Introduction to Deep Learning

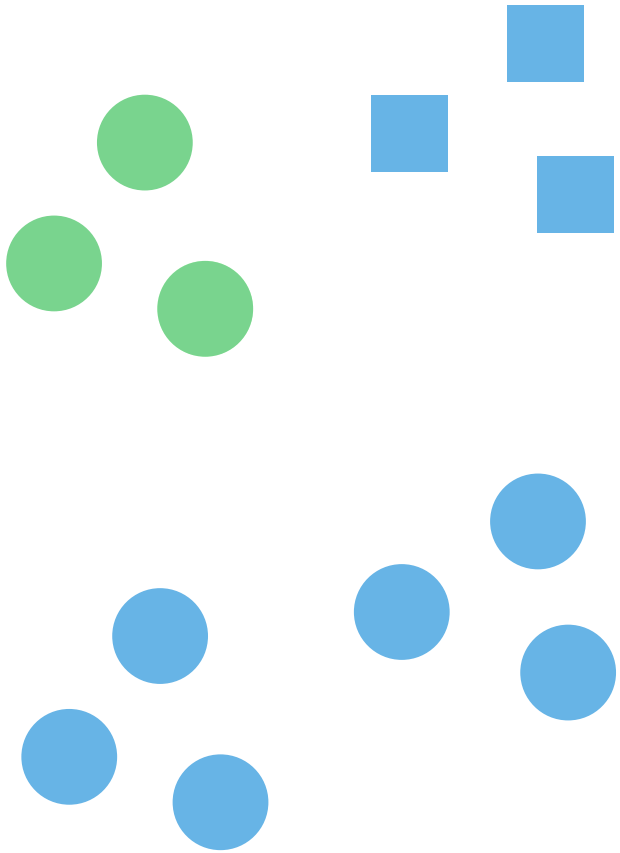
Supervised vs Unsupervised

Supervised learning

- Take (x,y) pairs

Unsupervised learning

- Take x alone



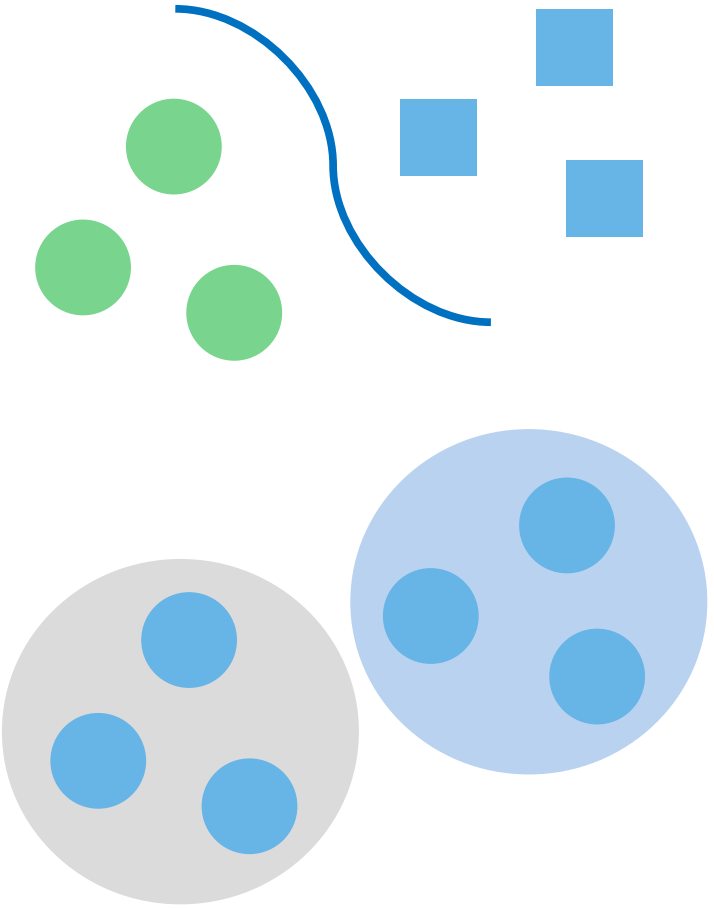
Supervised vs Unsupervised

Supervised learning

- Take (x,y) pairs
- Learn mapping x → y

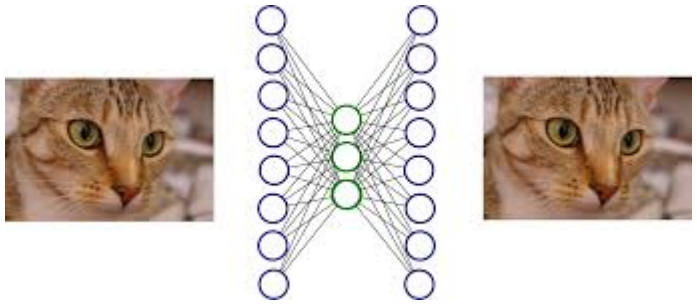
Unsupervised learning

- Take x alone
- Learn hidden structure
- Behind the data



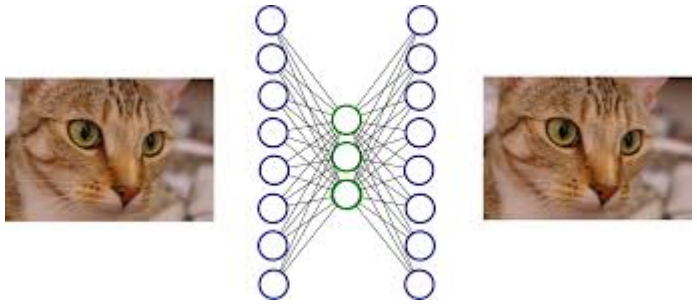
Why bother?

- Find most relevant features
- Compress information



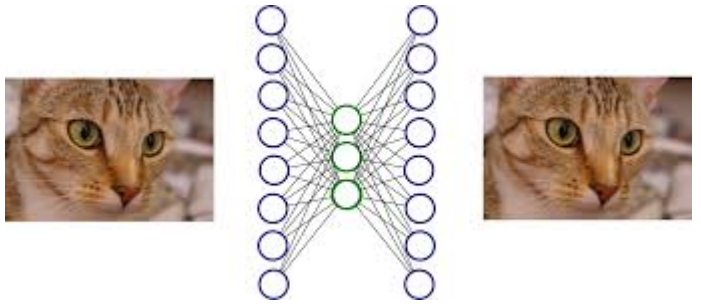
Why bother?

- Find most relevant features
- Compress information
- Retrieve similar objects



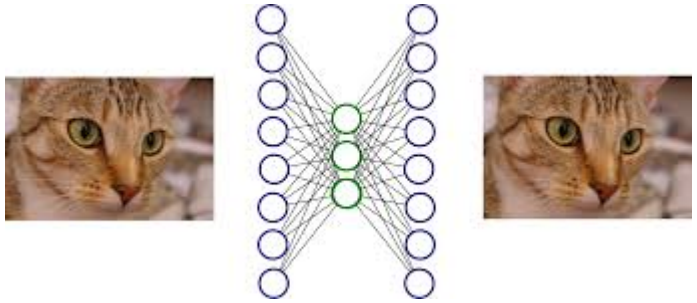
Why bother?

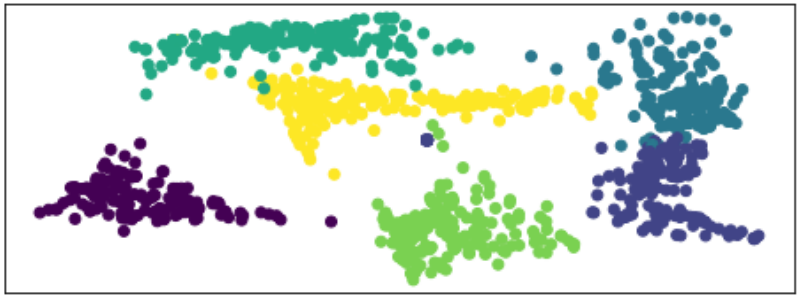
- Find most relevant features
- Compress information
- Retrieve similar objects
- Generate new data samples



Why bother?

- Find most relevant features
- Compress information
- Retrieve similar objects
- Generate new data samples
- Explore high-dimensional data





Autoencoders

Main idea:

Take data in some original (high-dimensional) space;

Project data into a new space **from which it can then be accurately restored**

"reconstruction"

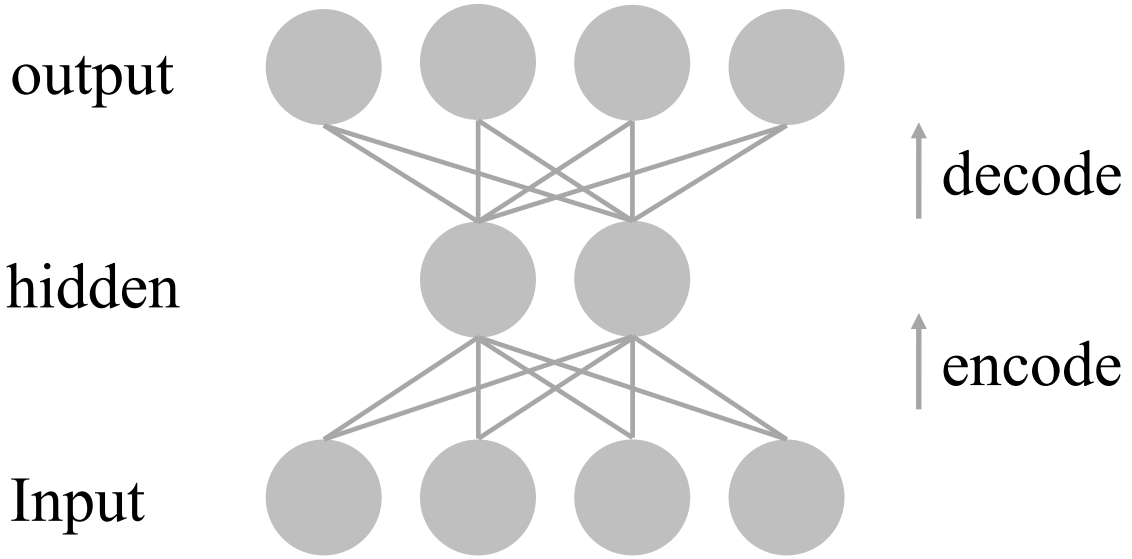
- aiming to learn a rep (encoding), typically for dim reduction
- Widely used in learning generative models

Autoencoders

Main idea:

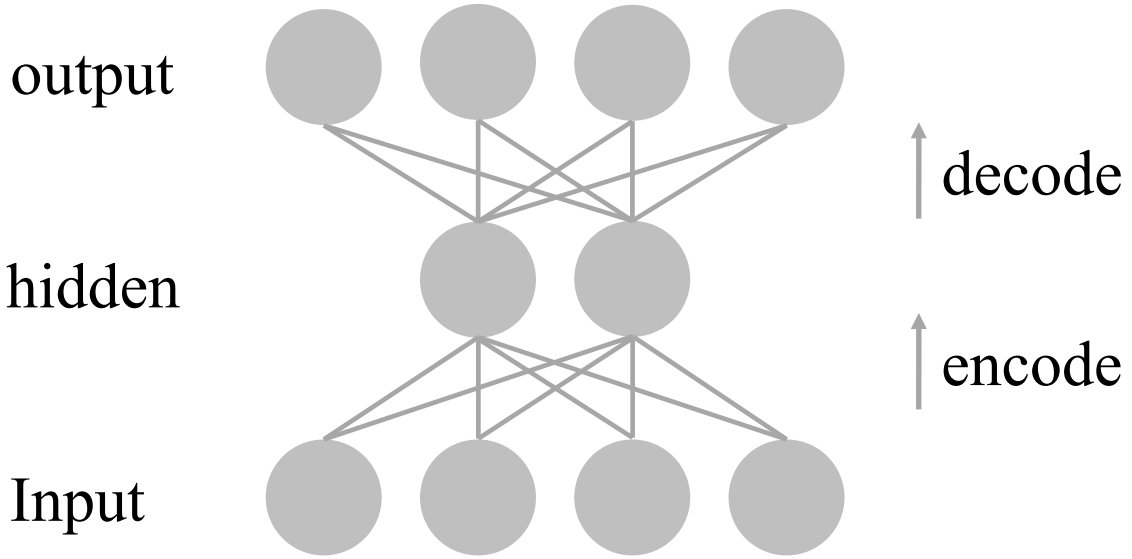
Take data in some original (high-dimensional) space;

Project data into a new space **from which it can then be accurately restored**



Autoencoders

- Encoder = data to hidden
- Decoder = hidden to data
- Decoder(Encoder(x)) ~ x



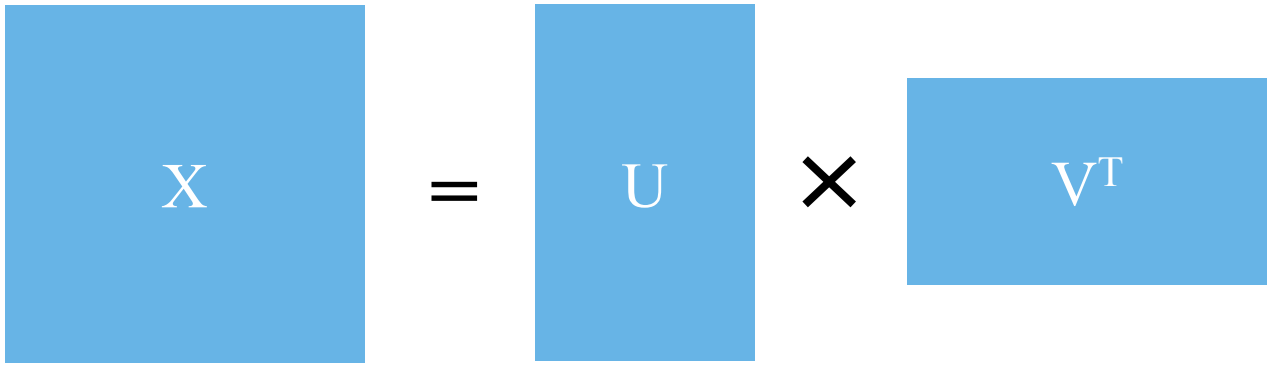
Why do we ever need that?

- Compress data
  - |code| << |data|
- Dimensionality reduction
  - Before feeding data to your XGBoost :)

- <to be continued>

Linear case

Example: matrix factorization

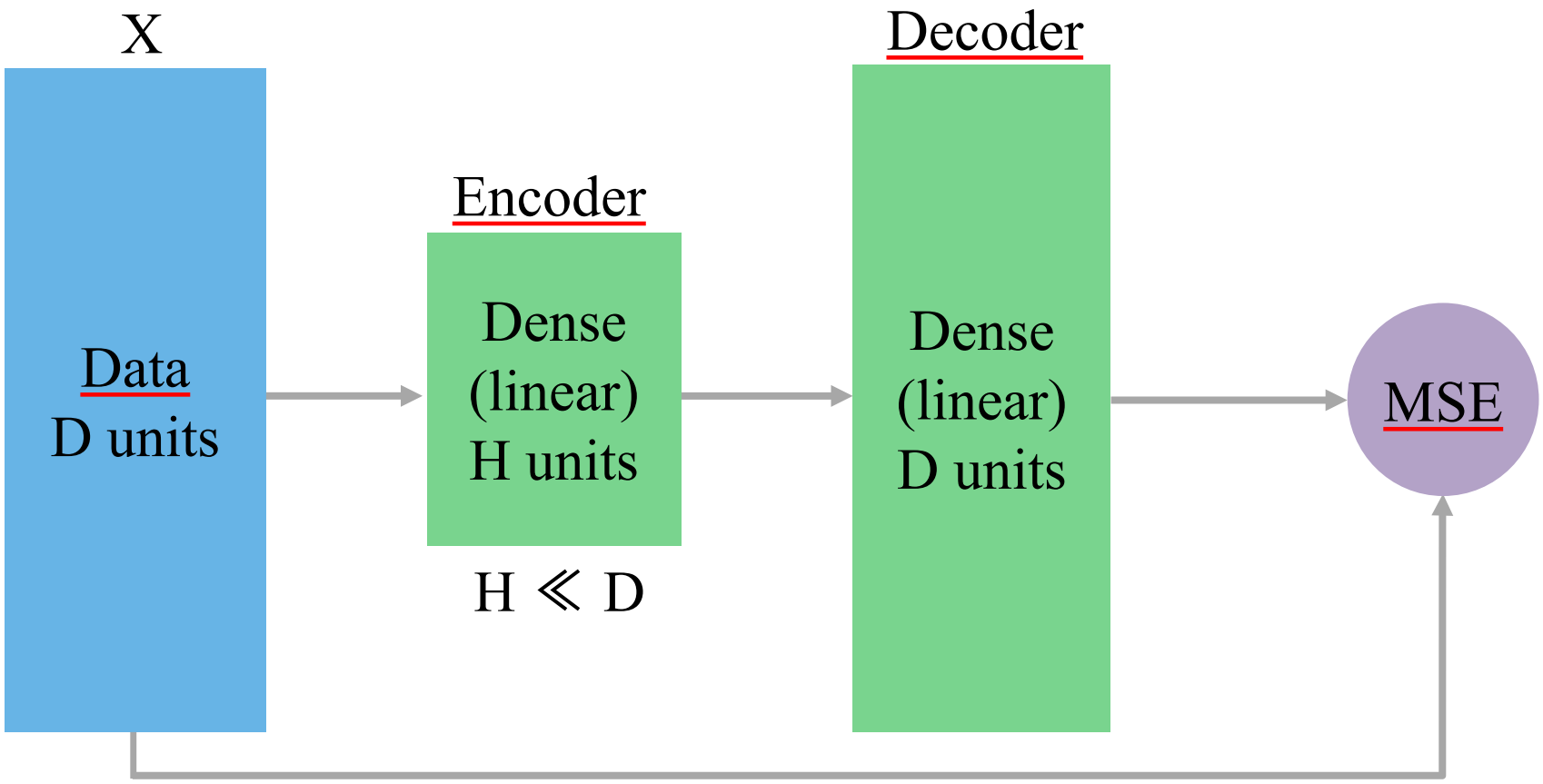


Minimizing reconstruction error

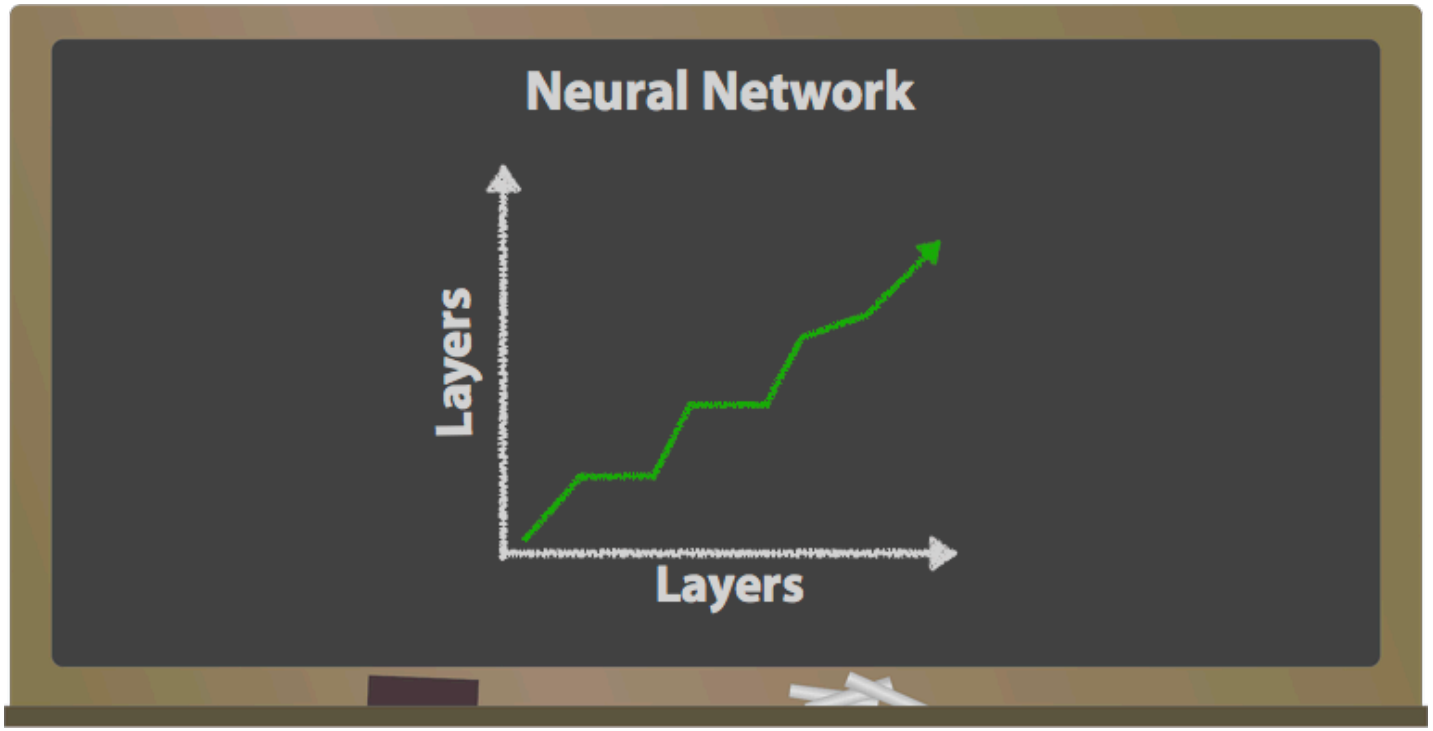
$$\| X - U \cdot V^T \| \rightarrow \min_{U,V}$$

Matrix decompositions

A different perspective



Matrix decompositions

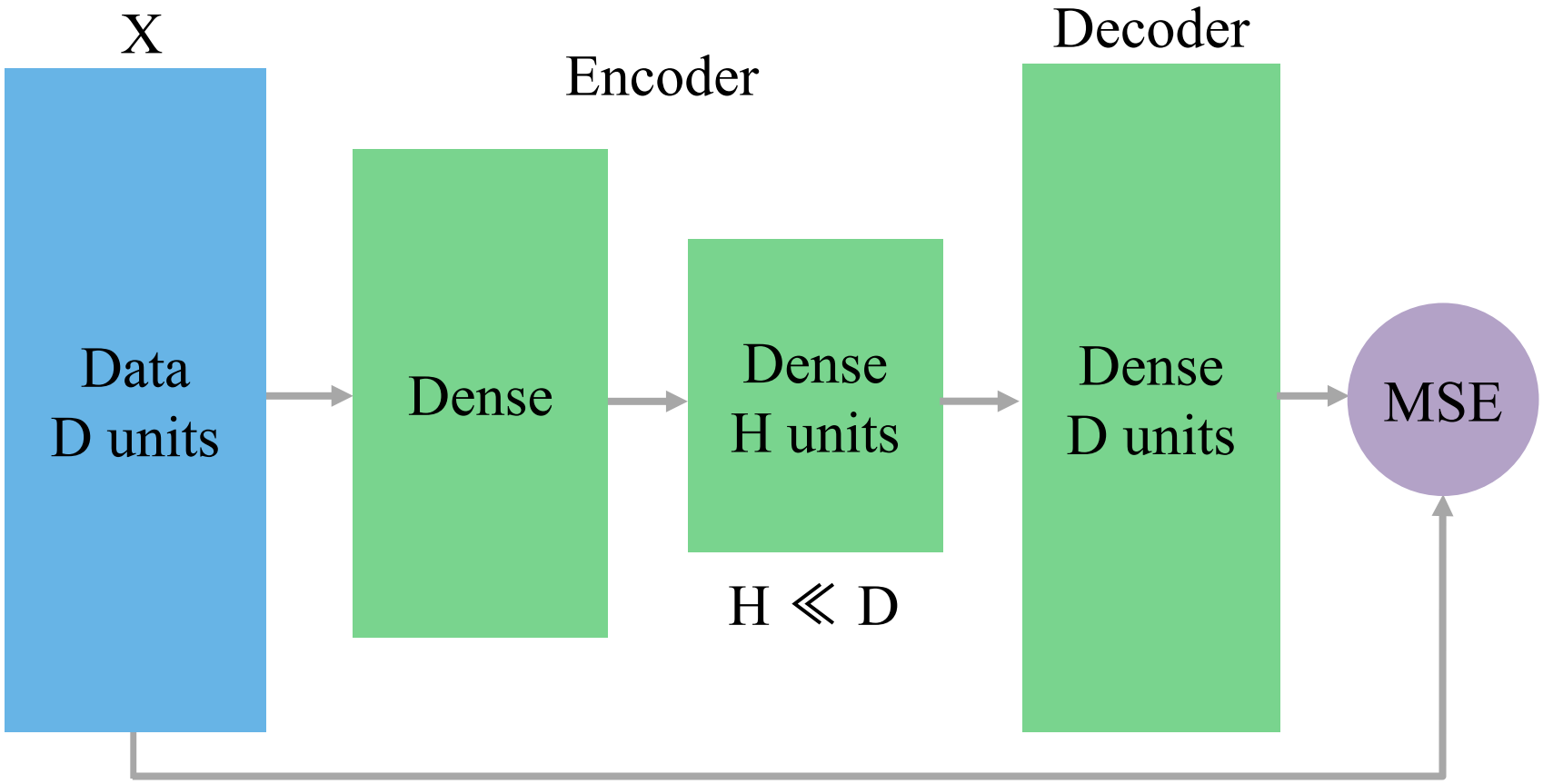


«STACK MORE LAYERS»

