spring 2020

Outline

Problem with Labelled Data

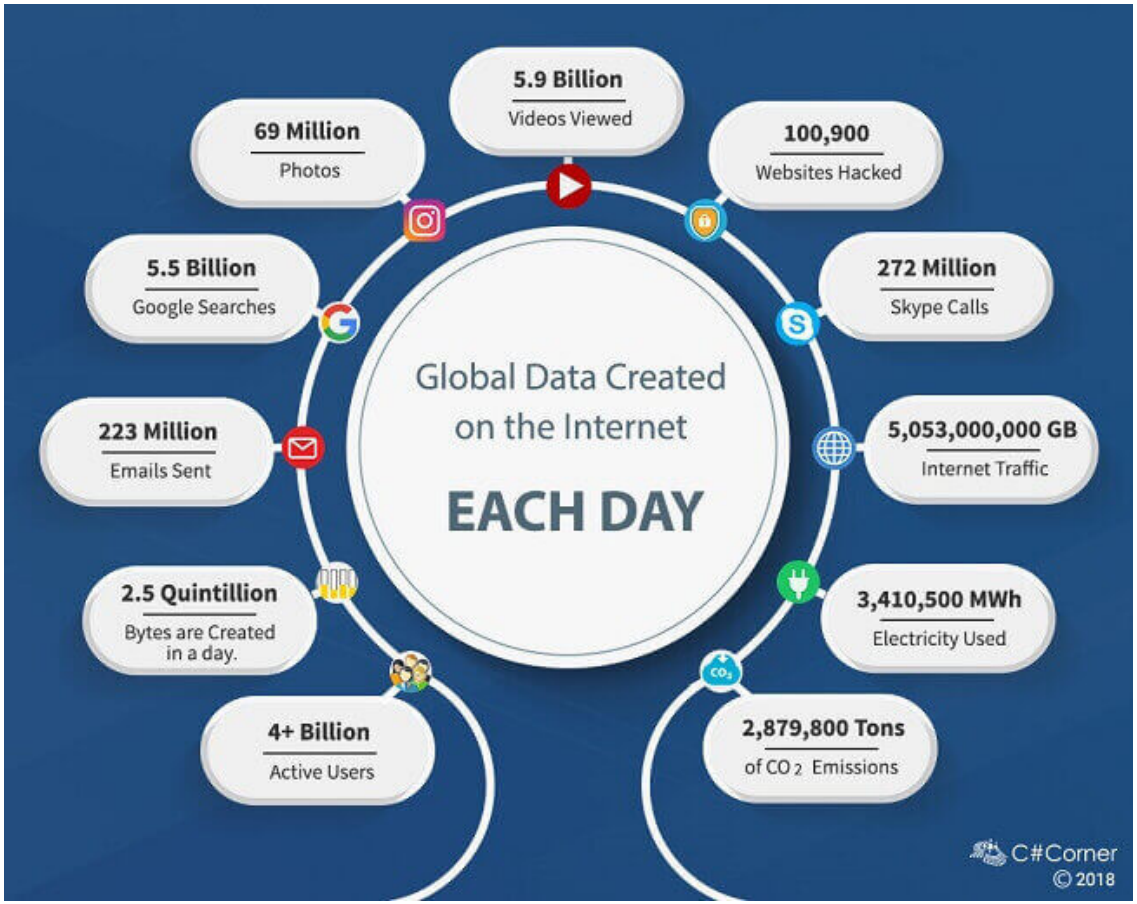




- Data is often carefully chosen,
- carefully cleaned and labelled,
- and usually *balanced*

Image credit: Google AI Blog

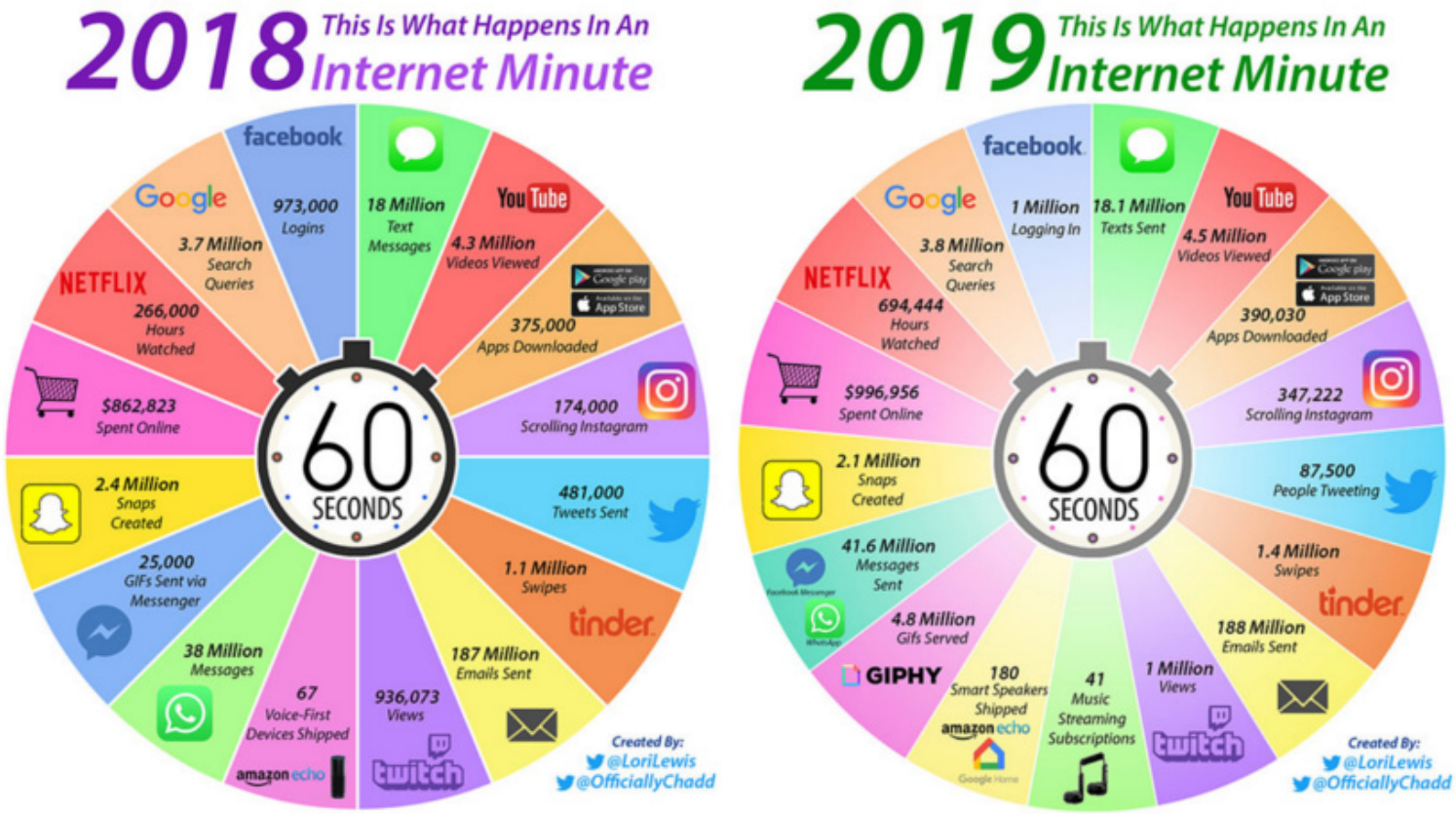
But, The World's Data is Enormous



- Data is fast increased every second

Image credit: C# Corner

Most Data is Unlabelled



- Also unstructured and unclean
- The cost to label them is prohibitively expensive.

Image credit: @LoriLewis & @OffiallyChadd - Twitter

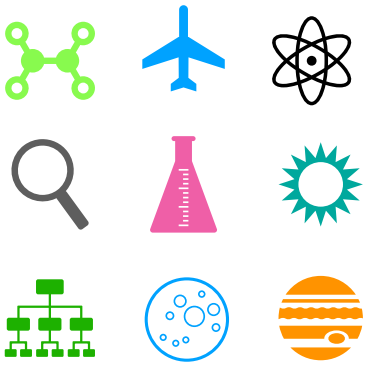
Outline

AI's Approach towards Big (Raw) Data

Carnegie Mellon

Raw Data

Model-based Features



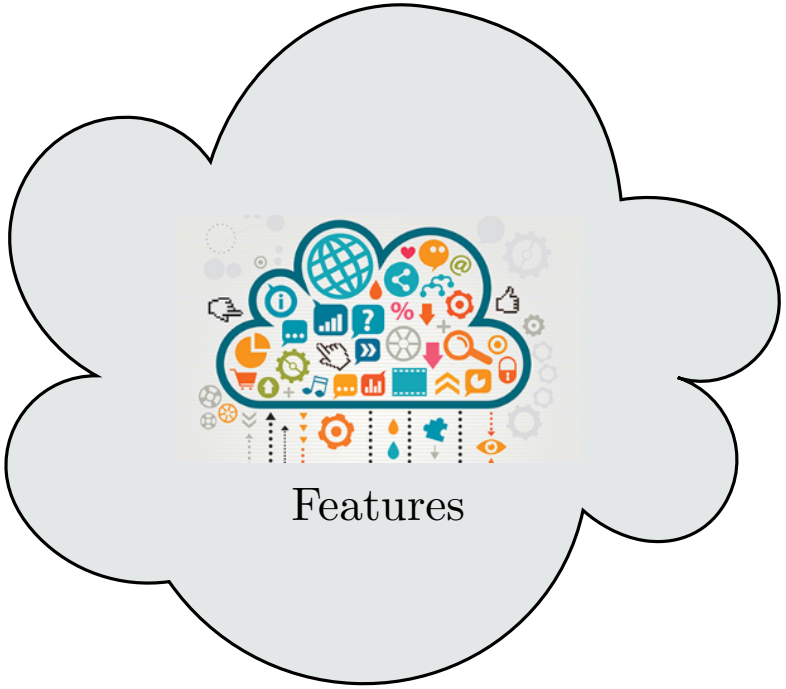
Tasks

- How about we not only focus on the task, but also put a proper focus on Data itself
- Difference from:
  - Fully labelled data: Supervised
  - Partially labelled data: Semi-supervised
- We have no labels thus no supervision → Self-supervised (Unsupervised).

