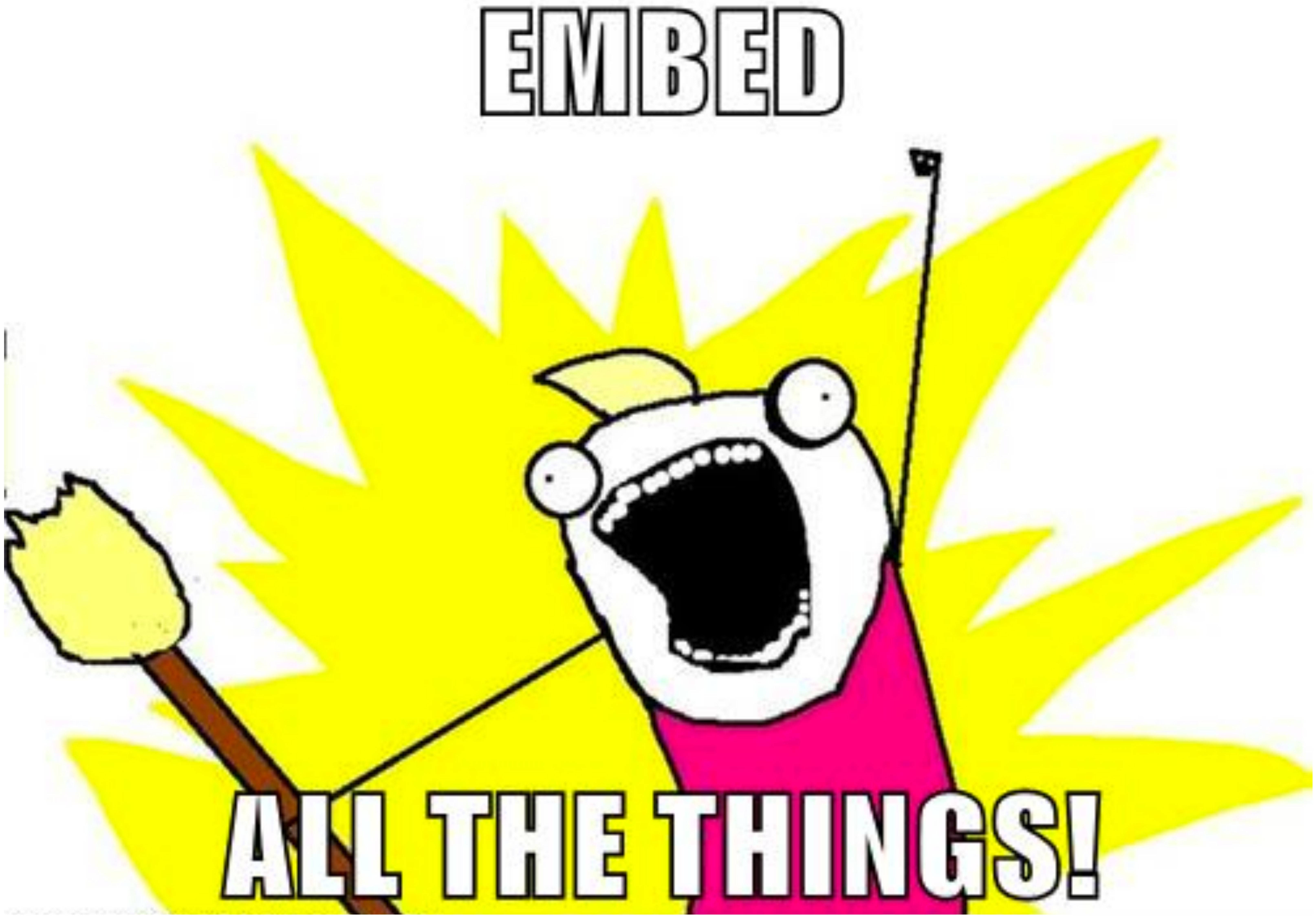
Embedding things

Dániel Varga (Rényi Institute)


EMBED
ALL THE THINGS!

Managing expectations

- There are no theorems and proofs. This is not a math talk. The four basic arithmetic operations are all that we will use.

- There is no “open problems” section. Just trying to start a conversation here between two fields that obviously share a lot of concepts. (Low rank matrix approximations, message passing on graphs, spectral graph methods, just to name a few.)

This talk is not an introduction to deep learning

- **For our purposes today**, an artificial neural network is a big black box that can learn continuous functions between two given vector spaces.

- This is how the learning happens: (not really)

- We give it a huge formula called the total loss function, and the mapping is repeatedly adjusted until a local minimum of the total loss function is found.

- The total loss function incorporates specific input vectors and their expected output.

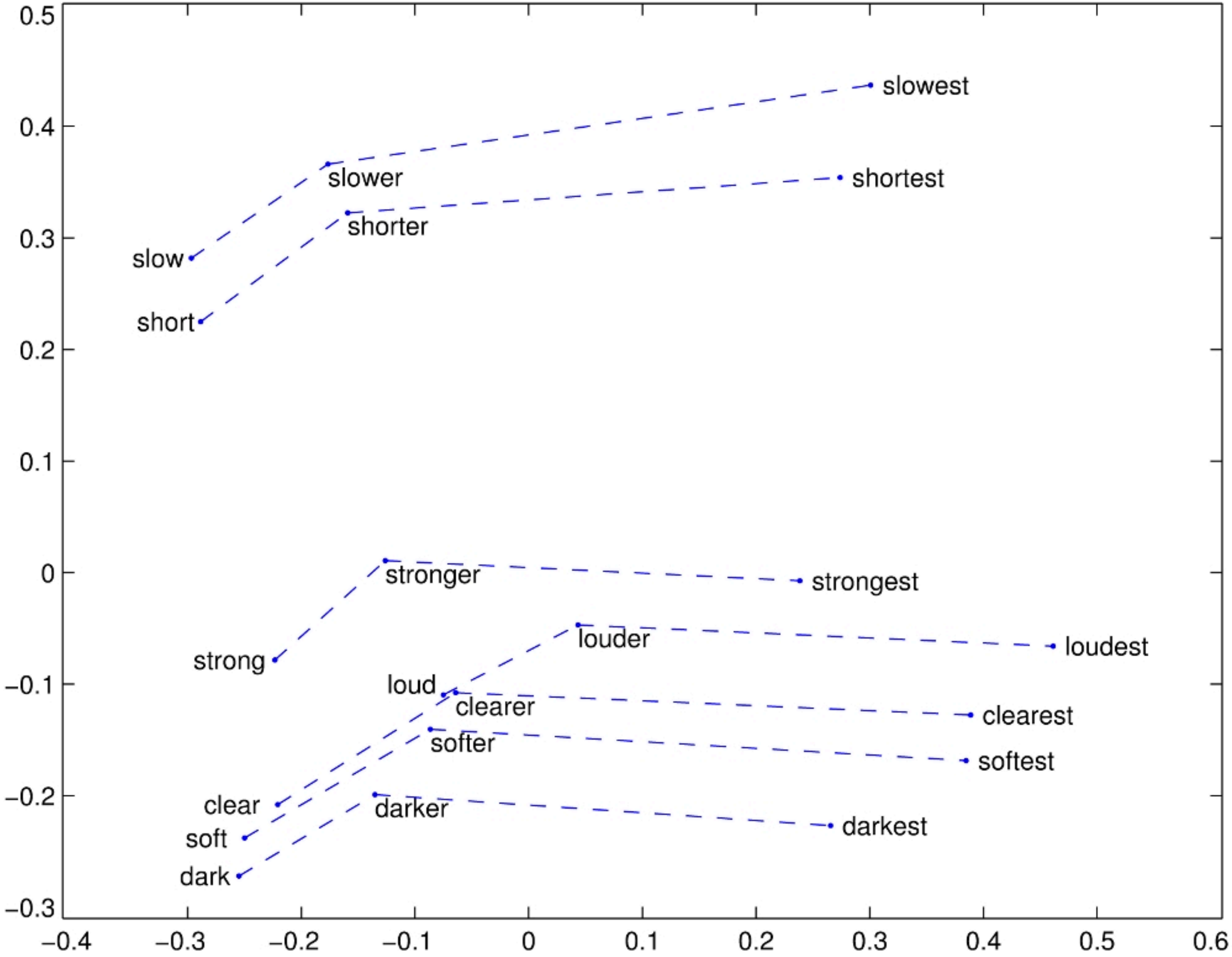
Overview

- Representation learning
  - Assigning vectors to objects so that useful complex relationships between the objects become simple (hopefully linear) relationships between the corresponding vectors.
  - I'll call these mappings embeddings or latent representations, and the vector space the latent space.
- Neural message passing
  - A simple but extremely successful class of methods where the objects to be embedded are nodes of a graph. My belief is that more theory-centered subfields of graph theory have a lot to add to this research.

Representation learning

- Assigning vectors to objects so that useful complex relationships between the objects become simple (hopefully linear) relationships between the corresponding vectors.

That’s a bit too abstract, let’s see some examples.

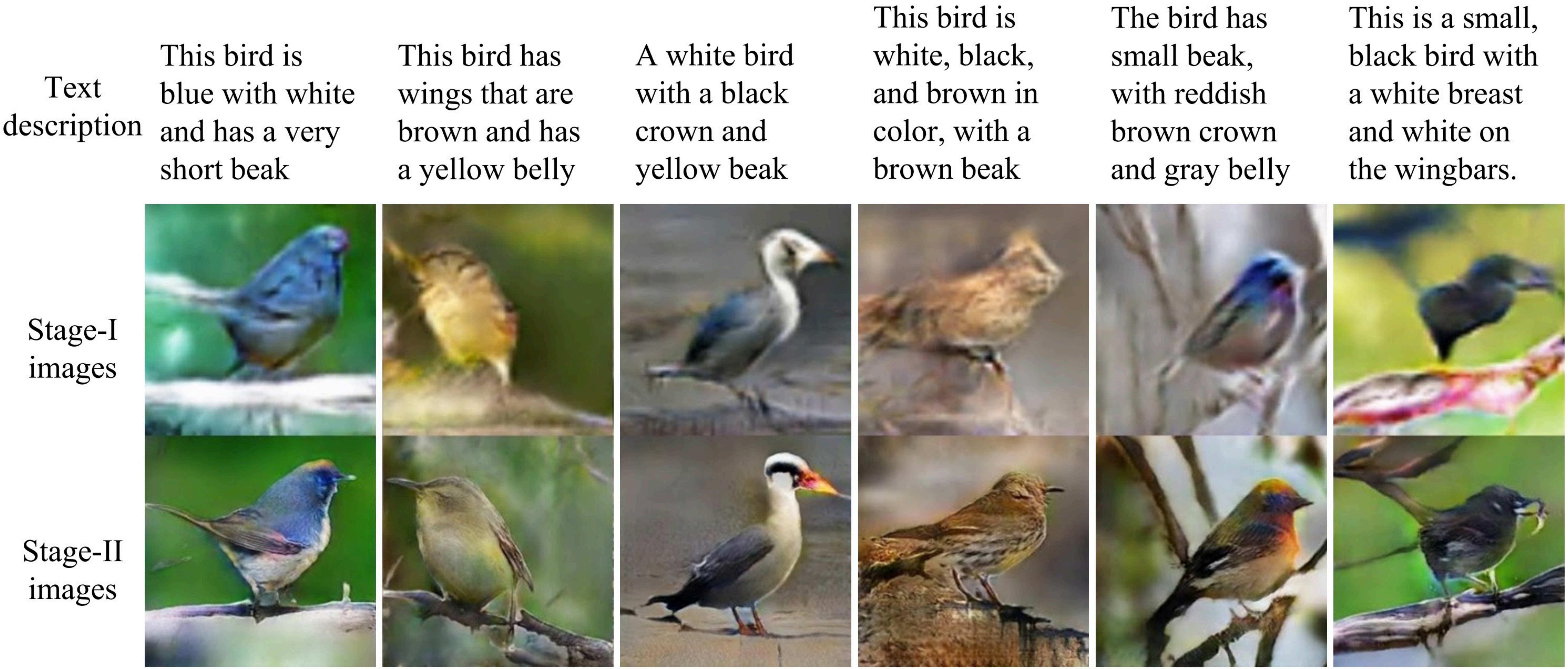

slowest
slower
shortest
shorter
slow
short
stronger
strongest
louder
loudest
strong
loud
clearer
clearest
softer
softest
clear
darker
darkest
soft
dark
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.3
−0.2
−0.1
0
0.1
0.2
0.3
0.4
0.5
0.6

Images of faces

move_and_show(daniel_varga + 0.6 * gender_direction, age_direction, np.linspace(-5, +2, 10))

StyleGAN

We can even embed sentences and images in the same vector space



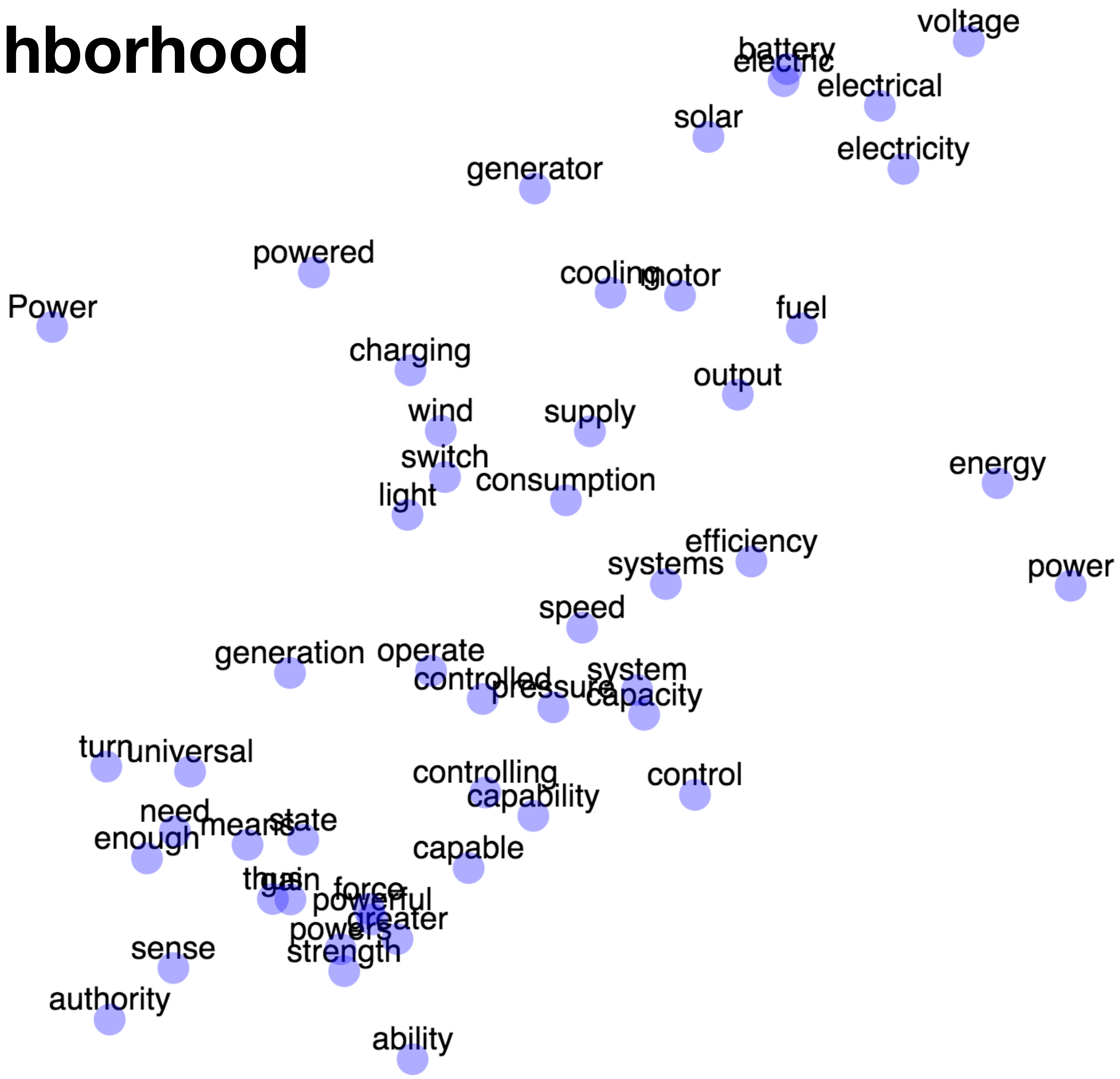
StackGAN

Word embeddings

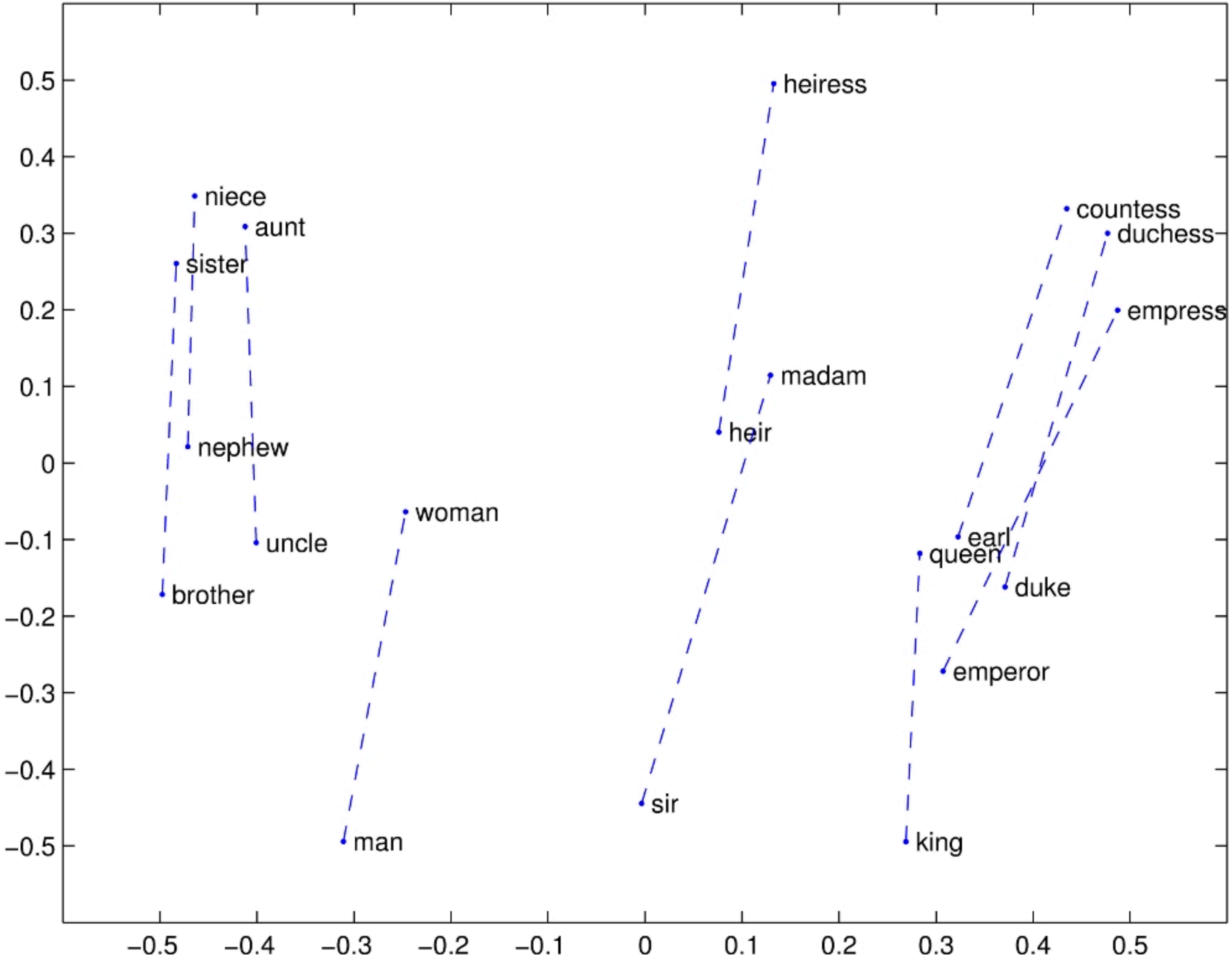
“You shall know a word by the company it keeps.”