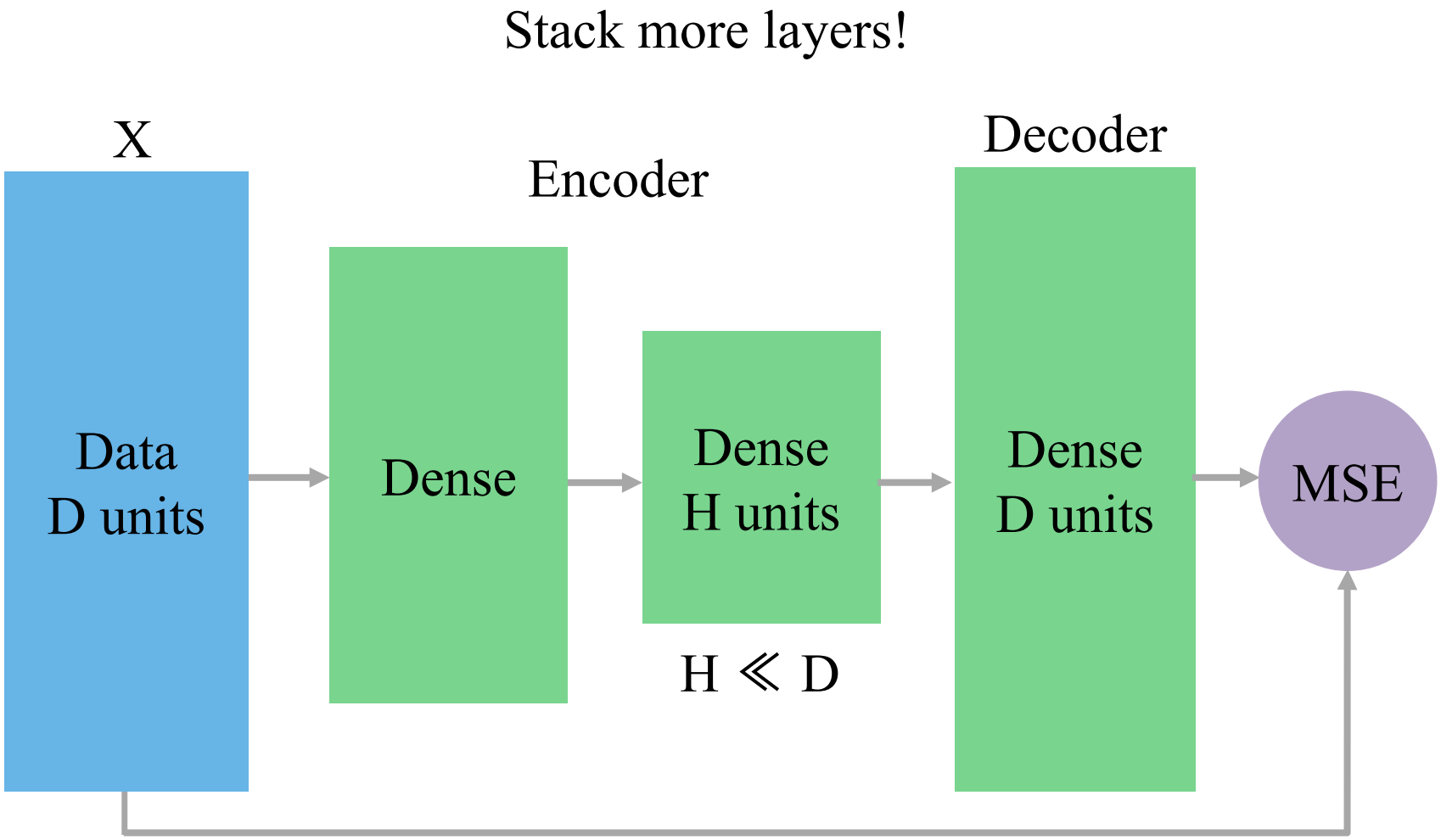
Deep autoencoder

CNN best fit here

Stack more layers!



Deep autoencoder



Quiz: What if data is an image?

Image2image: convolutional

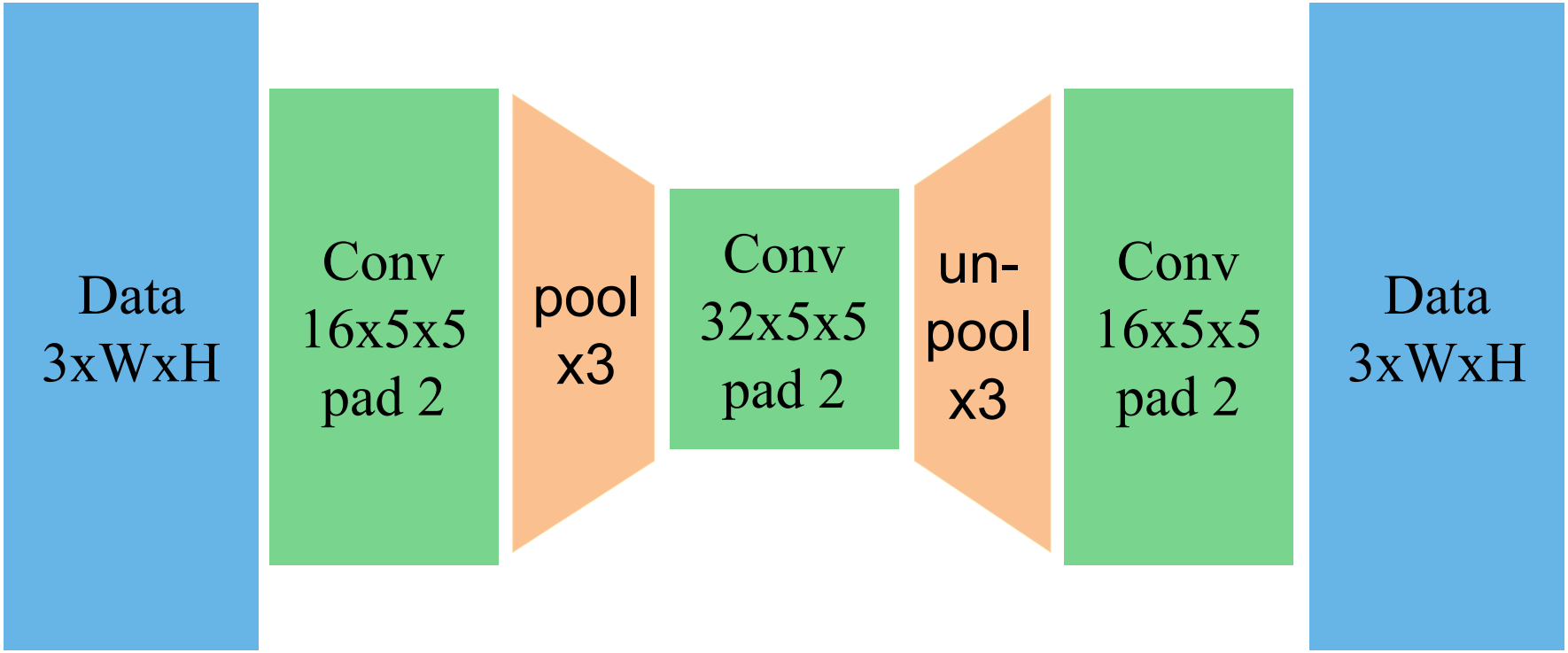
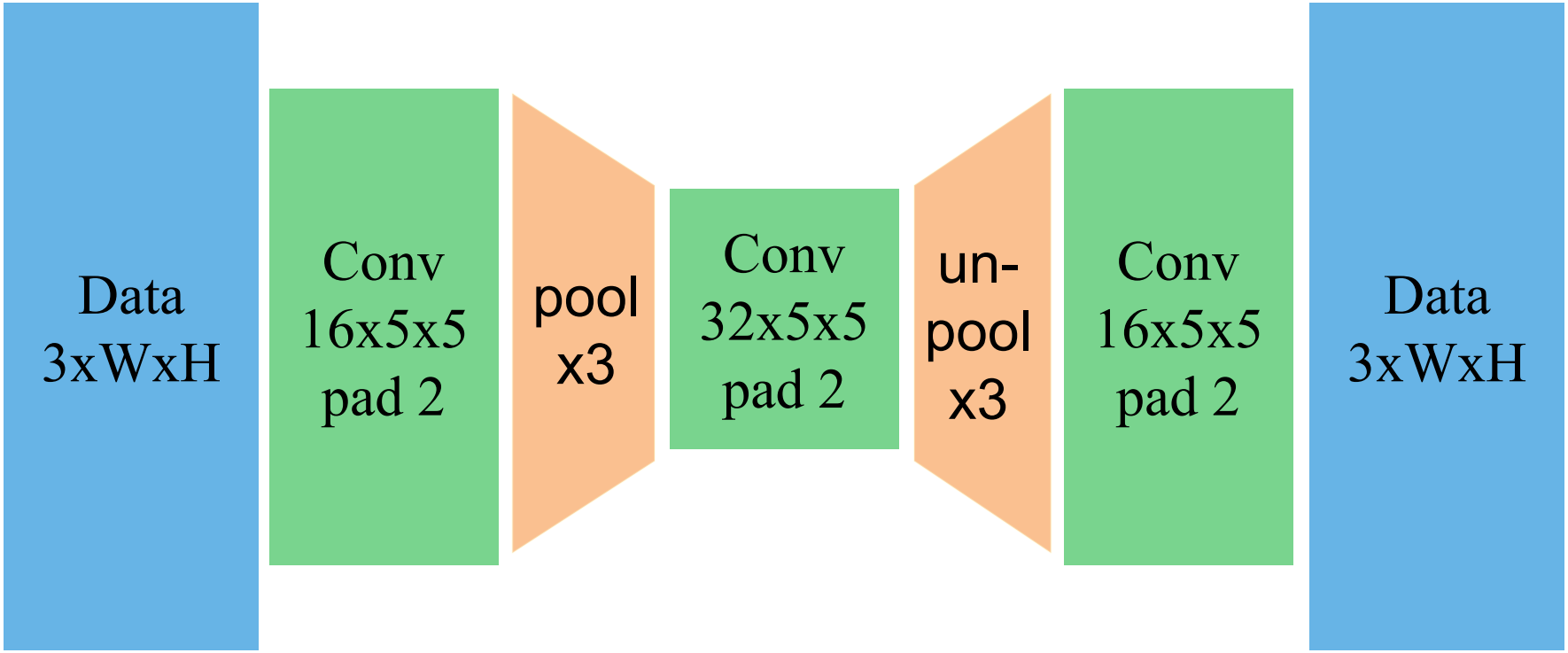


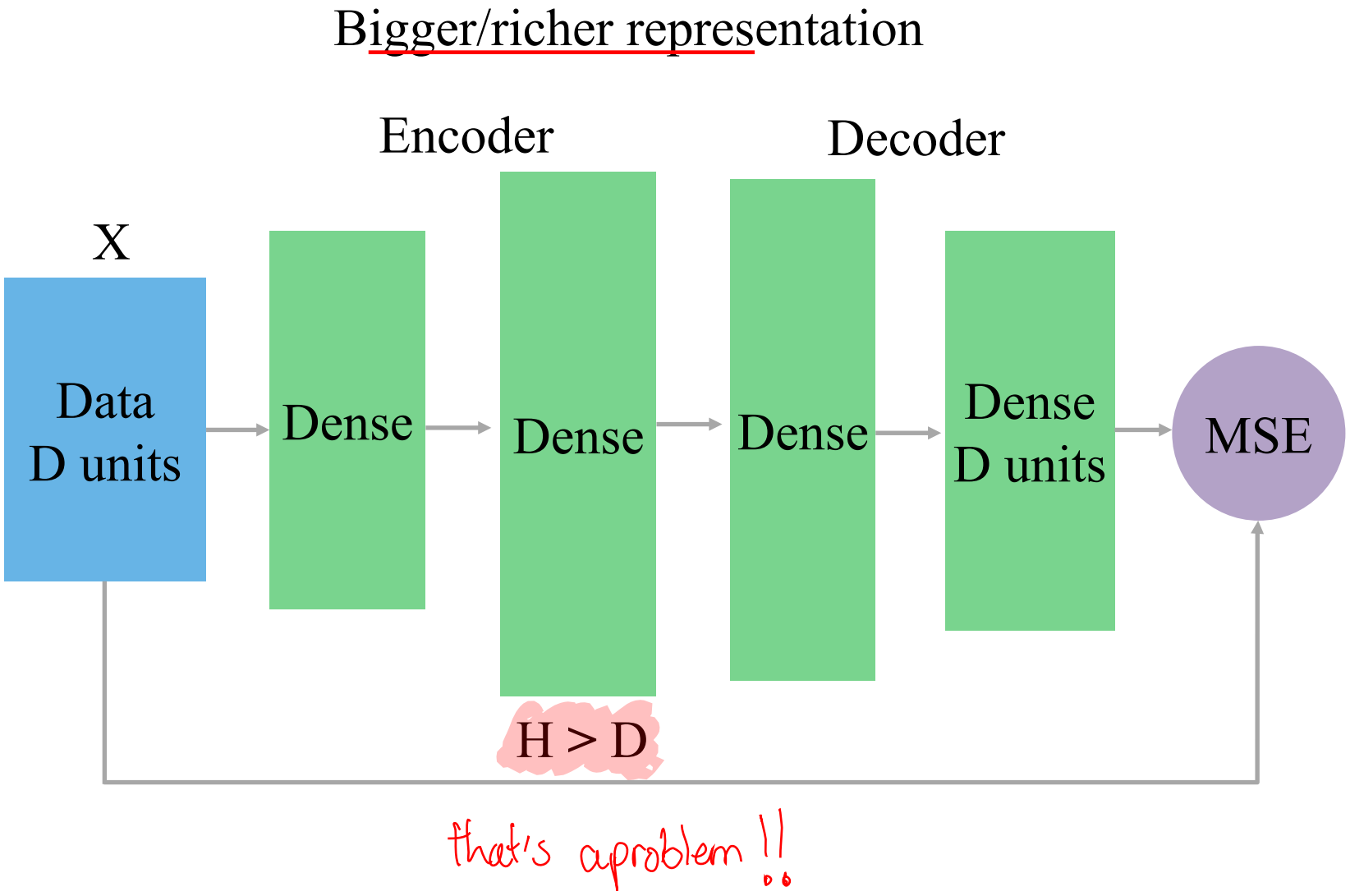
Image2image: fully-convolutional



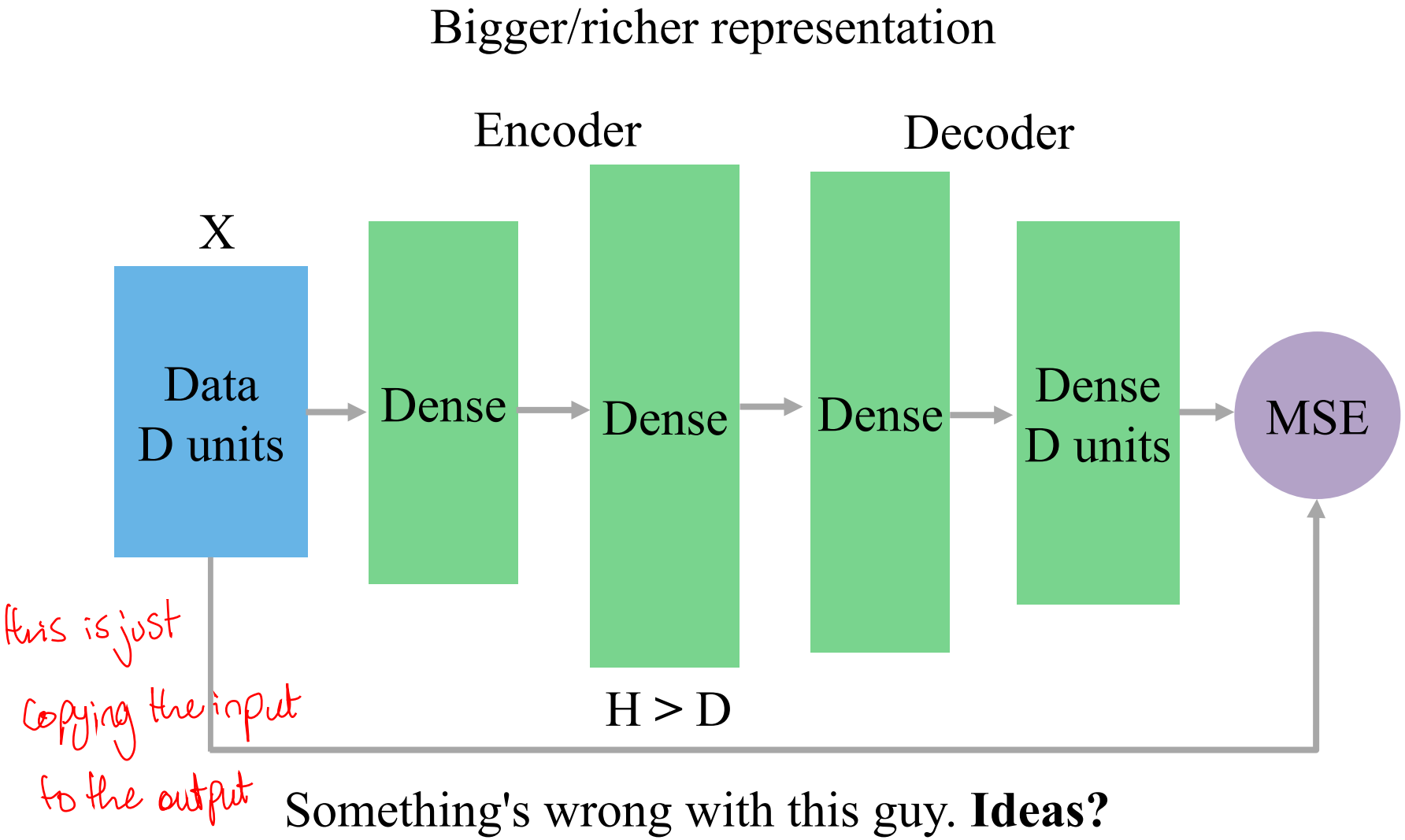
Why do we ever need that?

- Compress data
  - |code| << |data|
- Dimensionality reduction
  - Before feeding data to your XGBoost :)
- **Learn some great features!**
  - Before feeding data to your XGBoost
- **Unsupervised pretraining**
  - Large amounts of unlabeled data

Expanding autoencoder



Expanding autoencoder



Something's wrong with this guy. **Ideas?**

Expanding autoencoder

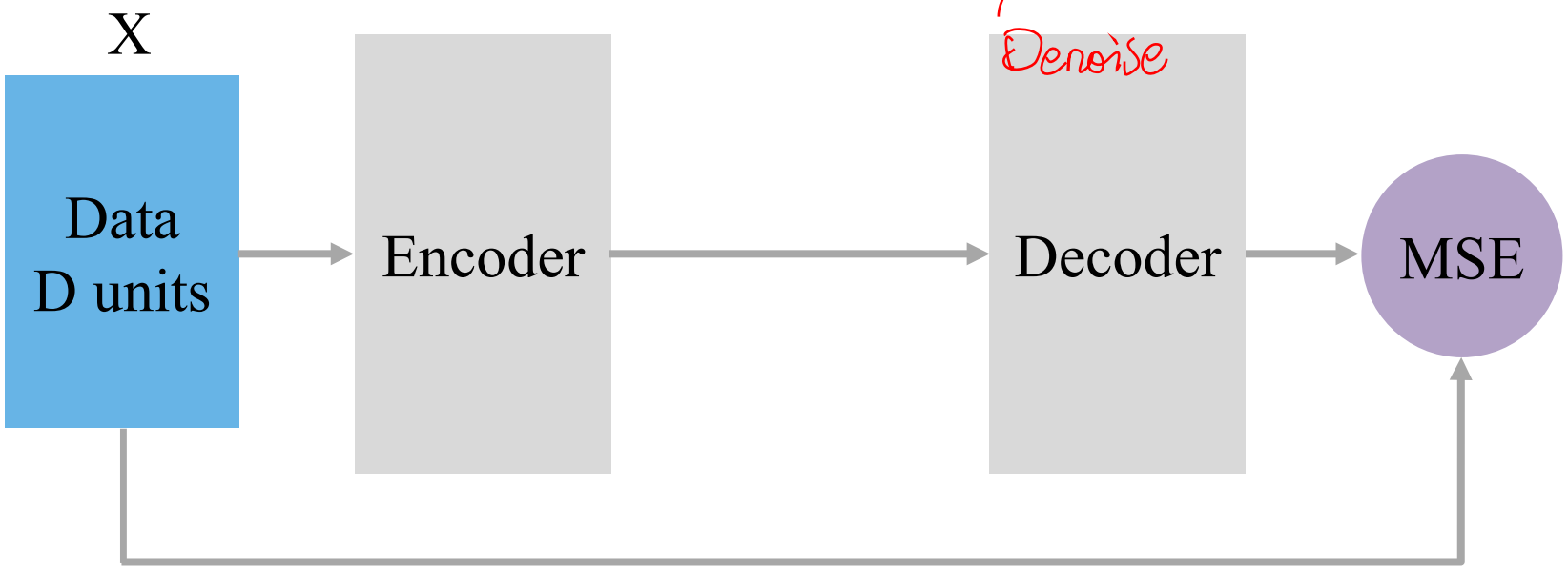
Naive approach will learn identity function!

Gotta regularize!