All is reduced to Feature Extraction

Carnegie Mellon

Raw Data



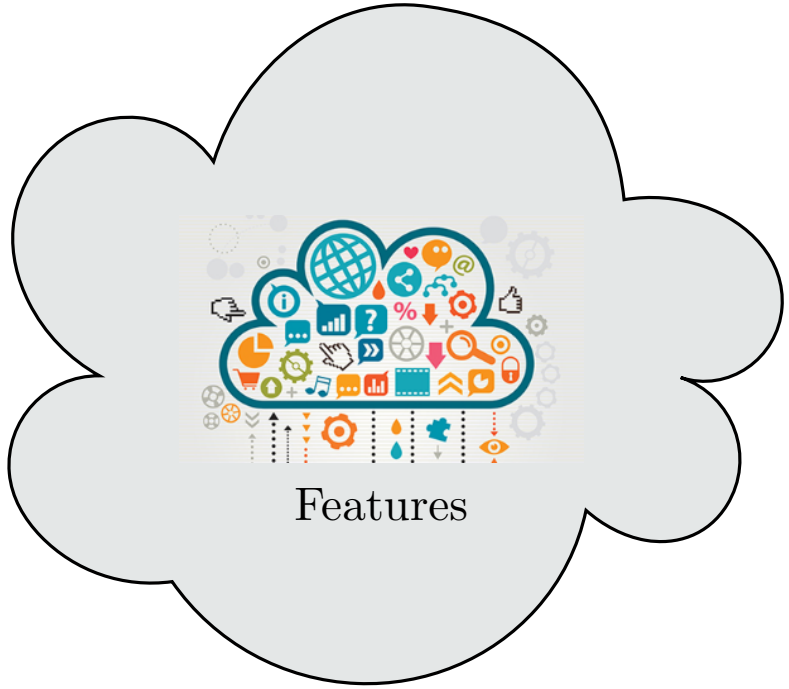
Features

Feature Space

- Classification/Regression: use features to predict labels/numbers
- Required: ground-truth labels/numbers

All is reduced to Feature Extraction

Raw Data



Features

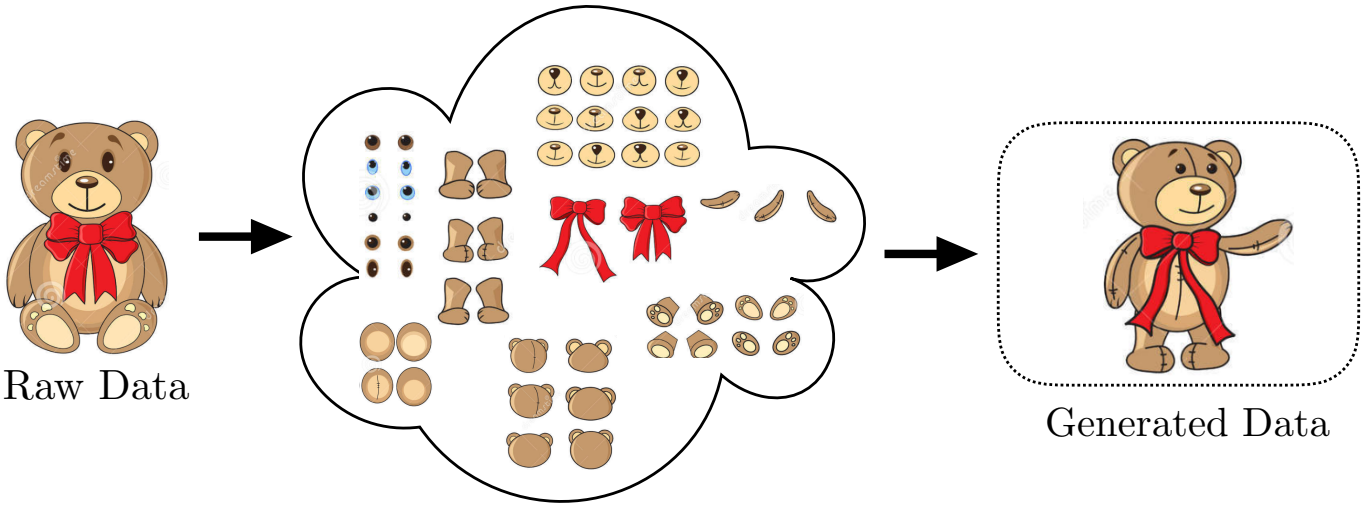
Feature Space

- Classification/Regression: use features to predict labels/numbers
- Required: ground-truth labels/numbers
- Unfortunately, we don't have such ground truths

Not really: Data are labels themselves

- Use each data sample itself as a label
  - Extract Features from x: $z = f_1(x)$
  - Reconstruct data $\hat{x}$: $\hat{x} = f_2(z)$
  - Minimize the distance $d(x, \hat{x})$, *a.k.a* reconstruction error.
- But, data itself are often highly-dimensional $\rightarrow$ (very) **hard!**
- Inject prior human knowledge/expertise about data, *e.g.*
  - Vision: use CNN as feature extractor
  - Text, sequential data: use RNN instead

Generate a bear to just classify a bear?

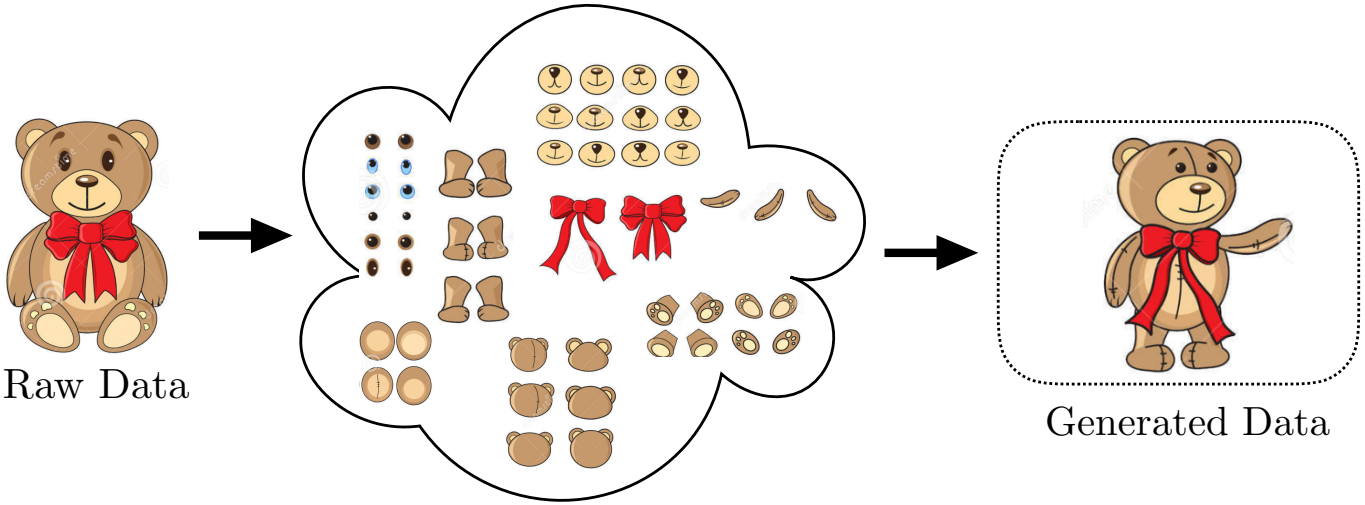


- No
  - No in cases such as classification
  - In many cases, yes, we need to actually generate data
- Regardless the task, we need to have good data features[1]
- Remind: for supervised, we don't need to generate data
  - Features needn't be well-structured to predict labels

[1] https://arxiv.org/pdf/1206.5538.pdf

Icons credit: Dreamstime.com

Generate a bear to just classify a bear?

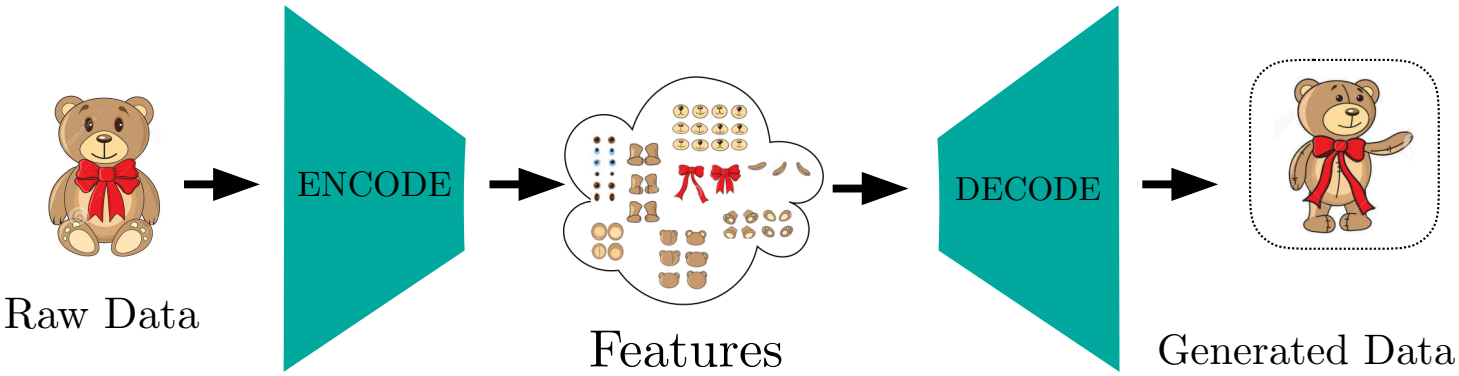


- Remind: for supervised, we don't need to generate data
  - Features needn't be well-structured to predict labels
- Comparison to Unsupervised in the classification/regression:
  - Supervised: we model $p(y|x)$
  - Unsupervised: $p(x, y)$