John Rupert Firth

power's neighborhood

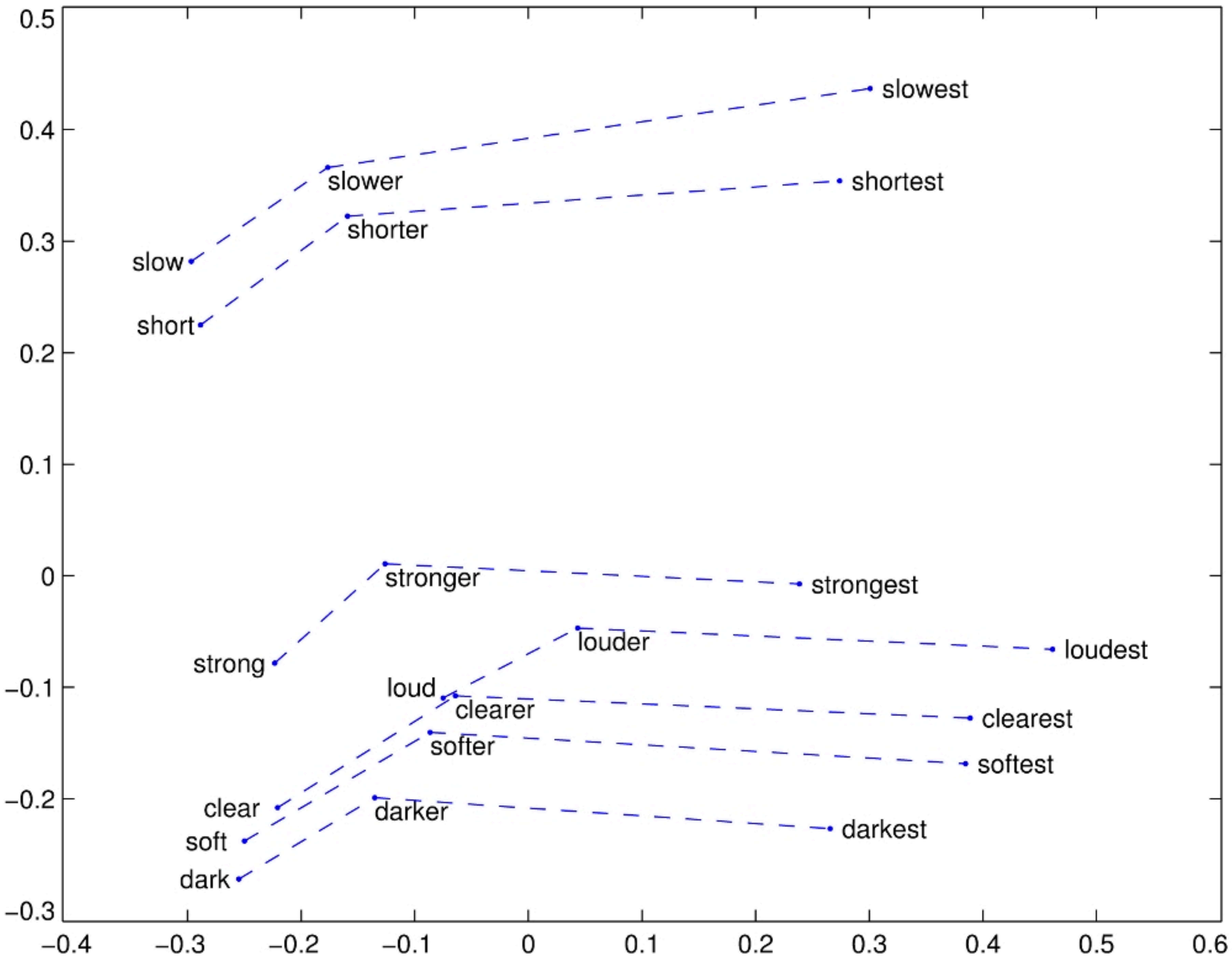


Words



heiress
niece
aunt
sister
countess
duchess
empress
madam
heir
nephew
woman
uncle
brother
man
sir
earl
queen
duke
emperor
king
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.5
−0.4
−0.3
−0.2
−0.1
0
0.1
0.2
0.3
0.4
0.5

GloVe



0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.3
−0.2
−0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
slowest
slower
shortest
shorter
slow
short
stronger
strongest
strong
louder
loudest
loud
clearer
clearest
softer
softest
clear
soft
dark
darker
darkest

Vector arithmetic plus nearest neighbor lookup

Rome - Italy + China = Beijing
China - Taiwan + Ukraine = Russia
house - roof + castle = dome
knee - leg + elbow = forearm
love - indifference + fear = apathy

https://deeplearning4j.org/word2vec.html

Ratio of co-occurrences convey meaning

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Here I sketch the log-bilinear model at the blackboard

6 years is a very long time in deep learning land

- Log-bilinear was enough to kickstart the deep learning revolution in natural language processing, by giving neural networks the ability to take text as input and output.
- (And earning the two word2vec papers 20000 citations in the process.)
- Nowadays character-based deep context-sensitive word embedding models dominate.

GPT-2, a state-of-the-art language model

SYSTEM PROMPT (HUMAN-WRITTEN)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES)

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

Cherry-picked

Kawasaki Z1000

I recently acquired this bike from its original owner. It has always been a good bike in every regard, but I found some problems along the way: It has some wear and tear on the engine bay, the front end has been modified to a wider track and the suspension was modified as well. The bike has been running very well, and the engine is still working perfectly (I have changed oil and filters and both of my oil changes since acquiring this bike).

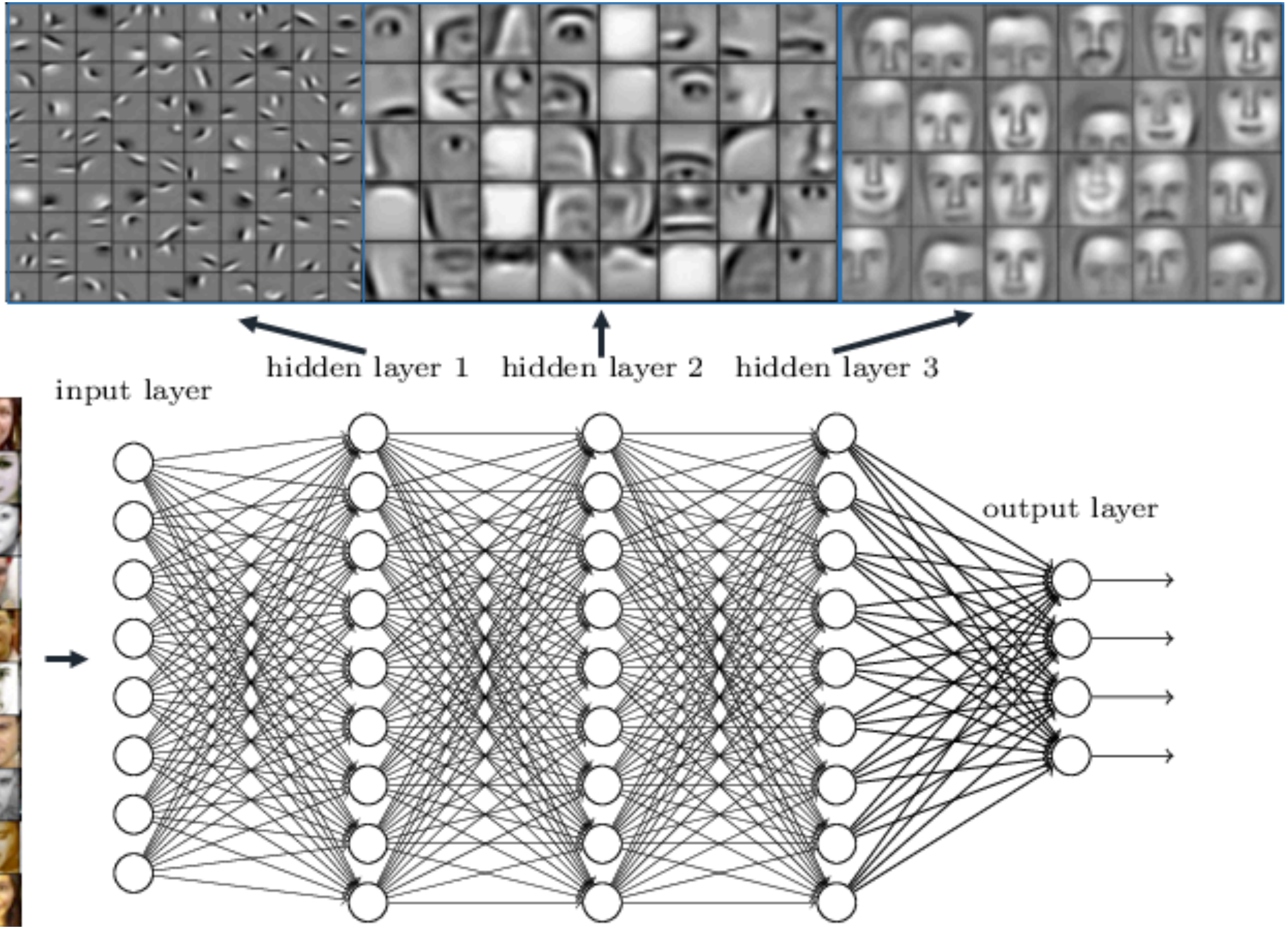
The bike is still in working order and it has all the original items including the oil filter. I am giving this kz1000 a thorough overhaul with a complete overhaul in the oil system and front end as well as some complete brakes and exhaust work.

As a side project, I also have a very old motorbike that is a great engine swap to take these bikes to the next level.

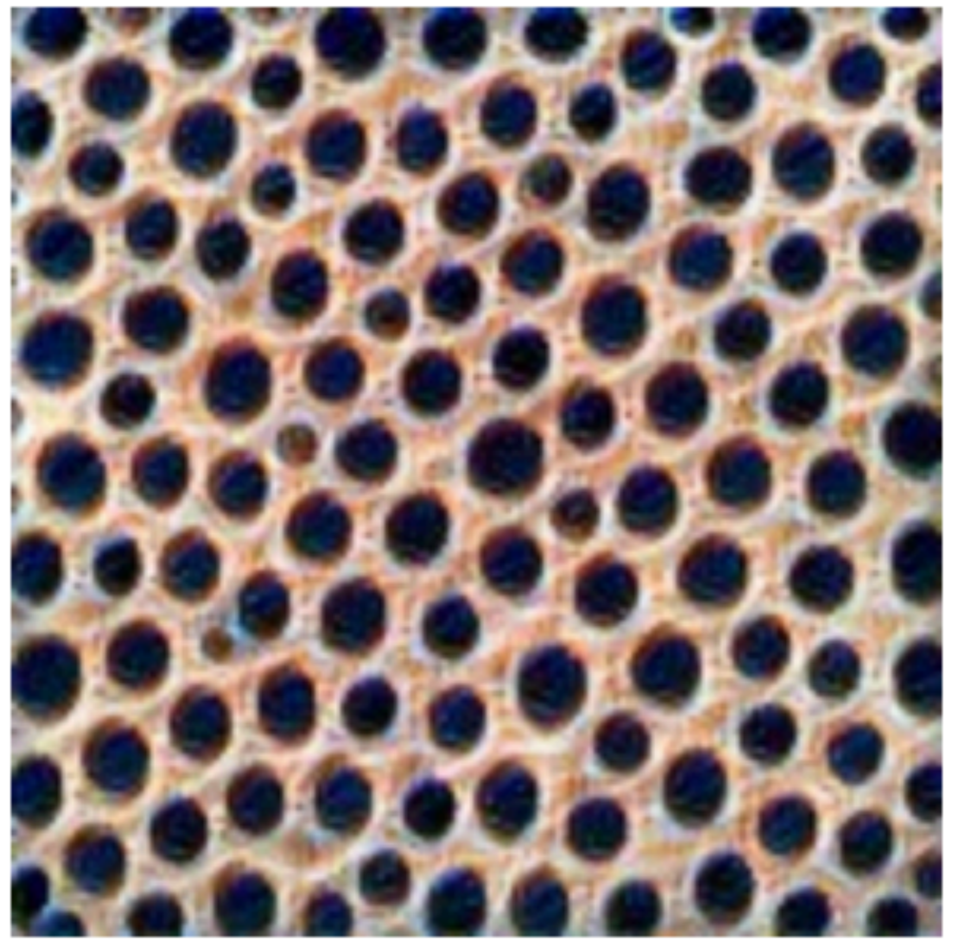
Non-cherry-picked

Supervised representation learning

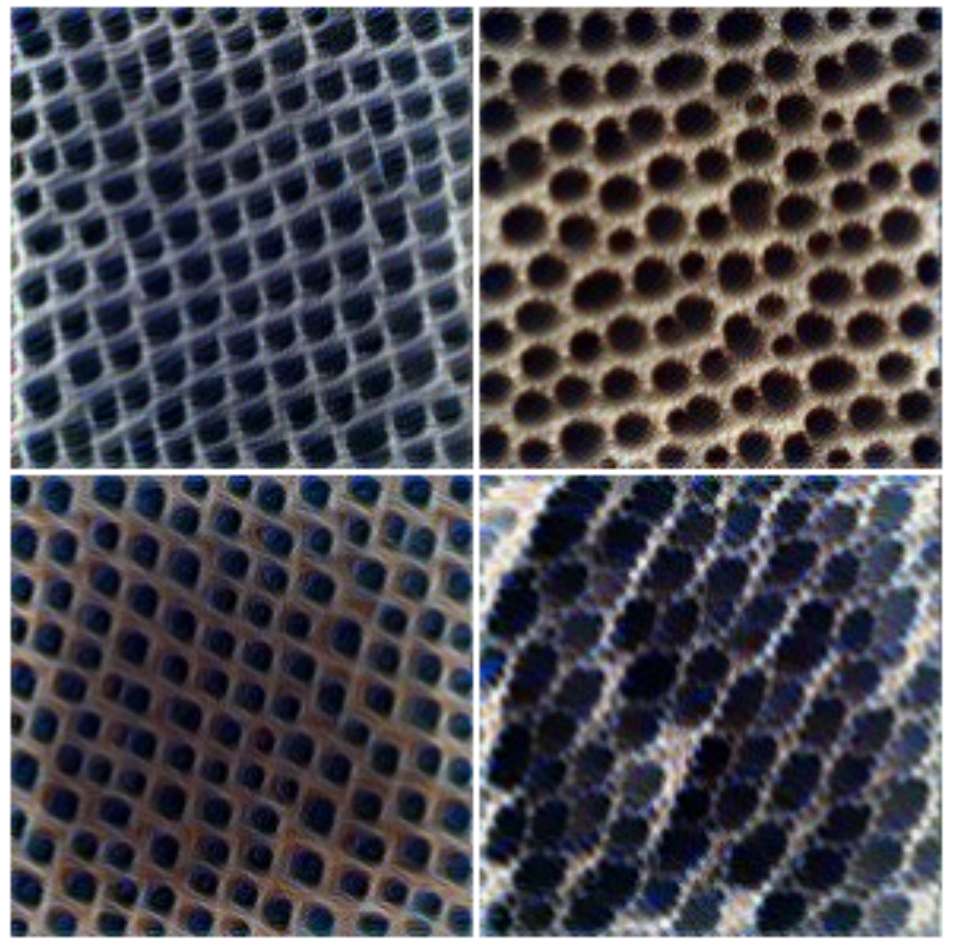
Deep neural networks learn hierarchical feature representations