Sparse

Redundant

Denoise



* Previously, we regularized weights

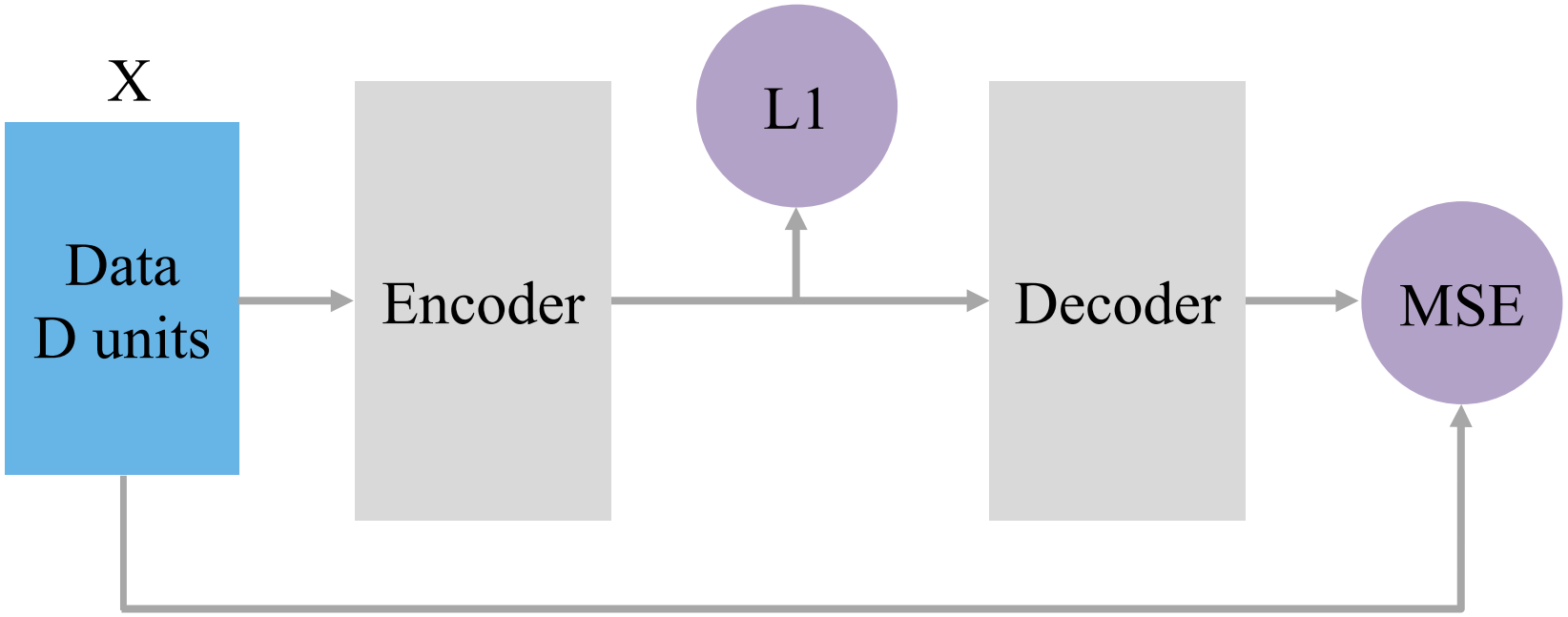
$$L = \|X - Dec(Enc(X))\|$$

Sparse autoencoder

model dropping out features → zero
i.e weights ending up being zero

Naive approach will learn identity function!

Idea 1: L1 on **activations**, sparse code

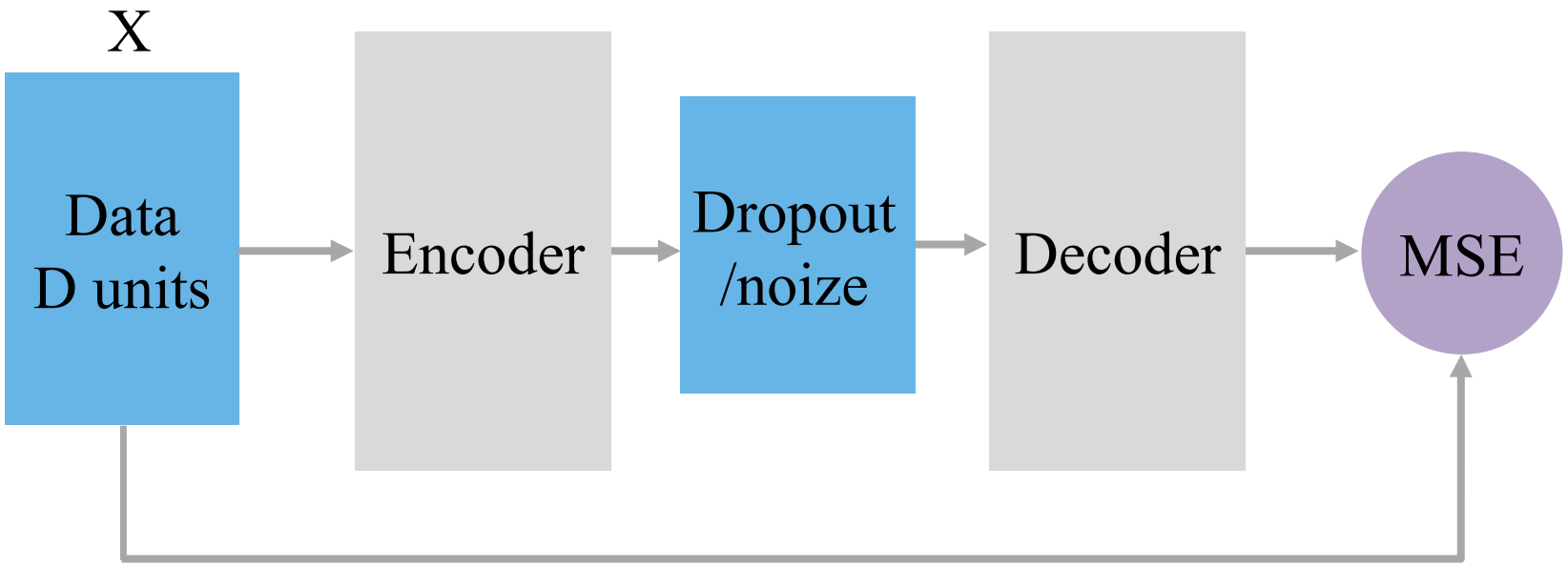


$$L = \|X - Dec(Enc(X))\| + \sum_i |Enc_i(X)|$$

Redundant autoencoder

Naive approach will learn identity function!

Idea 2: noize/dropout, redundant code



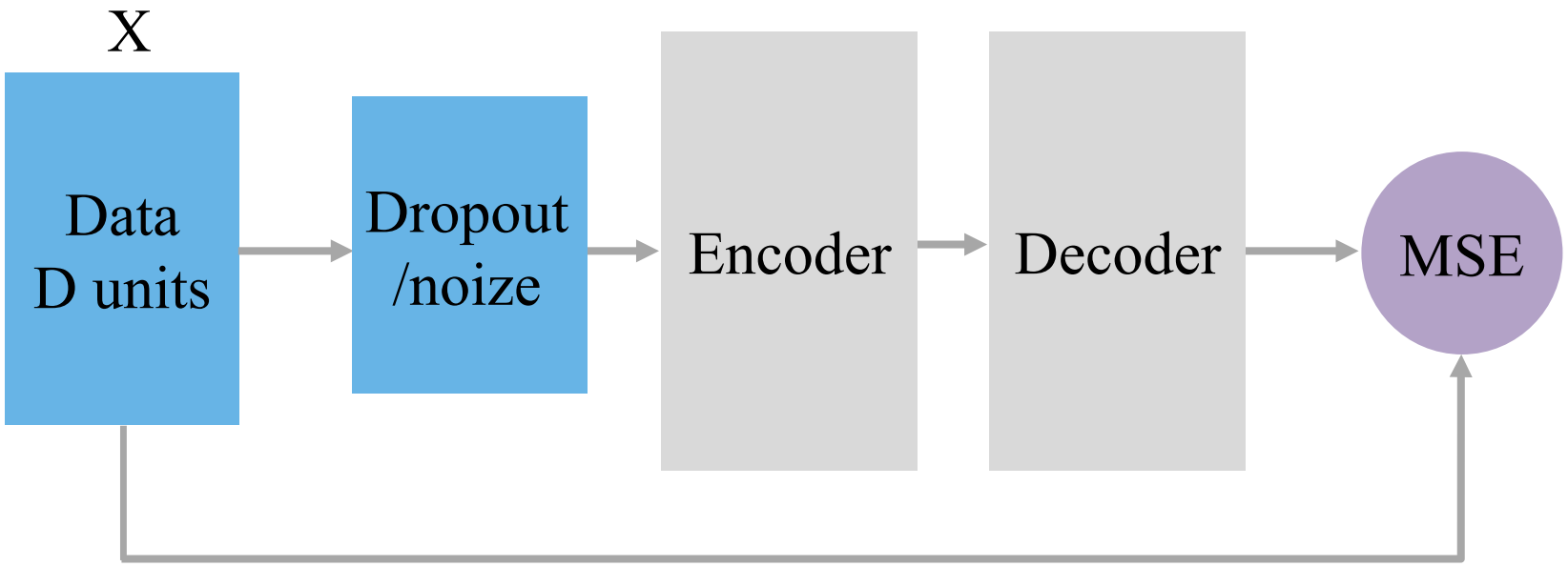
$$L = \left\| X - Dec\left(Enc\left(Noize(X)\right)\right) \right\|$$

Denoizing autoencoder

model extrapolating features for itself

Naive approach will learn identity function!

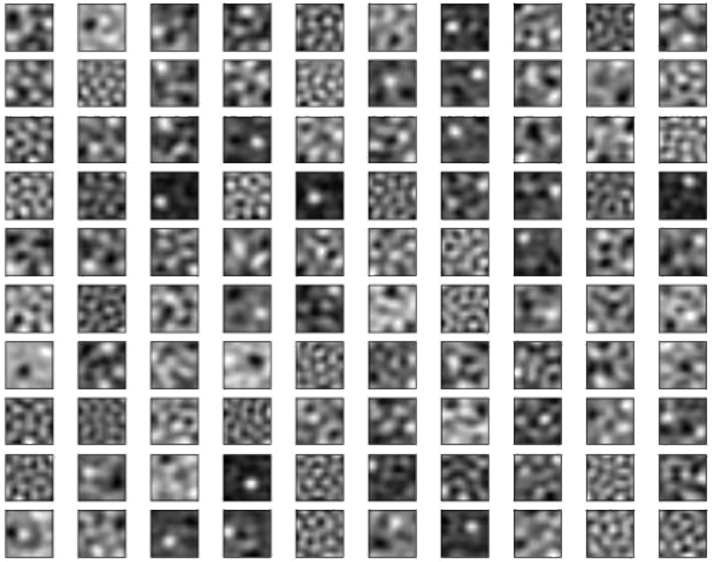
Idea 3: distort input, learn to fix distorsion



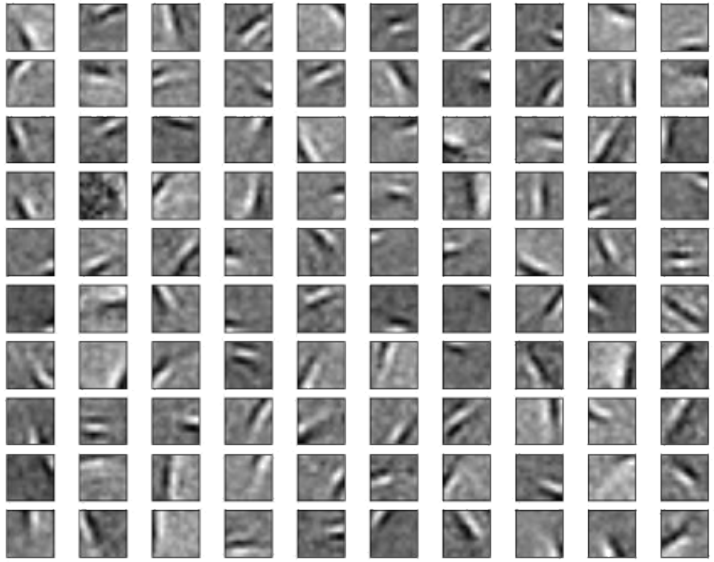
$$L = \left\| X - Enc\left(Dec\left(Noize(X)\right)\right) \right\|$$

Sparse Vs Denoizing

Filter weights, 12x12 patches



Sparse AE



Denoizing AE

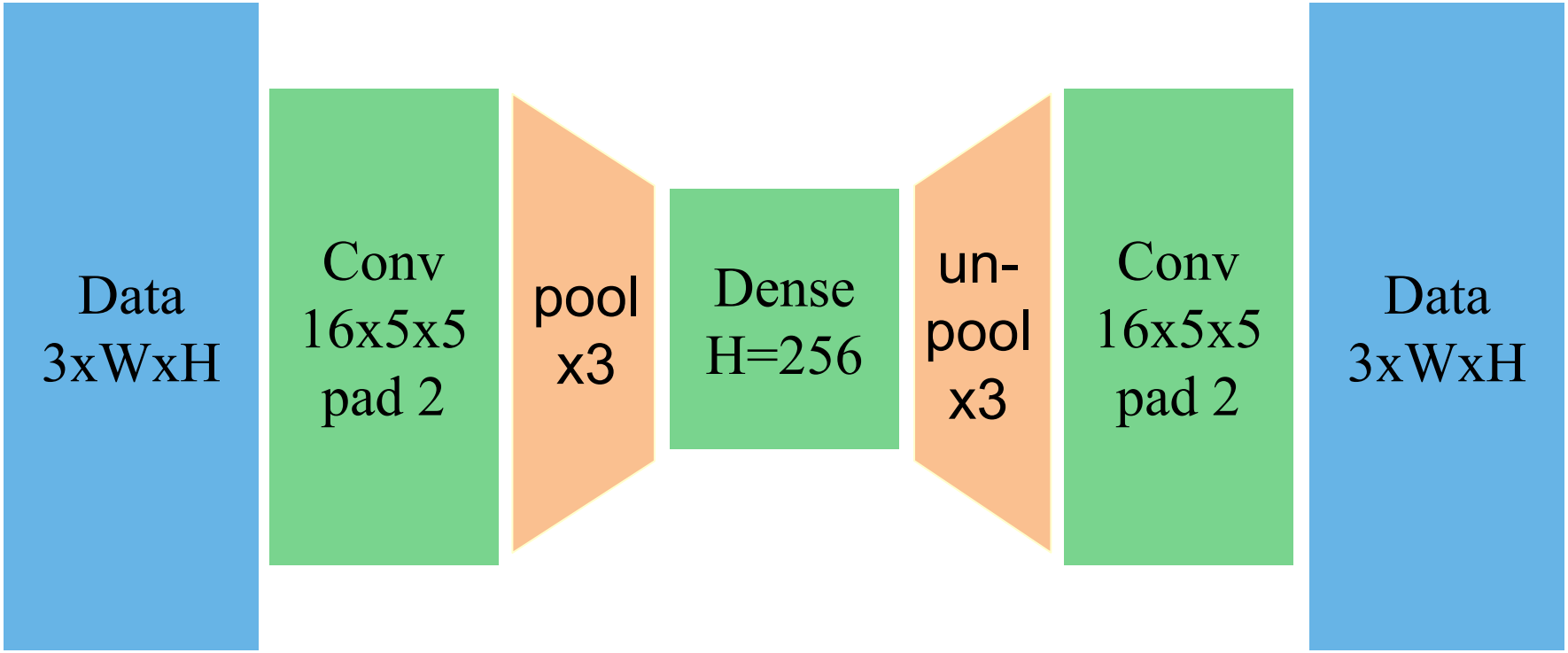
These images are actually clueless :)

Why do we ever need that?

- Compress data
  - |code| << |data|
- Dimensionality reduction
  - Before feeding data to your XGBoost
- **Learn some great features!**
  - Before feeding data to your XGBoost
- **Unsupervised pretraining**
  - Large amounts of unlabeled data

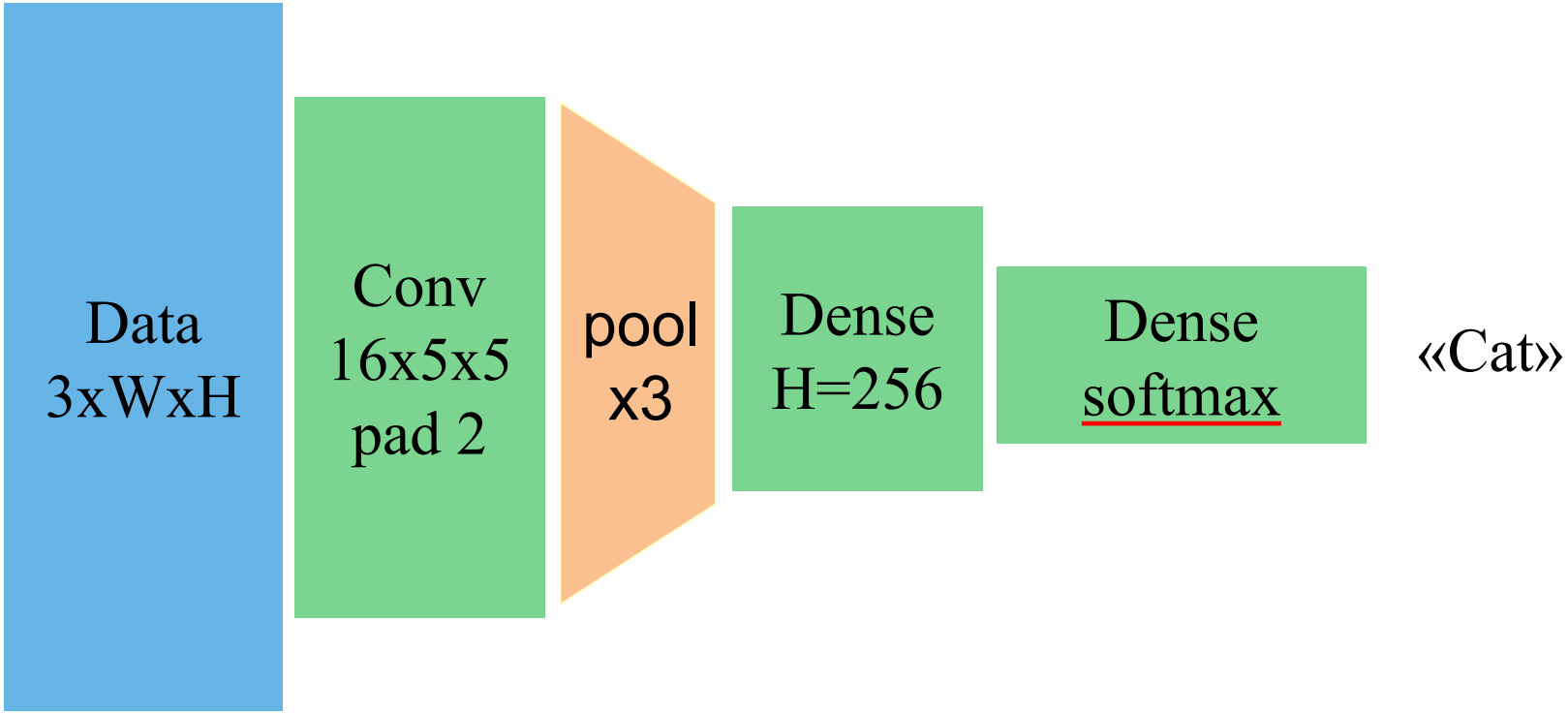
Pretraining

Use autoencoder as initialization



Pretraining

Use autoencoder as initialization



Pretraining

Supervised pre-training (on similar task)

- Needs labels for similar problem
- Luckily, we have Imagenet and Model Zoo
  - Alas, it's only good for popular problems

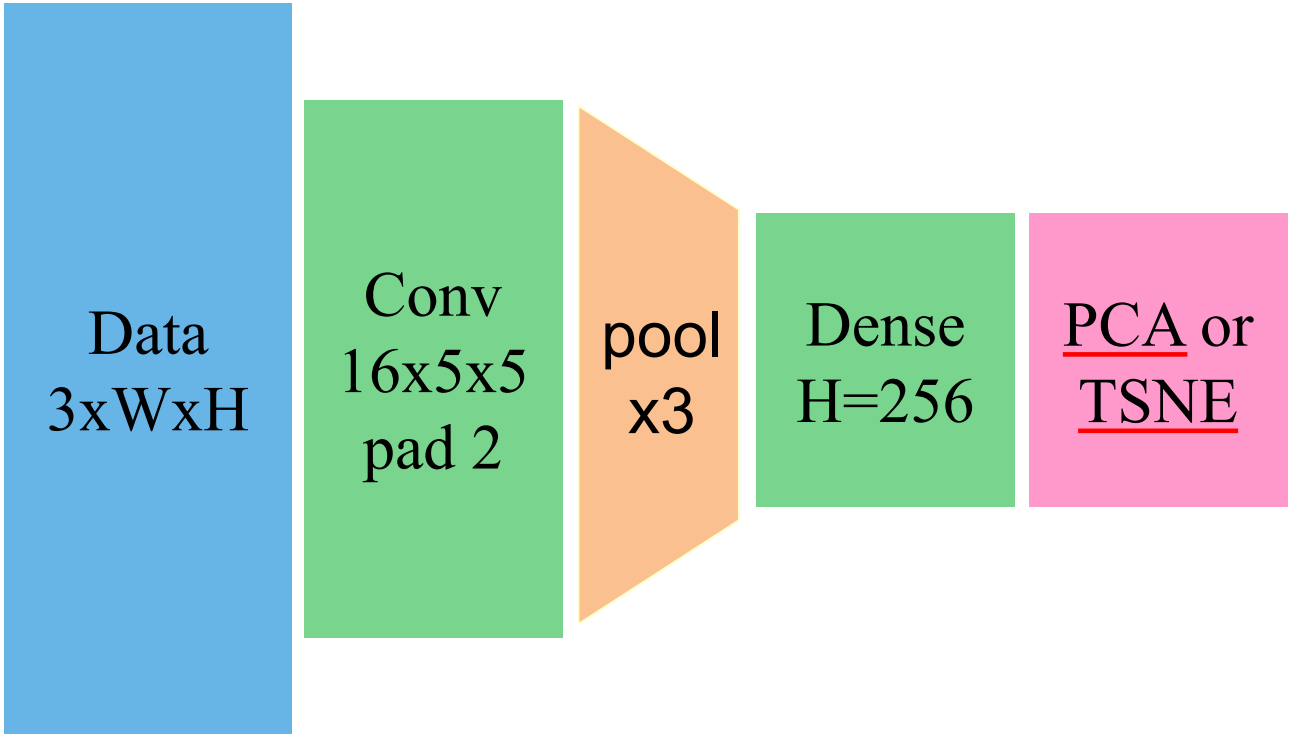
Unsupervised pretraining (autoencoder)

- Needs no labels at all!
- ✗ May learn features irrelevant to your problem
- e.g. background sky color for object classification

Exploratory data analysis

Again, no labels

Visualize data in hidden space

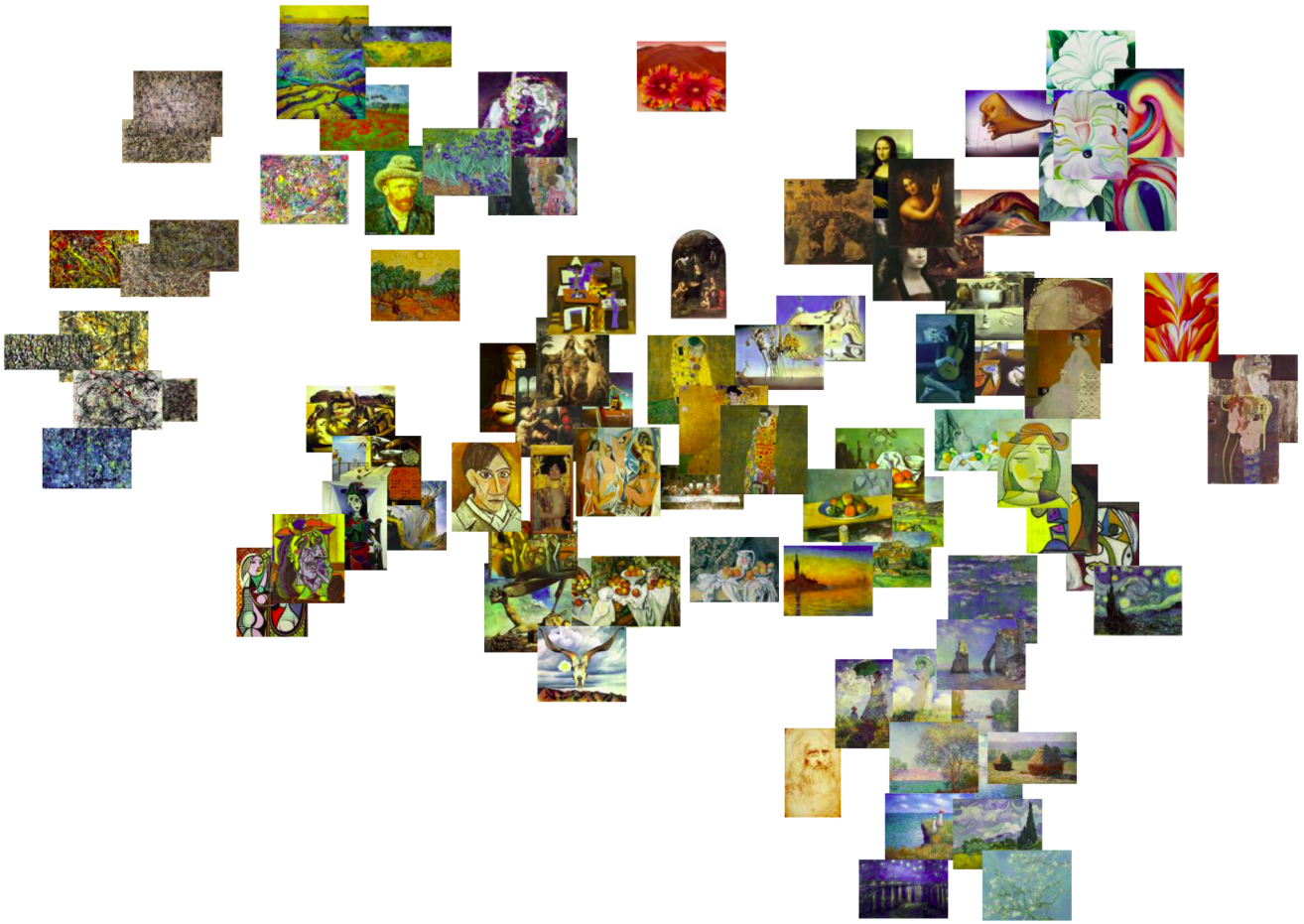


Encoder here mainly reduce dim;

helping to visualize high dim data

Exploratory data analysis

Visualize data in hidden space



Razi Shaban, https://razi.xyz/vgg2vec/picasso

Why do we ever need that?