Icons credit: Dreamstime.com

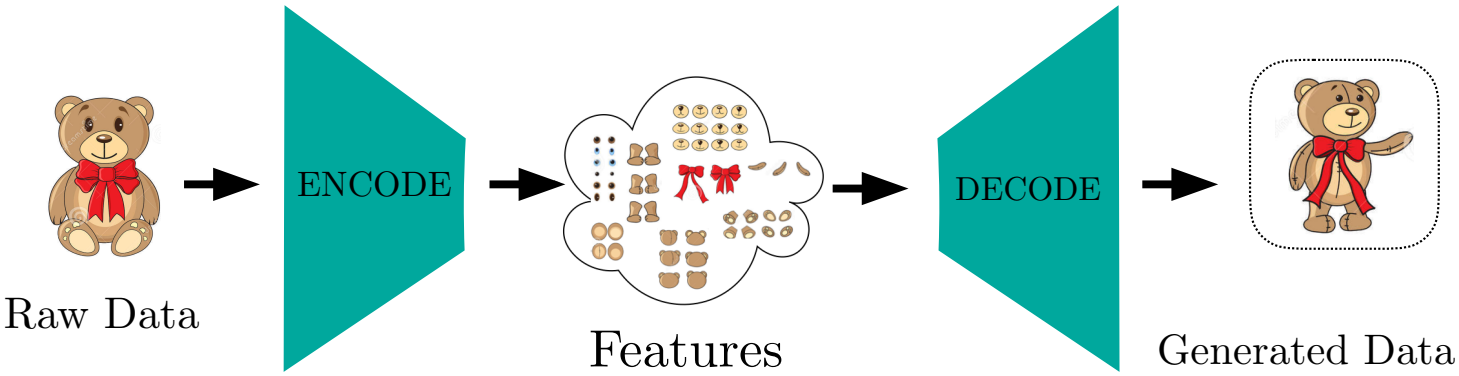
Outline

AutoEncoder



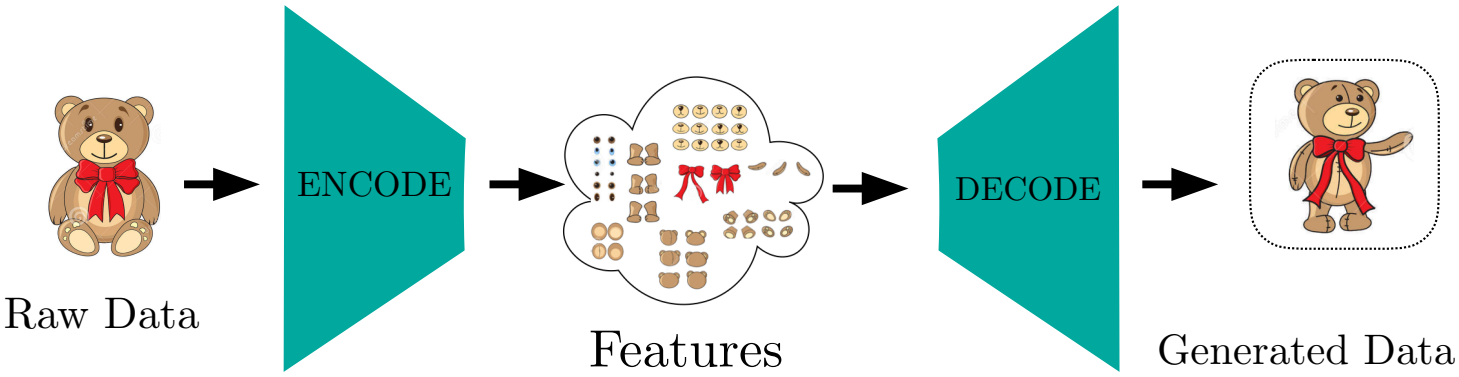
- Extract and Reconstruct *itself* → *Auto*-Encoder + Decoder
- What should the dimensionality of Features be?

AutoEncoder



- Extract and Reconstruct *itself* → *Auto*-Encoder + Decoder
- What should the dimensionality of Features be?
  - Smaller than Input! Why?
  - So encoding (feature extraction) process is a compression

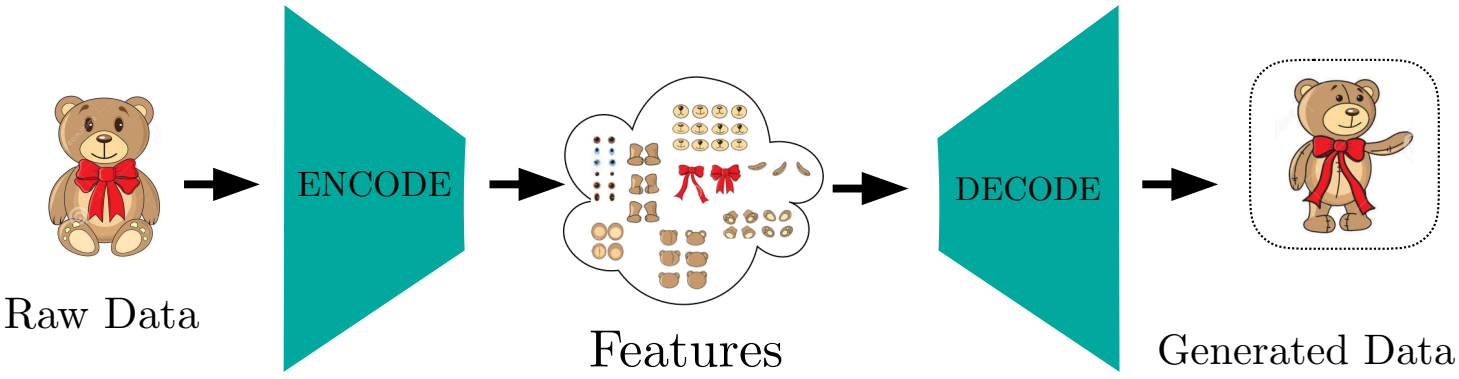
AutoEncoder



- Extract and Reconstruct *itself* → *Auto*-Encoder + Decoder
- What should the dimensionality of Features be?
  - Smaller than Input! Why?
  - So encoding (feature extraction) process is a compression
- Compression: we need to learn prominent features (factors) only
  - Yet should be enough to reconstruct data

AutoEncoder

Carnegie Mellon



- Extract and Reconstruct *itself* → *Auto*-Encoder + Decoder
- What should the dimensionality of Features be?
  - Smaller than Input! Why?
  - So encoding (feature extraction) process is a compression
- Compression: we need to learn prominent features (factors) only
  - Yet should be enough to reconstruct data
  - How? Unless for trivial cases, we need params θ

Reduced to Feature Extraction (again)

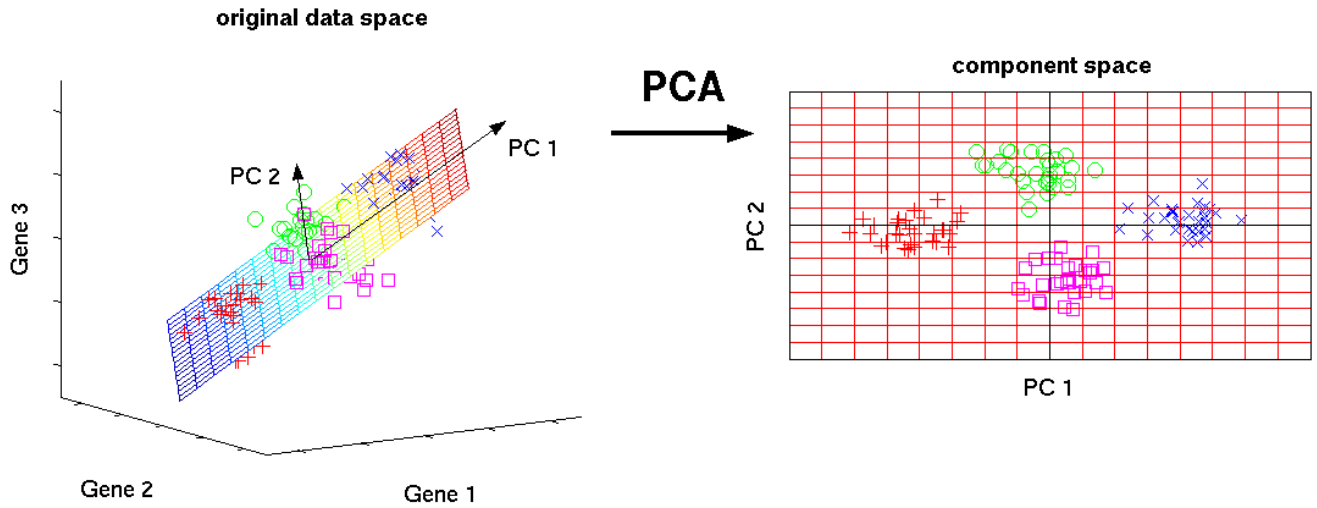
Carnegie Mellon

How to *best* extract features from data?