POSITIVE CHANNEL



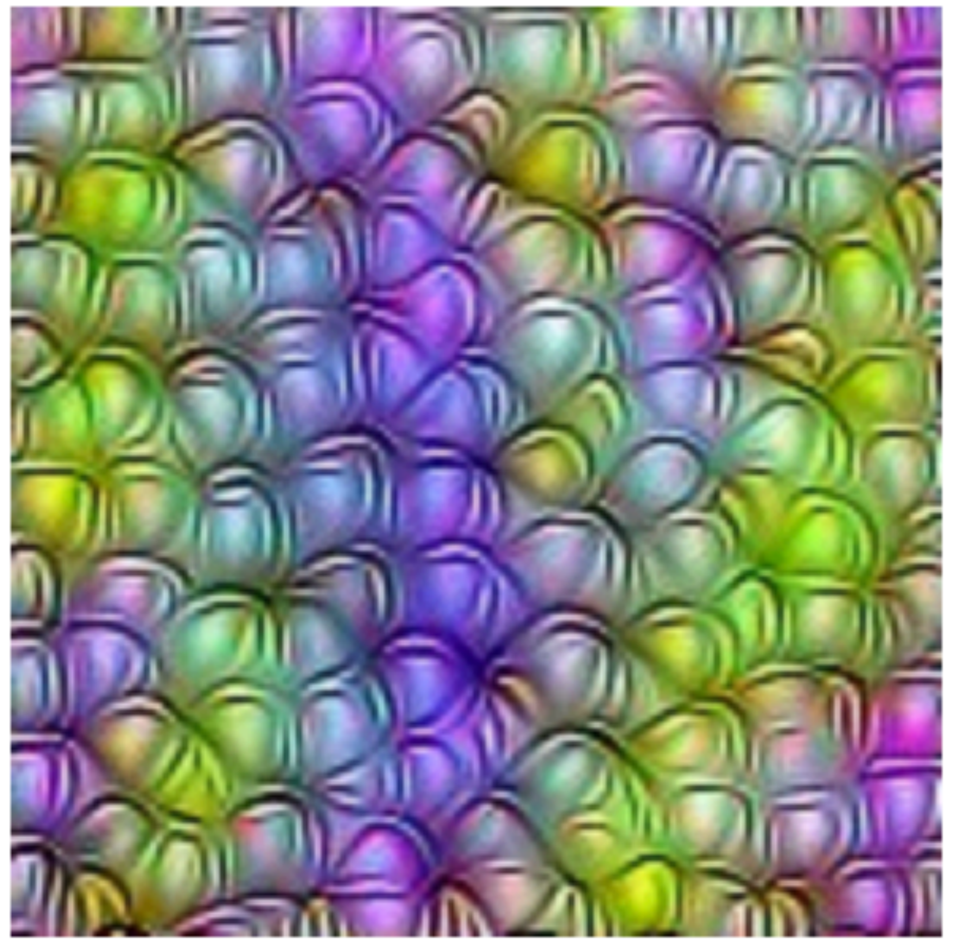
Channel Objective



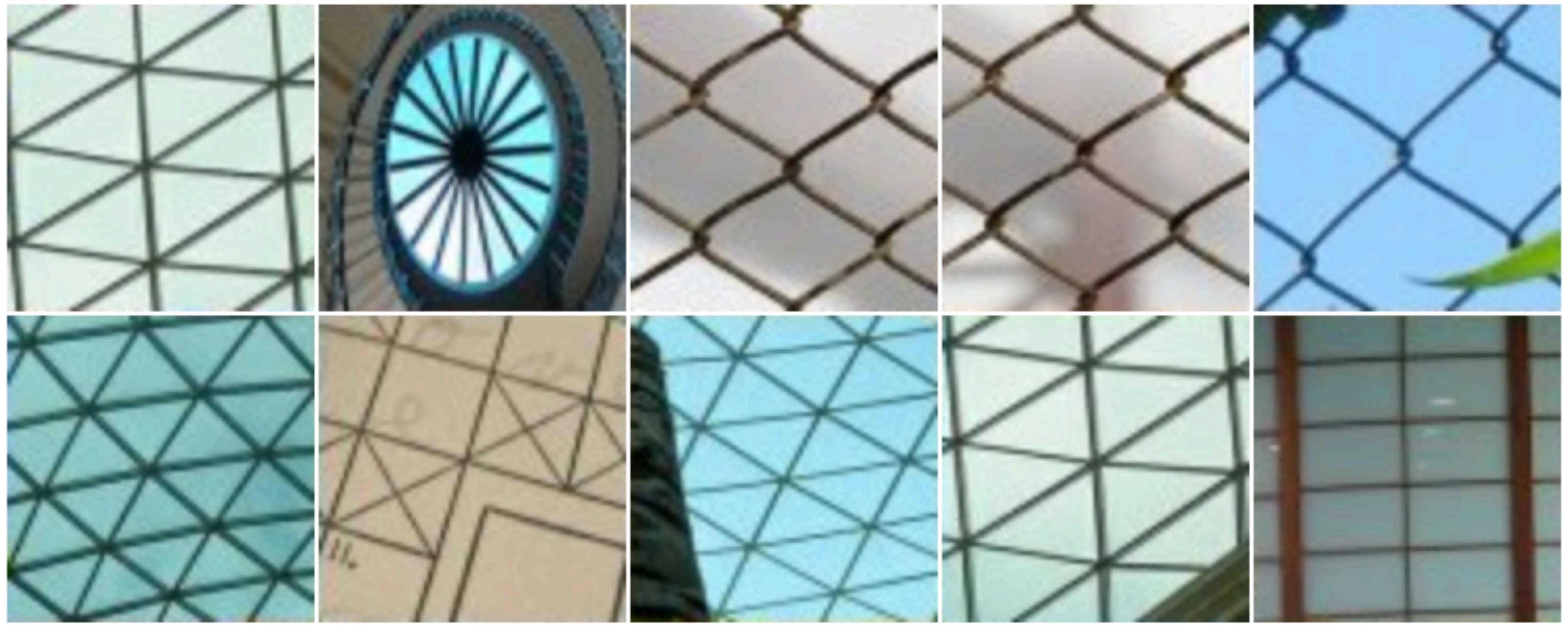
Diversity

Dataset examples

NEGATIVE CHANNEL



Negative Channel



Negative dataset examples

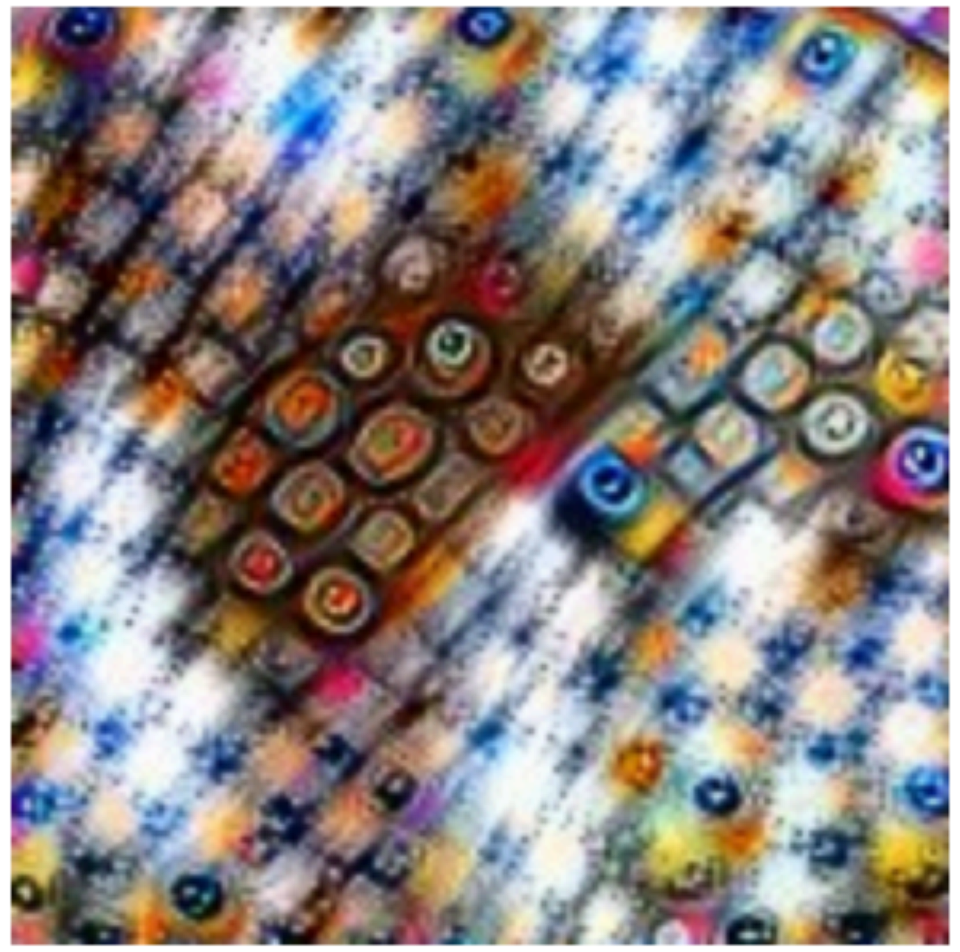
POSITIVE CHANNEL

Channel Objective

Diversity

Dataset examples

NEGATIVE CHANNEL



Negative Channel

Negative dataset examples

POSITIVE CHANNEL

Channel Objective

Diversity

Dataset examples

NEGATIVE CHANNEL

Negative Channel

Negative dataset examples

POSITIVE CHANNEL

Channel Objective

Diversity

Dataset examples

NEGATIVE CHANNEL

Negative Channel

Negative dataset examples

POSITIVE CHANNEL

Channel Objective

Diversity

Dataset examples

NEGATIVE CHANNEL

Negative Channel

Negative dataset examples

POSITIVE CHANNEL

Channel Objective

Diversity

Dataset examples

NEGATIVE CHANNEL

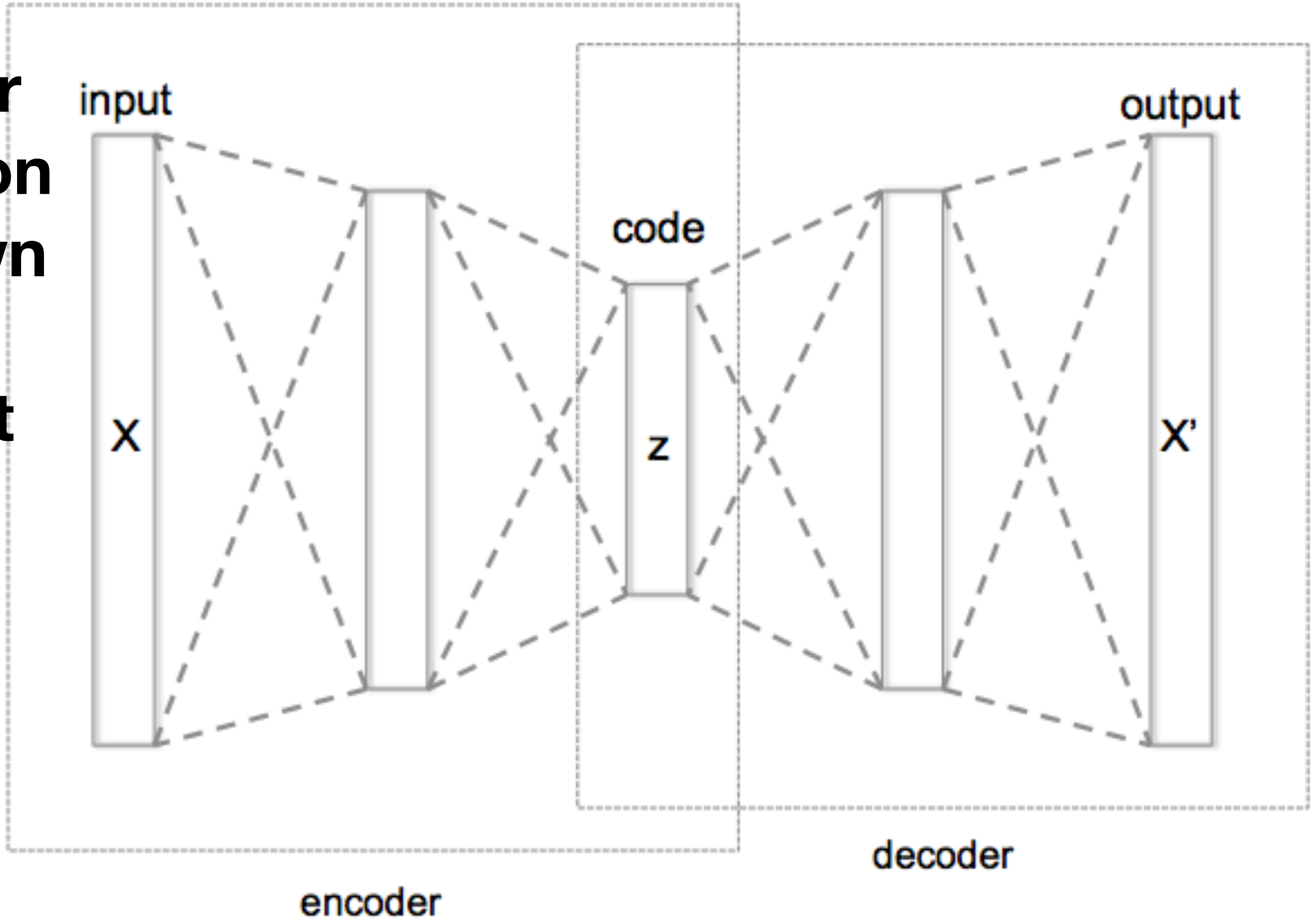
Negative Channel

Negative dataset examples

Unsupervised representation learning

Autoencoder

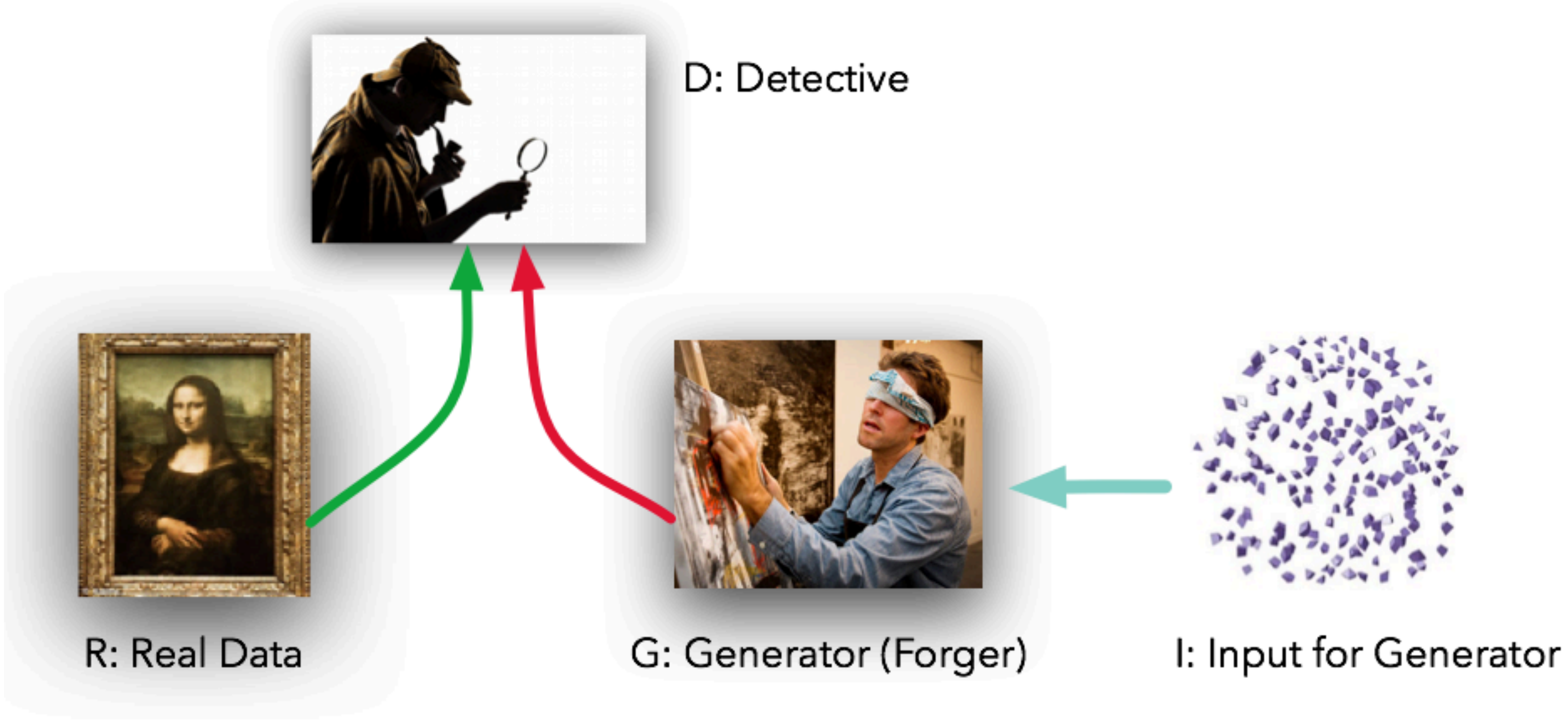
A non-linear generalization of well known Principal Component Analysis