- Compress data
  - |code| << |data|
- Dimensionality reduction
  - Before feeding data to your XGBoost
- Learn some great features!
  - Before feeding data to your XGBoost
- Unsupervised pretraining
  - Large amounts of unlabeled data
- **Generate new data!**

Image morphing with AE

Idea:

If Enc(image1) = c1

Enc(image2) = c2

Than

maybe (c1+c2)/2 is a semantic average of the two images

Decoder here mainly alter some features

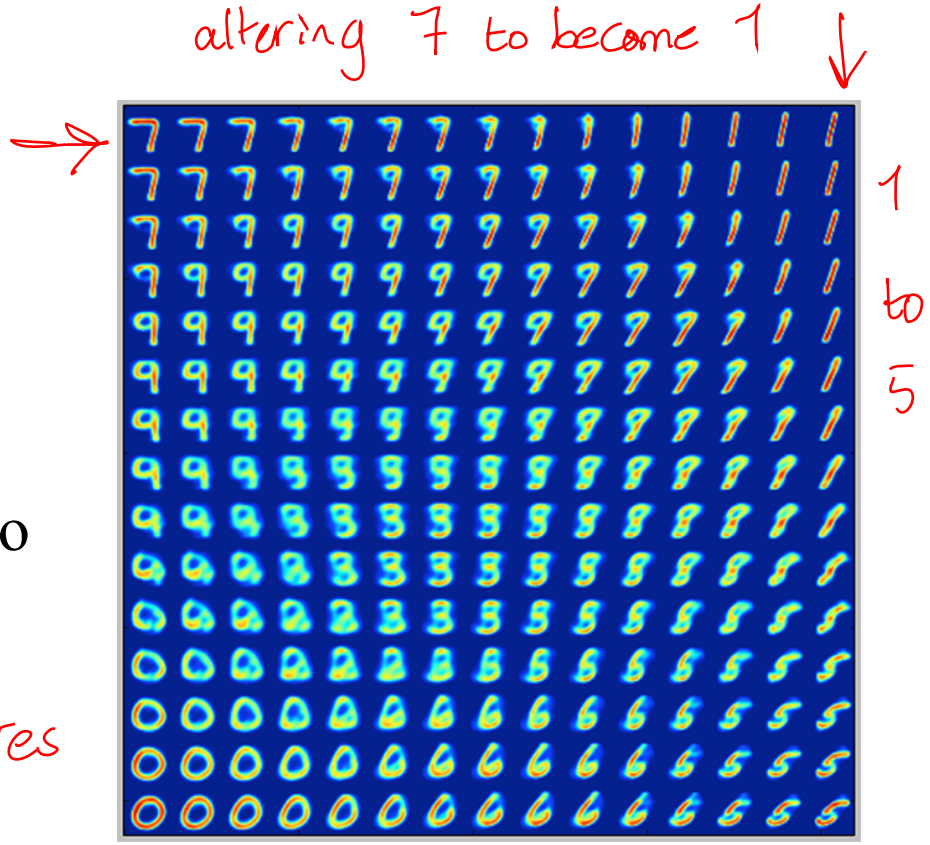
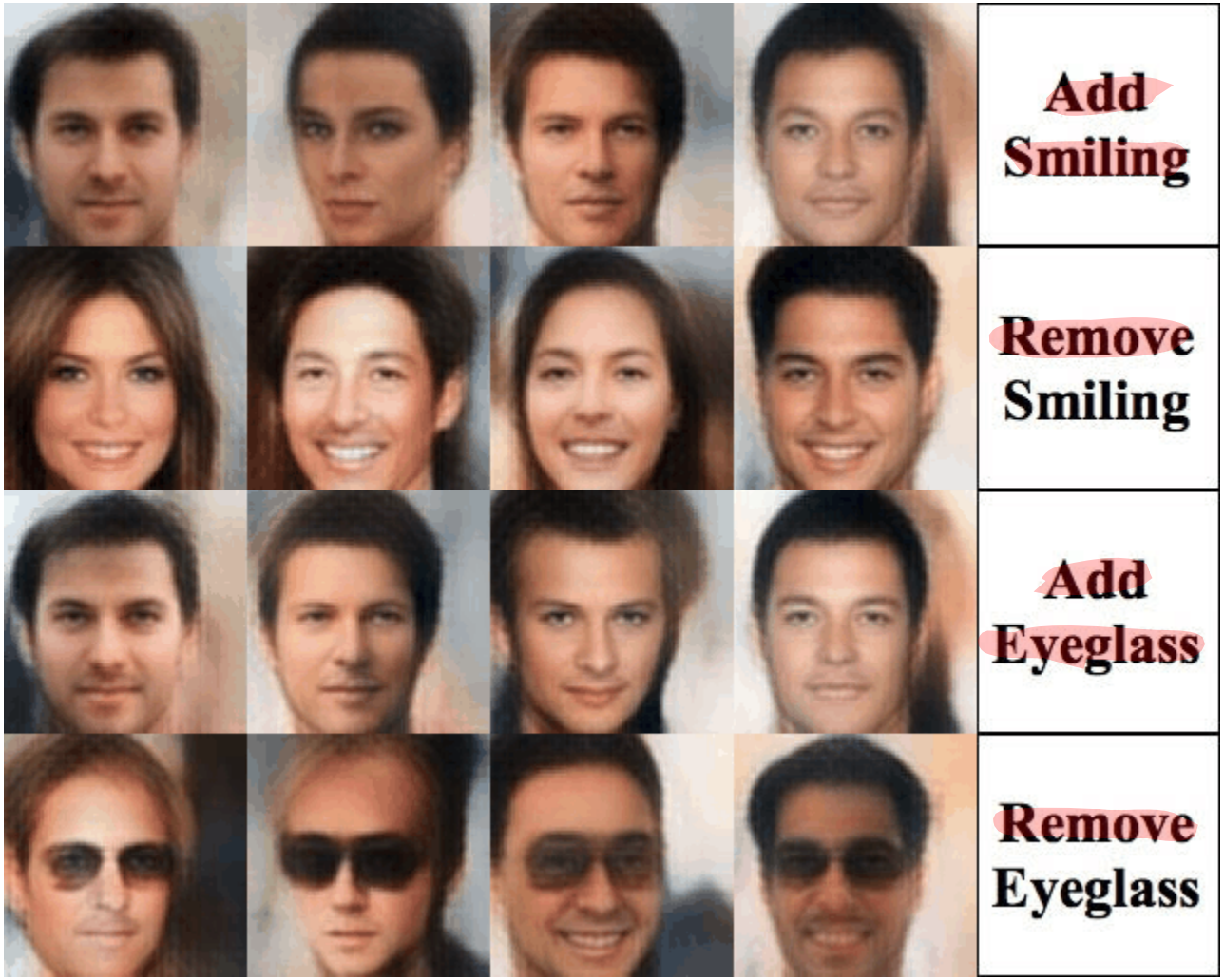


Image morphing with AE

Idea:

- Look for a common direction vector for “add mustache” or “add age” changes.
- Apply to new images

Image morphing with AE



Xianxu Hour, https://arxiv.org/abs/1610.00291

Text 101: tokens

unsupervised learning

Text:

A sequence of tokens(words).

Token/word:

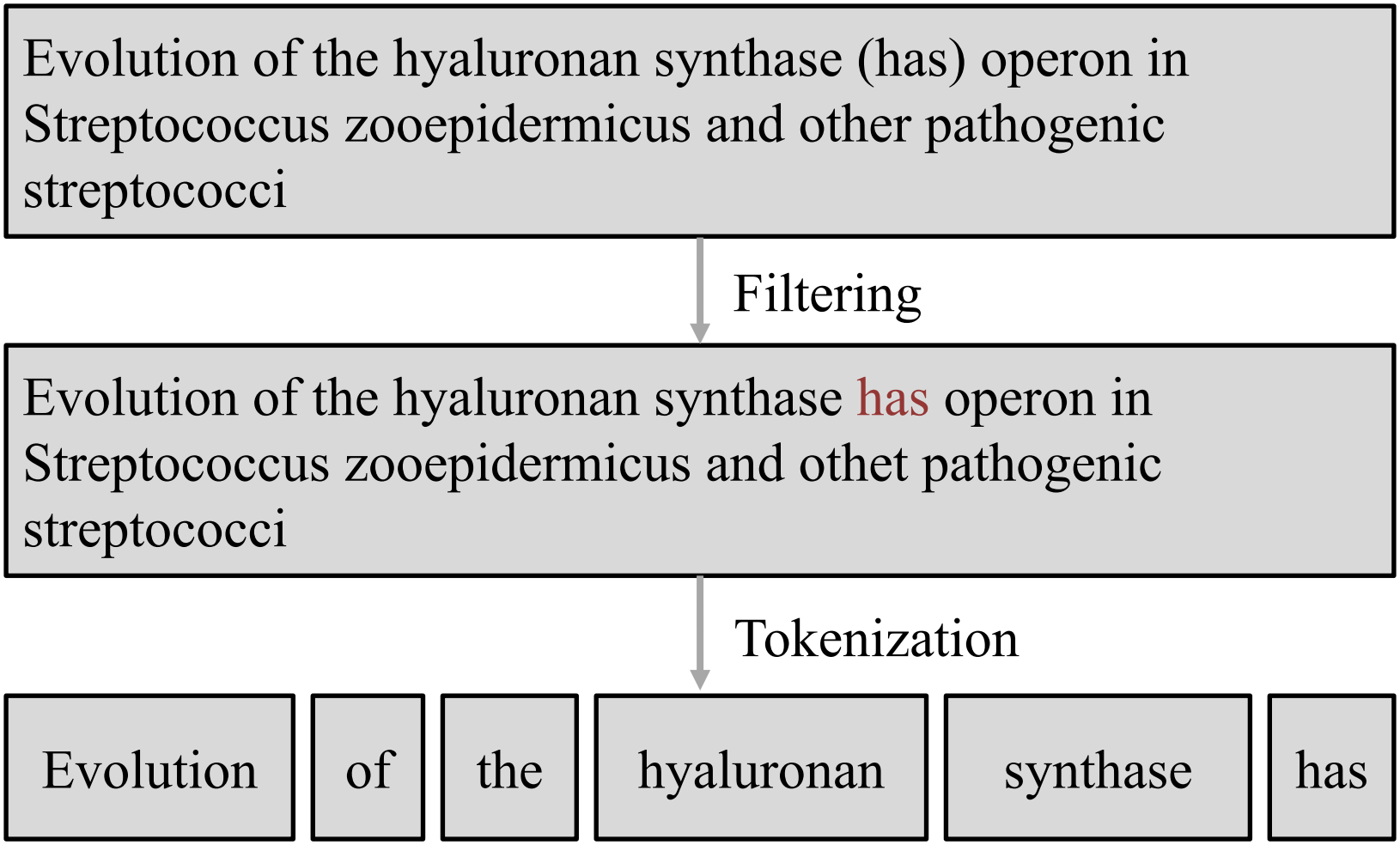
A sequence of characters.

Character:

An atomic element of text.

¯_(ツ)_/¯

Text 101: tokens



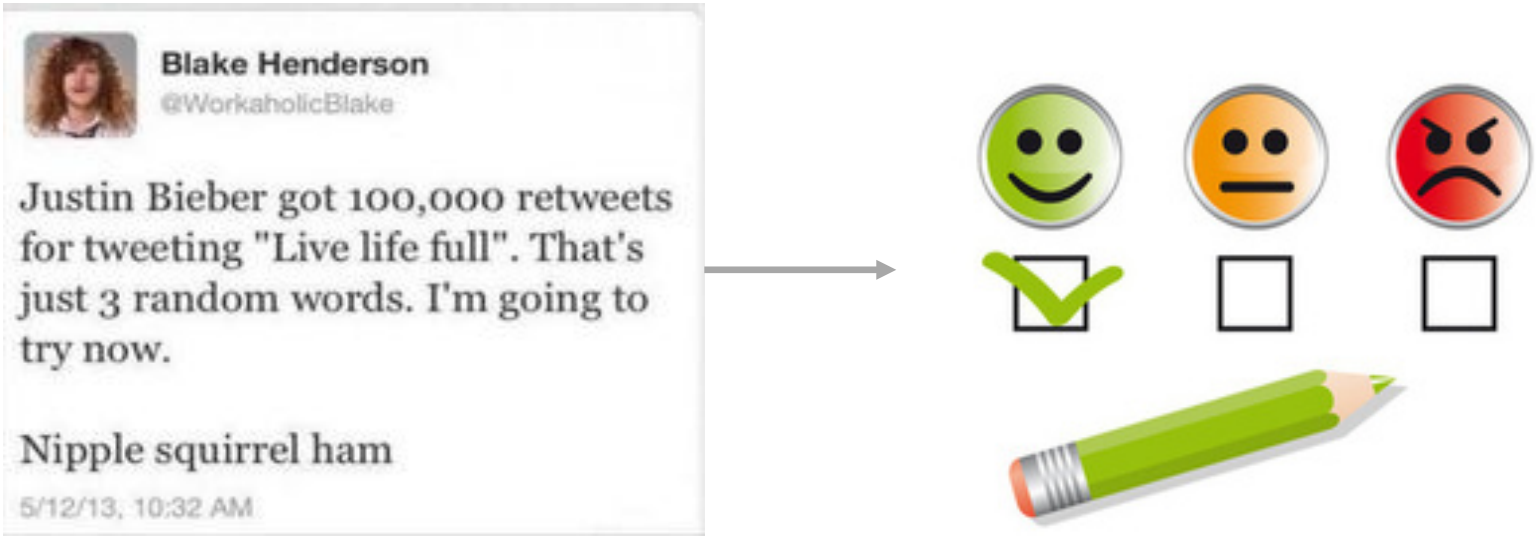
What is a text 101: bag of words

is a simplifying tech in NLP, where grammar & word order are ignored, while multiplicity is kept

Journal of Artificial Intelligence Research JAIR is a refereed journal, covering the areas of Artificial intelligence, which is distributed free of charge over the Internet. Each volume of the journal is also published by Morgan Kaufmann

0	learning
3	journal
2	intelligence
0	text
1	Internet
…	…

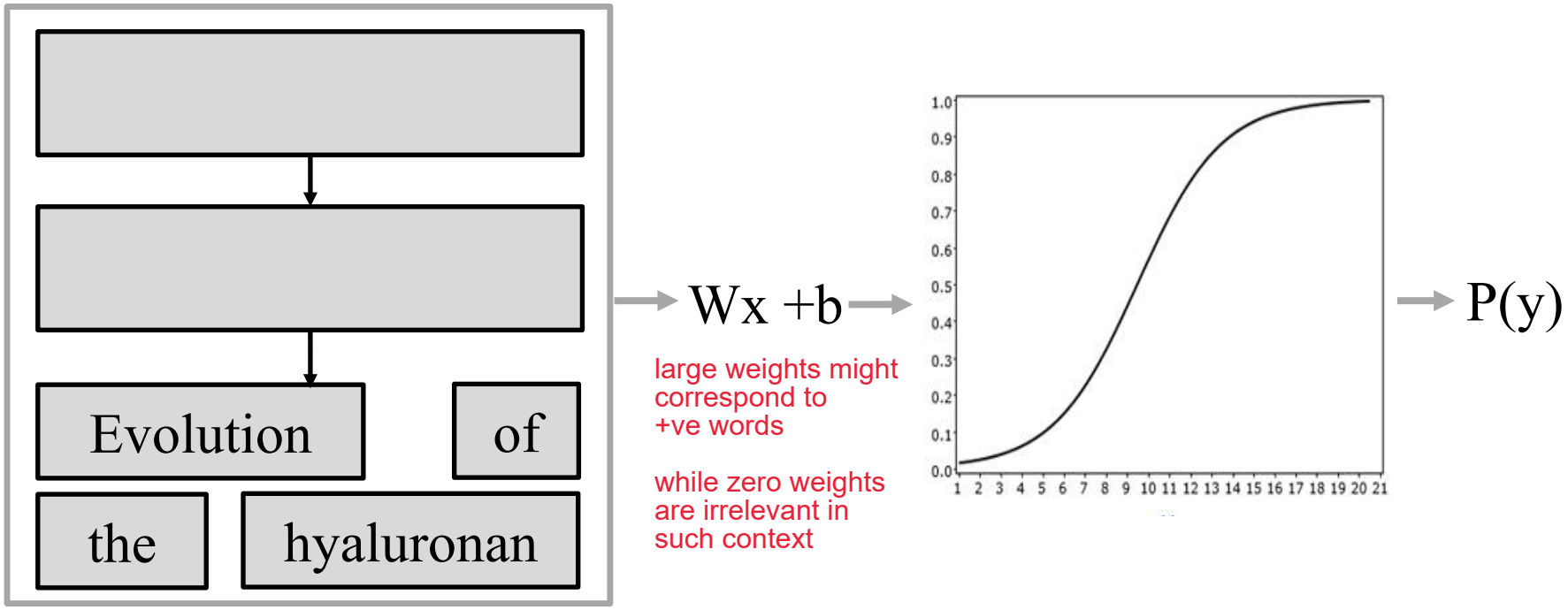
Text classification/regression



Other usage:

- Adult content filter (safe search)
- Detect age/gender/interests by search queries
- Convert movie review into “stars”
- Survey public opinion for the new iphone Vs old one (SNA)
- ...

Text classification: BoW + linear

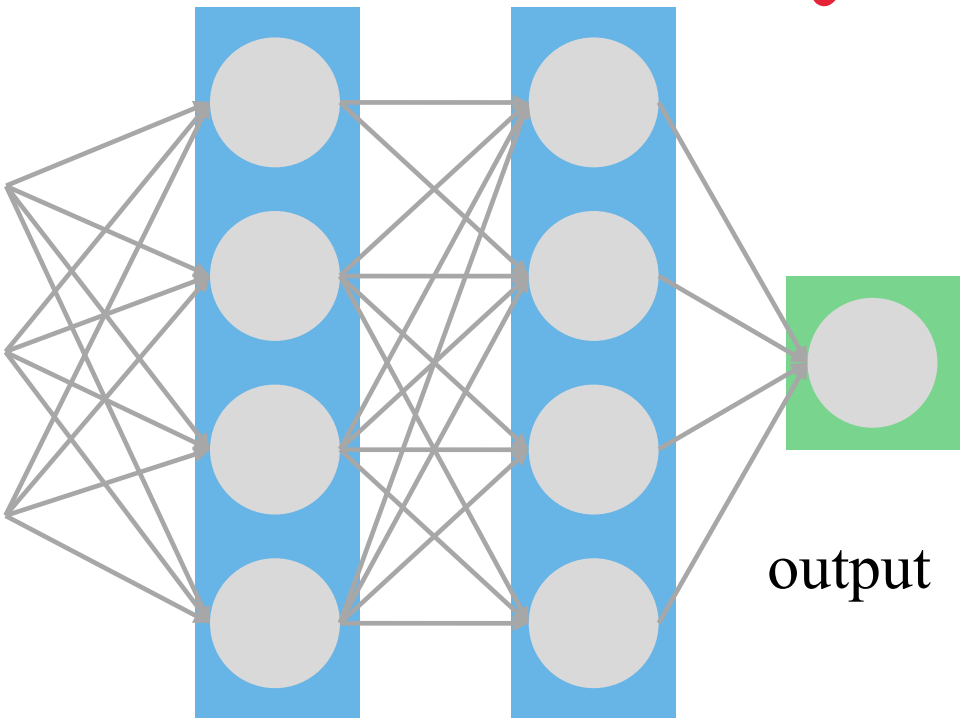


Divination:

How many features (approx.) will such model have?