Outline

Principal Components Analysis (PCA)

Carnegie Mellon

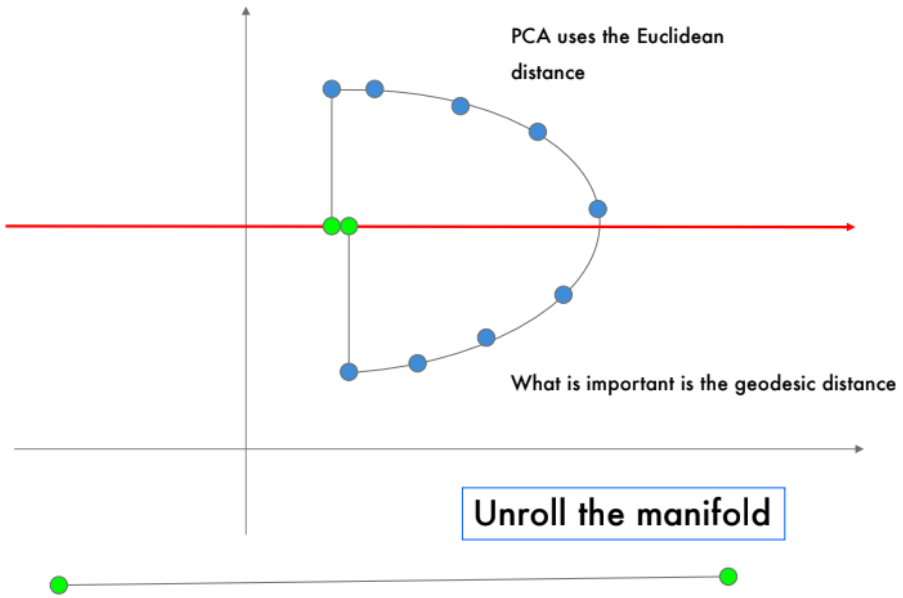


- Each Principal Component is a feature, and orthogonal to the rest
- Find a $s.t.$ $x^*a = \lambda a$ where λ is a scalar
  - One solution: perform SVD for x
  - Take only first few eigenvectors (hence approximation)

Image credit: Matthias Scholz, Ph.D. thesis

Principal Components Analysis (PCA)

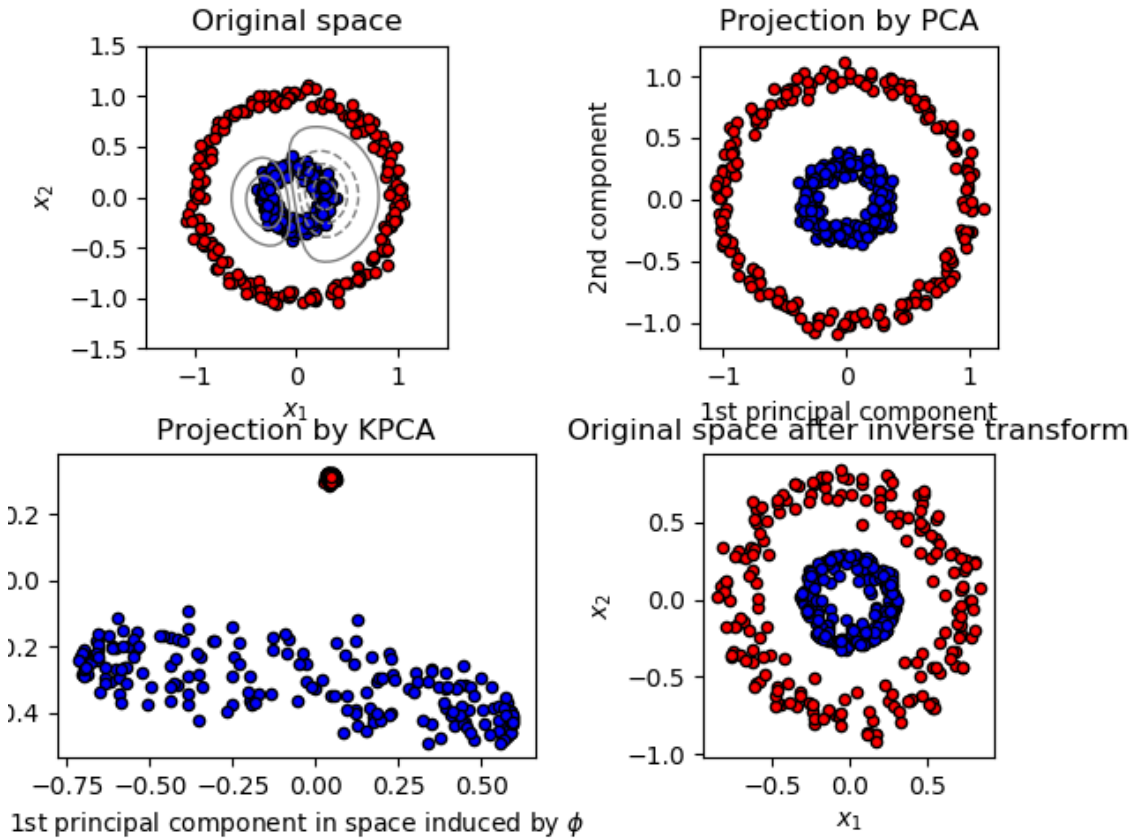
Carnegie Mellon



- Limitation: it's a linear method

Image credit: Barnabas Poczos

Kernel PCA[2]



- Recall: why do we use a kernel?
- Still a PCA, and have to carefully choose the kernel

[2] https://en.wikipedia.org/wiki/Kernel_principal_component_analysis

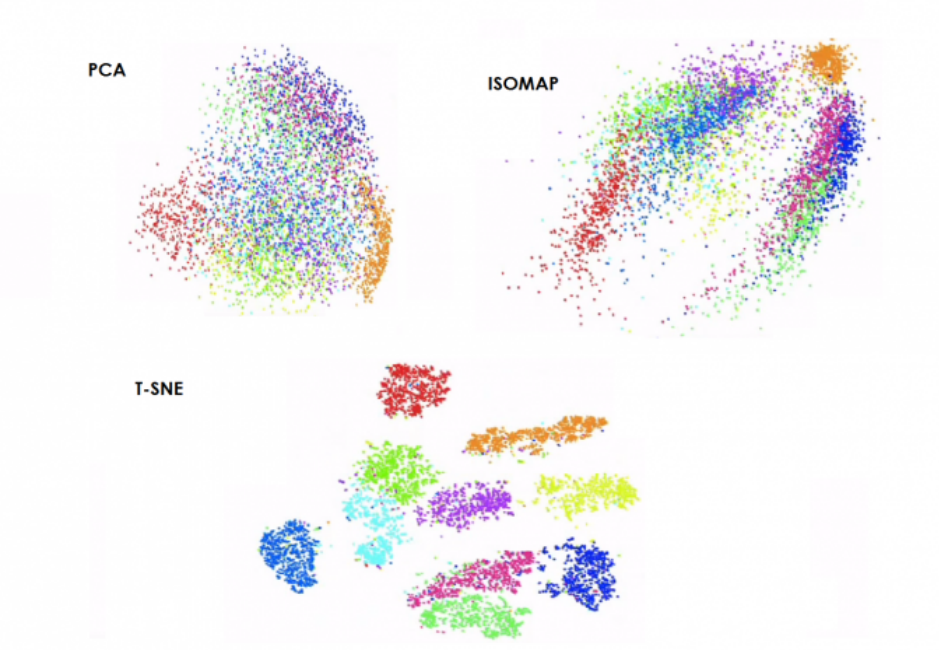
Image credit: scikitlearn

Decoded PCA (Generated Data)

- Number of eigenvectors taken: 2000, 1000, 500, 100, 50, 10, 2, 1

Image credit: Gene Kogan

t-SNE[3]

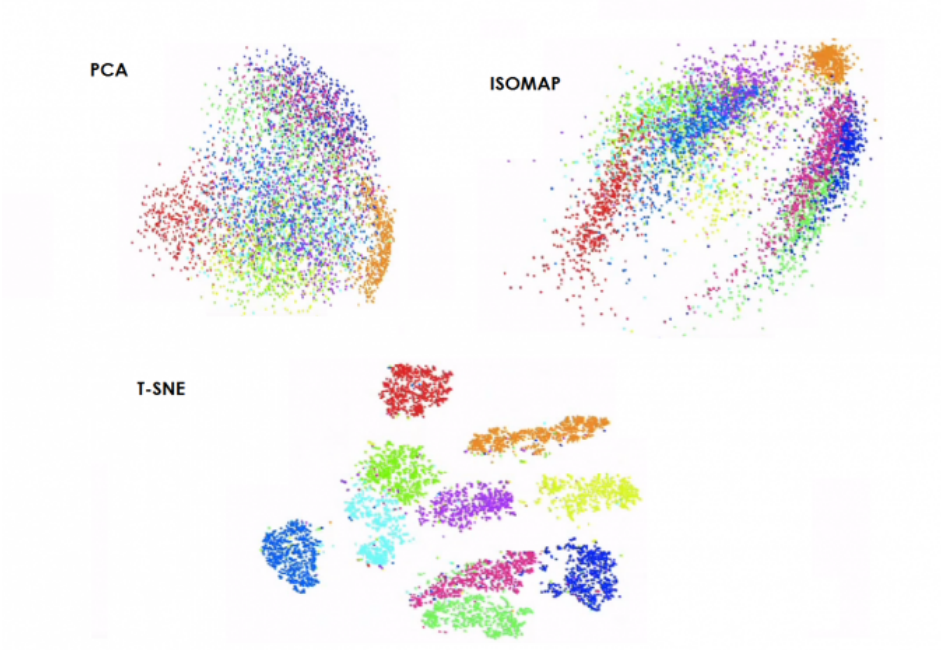


- Take into account relative distances between data points
- Is a non-linear method
- Both PCA and t-SNE suffer in high dimensional space.