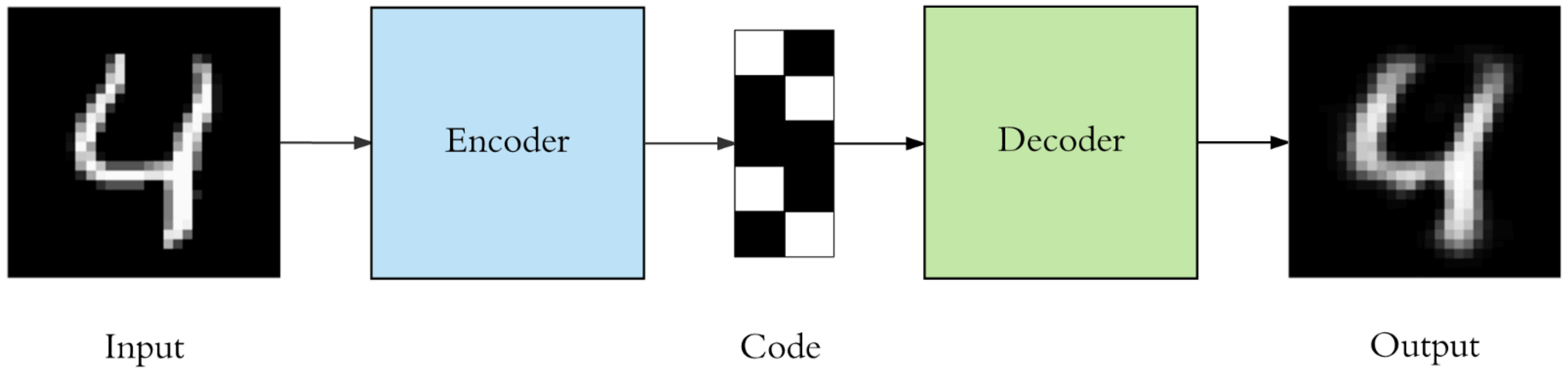
Generative neural networks

Generative neural networks

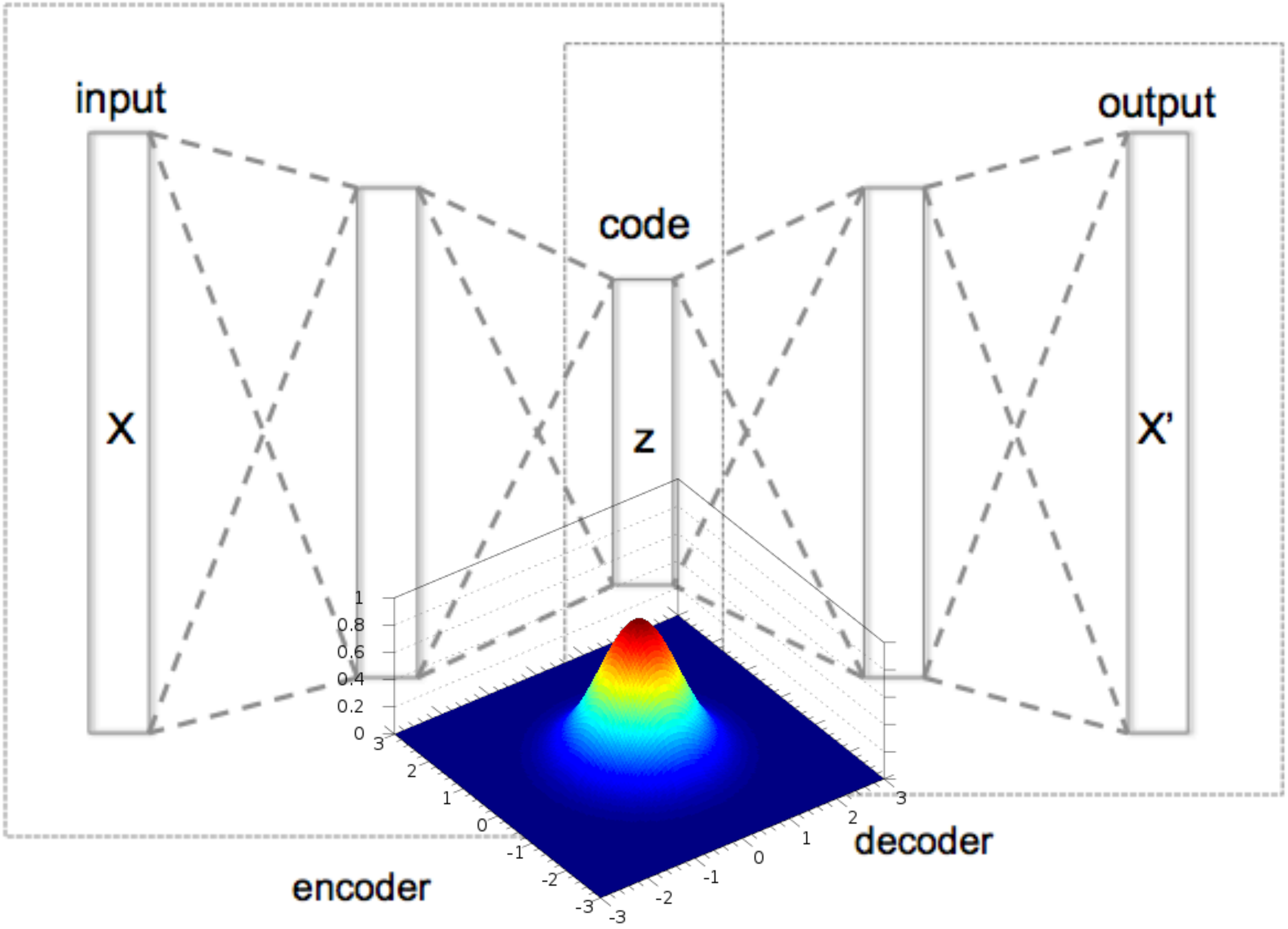
Generative Adversarial Network



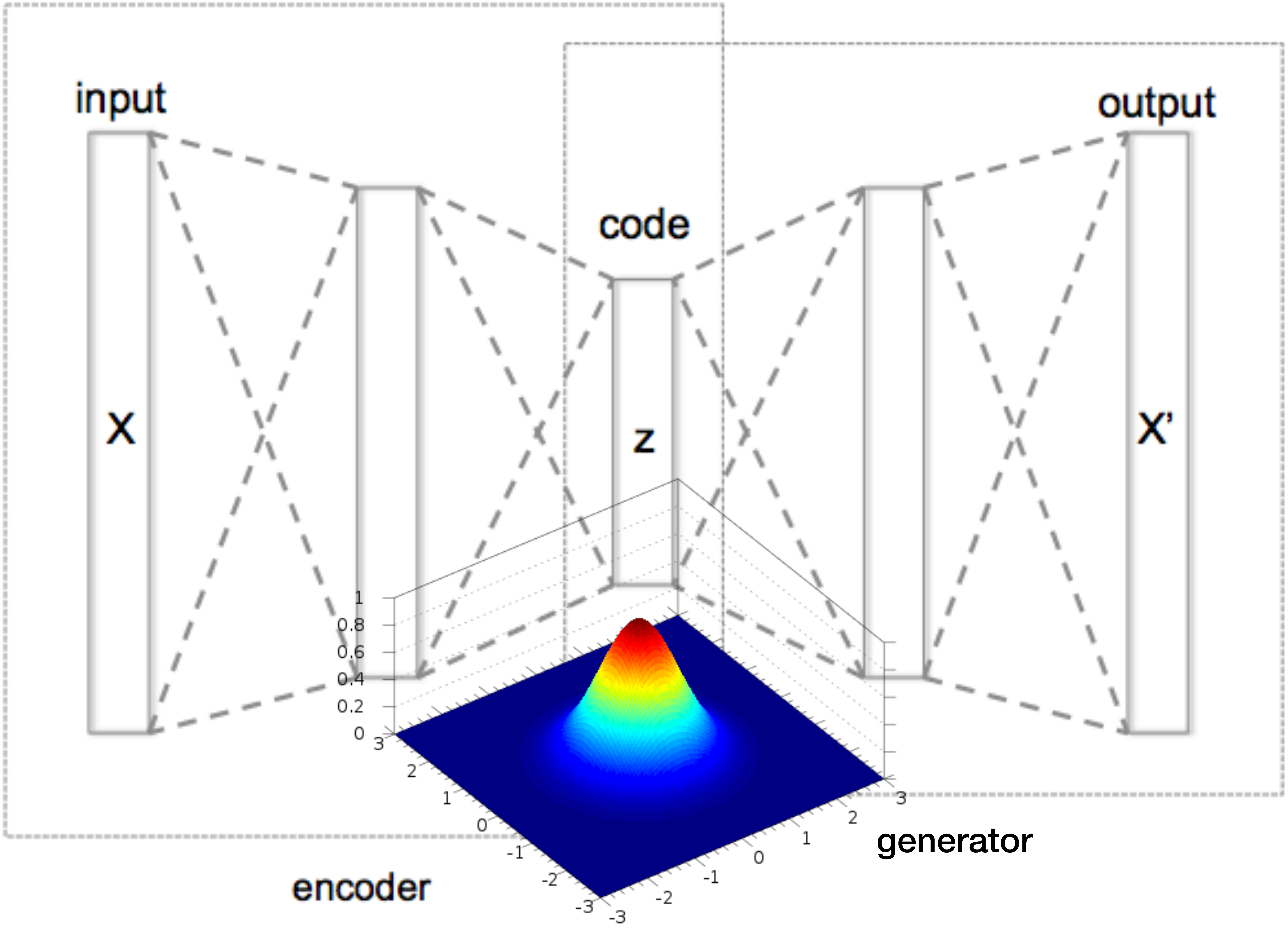
Variational Autoencoder



Generative autoencoder



Generative autoencoder



StyleGAN

Close-up from previous slide. Learning optics, but not really there yet.

StyleGAN

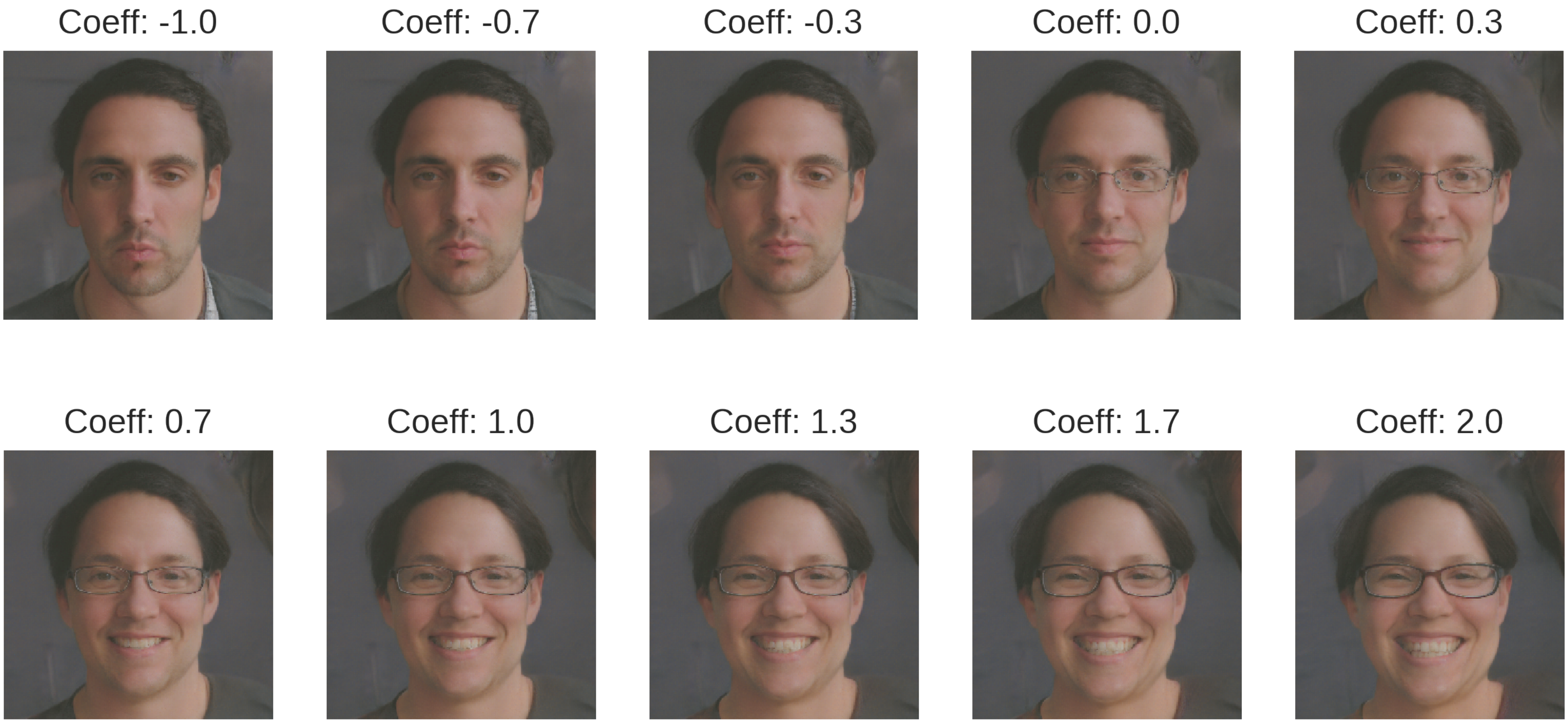
Inverting the Generator

- Nowadays, Generative Adversarial Networks beat the hell out of Autoencoders in generative quality.
- But their big handicap is that they don't have an encoder, they cannot be used for embedding.
- My main current research interest is making generative autoencoders just a bit better at generation, by shaping the point cloud of the embedded example points.

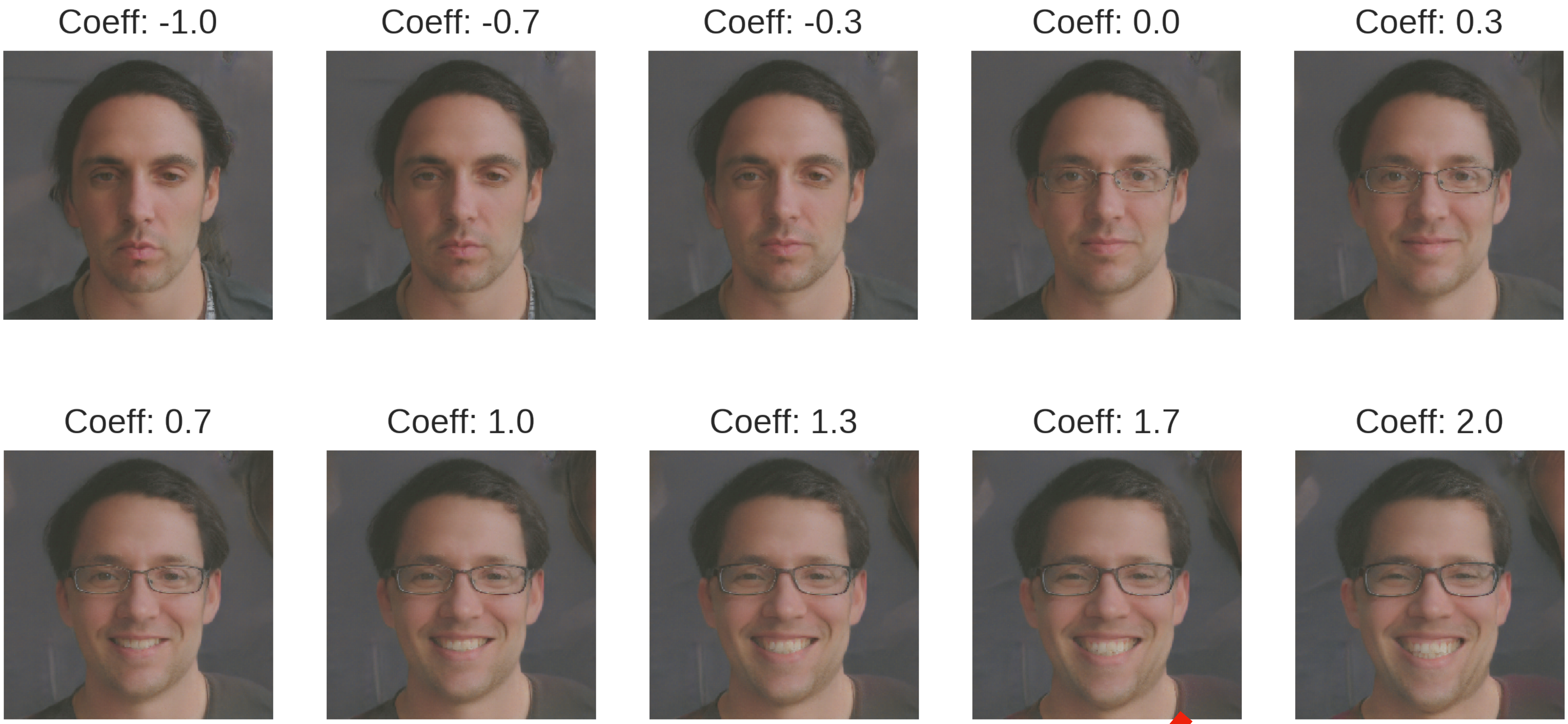
- But we are definitely not there yet, so today, I’ll just find the embedding through optimization.

- For today, the important part is that these models all have the nice linear latent structure that we are interested in.

move_and_show(daniel_varga + 0.6 * gender_direction, age_direction, np.linspace(-5, +2, 10))




```
move_and_show(daniel_varga + 0.6 * gender_direction, smile_direction, np.linspace(-1, +2, 10))
```

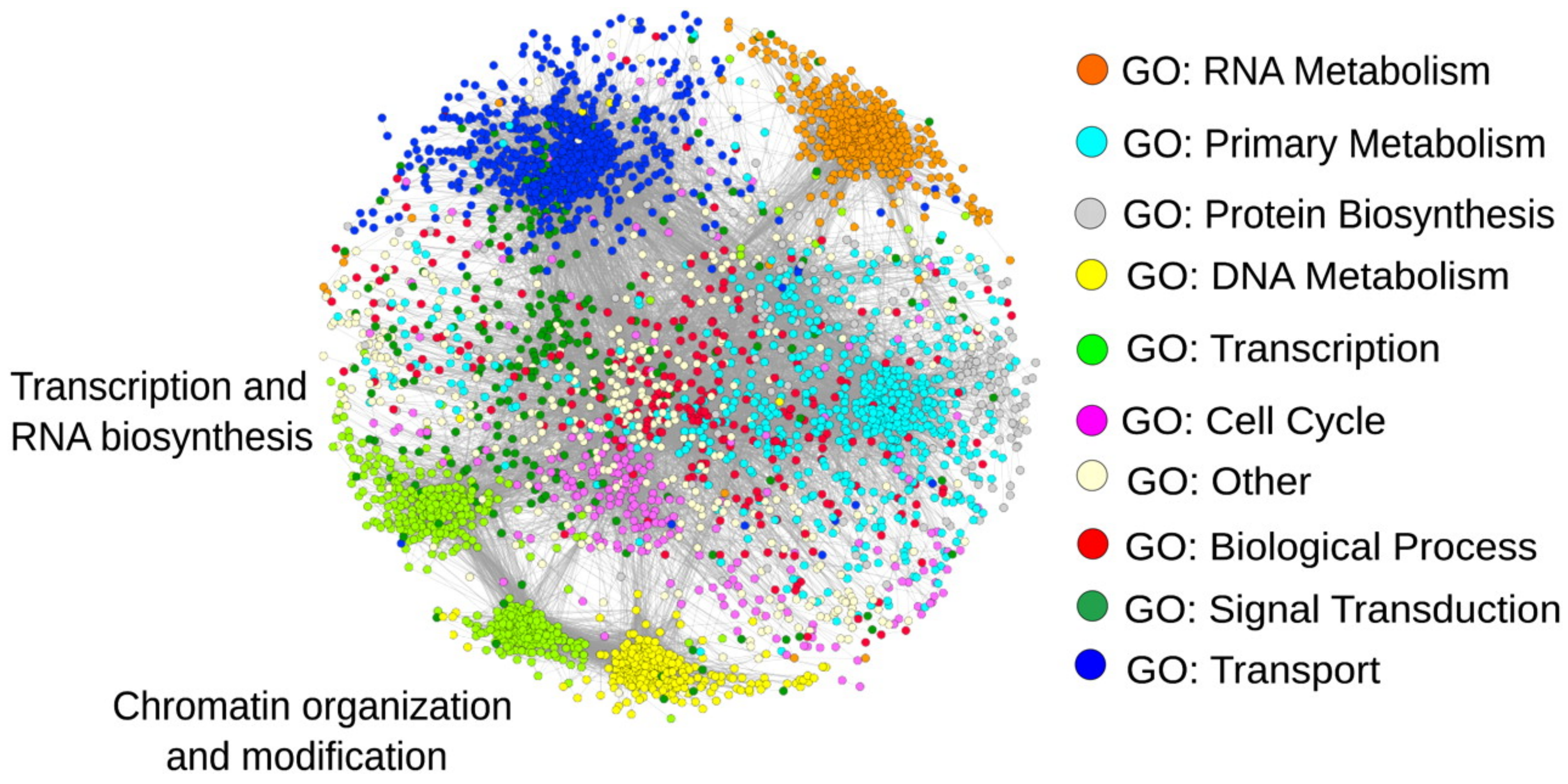



```
move_and_show(daniel_varga + 0.6 * gender_direction, smile_direction + 0.3 * gender_direction, np.linspace(-1, +2, 10))
```