Text classification: BoW + linear

input

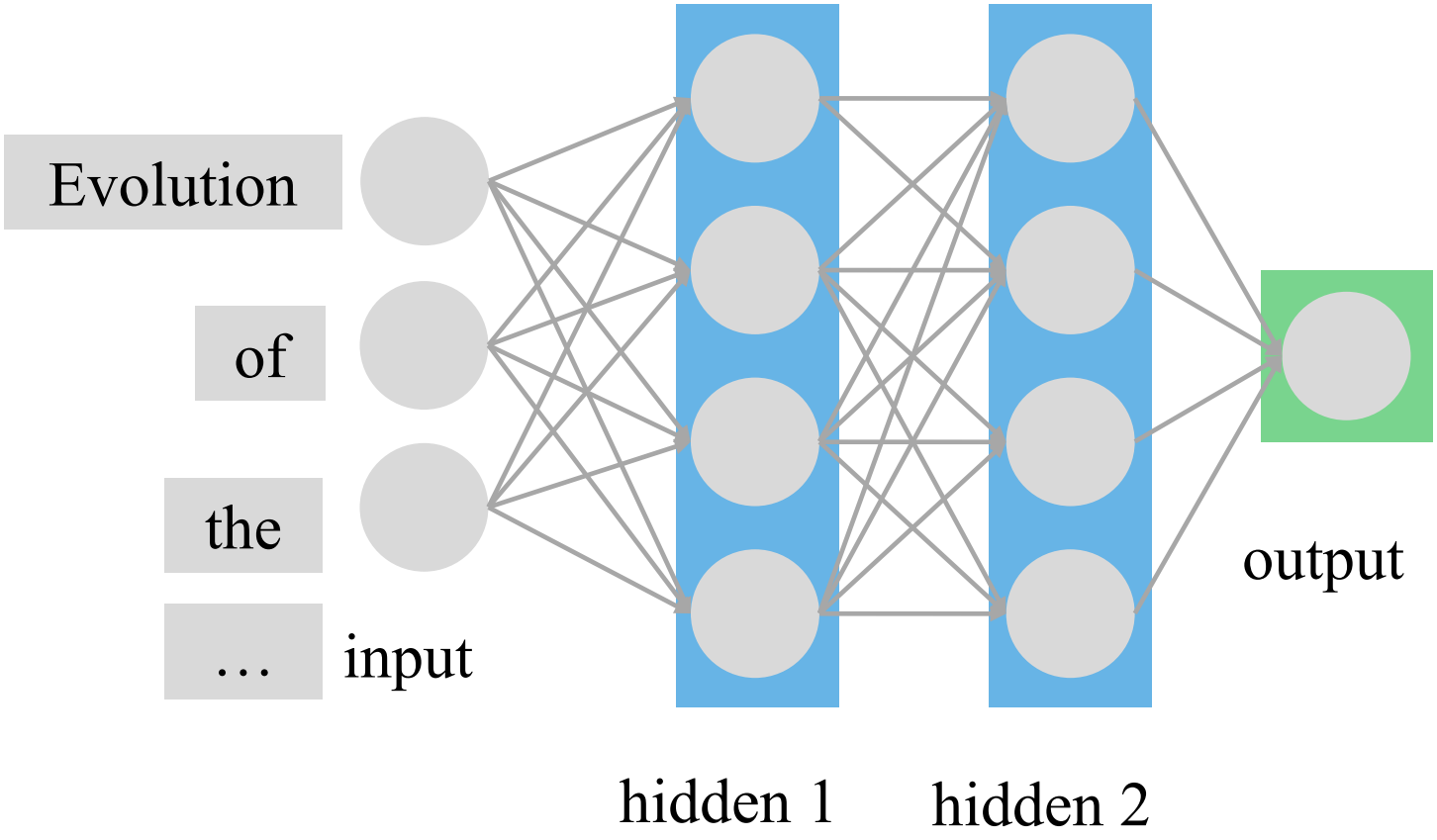


output

hidden 1 hidden 2

Text classification: BoW + linear

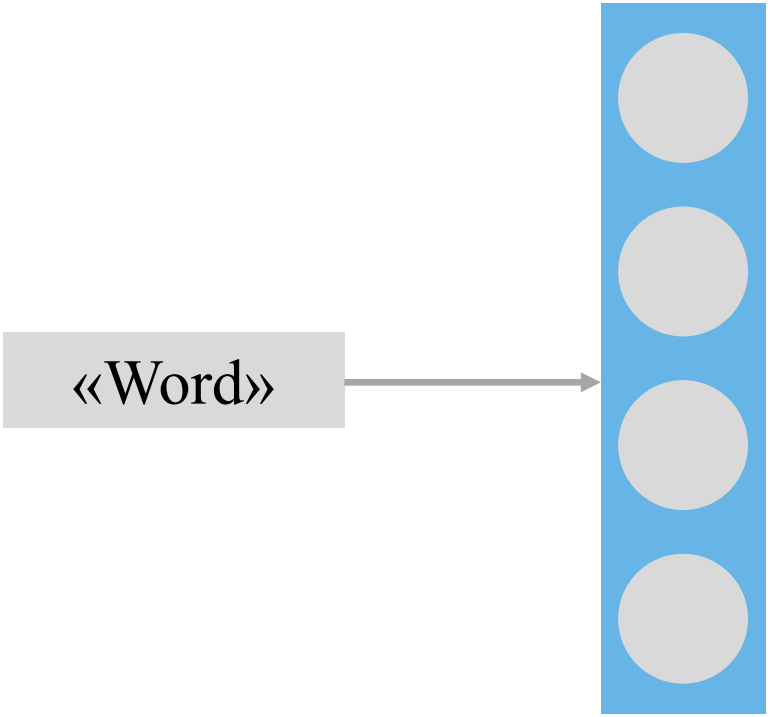
DNN can be used in such context as well, however it won't solve the main problem of sentiments



Word embeddings

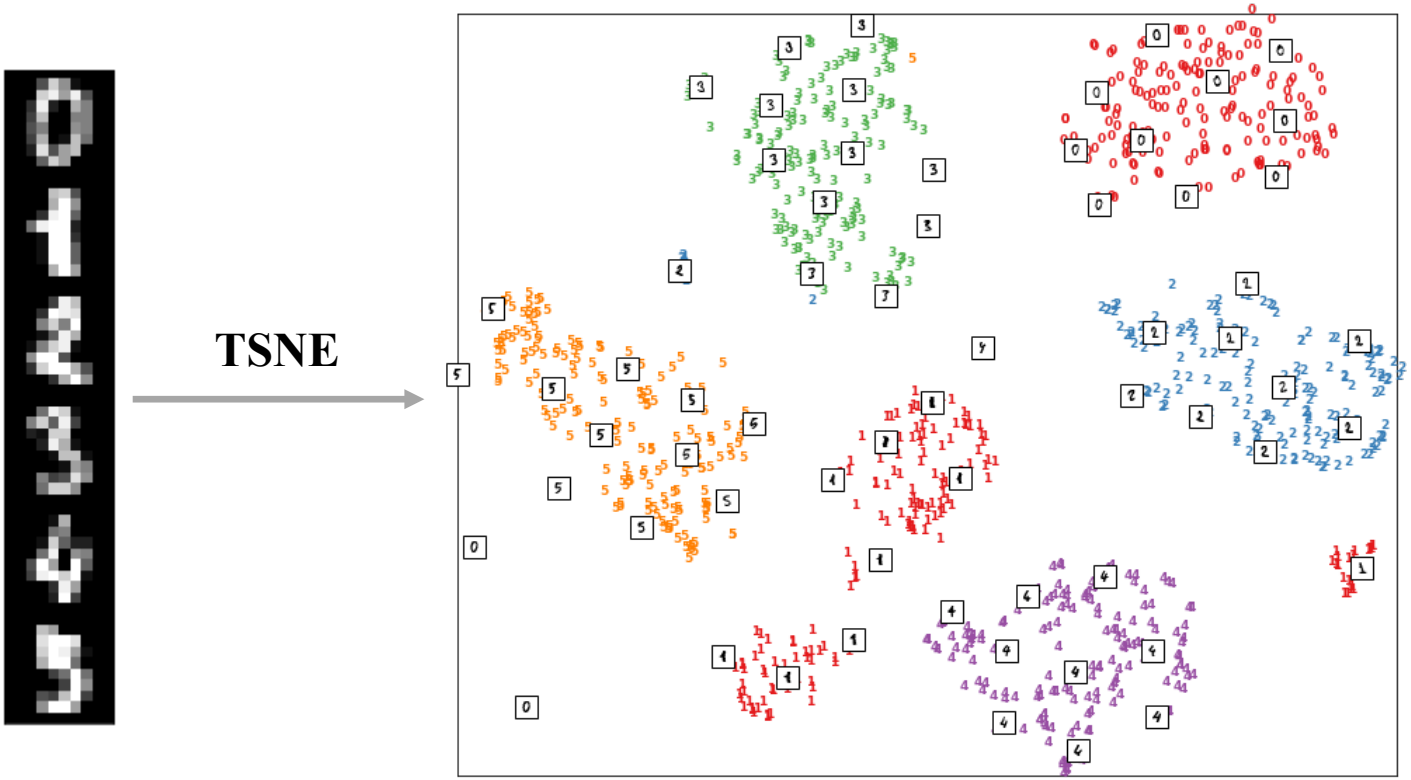
i.e. words like like & dislike shall be distanced apart

We want a compact representation of text so that we could use it for neural nets!



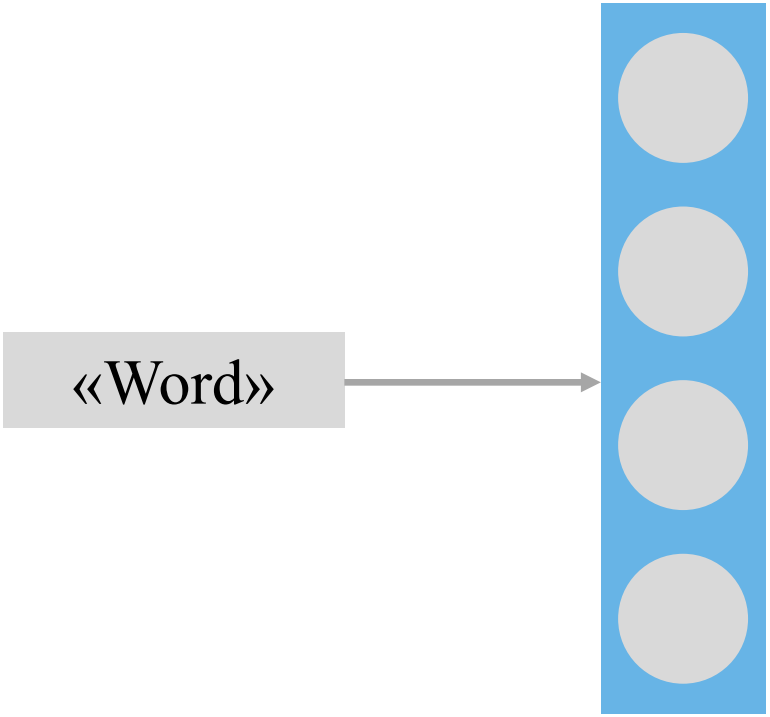
Recap: embeddings

Map data into a lower dimensional space while preserving structure. MDS, LLE, **TSNE**, etc.



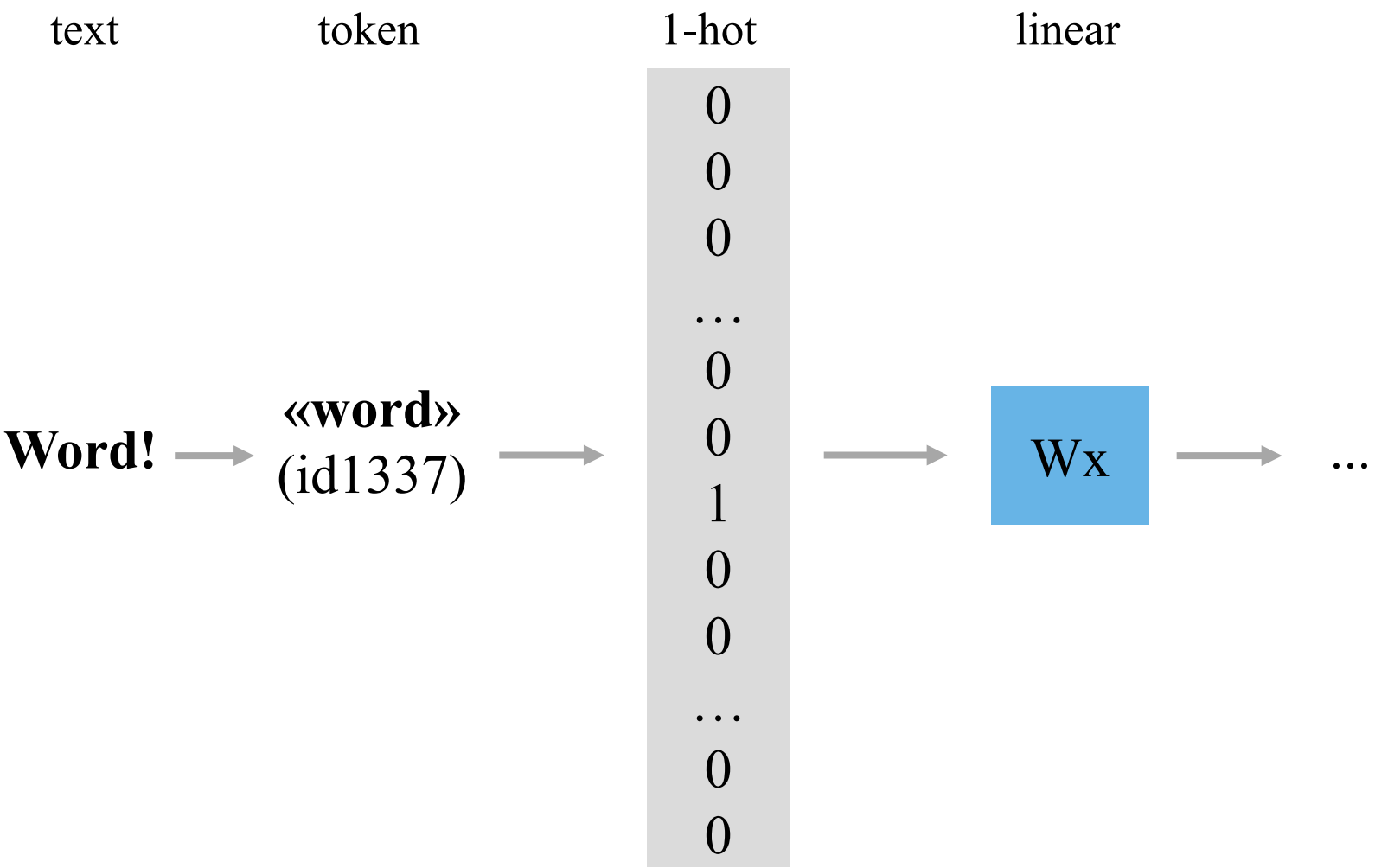
Word embeddings

We want a compact representation of text so that we could use it for neural nets!