[3] https://lvdmaaten.github.io/publications/papers/JMLR_2008.pdf

Image credit: quantdare.com

t-SNE[3]



- Take into account relative distances between data points
- Is a non-linear method
- Both PCA and t-SNE suffer in high dimensional space.
- *It's time for Deep NN to be used!*

[3] https://lvdmaaten.github.io/publications/papers/JMLR_2008.pdf

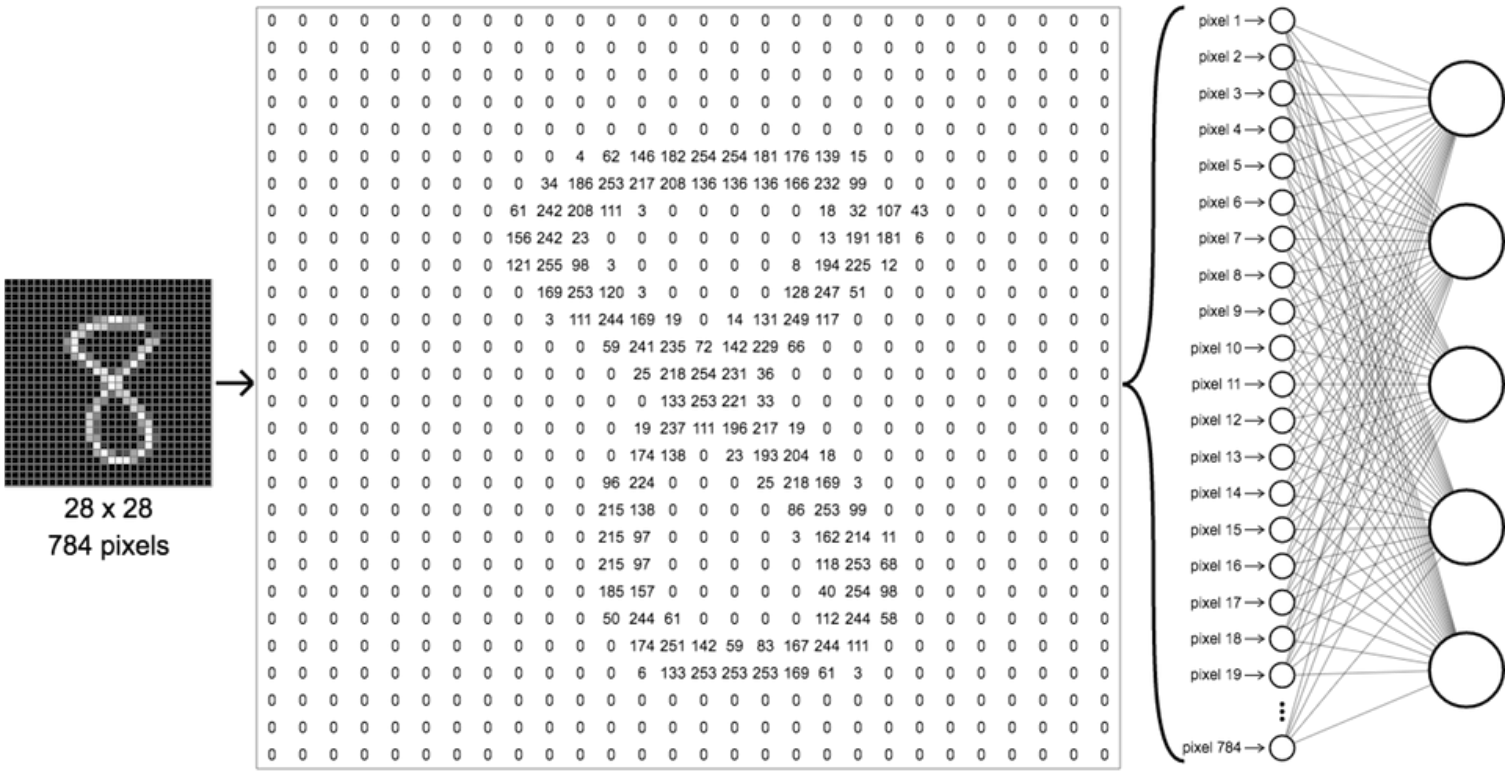
Image credit: quantdare.com

More background on t-SNE

Carnegie Mellon

- Original space:
  - one-sided distance: $p_{i|j} = \frac{exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)}$
  - two-sided (gaussian centered density): $p_{ij} = \frac{p_{i|j} + p_{j|i}}{2}$
- Embedded space: similar distance of 2 spaces:
  $q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq i} (1 + \|y_i - y_k\|^2)^{-1}}$
- Target: by mimimizing the KL distance:
  $y_1, ..., y_D = \text{argmin}\, KL(P\|Q) = \text{argmin} \sum_{i \neq j} p_{ij}\, log \frac{p_{ij}}{q_{ij}}$

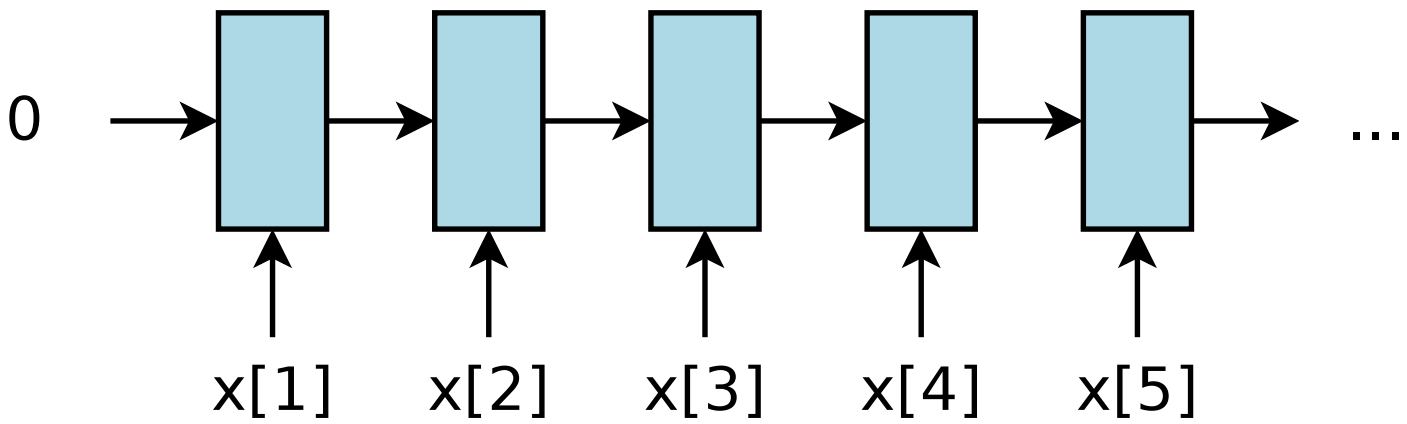
General Case: Use MLP



- Caveat: need to flatten the input
- With data knowledge, we can do a better job

Image credit: ml4a.github.io

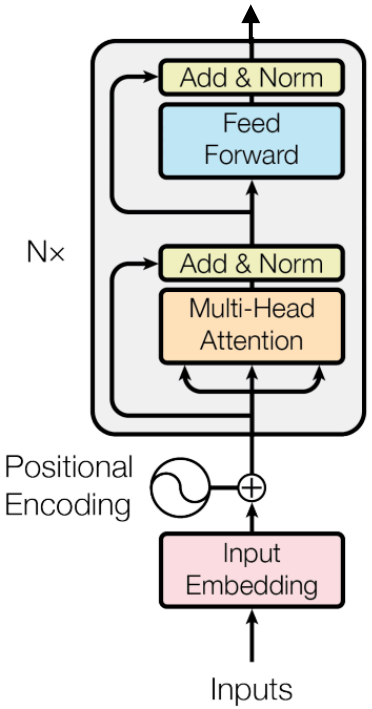
Sequential: Use RNN



- Representation has sequential correlations

Sequential: *Transformer*[4]

Carnegie Mellon



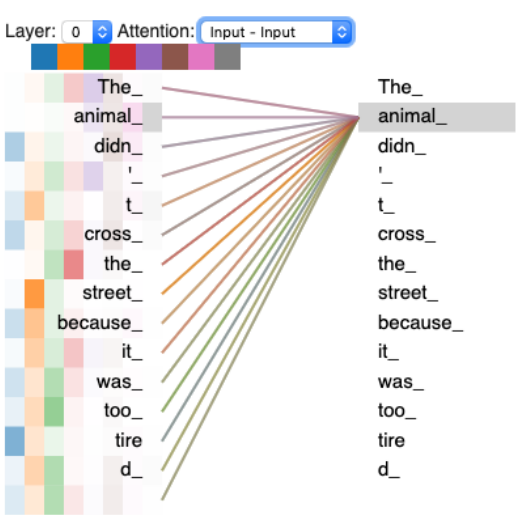
- We don't rely on RNN function anymore
- Instead, rely on $N*$ (multi-head attentions + FNN) blocks
- What's more ...