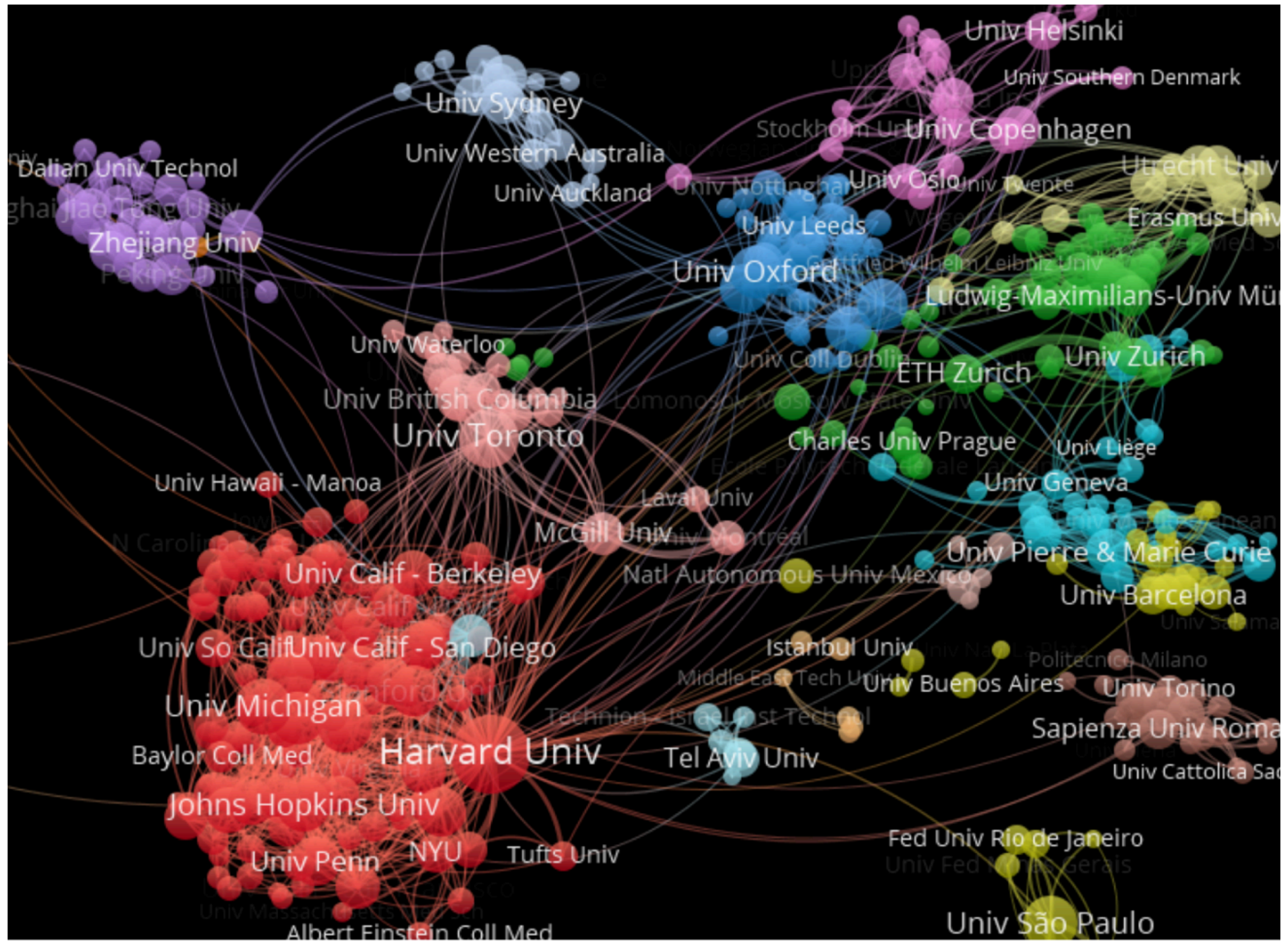
Yes, that’s frivolous, but the underlying idea of deep latent representations is not. It’s at the core of all modern machine translation, question answering, image processing and more.

Message passing

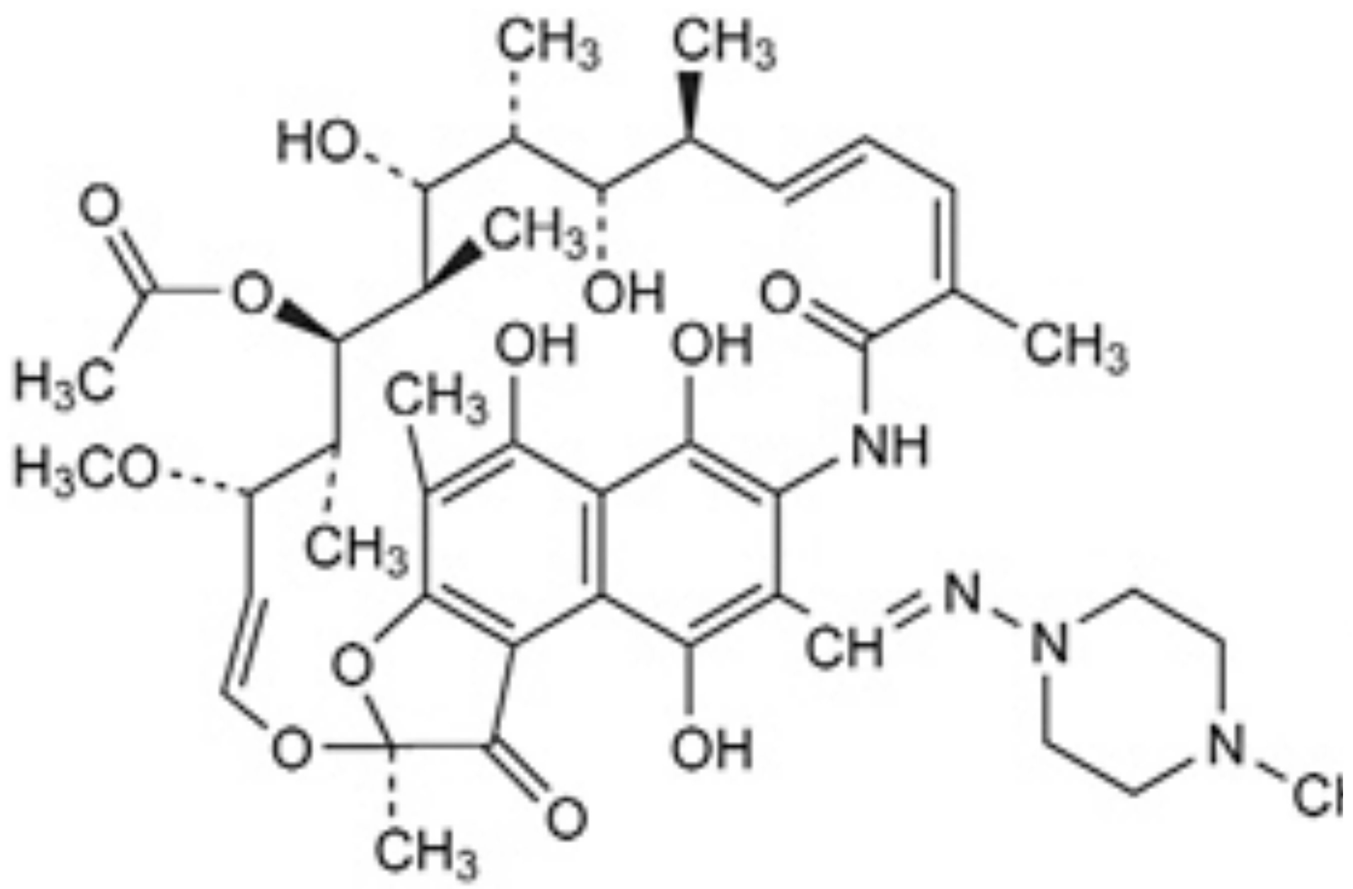
Networks



Protein interaction network

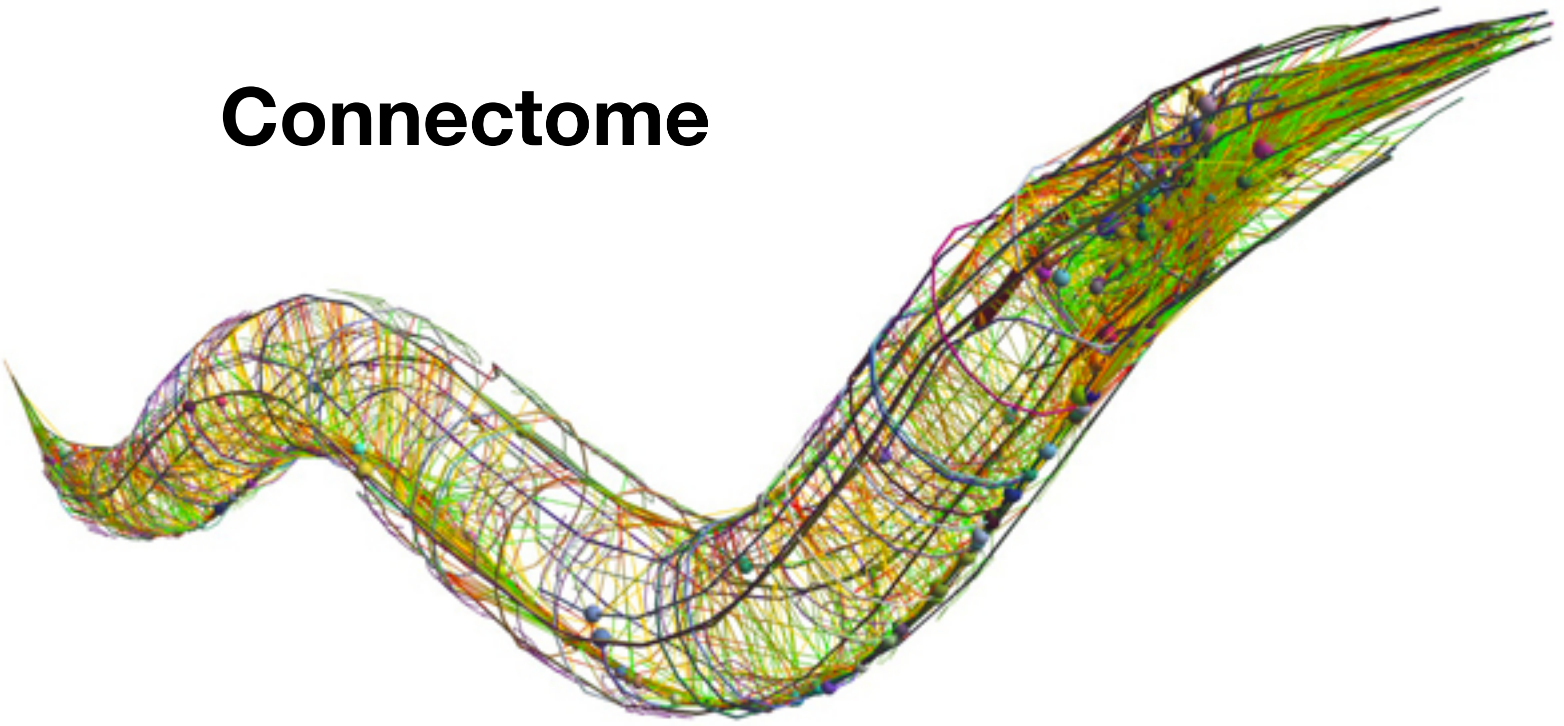


Citation network



Molecule

Connectome



Social network

PageRank

the $25,000,000,000 eigenvector

"Better teams are those teams that beat better teams."

Michael Mahoney

"One man's vicious circle is another man's successive approximation procedure."

Cosma Shalizi

The random surfer

```python
import numpy as np

def pagerank(M, eps=1.0e-8, d=0.85):
    N = M.shape[1]
    v = np.random.rand(N, 1)
    v = v / np.linalg.norm(v, 1)
    last_v = np.ones((N, 1), dtype=np.float32) * 100
    M_hat = (d * M) + (((1 - d) / N) * np.ones((N, N), dtype=np.float32))

    while np.linalg.norm(v - last_v, 2) > eps:
        last_v = v
        v = np.matmul(M_hat, v)
    return v
```

Power iteration

```python
def pagerank(M, eps=1.0e-8, d=0.85):
    N = M.shape[1]
    v = np.random.rand(N, 1)
    v = v / np.linalg.norm(v, 1)
    last_v = np.ones((N, 1), dtype=np.float32) * 100

    while np.linalg.norm(v - last_v, 2) > eps:
        last_v = v
        v = d * np.matmul(M, v) + (1 - d) / N
    return v
```

Message passing (with a VAT tax)

M is sparse

After all these years, I'm still surprised that I can actually edit Wikipedia

Browse history interactively

Revision as of 17:42, 28 February 2019 (edit)
Serols (talk | contribs)
m *(Reverted edits by 66.215.28.116 (talk) (HG) (3.4.6))*
(Tags: Huggle, Rollback)
← Previous edit

Latest revision as of 00:43, 7 March 2019 (edit) (undo)
80.99.84.109 (talk)
(for large sparse matrices explicitly computing the dense M_hat does not scale, and it is unnecessary.)

Line 252:

```
  v = v / np.linalg.norm(v, 1)
  last_v = np.ones((N, 1), dtype=np.float32) * 100
- M_hat = (d * M) + (((1 - d) / N) * np.ones((N, N), dtype=np.float32))

  while np.linalg.norm(v - last_v, 2) > eps:
      last_v = v
-     v = np.matmul(M_hat, v)
  return v

```

Line 252:

```
  v = v / np.linalg.norm(v, 1)
  last_v = np.ones((N, 1), dtype=np.float32) * 100

  while np.linalg.norm(v - last_v, 2) > eps:
      last_v = v
+     v = d * np.matmul(M, v) + (1 - d) / N
  return v

```

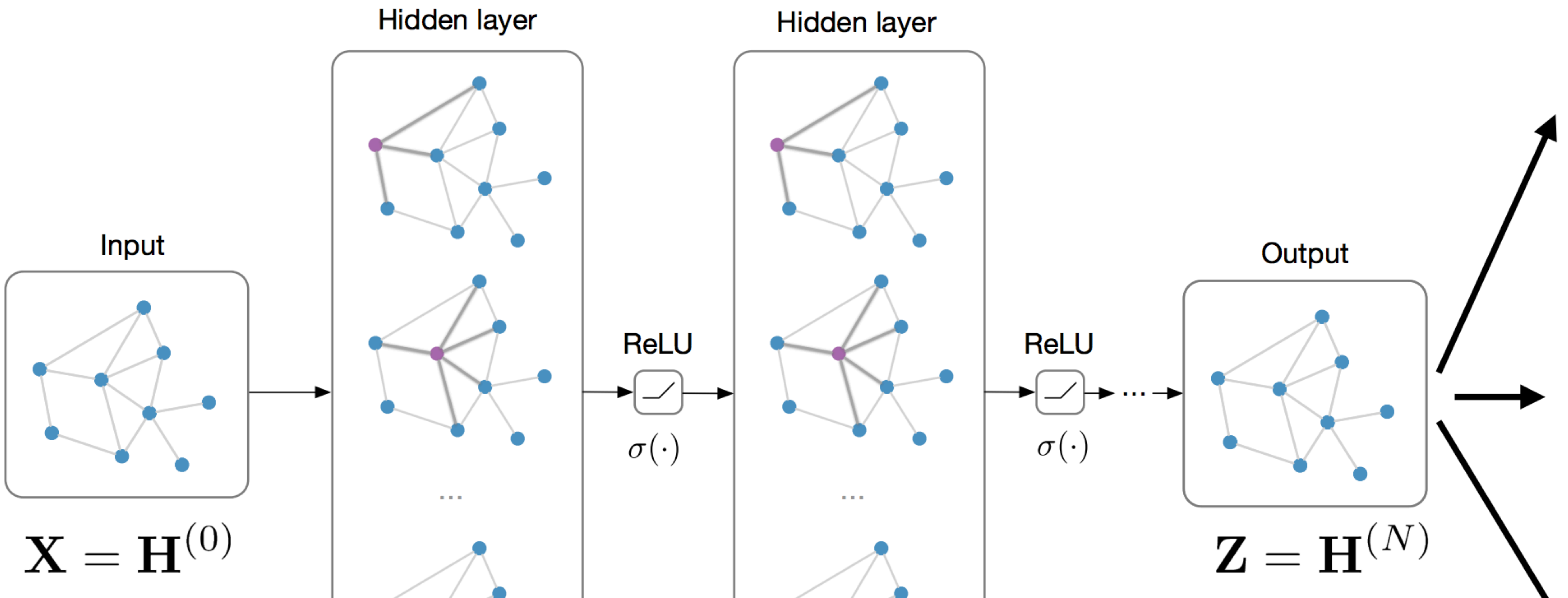
If this simple idea is worth $12,000,000,000, how much is its extreme generalization worth? :-)

Graph convolutional networks

Kipf and Welling, ICML 2017

One fits all: Classification and link prediction with GNNs/GCNs

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$



$$\mathbf{H}^{(l+1)} = \sigma\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$$

Node classification:

$$\mathrm{softmax}(\mathbf{z_n})$$

e.g. Kipf & Welling (ICLR 2017)

Graph classification:

$$\mathrm{softmax}(\textstyle\sum_n \mathbf{z_n})$$

e.g. Duvenaud et al. (NIPS 2015)

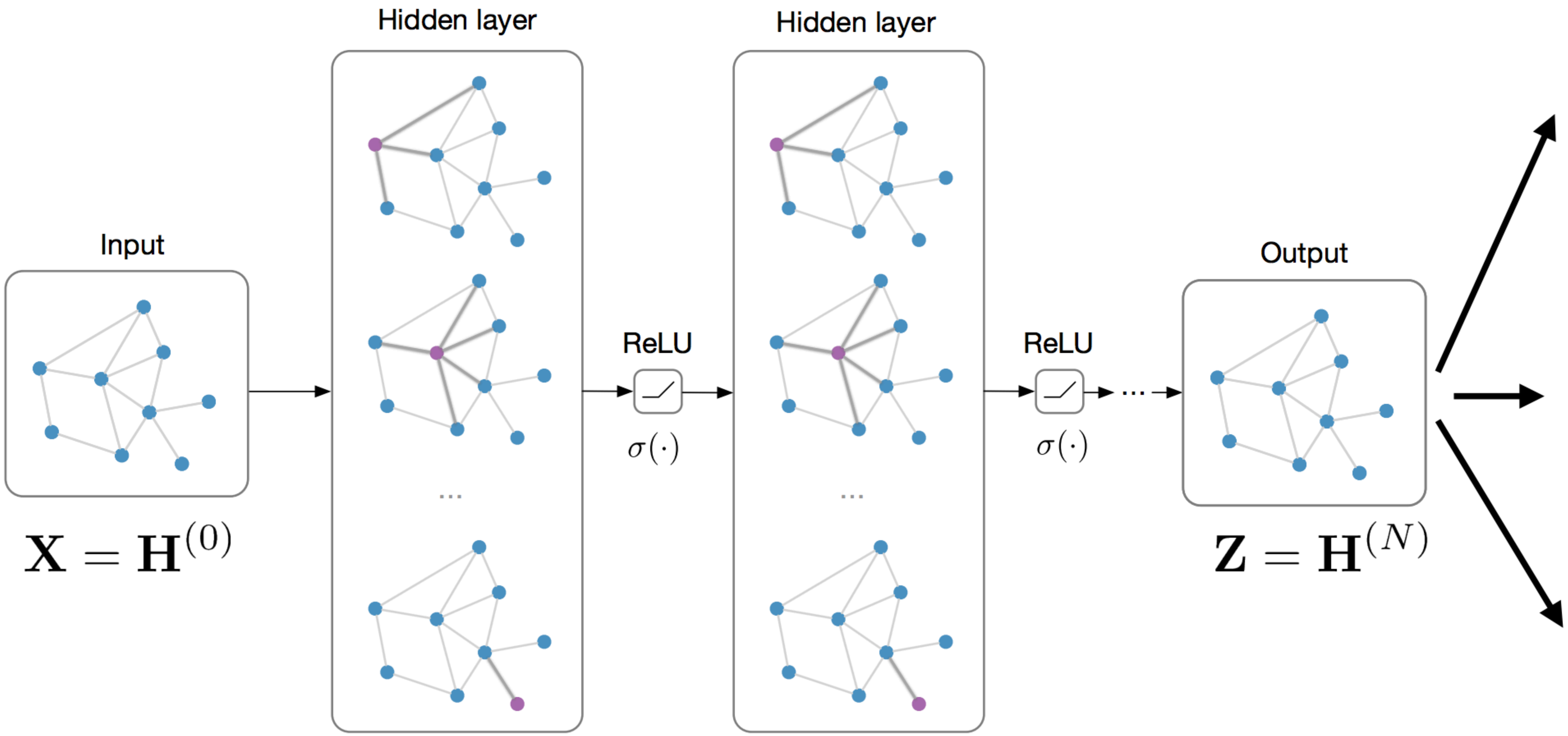
Link prediction:

$$p(A_{ij}) = \sigma(\mathbf{z_i}^{\mathbf{T}}\mathbf{z_j})$$

Kipf & Welling (NIPS BDL 2016)
"Graph Auto-Encoders"