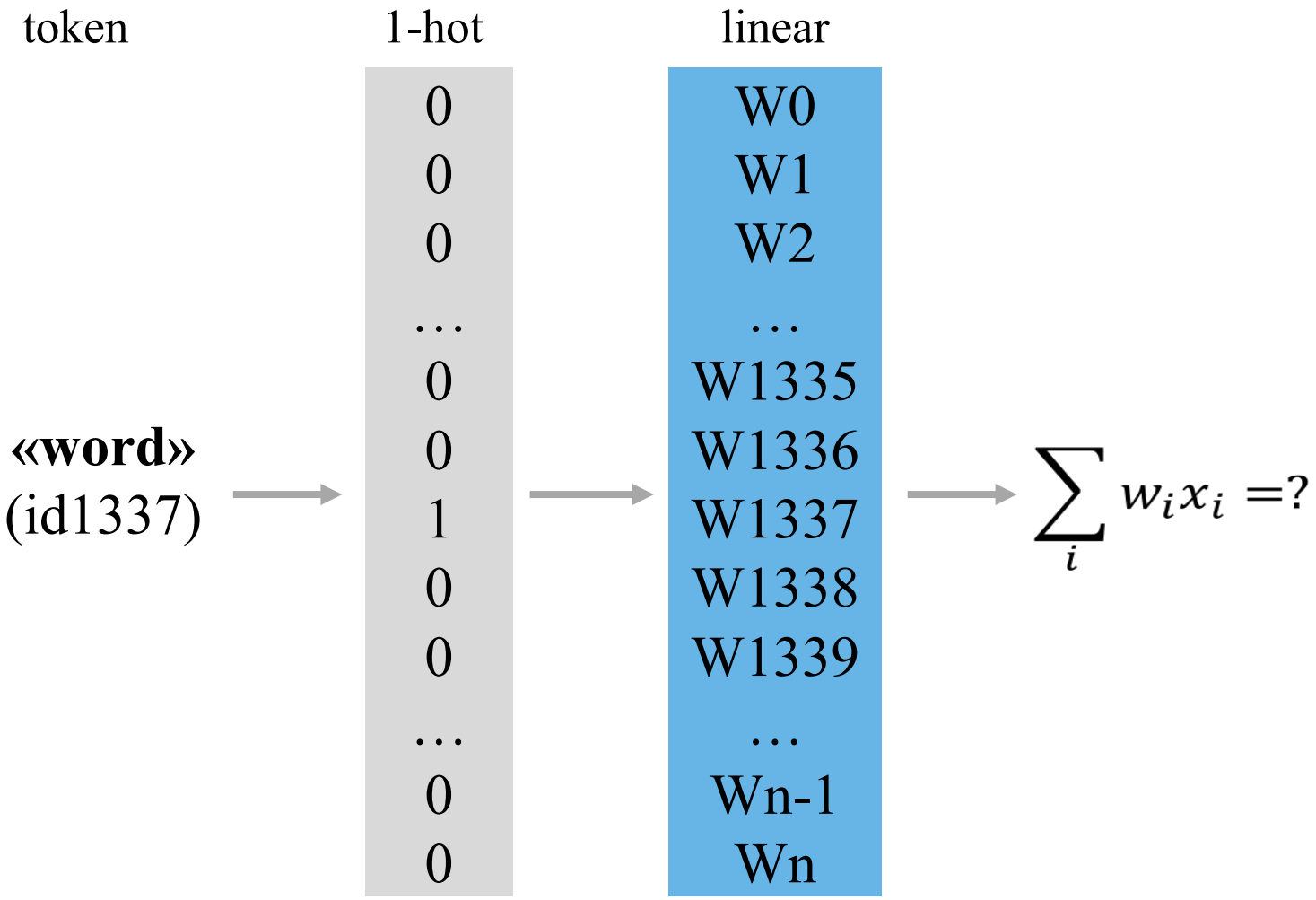


Sparse vector products

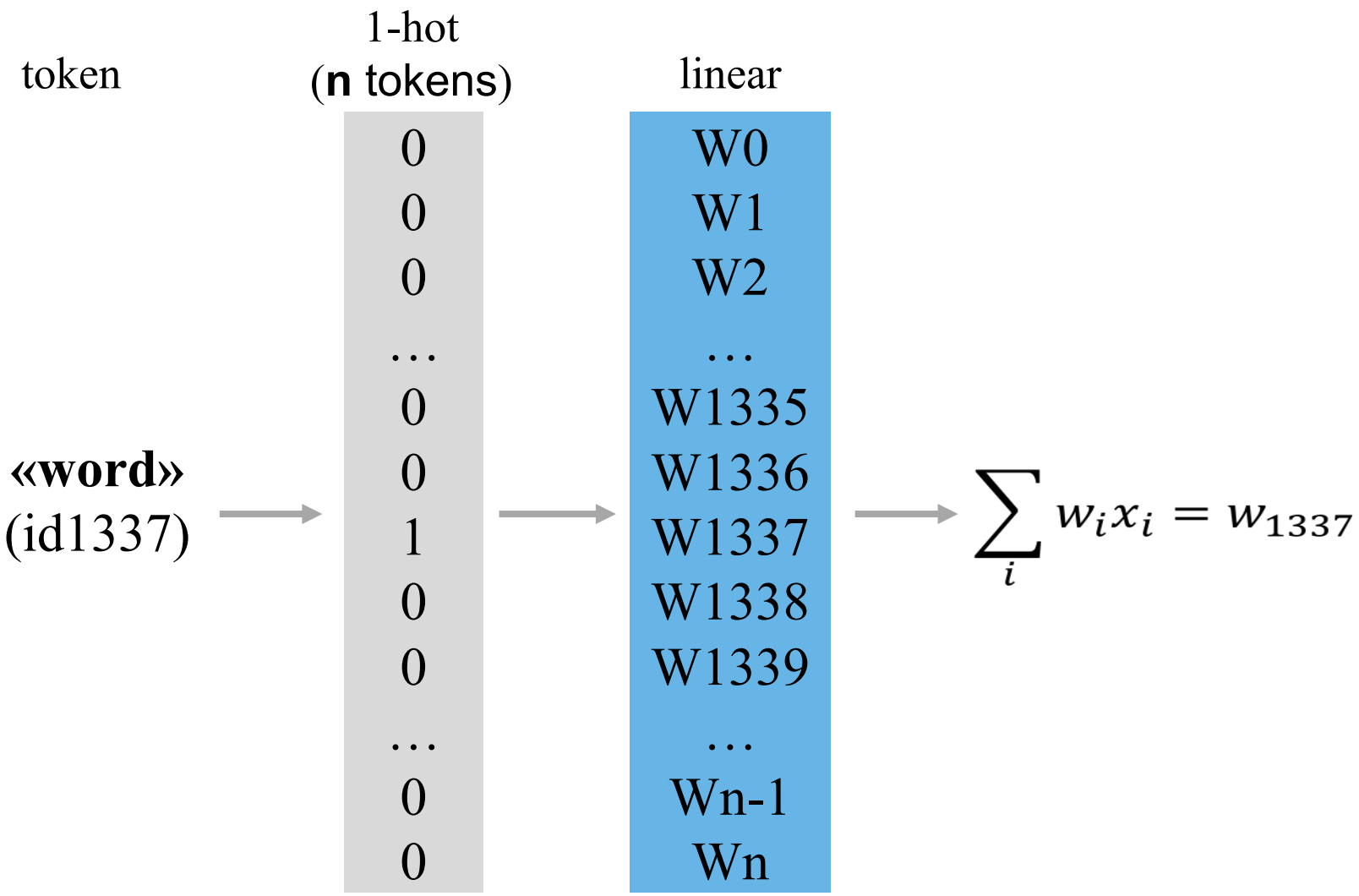
how to efficiently represent a word



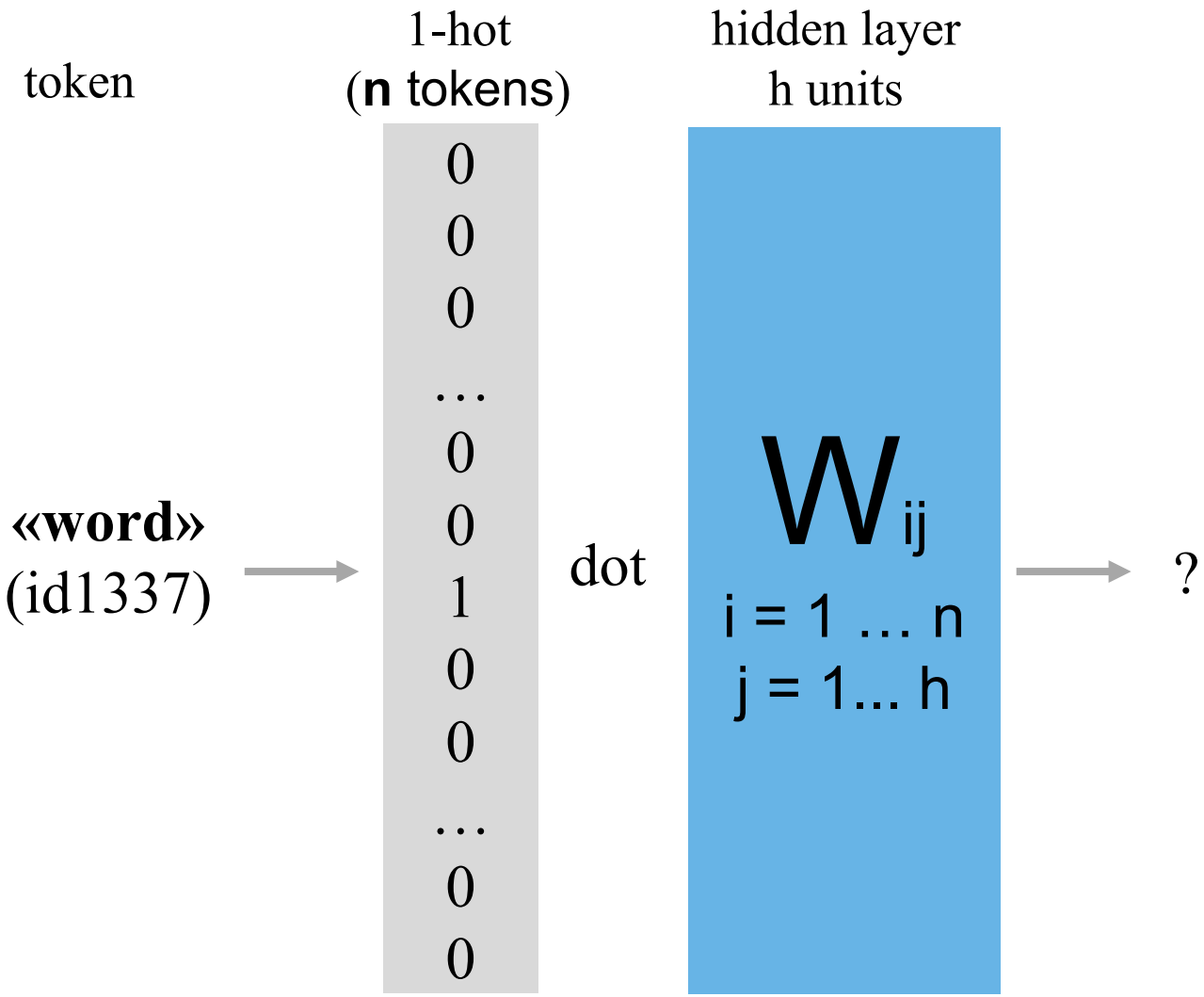
Sparse vector products



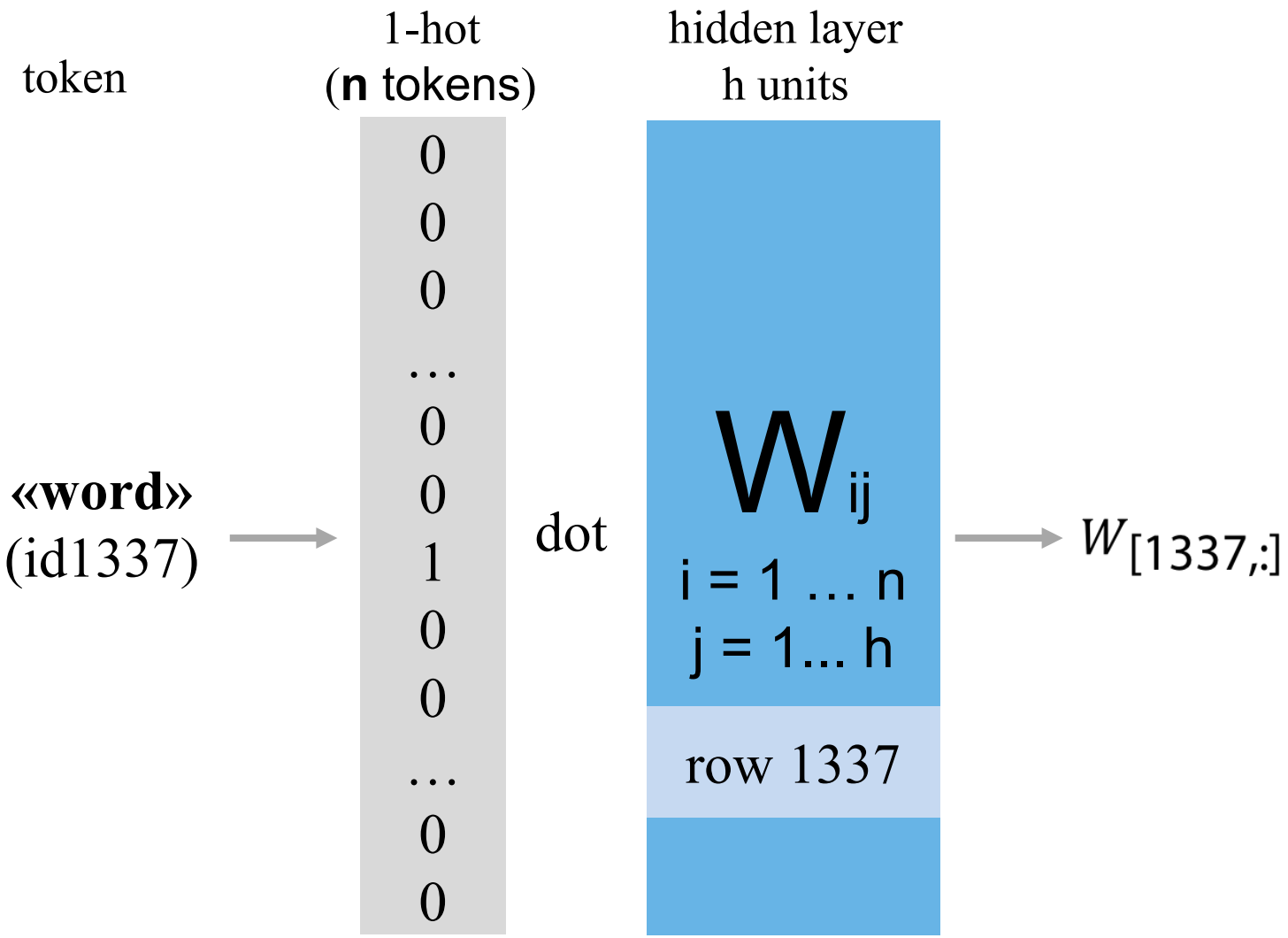
Sparse vector products



Sparse vector products



Embedding

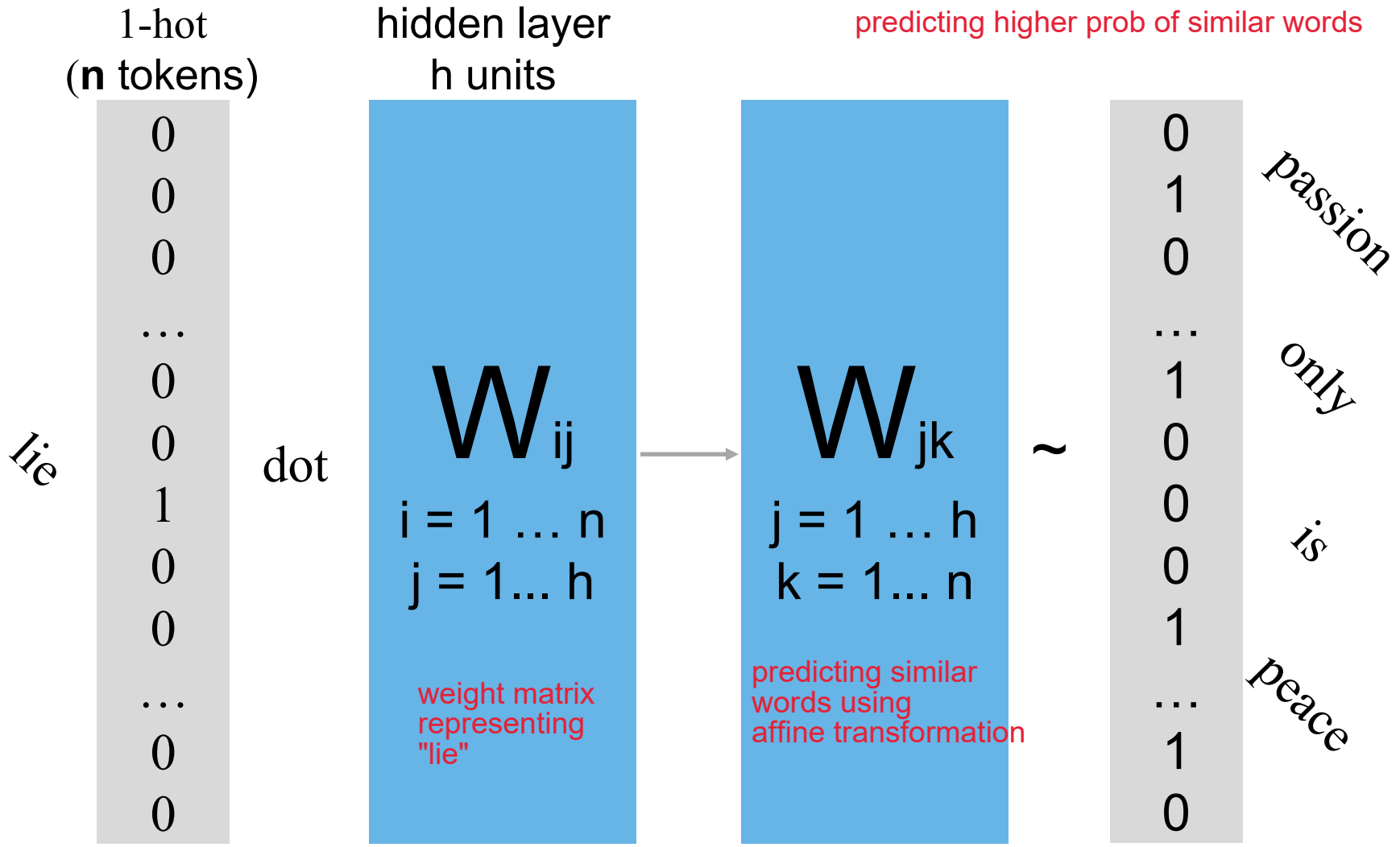

token
1-hot
(n tokens)
hidden layer
h units
«word»
(id1337)
0
0
0
…
0
0
1
0
0
…
0
0
dot
W_{ij}
i = 1 … n
j = 1... h
row 1337
$W_{[1337,:]}$

Embedding: word2vec

one way

taking one word & try predicting the next sequence of words

"Peace is a lie, there is only passion"

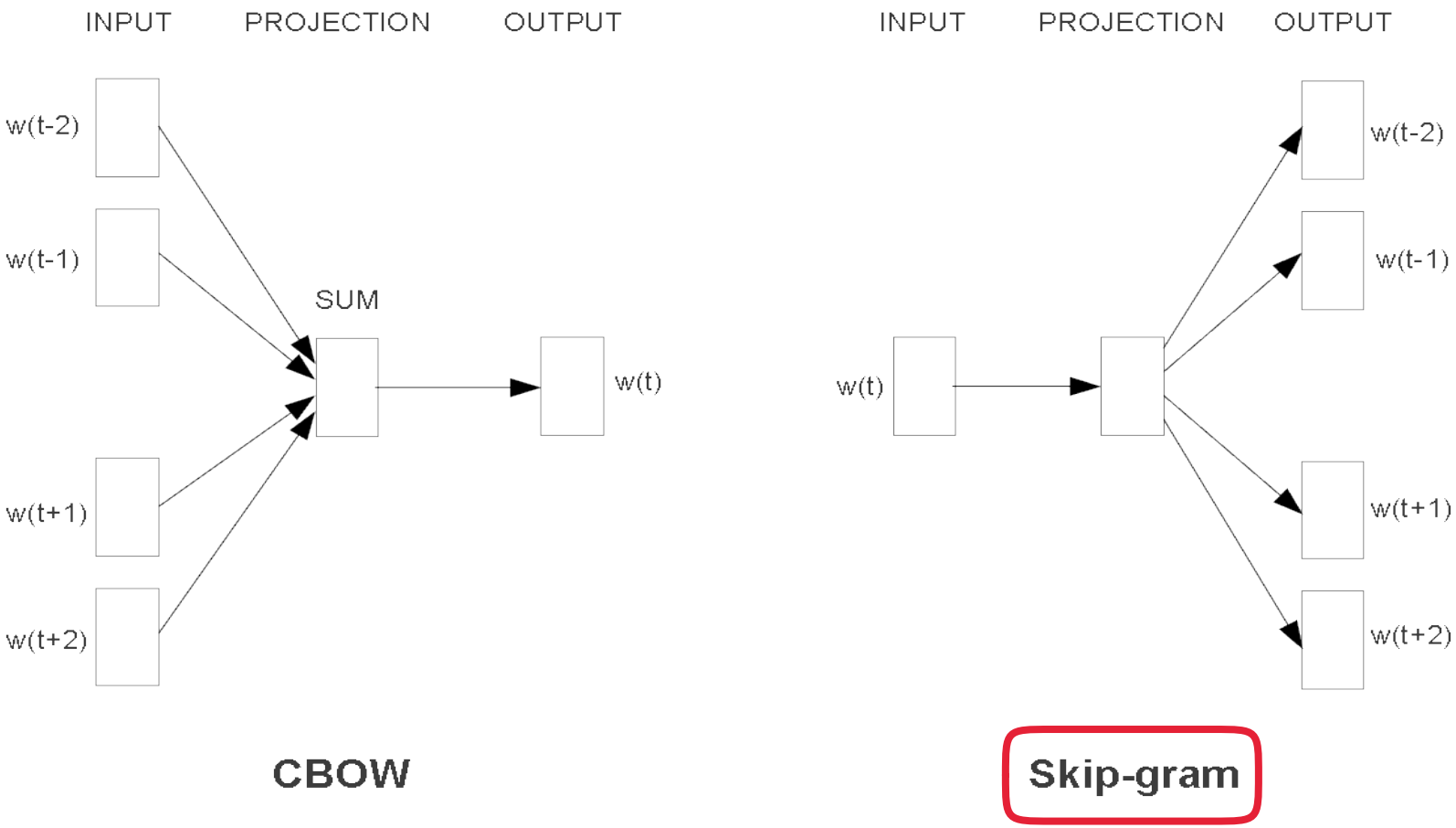


Embedding: word2vec

another way

take seq of words & try predicting the next word

the *distributional hypothesis* : similar context = similar meaning



Yang Huijeong, http://cscp2.sogang.ac.kr/CSE4187_02/index.php/%ED%8C%8C%EC%9D%BC:8.png

Embedding: word2vec

Side effect: synonyms

“nice” ~ “beautiful”

“hard” ~ “difficult”

Embedding: word2vec

not very side effect though

Side effect: synonyms

“nice” ~ “beautiful”

“hard” ~ “difficult”

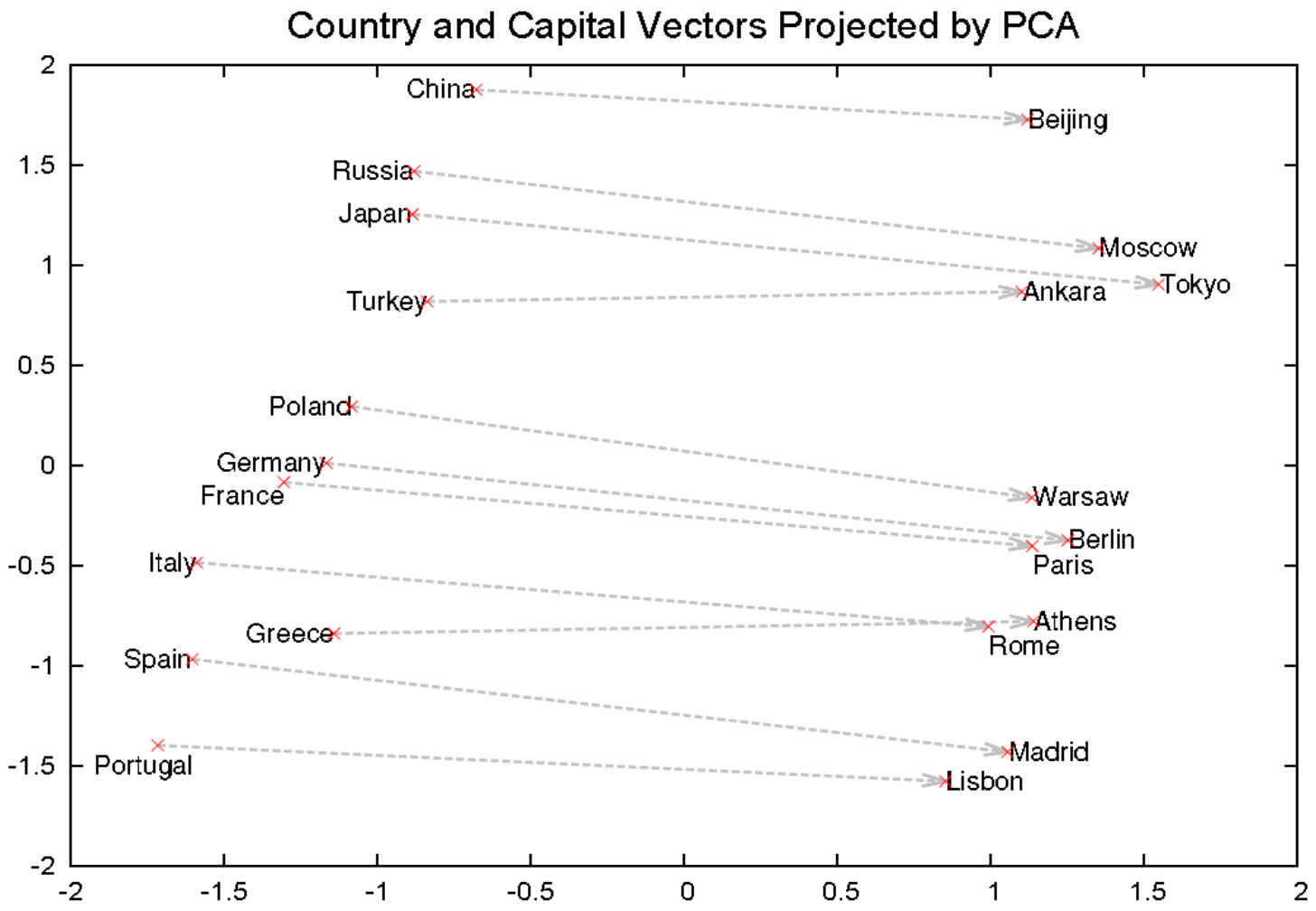
Side effect: word algebra

“king” - “man” + “woman” ~ “queen”

“moscow” - “russia” + “france” ~ “paris”

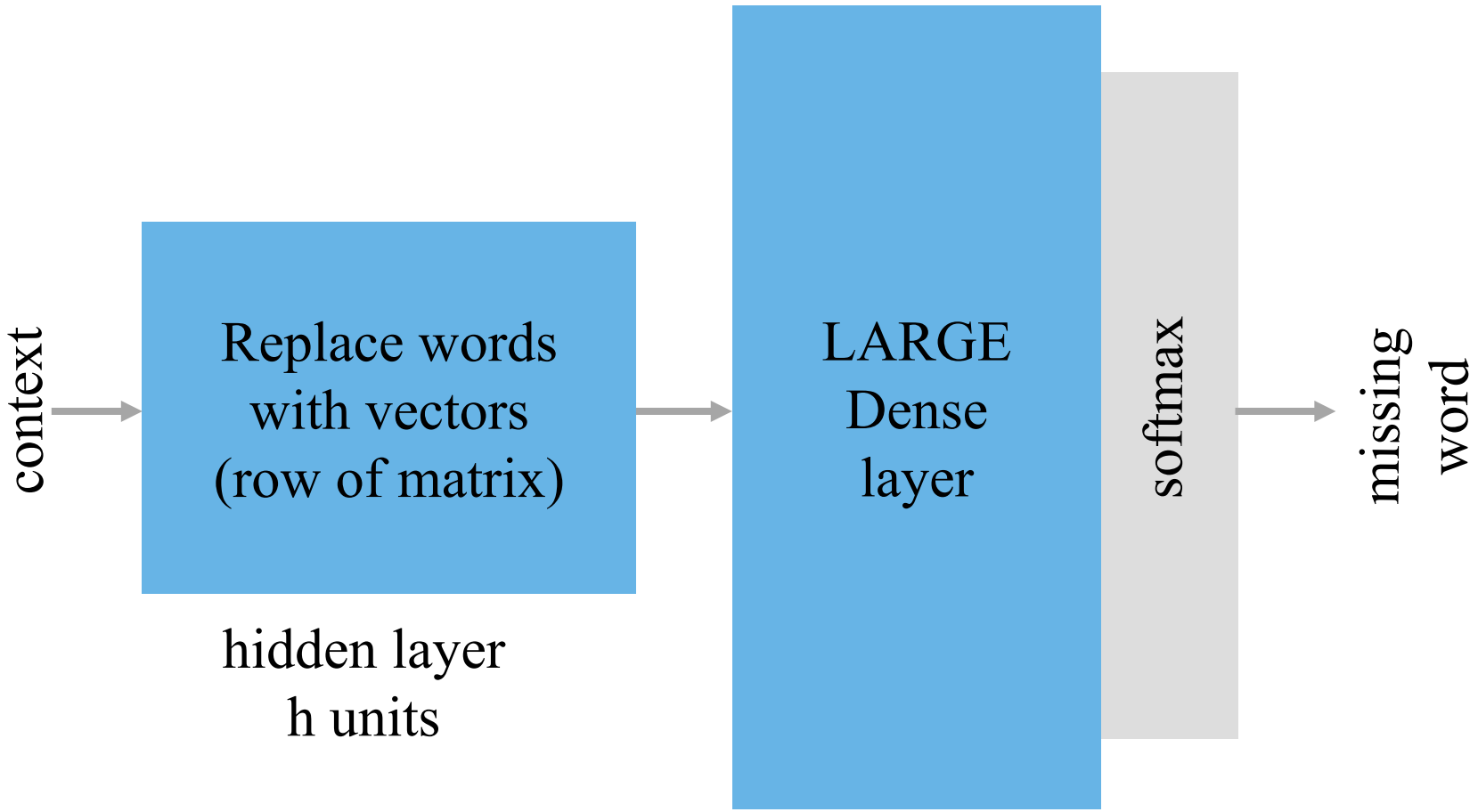
Embedding: word2vec

Side effect: word algebra

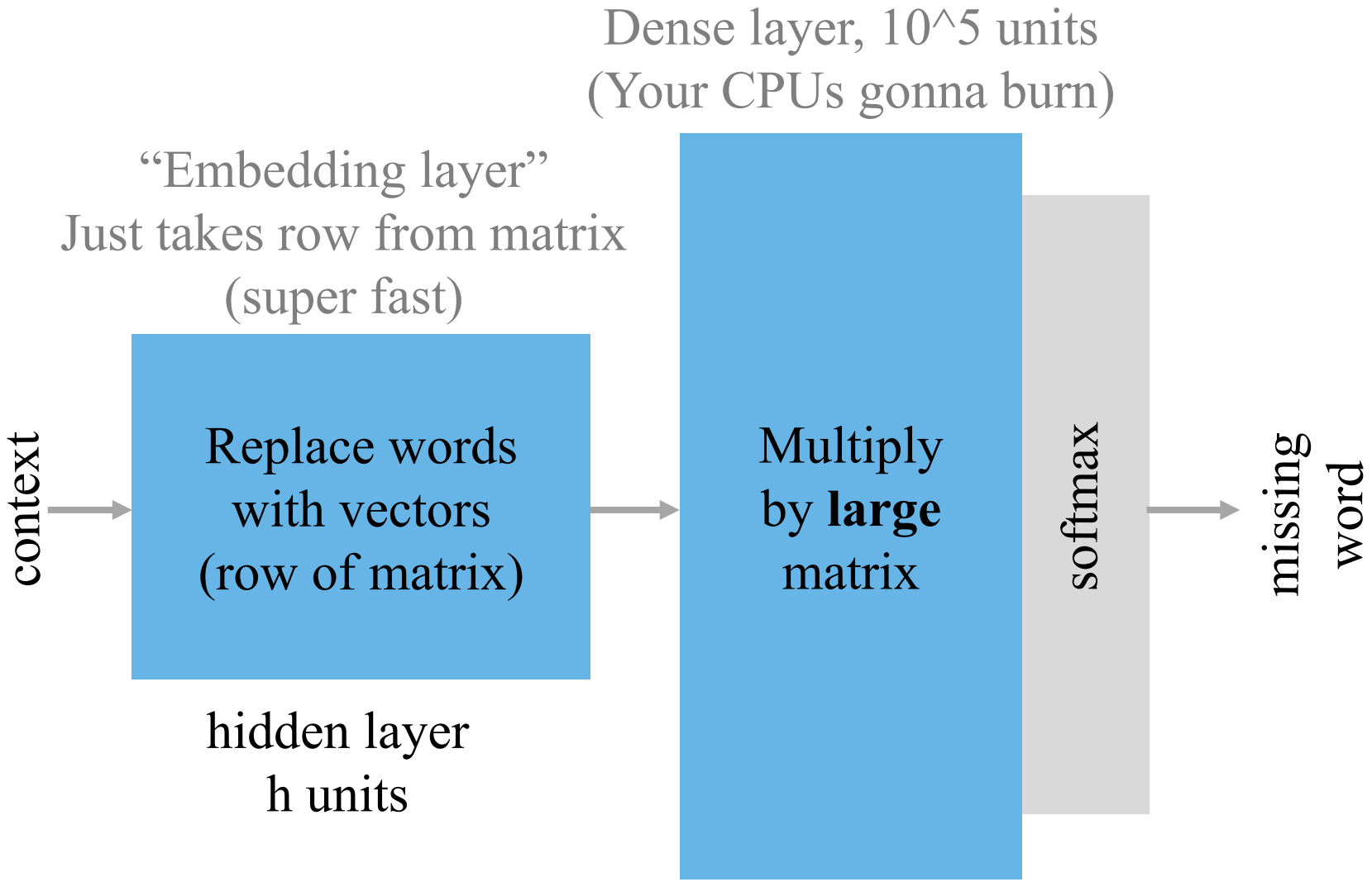


Softmax problem

how can we train just vectors



Softmax problem



More word embeddings

Faster softmax:

- Hierarchical softmax, negative samples, …
- learn more

More word embeddings

Faster softmax:

- Hierarchical softmax, negative samples, …
- learn more

Alternative models: GloVe

More word embeddings

Faster softmax:

- Hierarchical softmax, negative samples, …
- learn more

Alternative models: GloVe

Sentence level:

- Doc2vec, skip-thought (using rnn)

More word embeddings

because softmax would be
computationally expensive in this case

Faster softmax:

- Hierarchical softmax, negative samples, …
- learn more

Alternative models: GloVe

Sentence level:

- Doc2vec, skip-thought (using rnn)

To be continued...
in the NLP course

Generative models

Supervised vs Unsupervised

Supervised learning

- Take (x,y) pairs

Unsupervised learning

- Take x alone

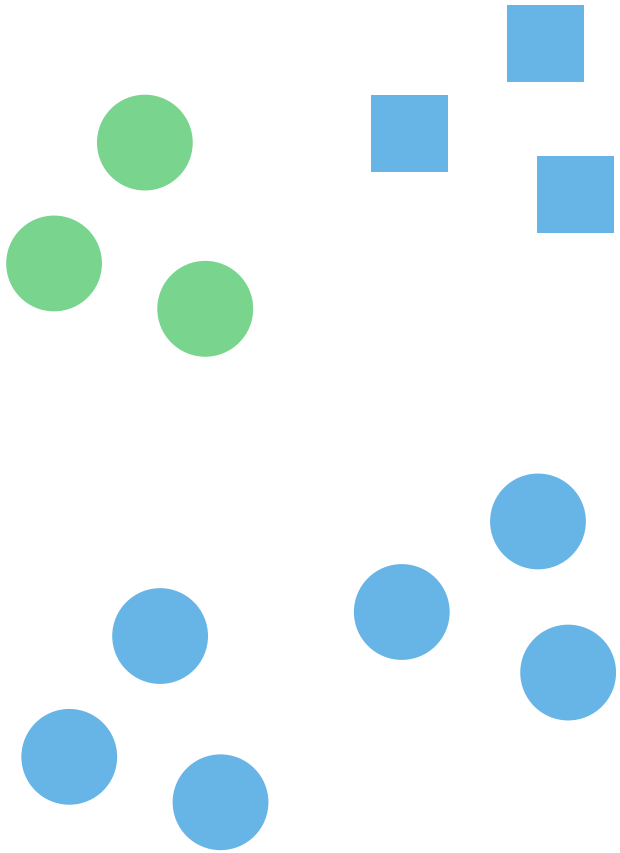
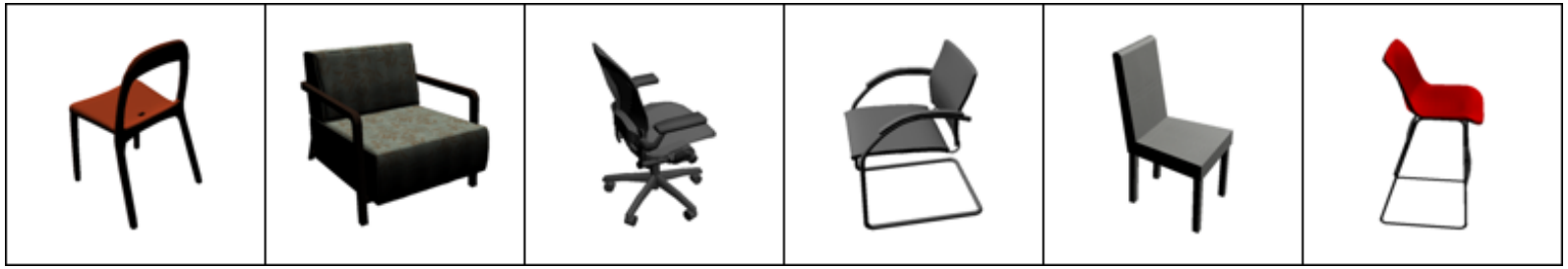


Image generation

Chairs (type, view, orientation)



Classifier

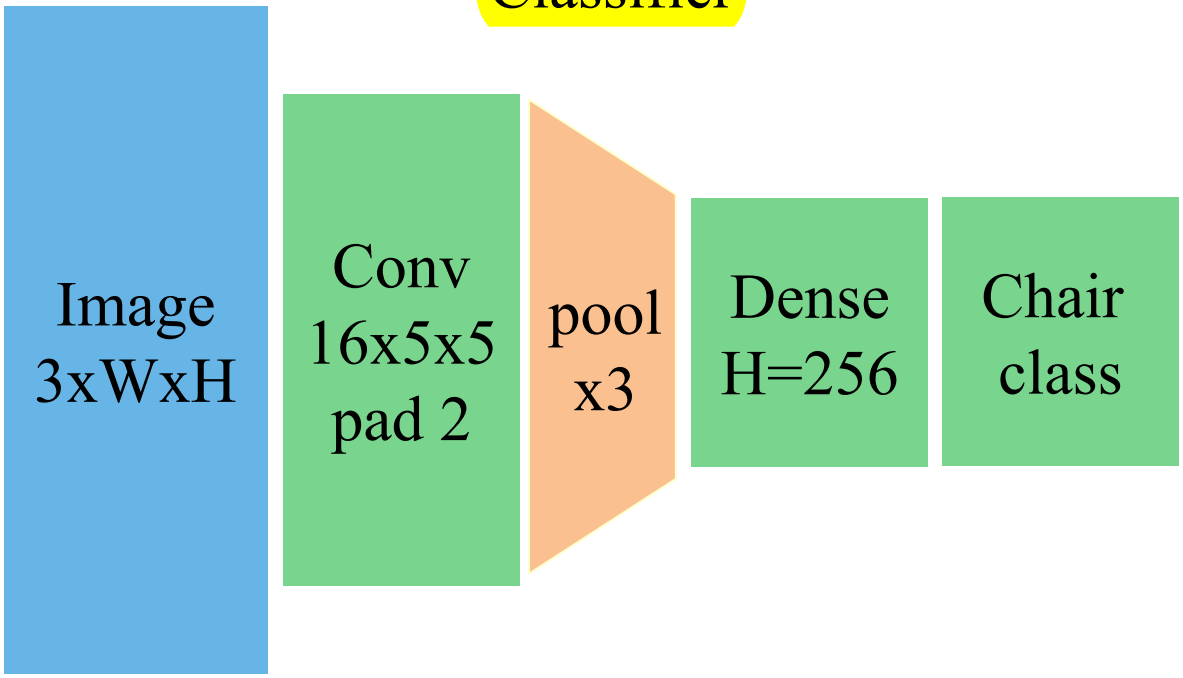
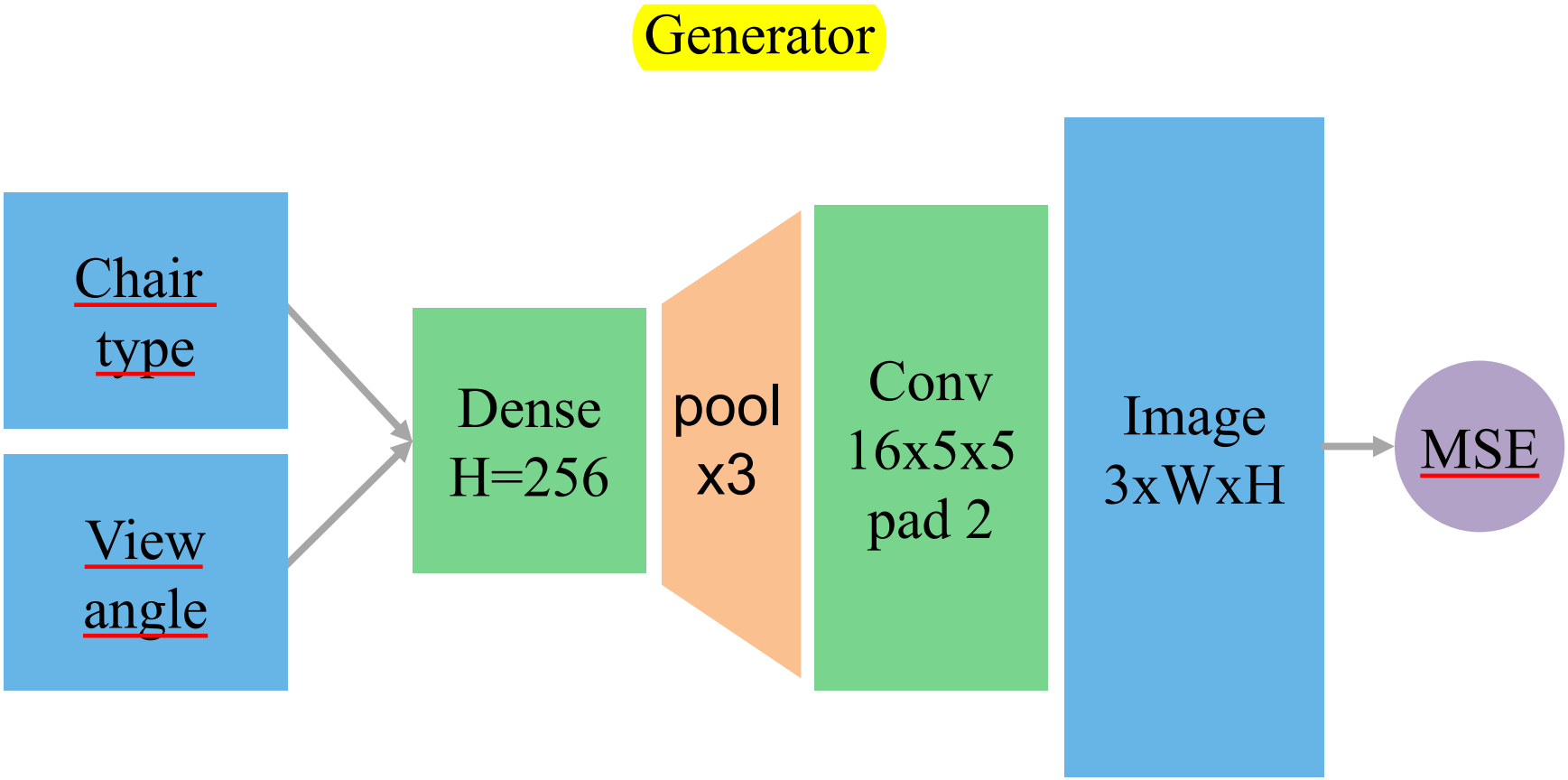


Image generation



Mean Squared Error

Pixelwise MSE:

- A **«cat on the left»** is closer to **«dog on the left»** than to **«cat on the right»**
- We may want to avoid that effect
- Can we obtain image representation that is less sensitive to small shifts?

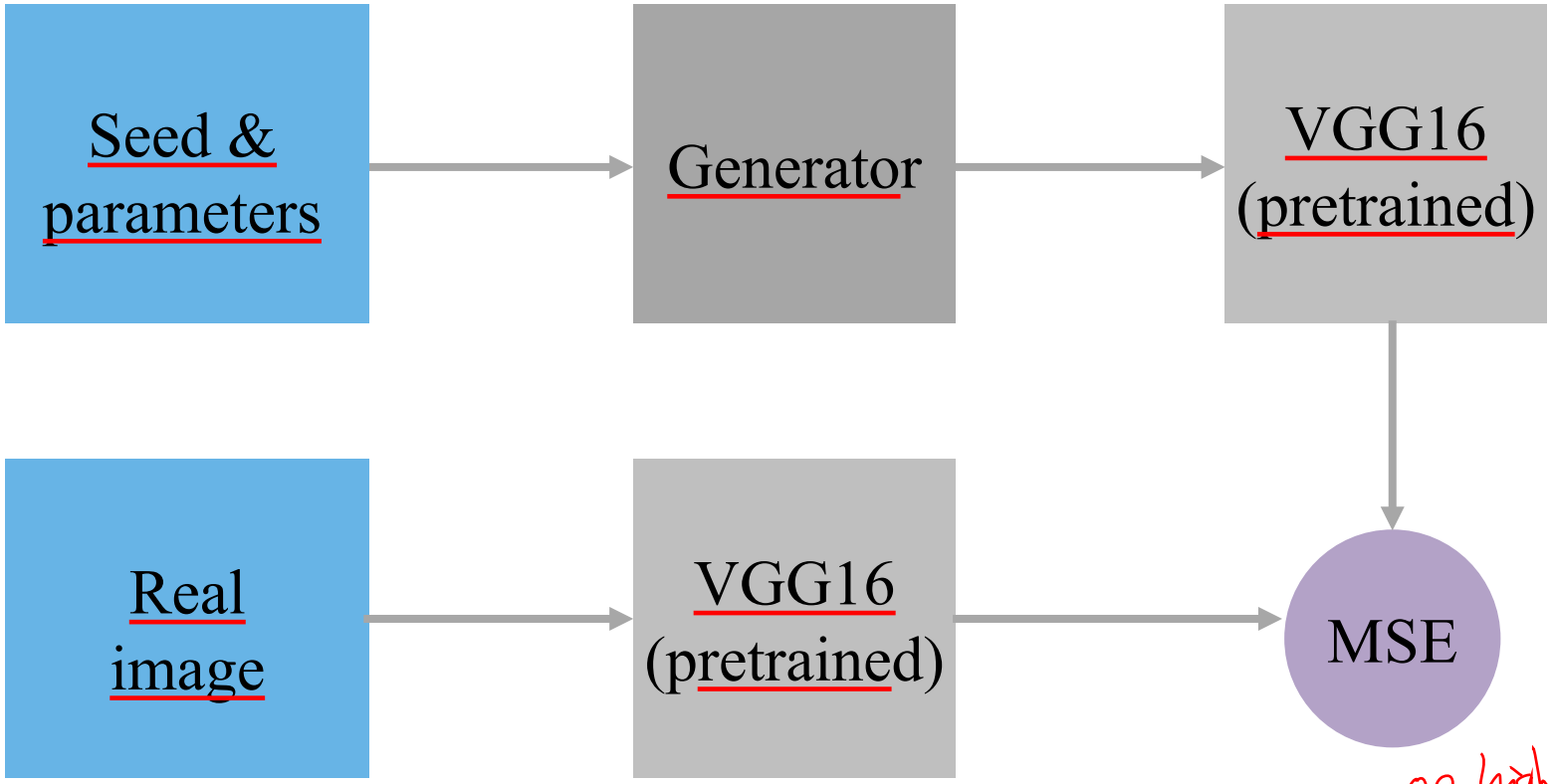
e.g activation of pre-final layer of CNN

would be least sensitive to small shifts

Problem: MSE sucks at this task.

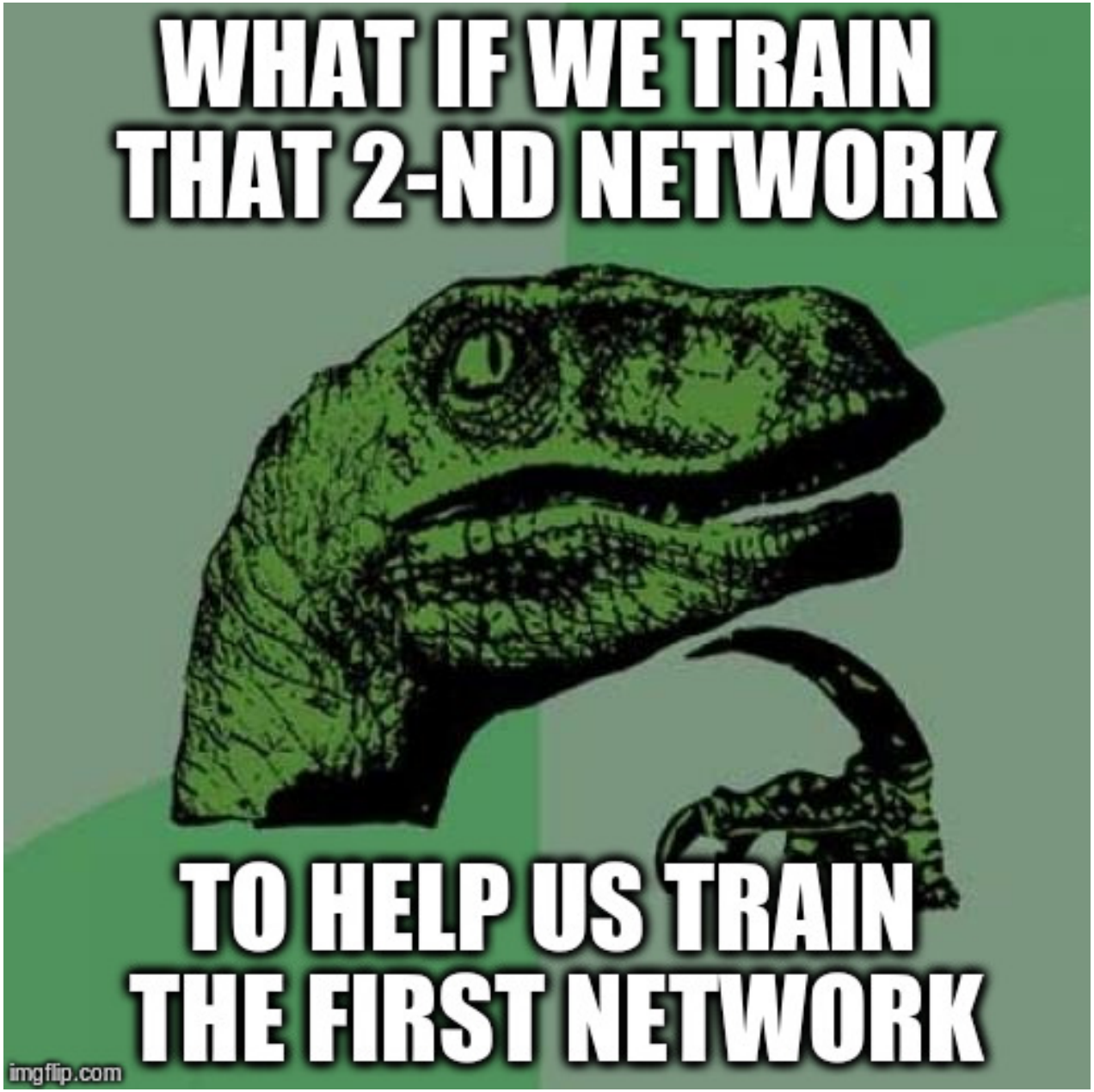
Ideas?

Sketch: using pre-trained nets



on high-level features

$$L = \|f(img) - f(Gen(seed))\|$$


WHAT IF WE TRAIN
THAT 2-ND NETWORK
TO HELP US TRAIN
THE FIRST NETWORK
imgflip.com

Generative Adversatial Networks

Generator

content

feedback

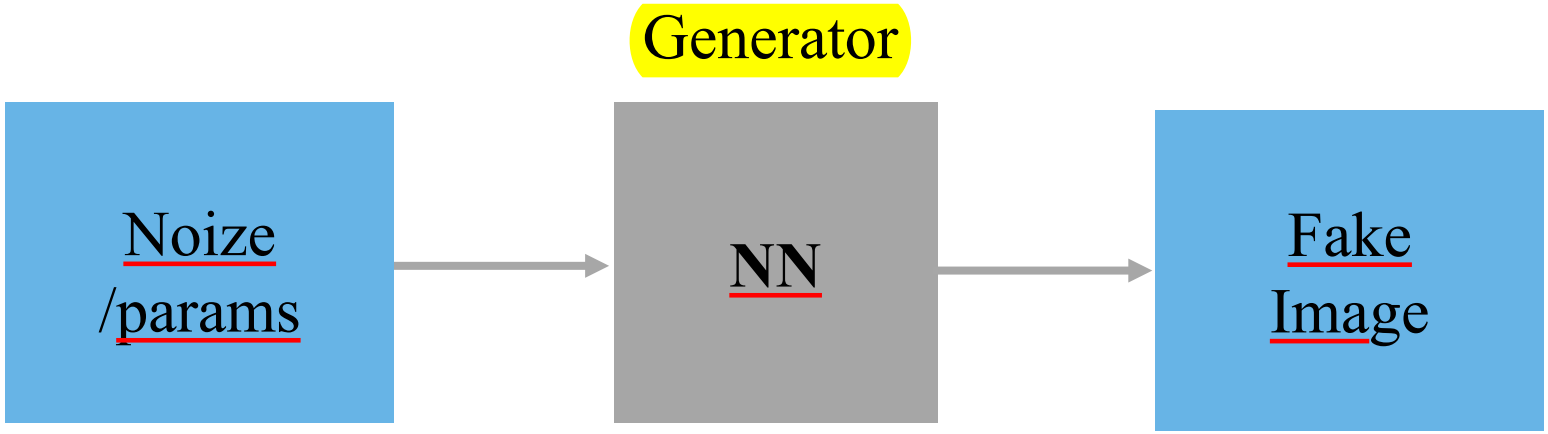
Discriminator

Generate image
(should be plausible)