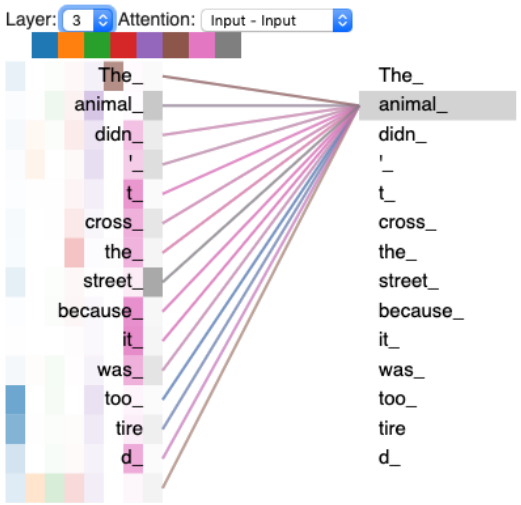
[4] https://arxiv.org/pdf/1706.03762.pdf

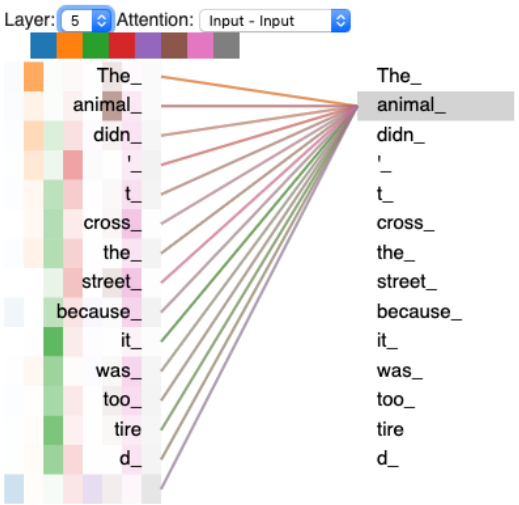
Image credit: Ashish Vaswani *et al.*

Self-Attention (of *it* to *itself*)

Carnegie Mellon



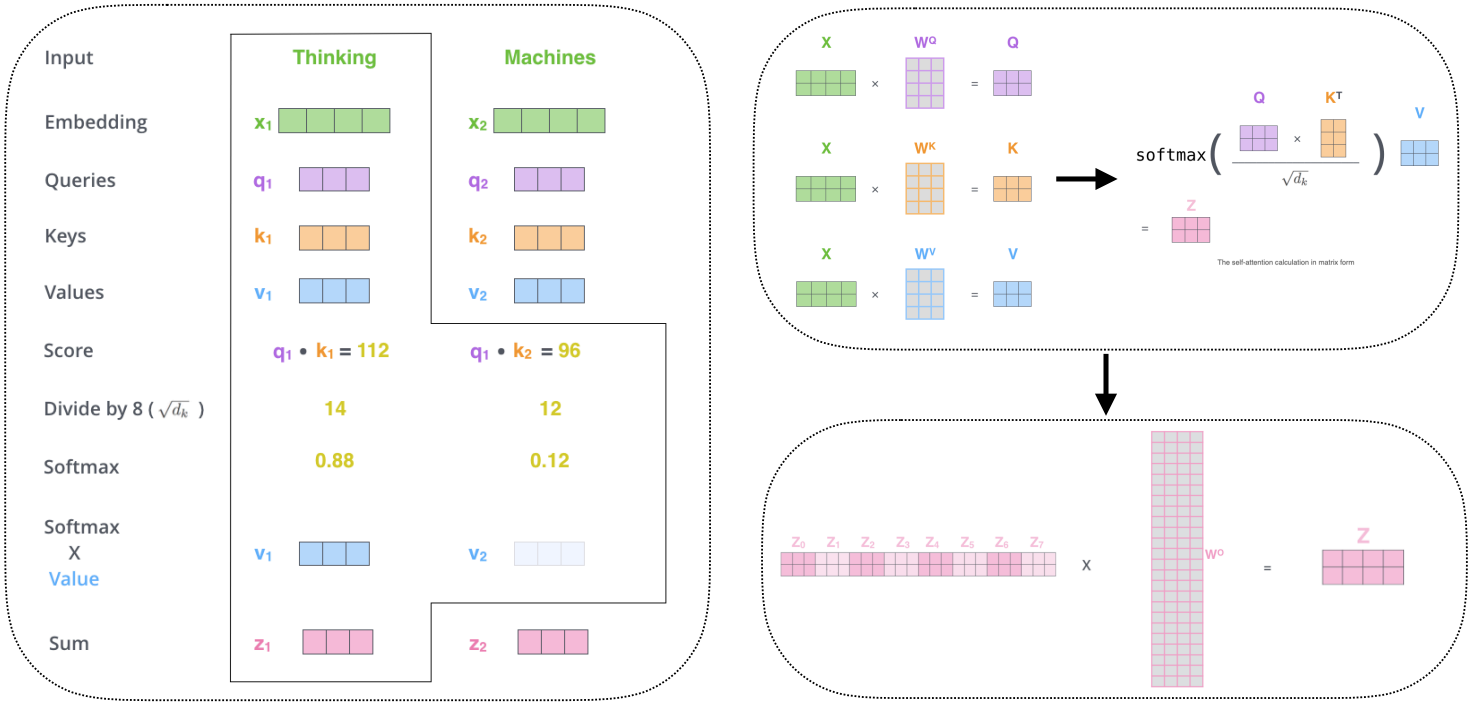




- Instead of encoding the sequence into a single hidden vector
- Now we calculate the relationship of each word vs. *every* another word
- Maybe a better idea for languages in which the order is very flexible

Image credit: Tensor2Tensor Ipython Notebook - Google

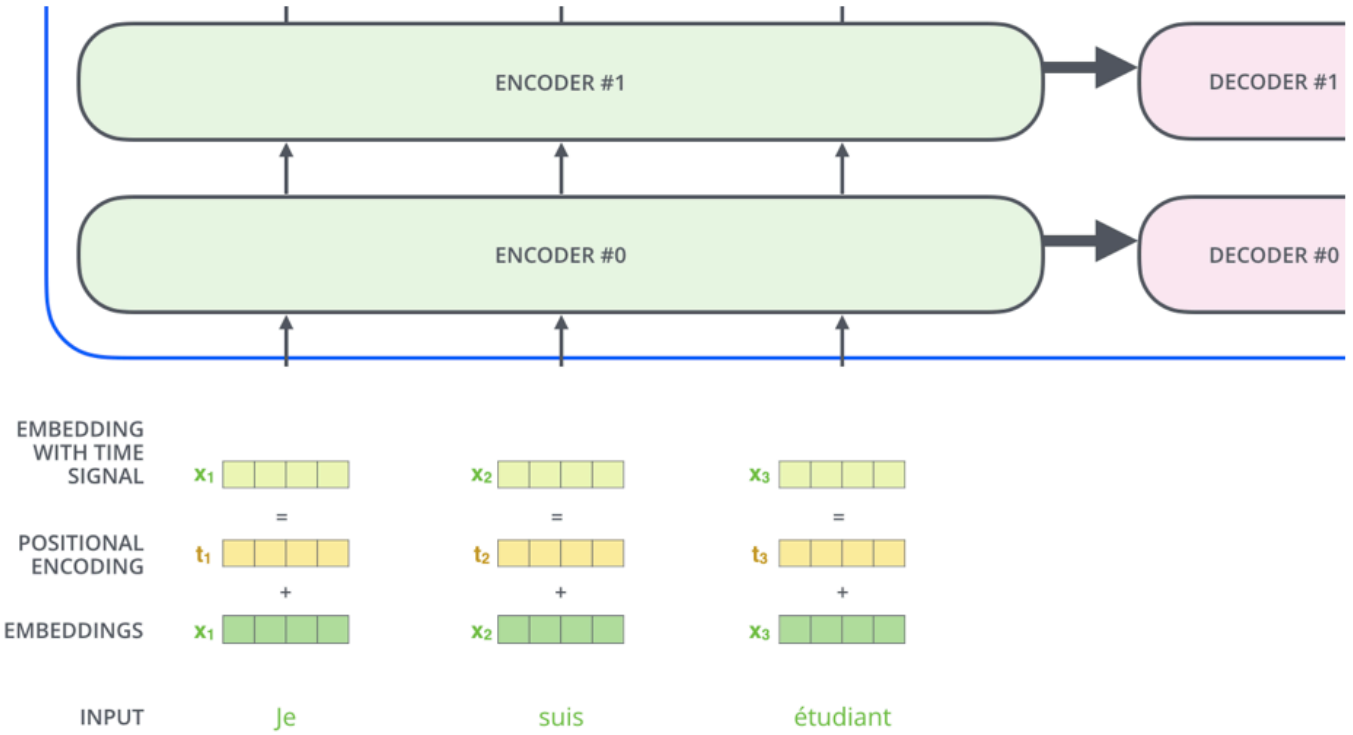
Multi-head Attention



- Instead of using a single FC, use a triple (q, k, v) (query, key, value)
- Normalized with input length and with softmax
- Merge all head using concatenation and a FC with weight W^0

Image credit: Jay Alammar

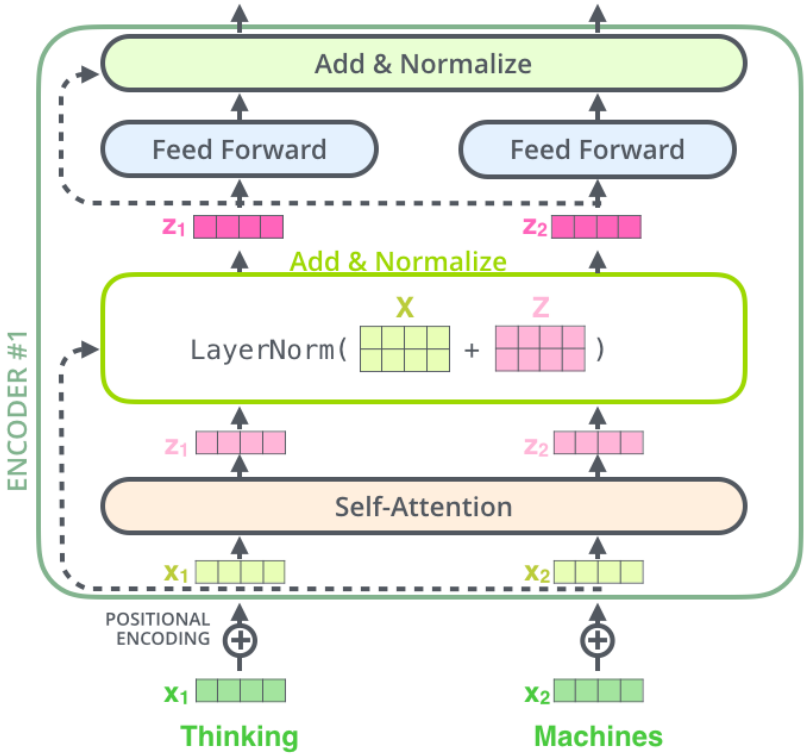
Multi-head Attn Able to Replace RNN?