One fits all: Classification and link prediction with GNNs/GCNs

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$



$$\mathbf{H}^{(l+1)} = \sigma\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$$

Node classification:

$$\mathrm{softmax}(\mathbf{z_n})$$

e.g. Kipf & Welling (ICLR 2017)

Graph classification:

$$\mathrm{softmax}(\textstyle\sum_n \mathbf{z_n})$$

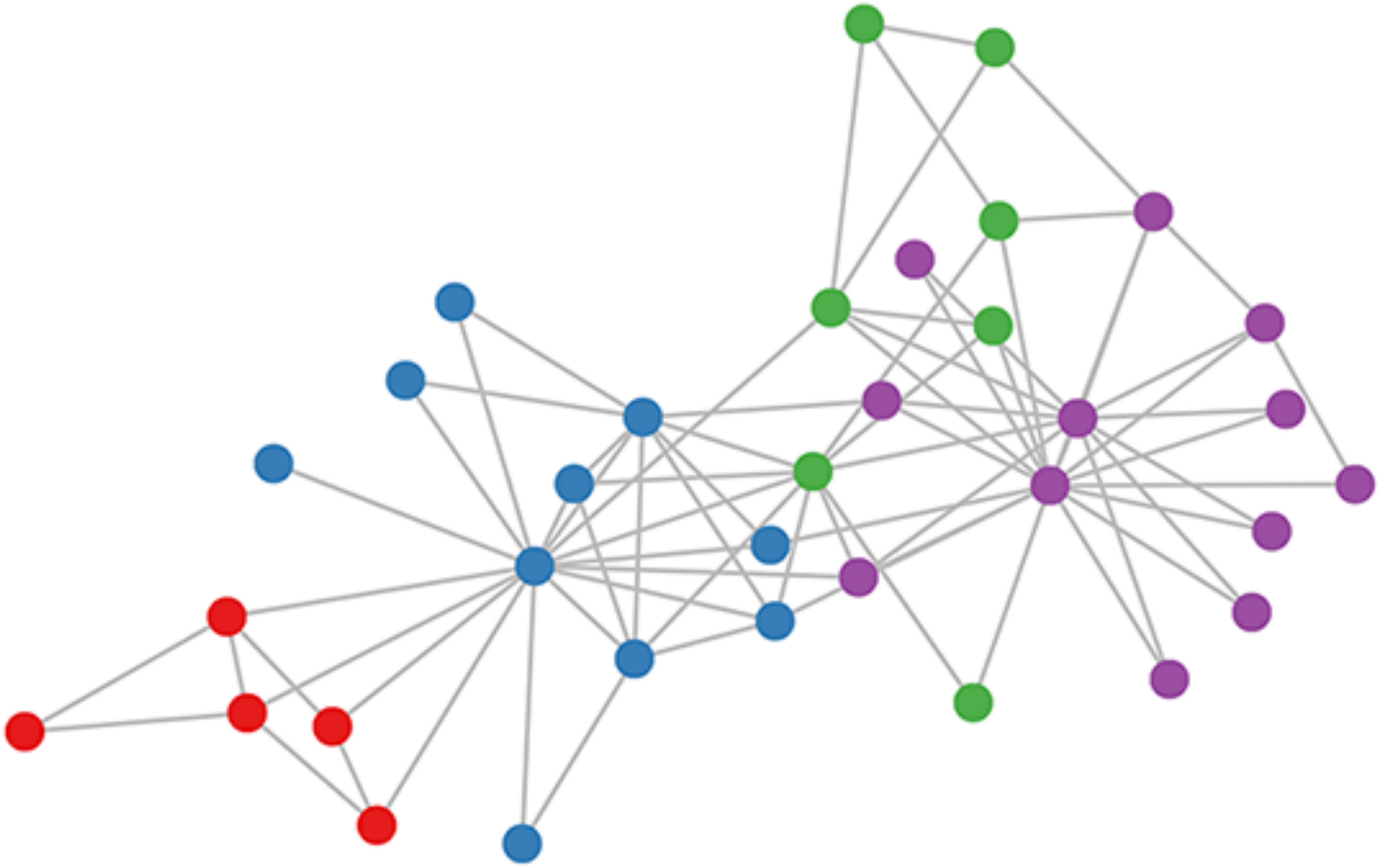
e.g. Duvenaud et al. (NIPS 2015)

Link prediction:

$$p(A_{ij}) = \sigma(\mathbf{z_i^T z_j})$$

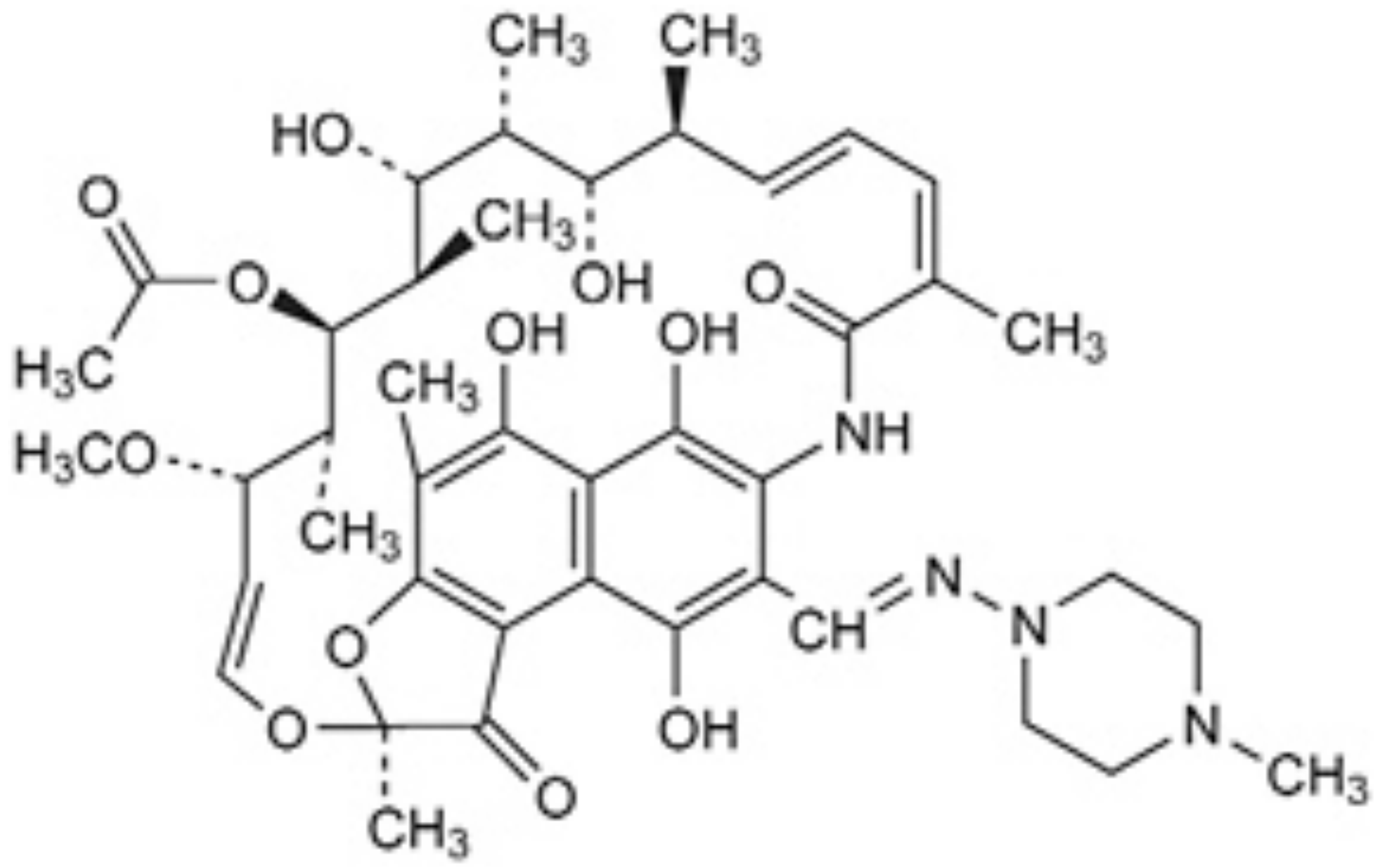
Kipf & Welling (NIPS BDL 2016)
"Graph Auto-Encoders"

Node classification



$$\mathrm{softmax}(\mathbf{z_n})$$

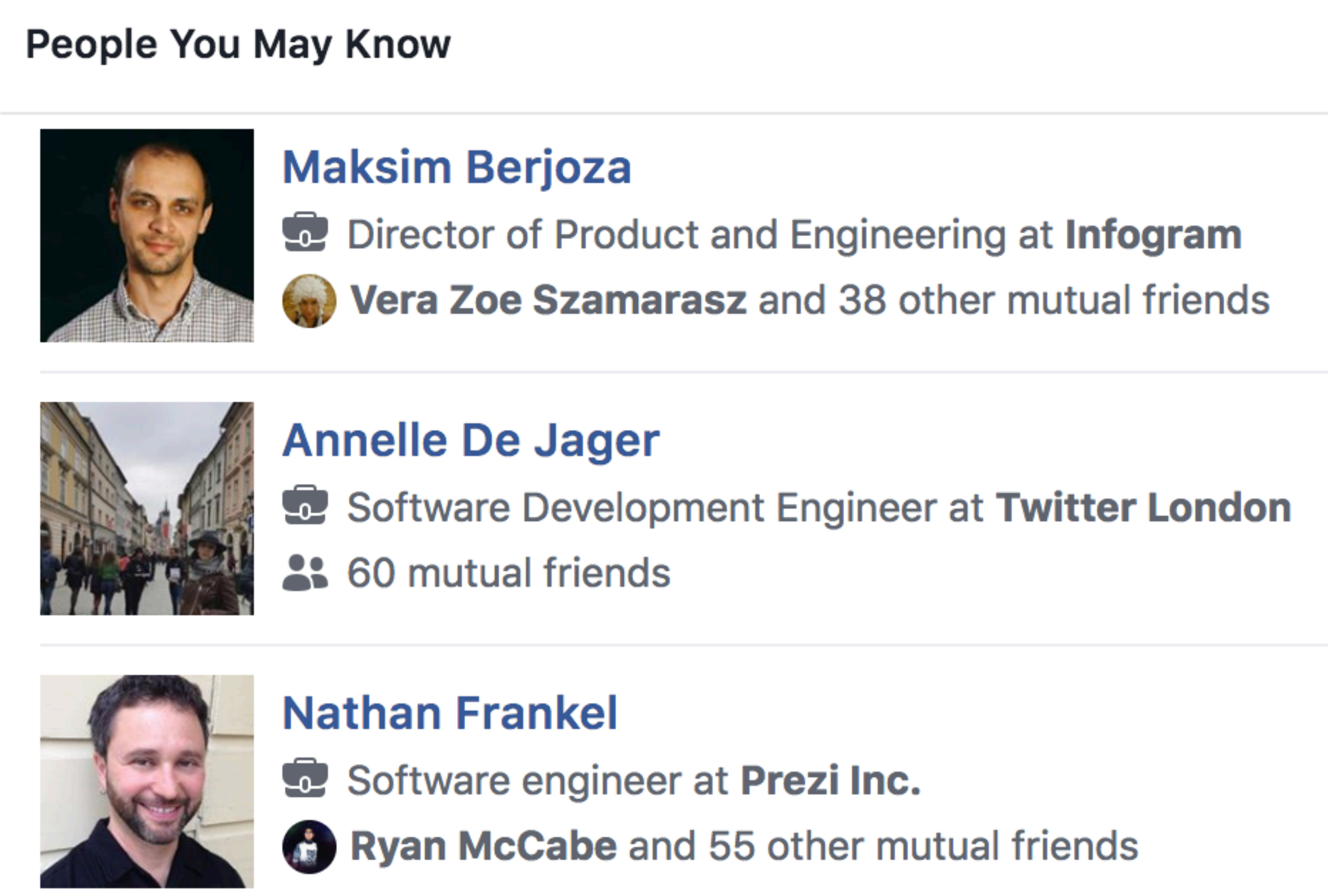
Graph classification,
Graph regression



- Is it poisonous?
- What is its boiling point?

$$\mathrm{softmax}(\textstyle\sum_n \mathbf{z_n})$$

Link prediction



General

Bipartite

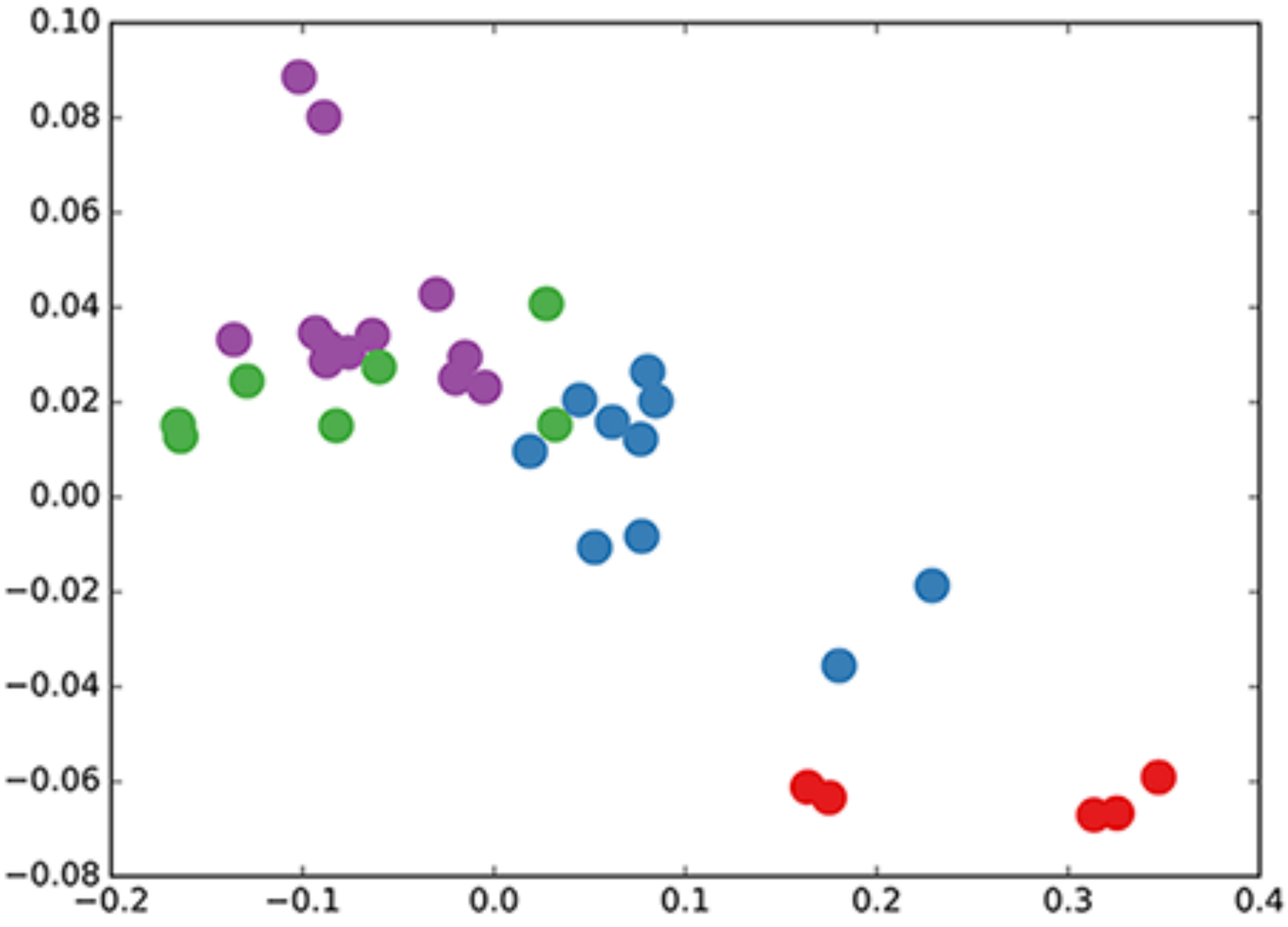
$$p(A_{ij}) = \sigma(\mathbf{z_i^T z_j})$$

Any kind of recommendation, really.