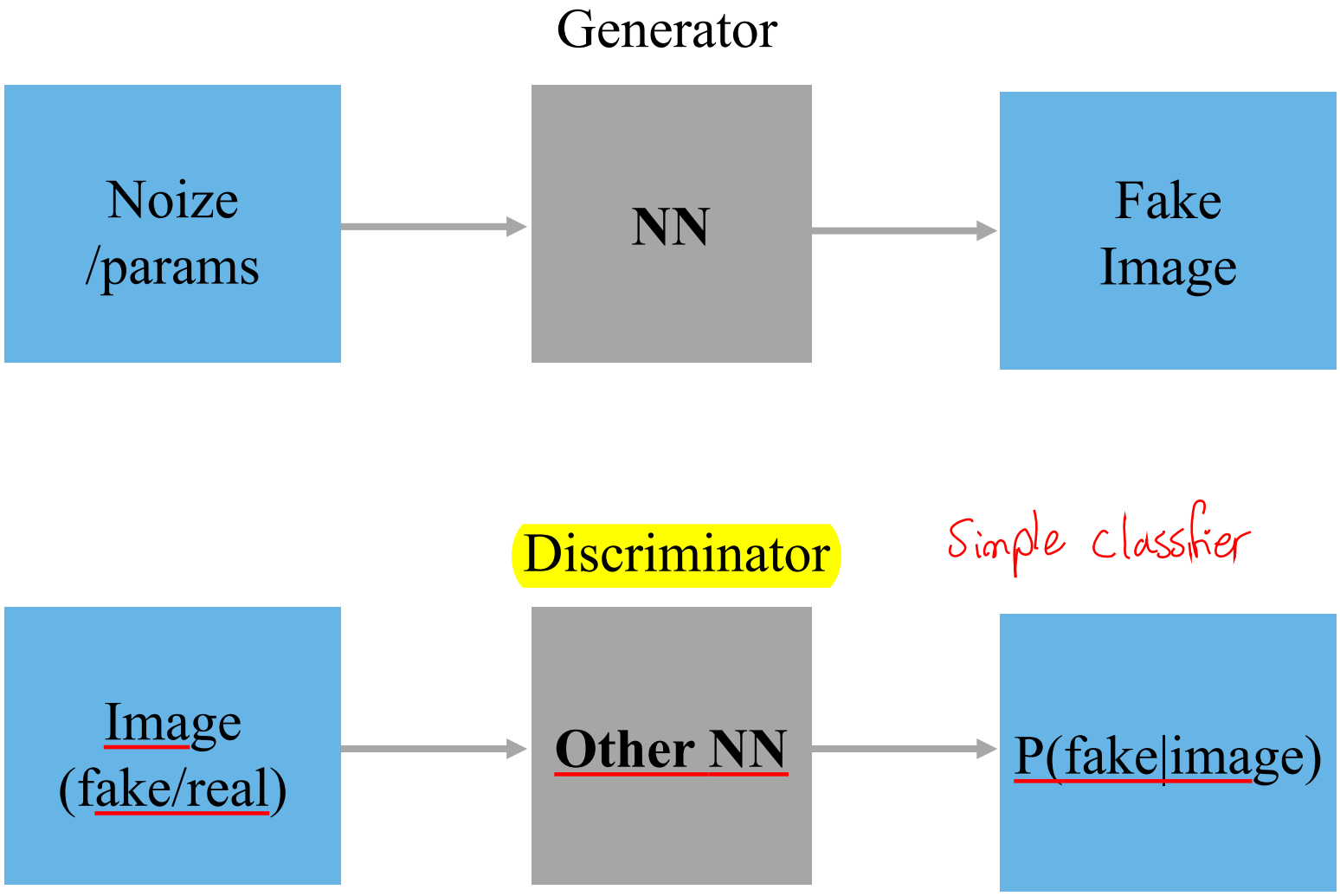
Tell if image is plausible
(image) → P(fake)

aims to fool the discriminator

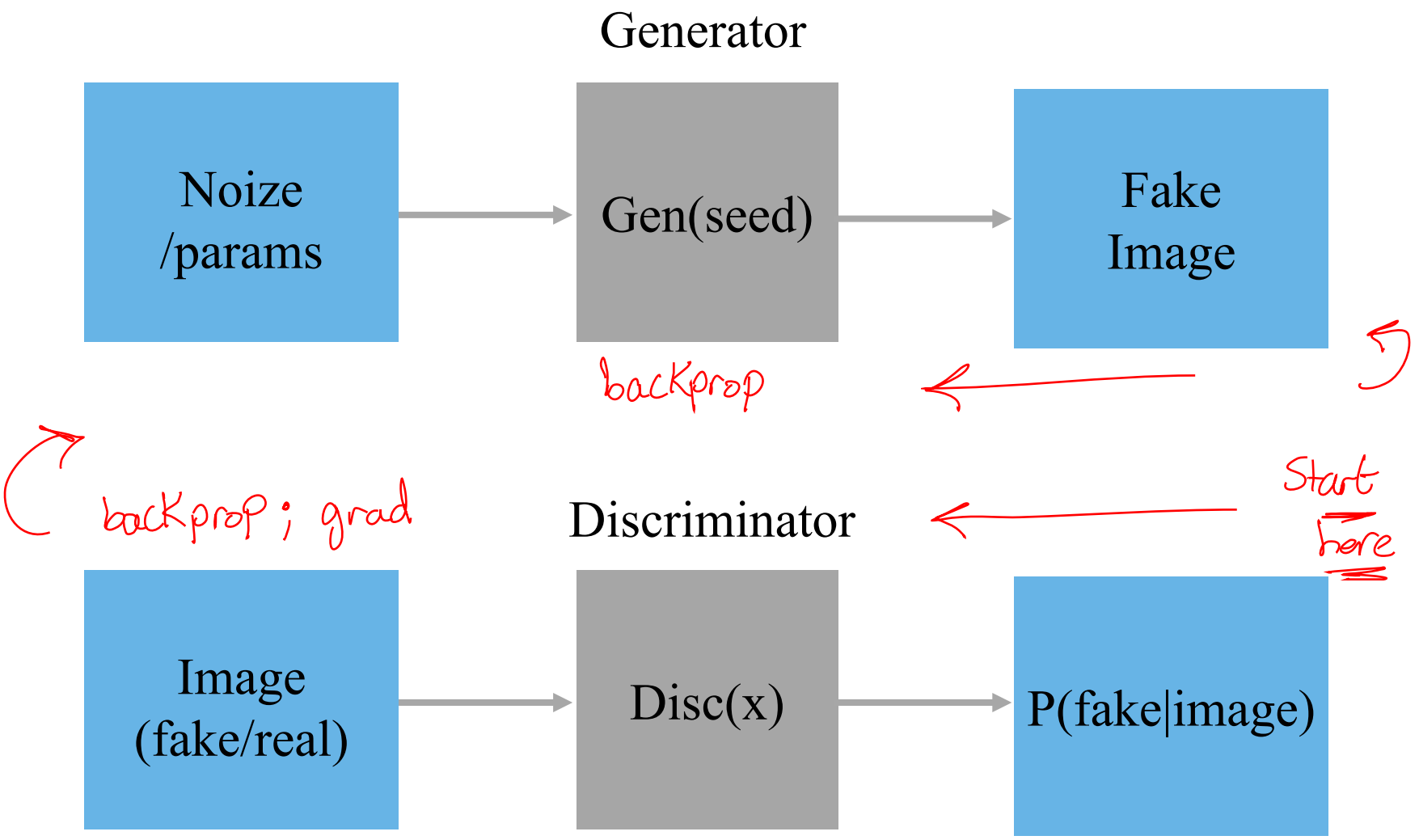
Generative Adversarial Networks



Generative Adversarial Networks

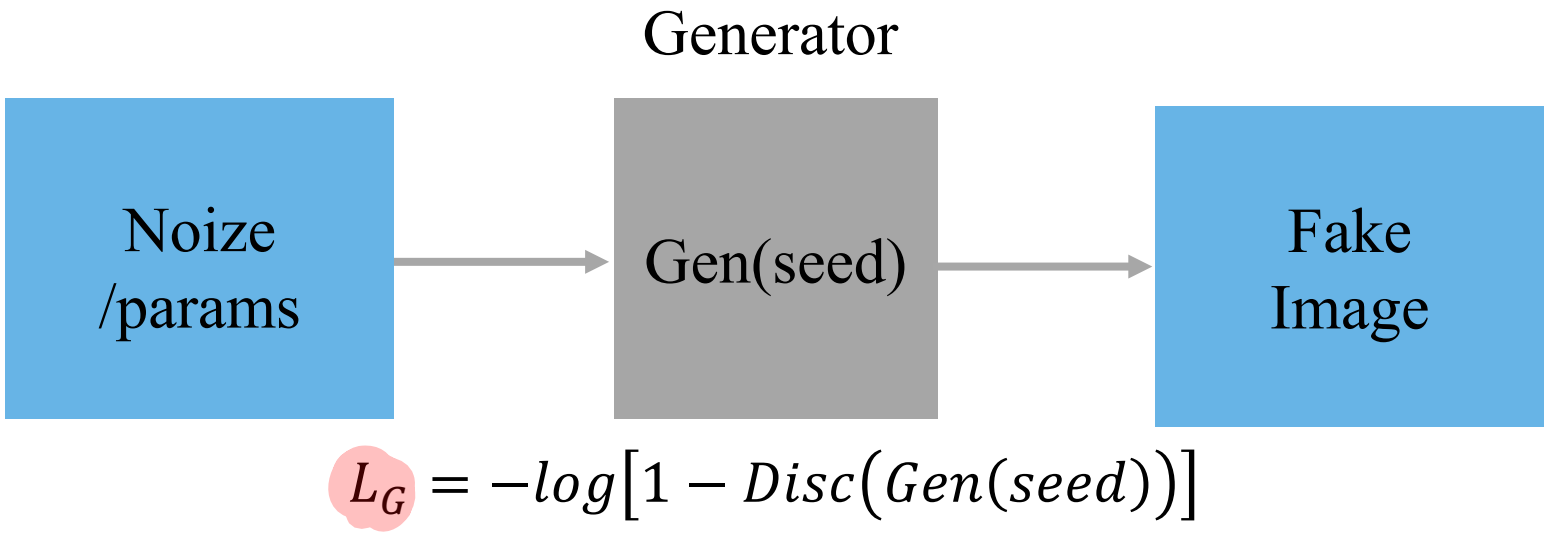


Generator aims to fool the discriminator



$$L_D = -log[1 - Disc(realdata)] - logDisc\big(Gen(seed)\big)$$

Generative Adversarial Networks



Generative Adversarial Networks

training both simultaneously

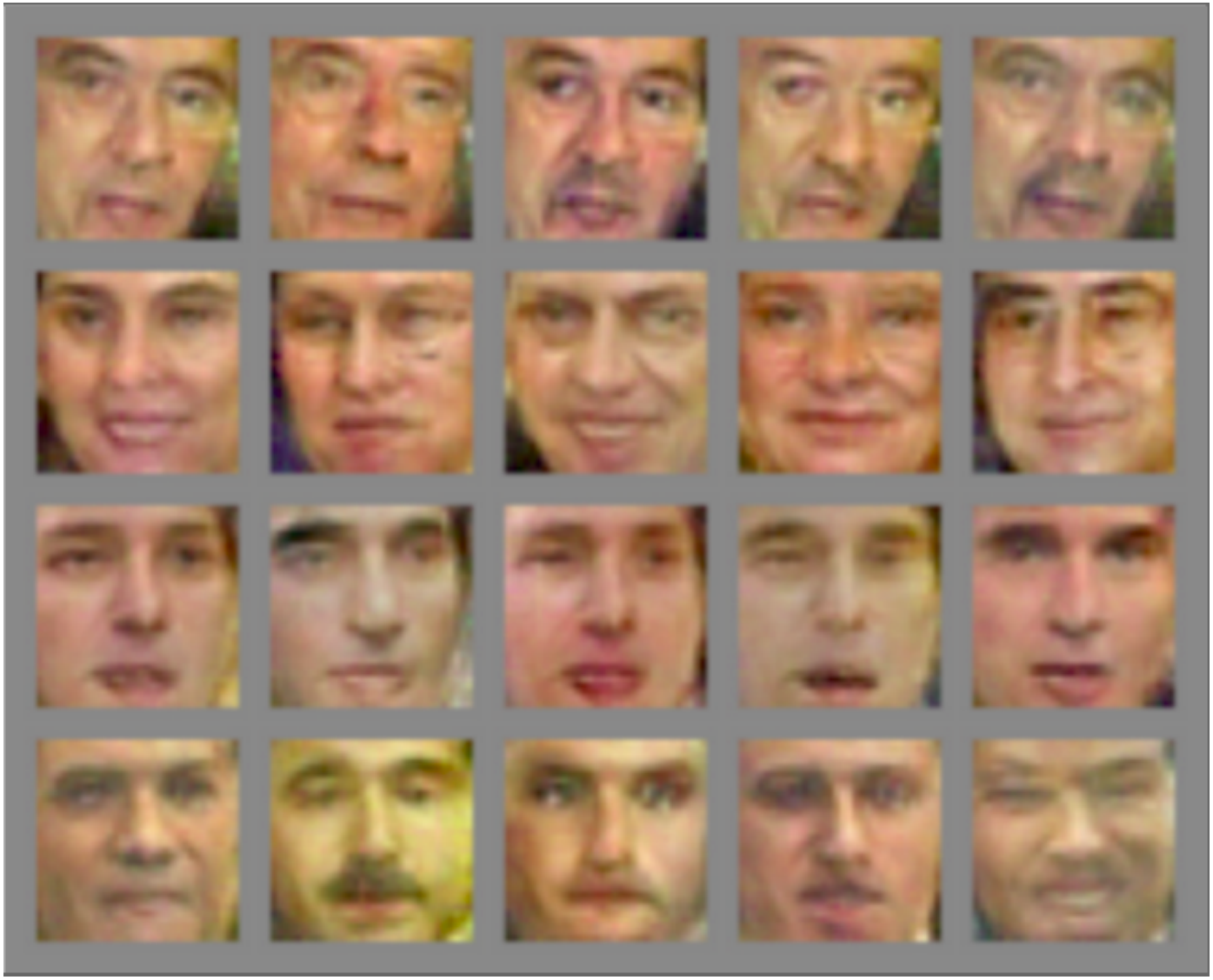
Algorithm

- sample noise **z** and images **x**
- for k in 1...K
  - Train discriminator(**x**), discriminator(generator(**z**))
- For m in 1...M
  - Train generator(z)

* training GANs is usually unstable
* Ideally, gen shall win after many iter

In practice however this is not always the case

Generative Adversarial Networks



Adversarial domain adaptation

- Two domains
  - e.g. mnist digits Vs actual digits on photos
- First domain is labeled, second isn't
- Wanna learn for the second domain

Adversarial domain adaptation

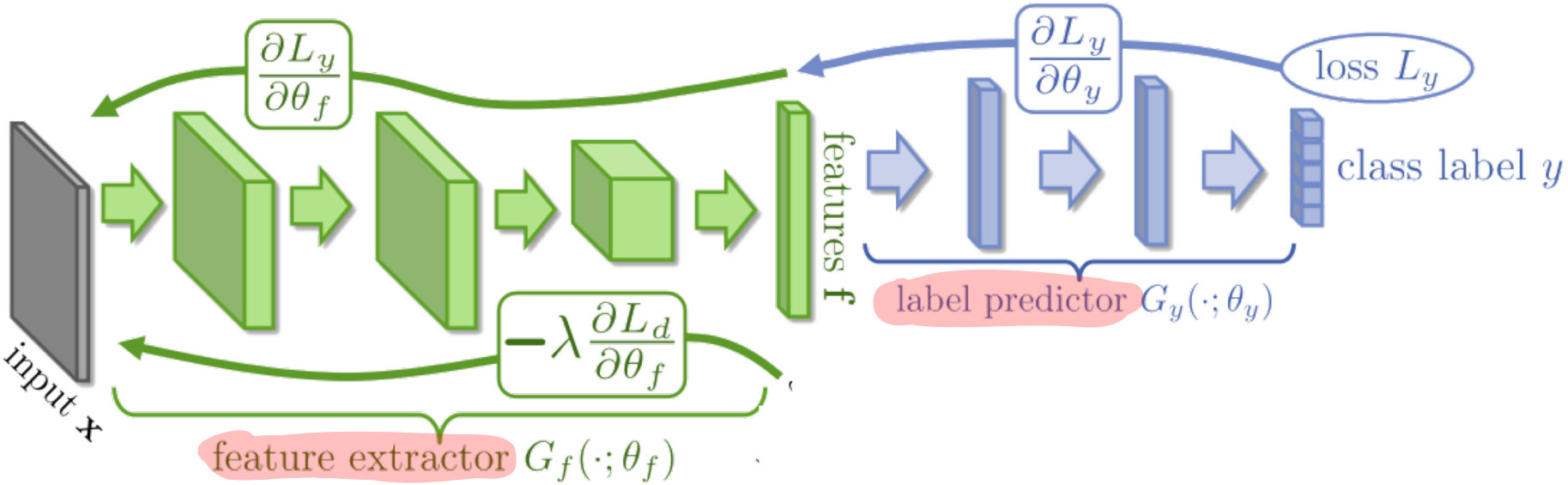
- Two domains
  - handwritten digits
  - house number digits
- First domain is labeled, second isn't
- You want your model to work on the second domain

Labeled data

Unlabeled data

Domain adaptation

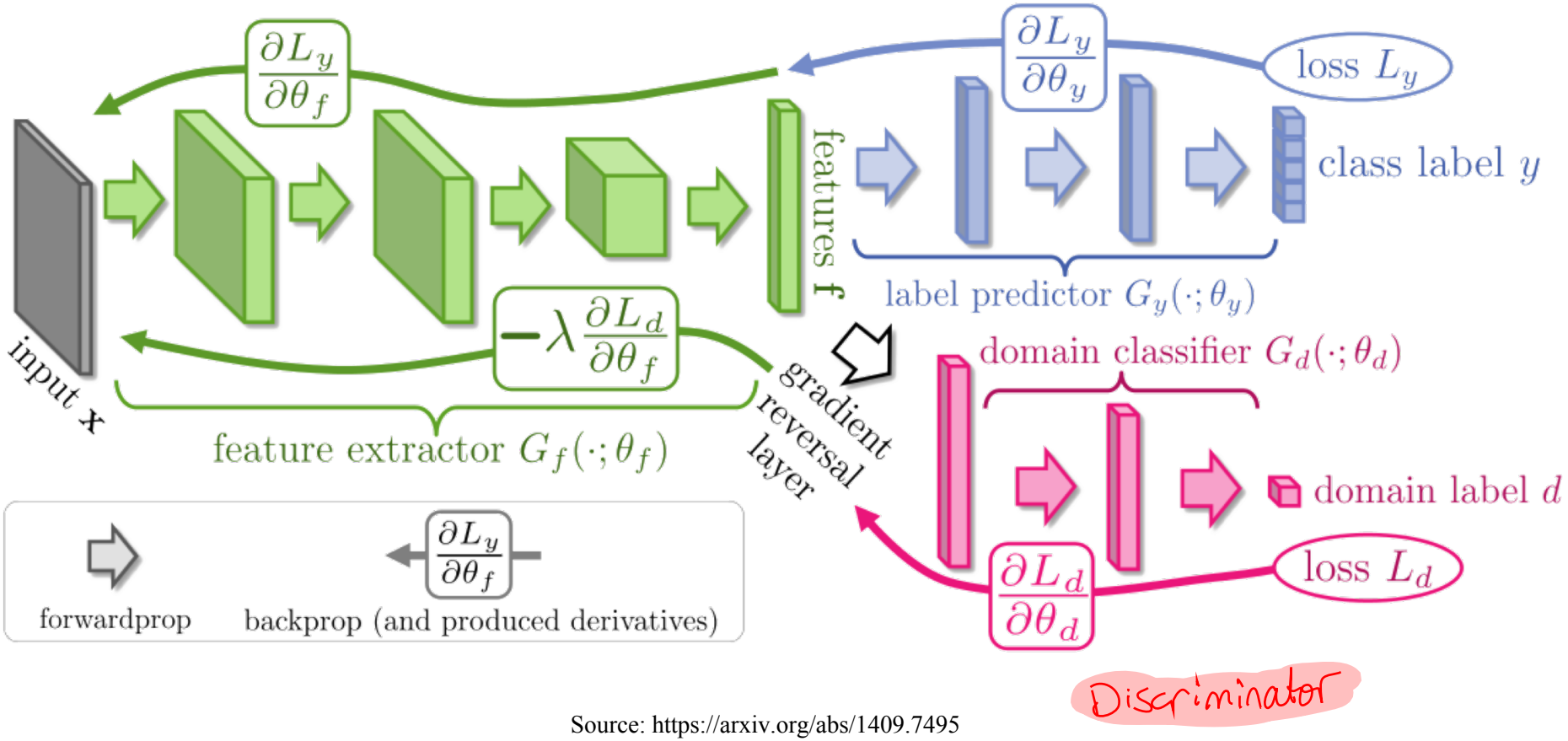
Idea: discriminator should not be able to distinguish features on two domains



Source: https://arxiv.org/abs/1409.7495

Domain adaptation

Idea: discriminator should not be able to distinguish features on two domains



Source: https://arxiv.org/abs/1409.7495

Domain adaptation

Idea: discriminator should not be able to distinguish features on two domains

$$-logP\big(real|h(x_{real})\big) - log\big[1 - P\big(real|h(x_{mc})\big)\big] \rightarrow \min_{discriminator}$$

$$L_{classifier}\left(y_{mc}, y\big(h(x_{mc})\big)\right) - logP\big(real|h(x_{mc})\big) \rightarrow \min_{classifier}$$

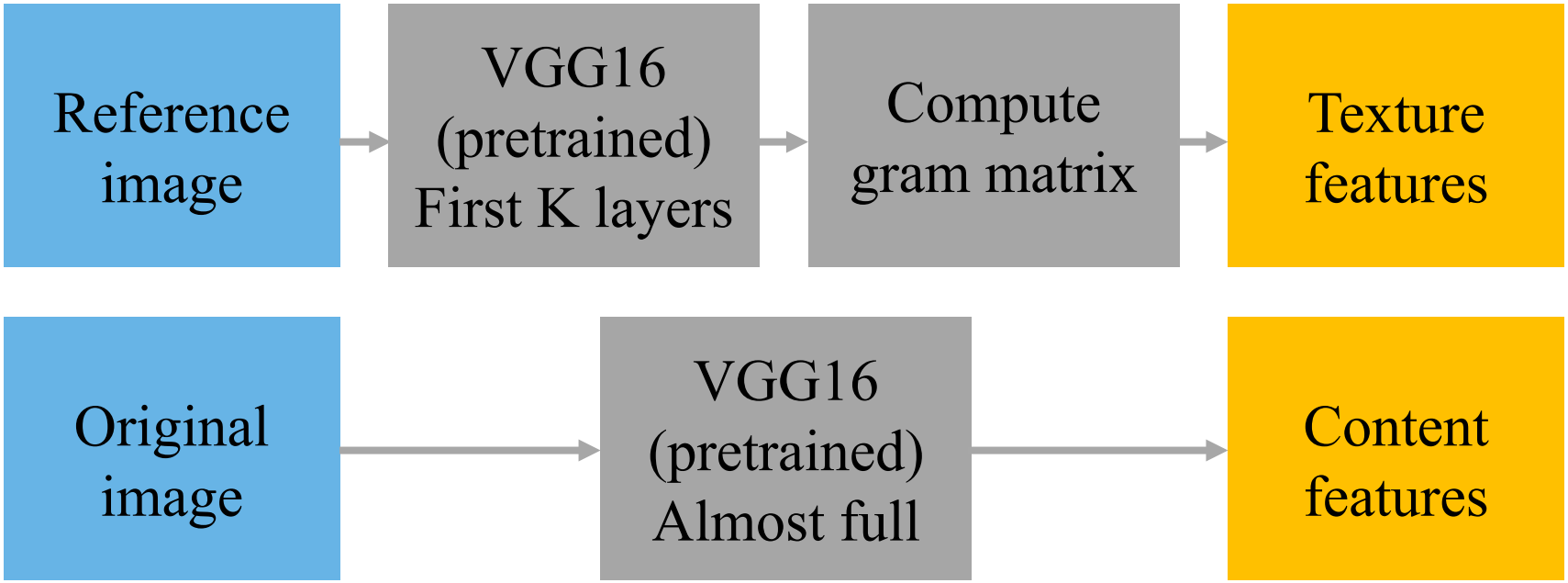
Art style transfer

Ideas?

Art style transfer

Formulate and optimize texture loss

$$L = \|Texture(x_{ref}) - Texture(x_{cand})\| + \|Content(x_{orig}) - Content(x_{cand})\|$$



Art style transfer

+

=