- Not yet, needs an alternative way to model sequence
- *Positional Encoding* is added to word embedding

Image credit: Jay Alammar

Putting together: Transformer Encoder

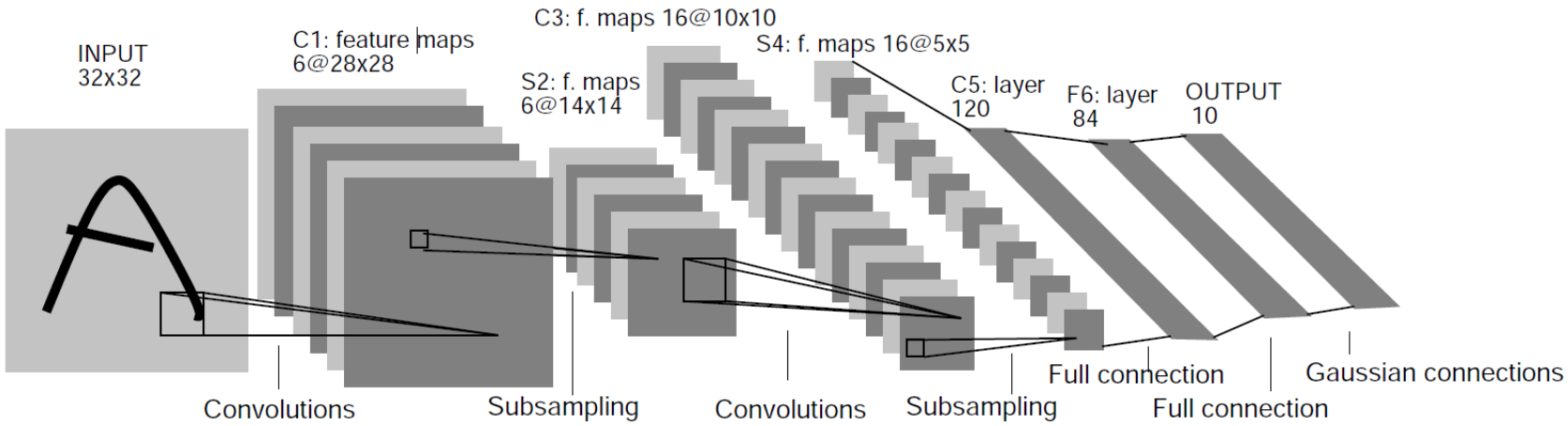


- Residual connection is used at self-attention and FNN
- Layer Norm[5] is also applied.
- Note: it also stacks multiple encoders (and decoders (later))

[5] https://arxiv.org/pdf/1607.06450.pdf

Image credit: Jay Alammar

Image/Video: Use CNN

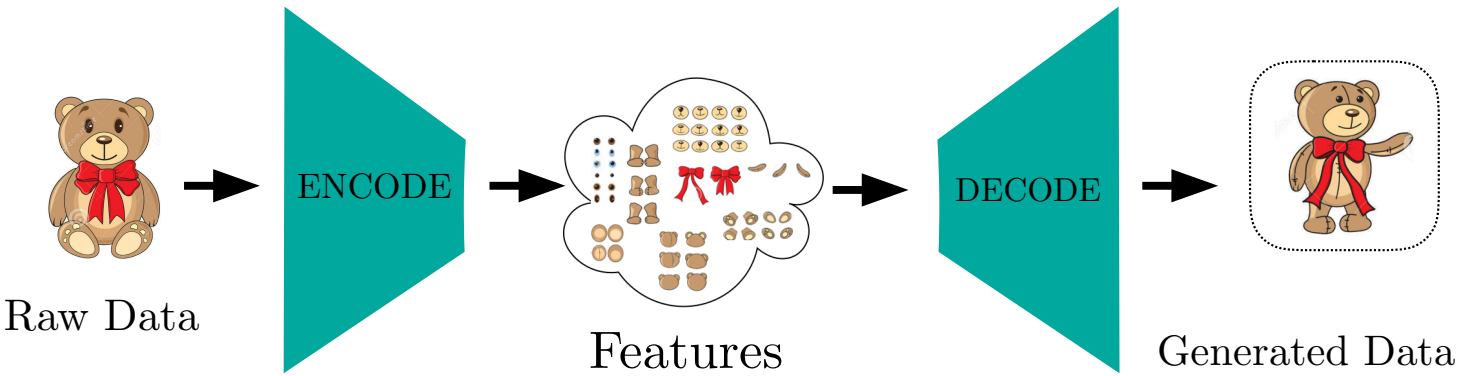


- Encoded representation has spatial correlations

Image credit: pythonmachinelearning.pro

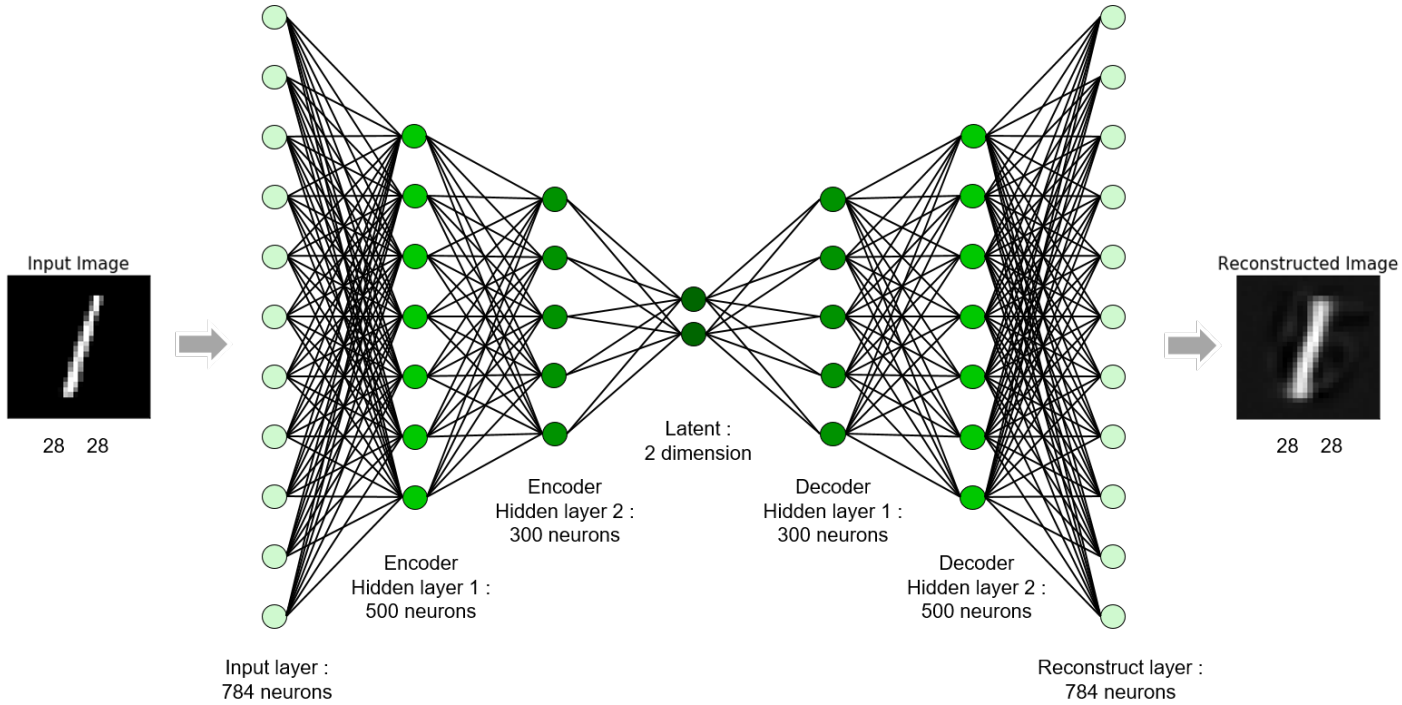
Outline

General case



- Perform the inverse process of decoder
  - Encoder: compress from dimension N to D for N > D
  - Decoder: uncompress from D back to N