This is when I write up the GCN formula at the blackboard

aggregation & local processing

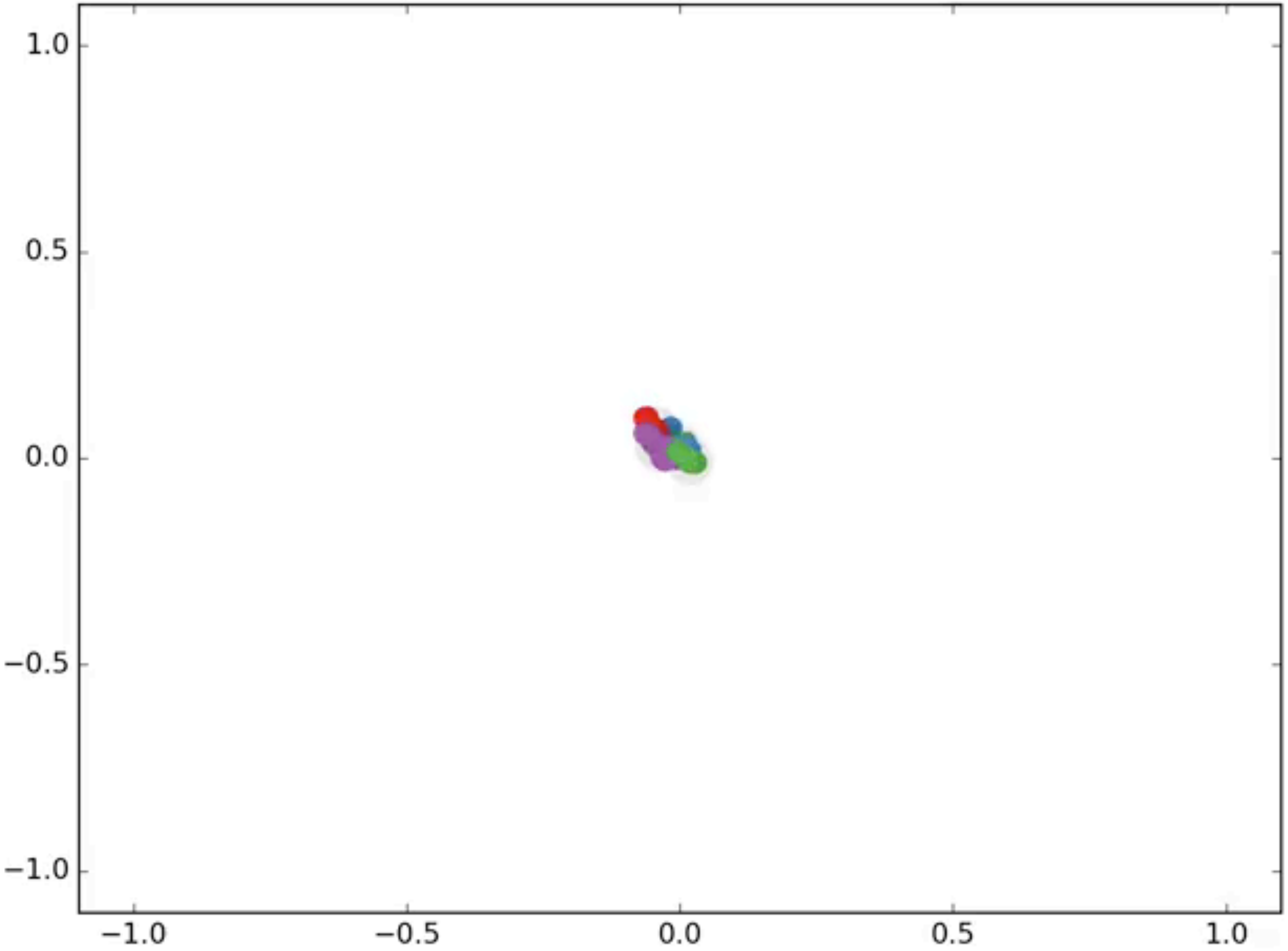
GCN with random weights



The Weisfeiler-Lehman parallel

- A classic message passing algorithm.
- Works by giving each node a fingerprint.
- Solves Graph Isomorphism for all but the most artificially symmetric graphs. (Actually, people once hoped that some version of it can put GI in P.)
- GCN with random weight matrices can be seen as a continuous analog.

GCN semi-supervised



Do we even need that ugly nonlinear operation in the GCN formula?

Simplifying Graph Convolutional Networks

Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr., Christopher Fifty, Tao Yu, Kilian Q. Weinberger

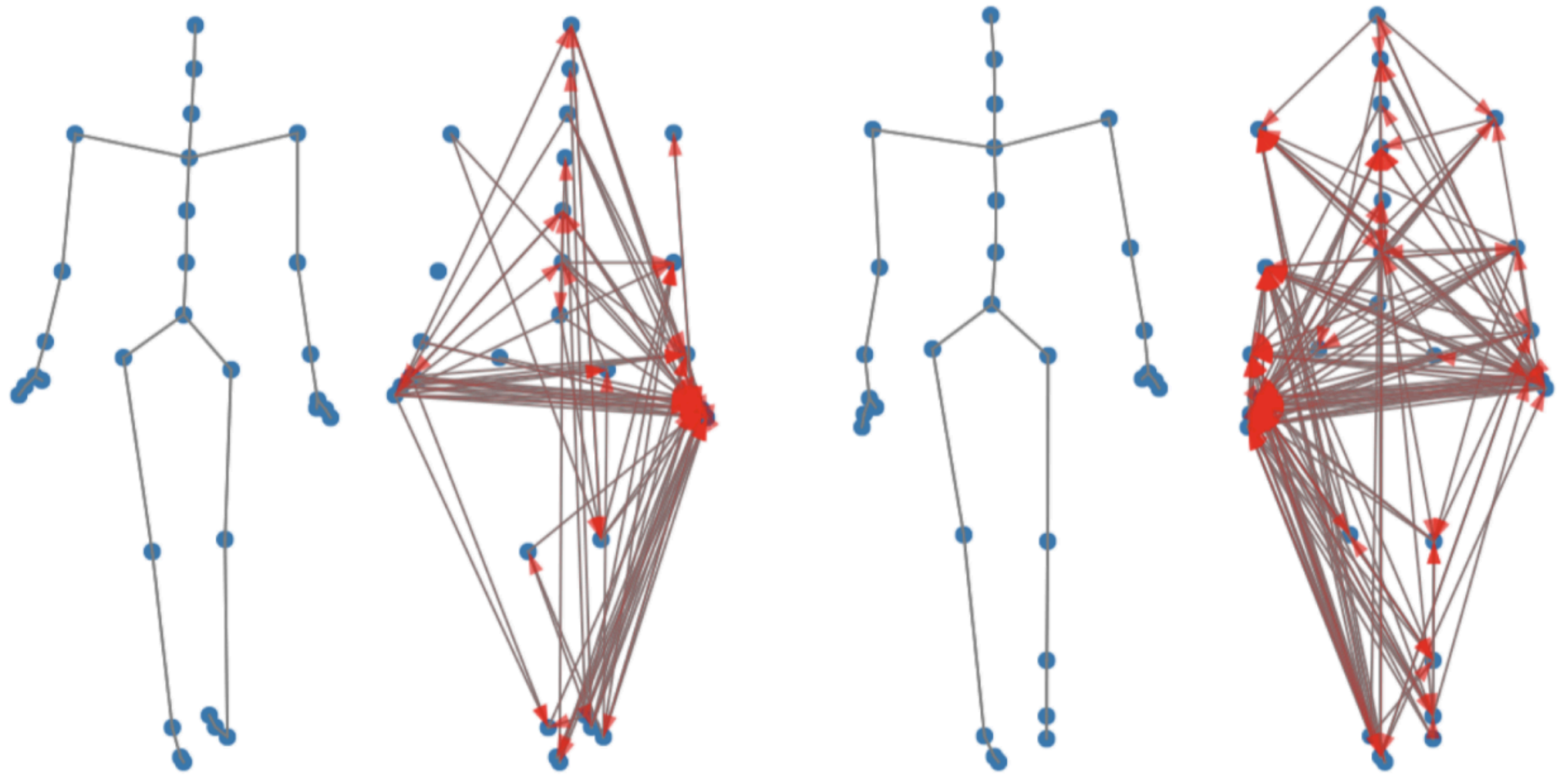
February 2019

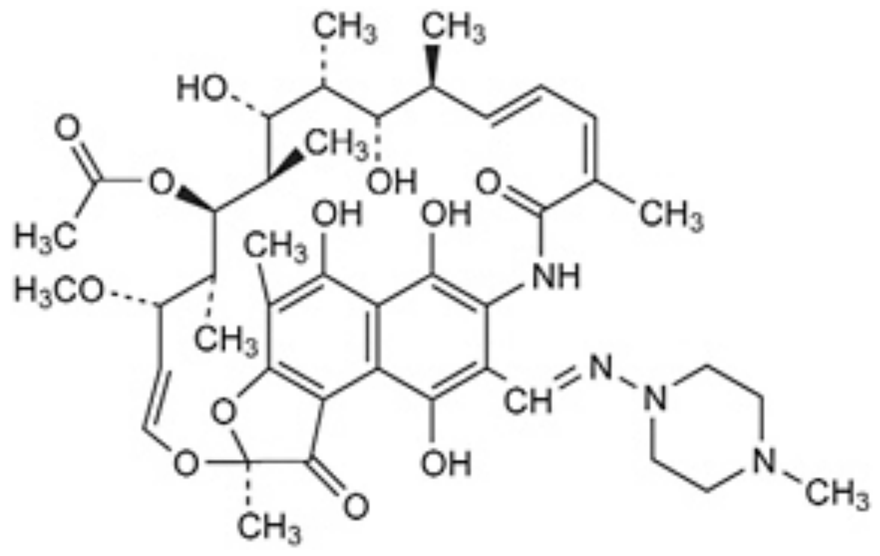
Do we even need that ugly nonlinear operation in the GCN formula?

- If we omit it, the GCN formula becomes linear,
- …and the whole right half of the chained matrix multiplication collapses to a single W weight matrix.
- We can still solve many of the tasks GCNs can solve.
- So it seems like the important part is the information propagation.

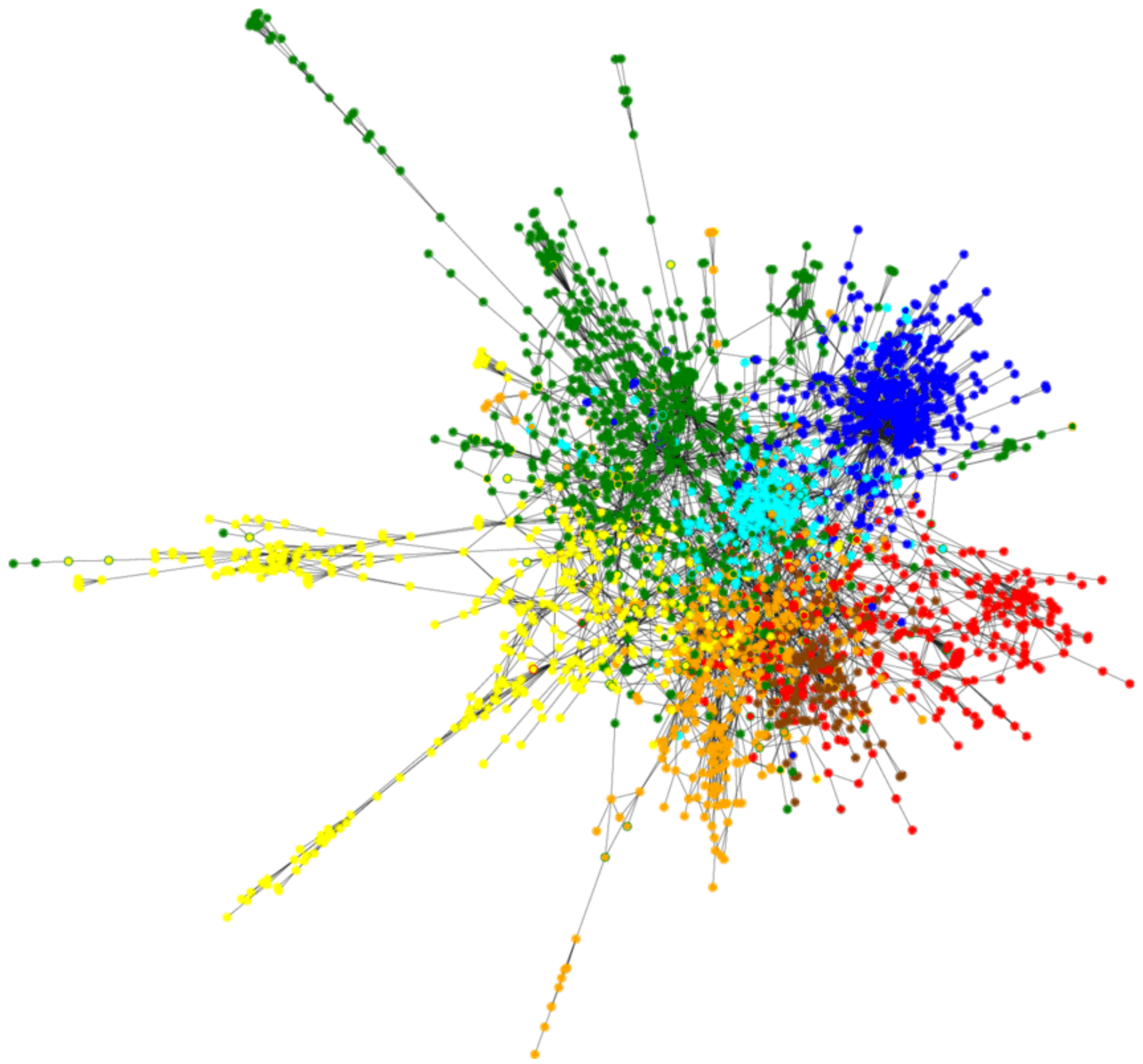
Do we even need that ugly nonlinear operation in the GCN formula?

- Well, we probably need it if we want to model complex nonlinear dynamics.





But as the example of PageRank shows, linearity can get you quite far with large, less structured networks.

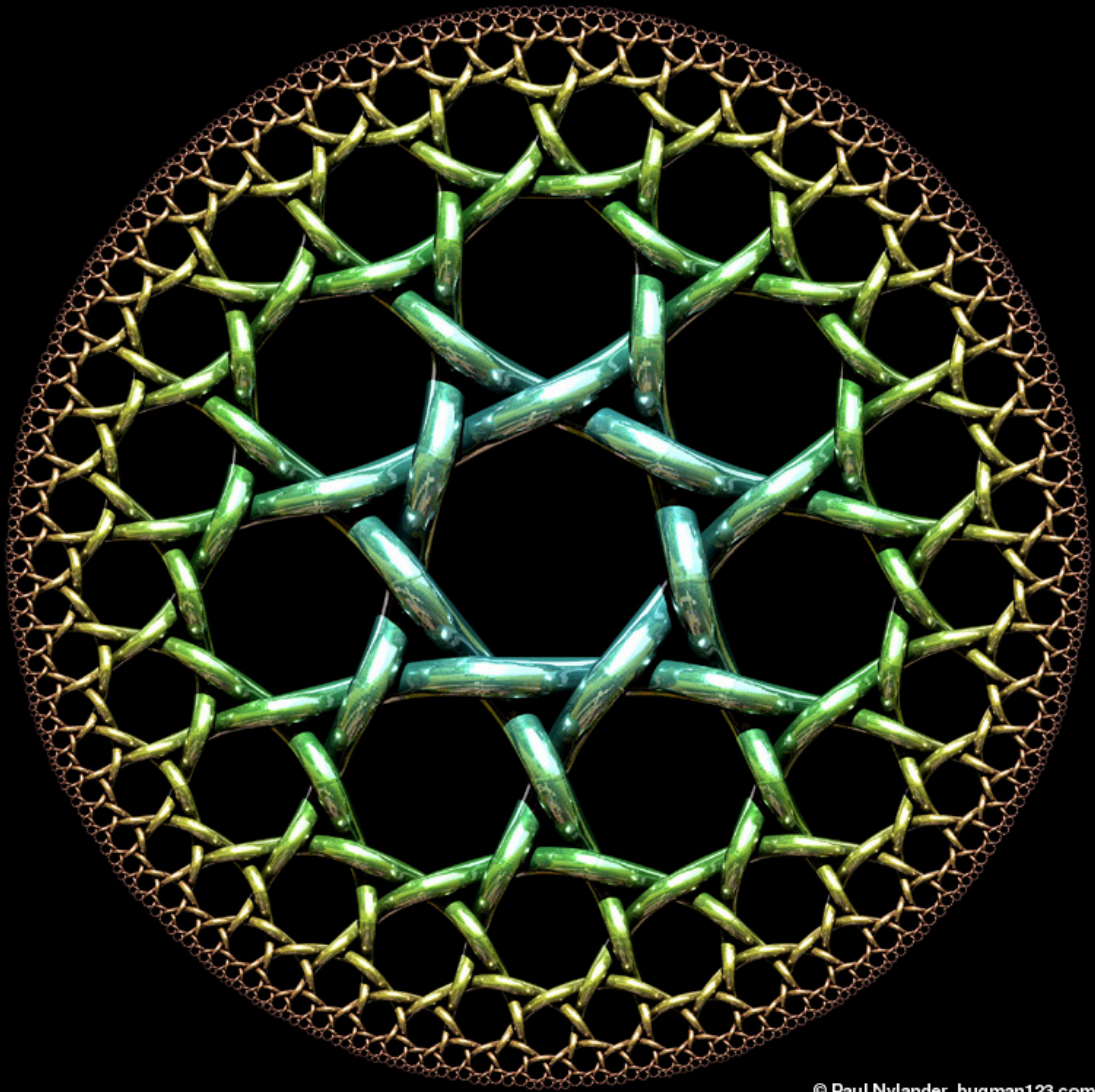


Is there a better way to propagate information?

- This $\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}$ propagation formula seems pretty ad hoc.
- Why does it work so well experimentally?
- People are experimenting with variations, like this orthogonal embedding-based version: Lovász Convolutional Networks, Yadav et al 2018.
- Can we do better? What does spectral graph theory say?