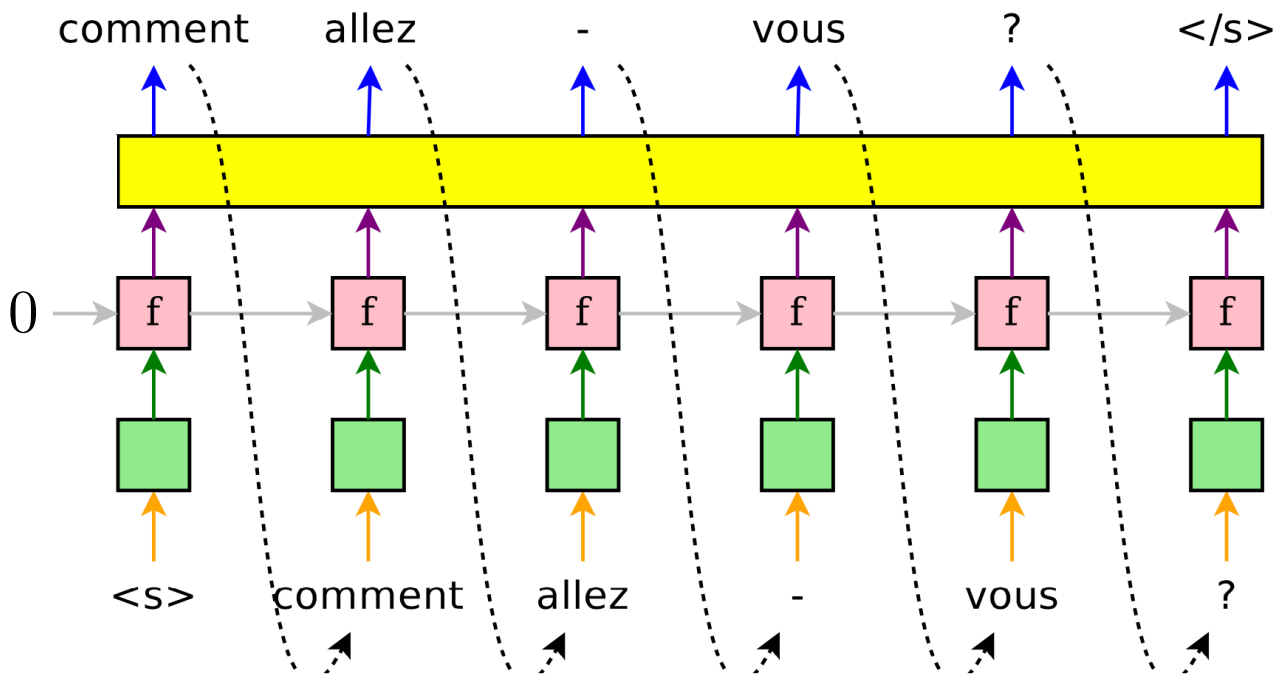
MLP Decoding



- Perform the inverse process of decoder
  - Encoder: MLP with decreasing dimensions
  - Decoder: MLP with increasing dimensions back to the original
- Tutorial: ▸ MNIST AutoEncoder Tensorflow-Keras
- Tutorial: ▸ MNIST Denoising AutoEncoder Tensorflow-Keras

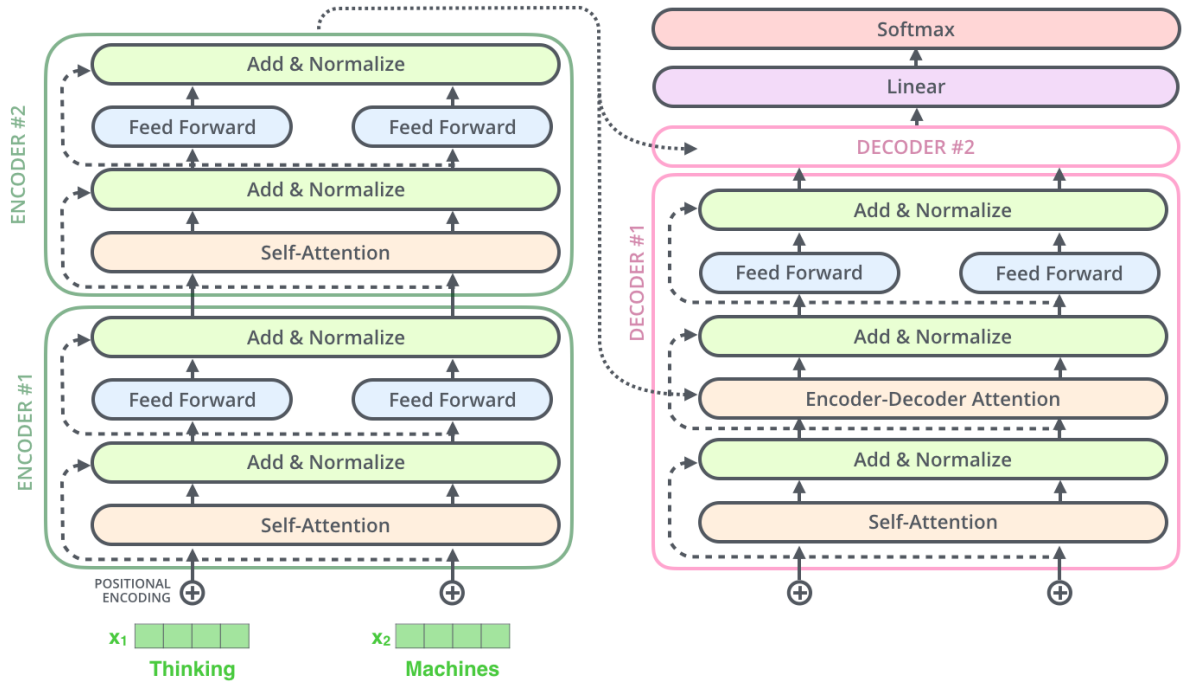
Image credit: i-systems.github.io

RNN Decoding



- Perform the inverse process of Encoder
  - Using previous output and previous hidden
  - It models $p(x_t|x_{<t})$ so sometimes it's called an *auto-regressive model.*
- Tutorial: ▸ RNN Text Generation with Tensorflow-Keras

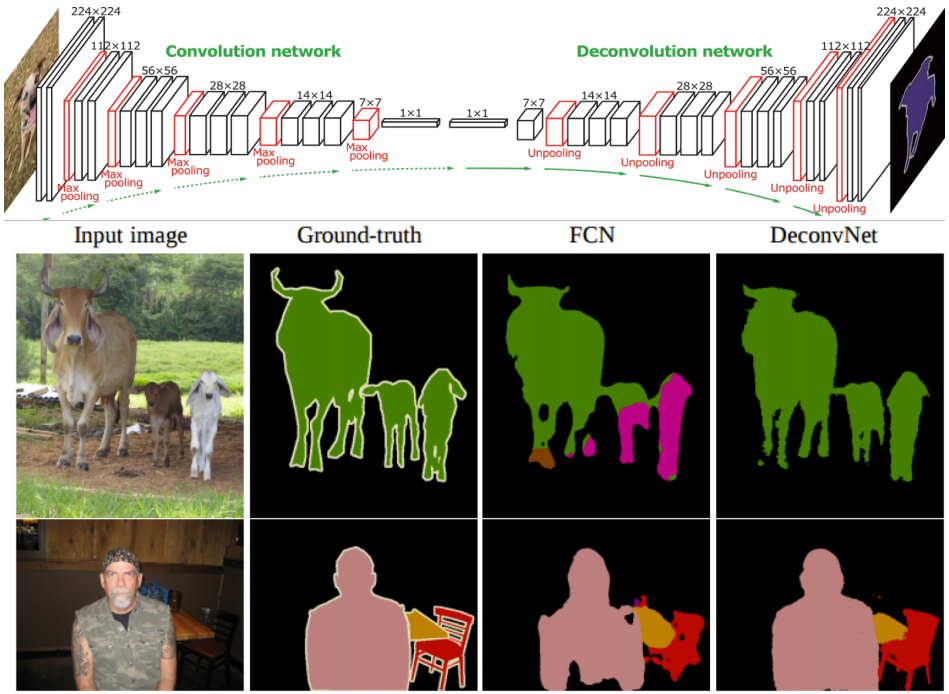
Transformer Decoding



- Self-Attention is used, but only for the seen sequence to that point
- Then normal Attention is used as in Seq2Seq
- Decoding as usual to find a word in vocabulary.
- Tutorial: ▸ Transformer NMT with Tensorflow-Keras
- Tutorial: ▸ NMT-Transformer with pre-trained weights

Image credit: Jay Alammar

CNN Decoding

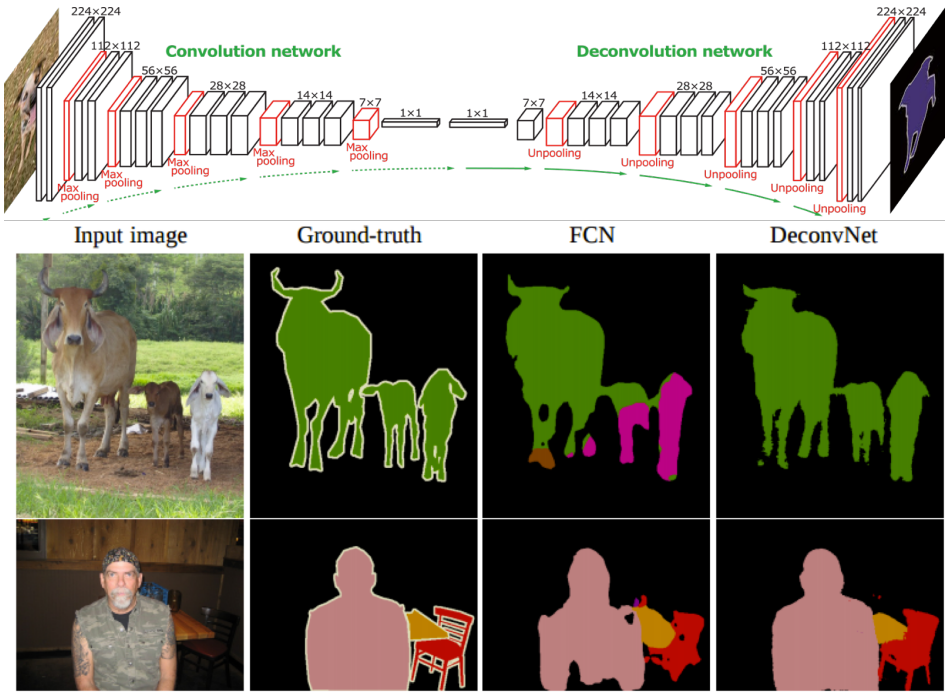


- Deconvolution is only good enough for such tasks as segmetation where detail is not the first priority
- Maybe we need to add some constraint into embedded space?
- Or maybe we should not use MLE-based method?

Image credit: Hyeonwoo Noh et al.

CNN Decoding: Beyond MNIST

Carnegie Mellon



- Maybe we need to add some constraint into embedded space?
- Or maybe we should not use MLE-based method?
- *Next lectures will help provide some appealing solutions to those questions.*

Image credit: Hyeonwoo Noh et al.