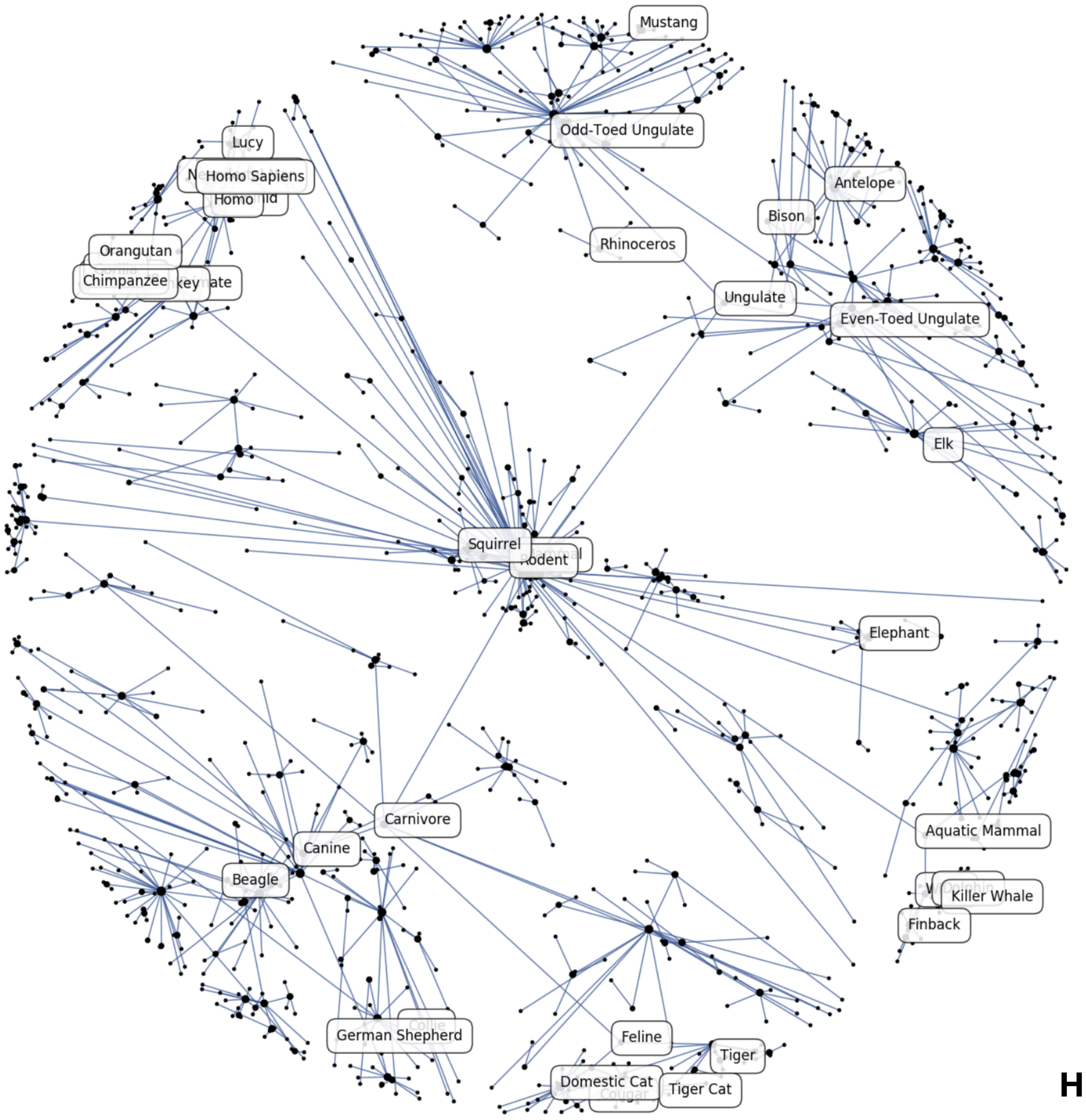
Thank you for your attention!

Hyperbolic embeddings

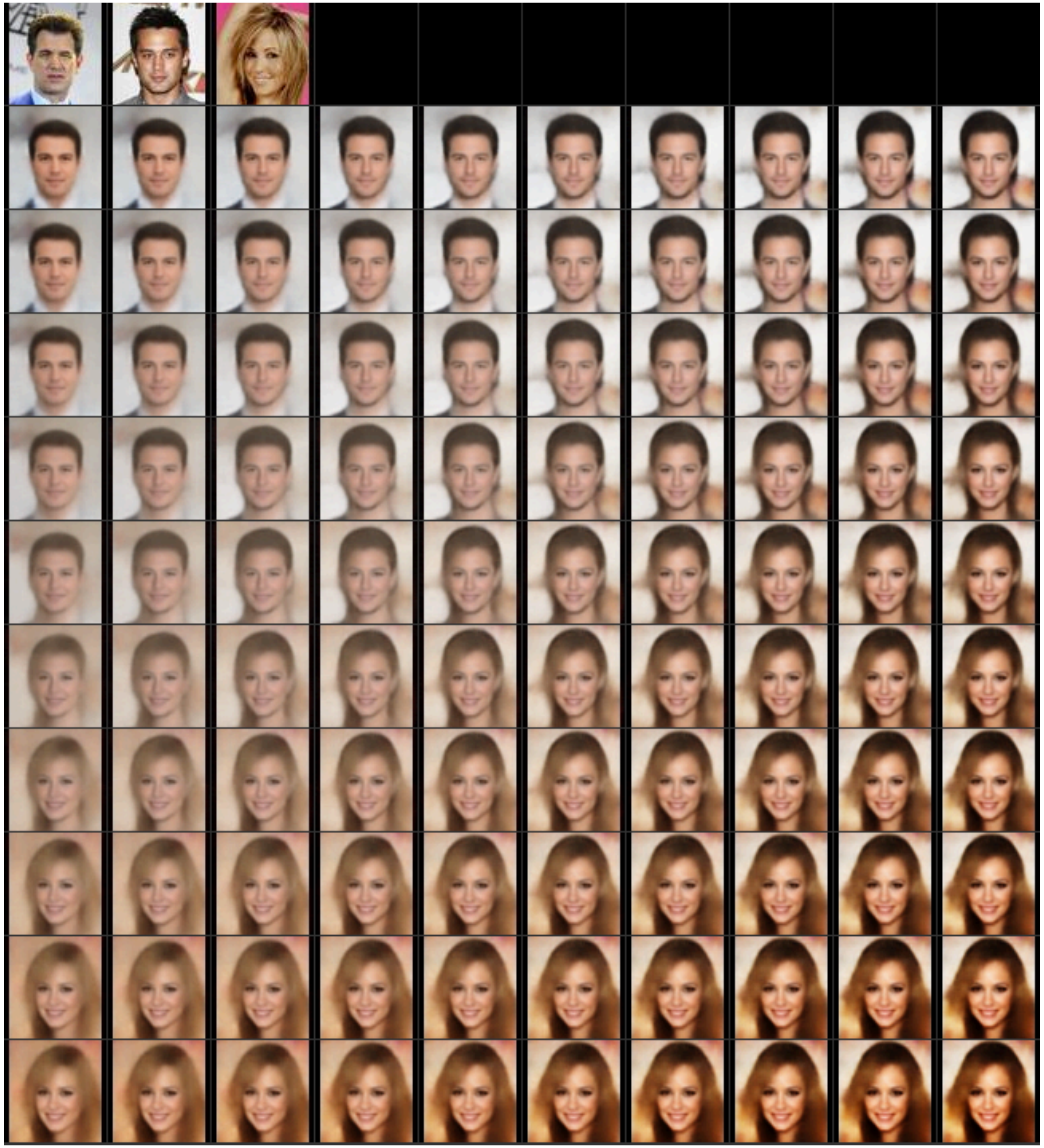


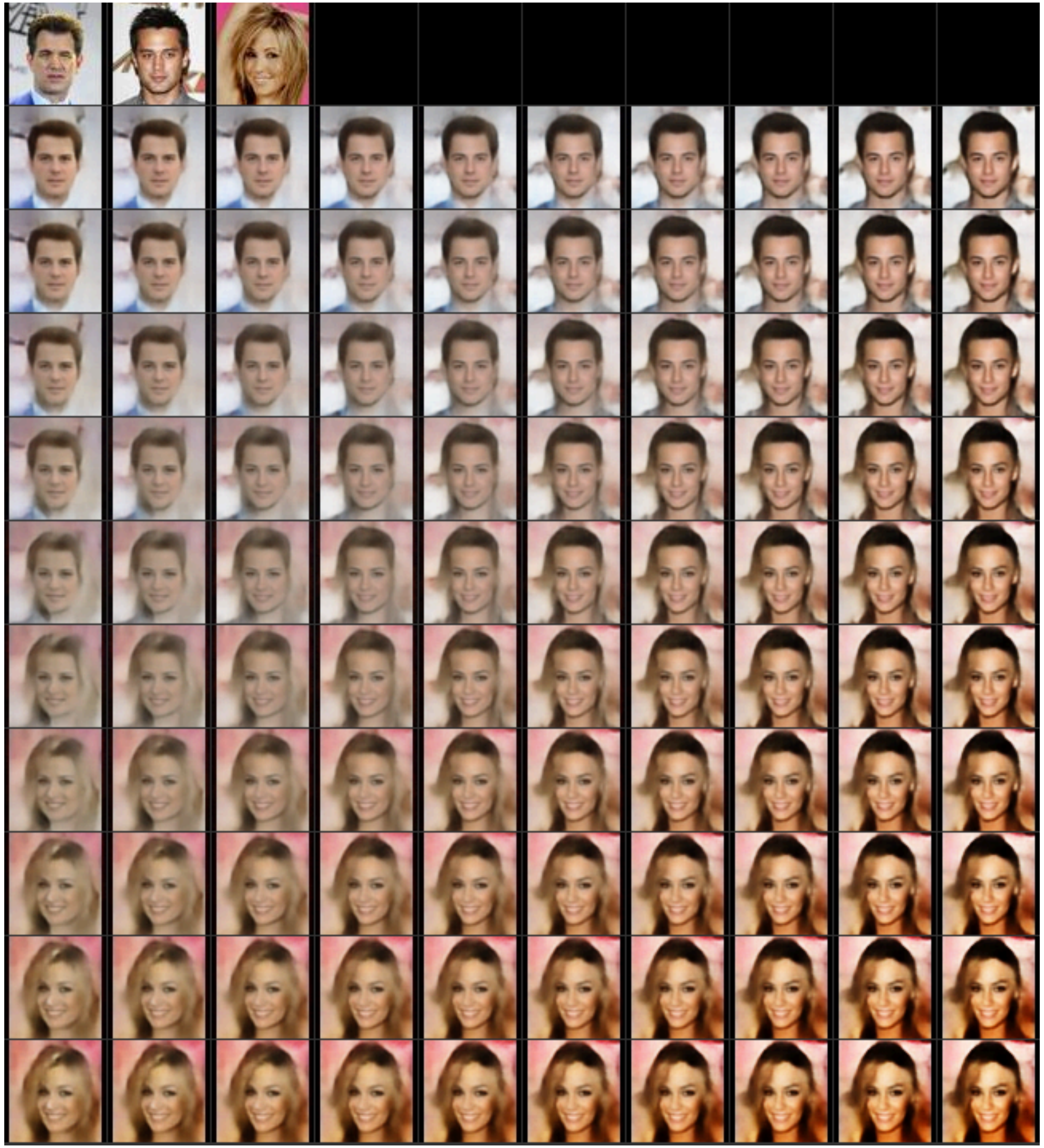
“[…] random geometric graphs in hyperbolic spaces are an adequate model for complex networks. The high-level explanation of this connection is that complex networks exhibit hierarchical, tree-like organization, while hyperbolic geometry is the geometry of trees. Graphs representing complex networks appear then as discrete samples from the continuous world of hyperbolic geometry.”

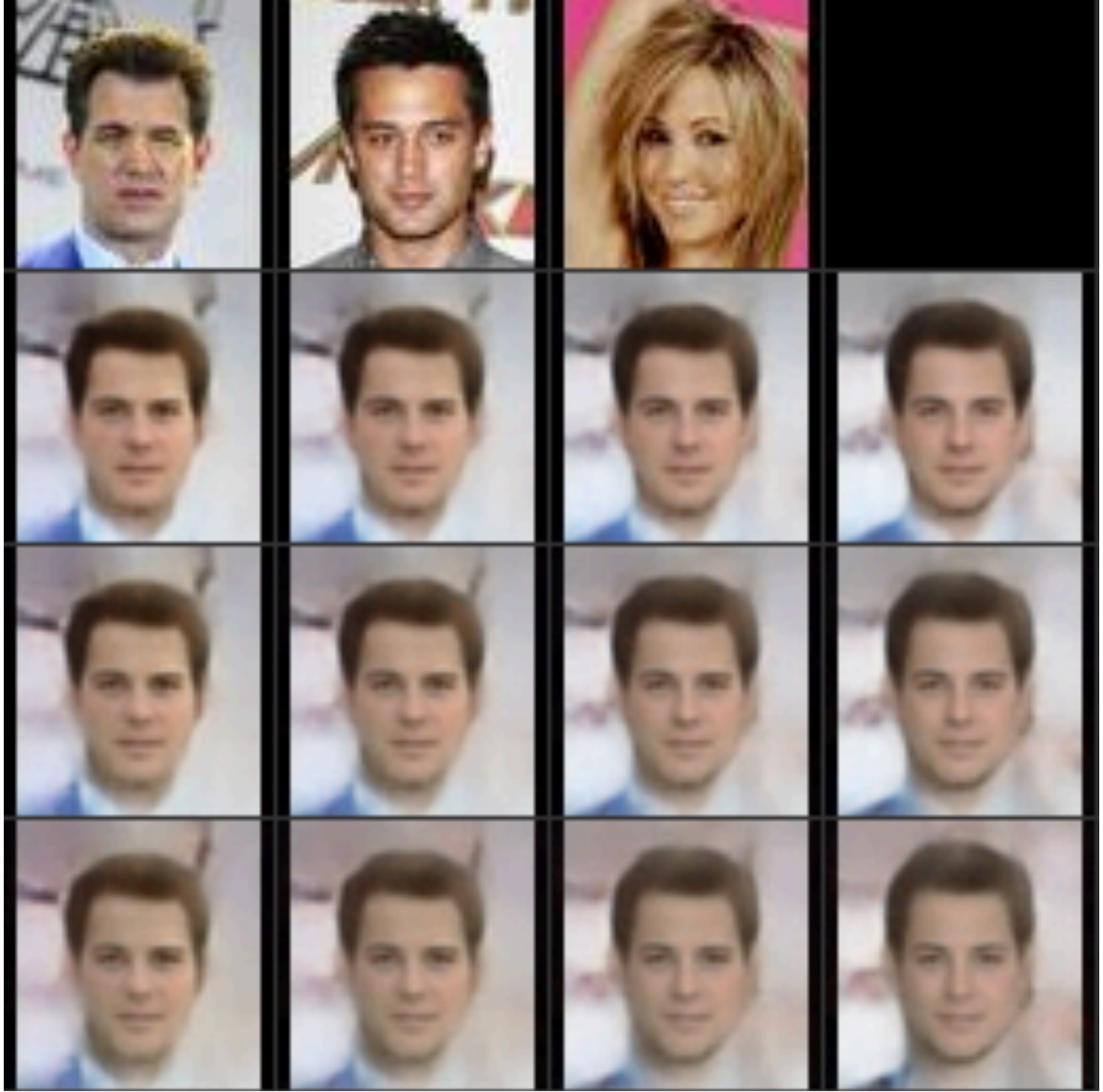
(Fragkiskos Papadopoulos)



Nickel & Kiela - Poincaré Embeddings for Learning Hierarchical Representations





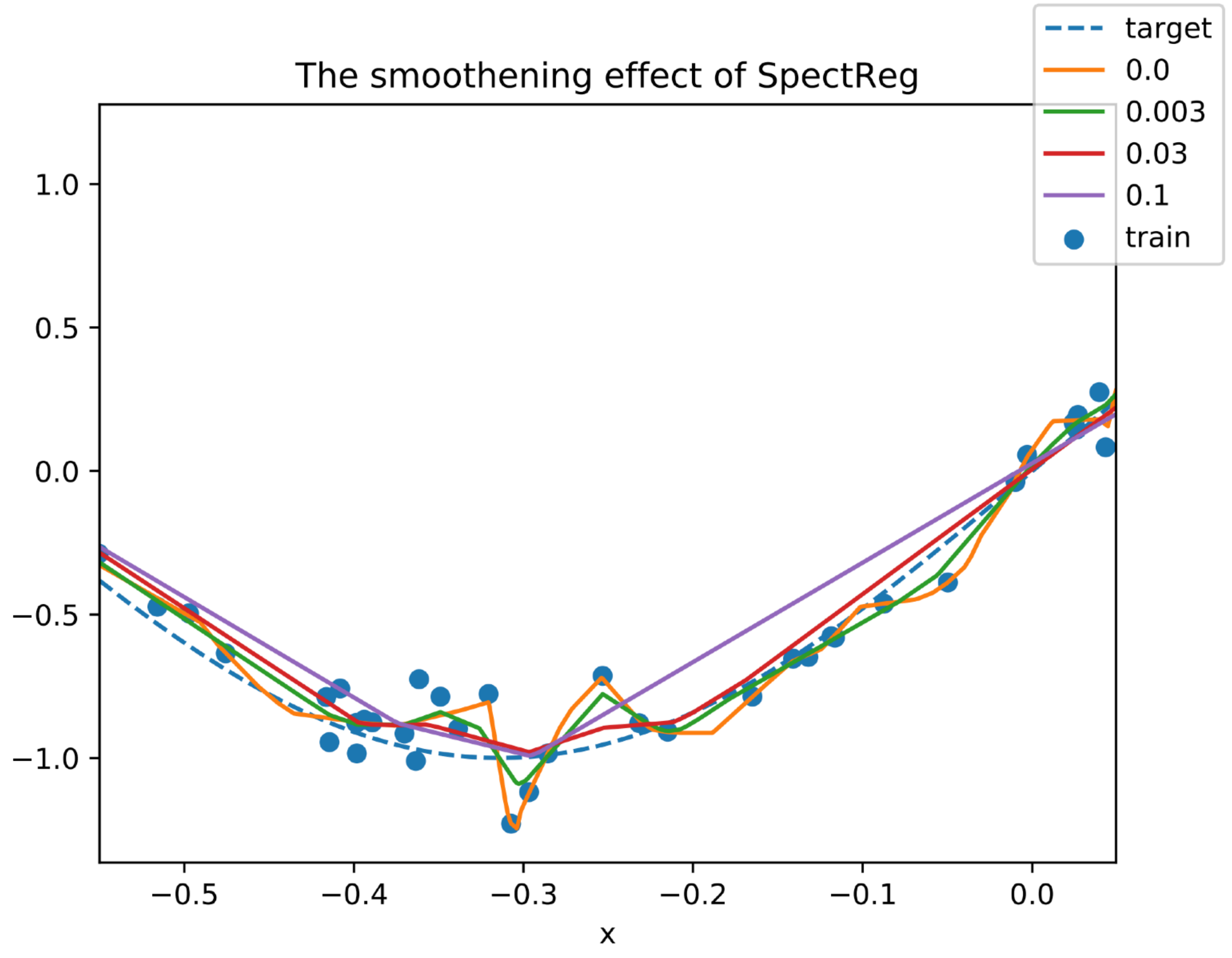


Smoothness priors

The **Spectral Regularizer** approximates the Frobenius norm of the Jacobian of the input-logit mapping at the training examples:

$$L_{SpectReg}(x, \Theta) = \|\frac{\partial}{\partial x} f_\Theta(x)\|_F^2 = \mathbb{E}_{\epsilon \sim \mathcal{N}(0, I^m)} [\|\epsilon^T \frac{\partial}{\partial x} f_\Theta(x)\|_2^2]$$

This helps the embedding take a smoother shape, helping generalization to unobserved objects.



…but I will not talk about it today.

They don’t have an encoder

- Ouch. I should have put more effort into solving the inverse optimization task.
- I used my own face because I didn’t want to do this to friends.
- I tried David Hilbert, but it ruined his face completely.

But at least the latent space has a nice linear structure