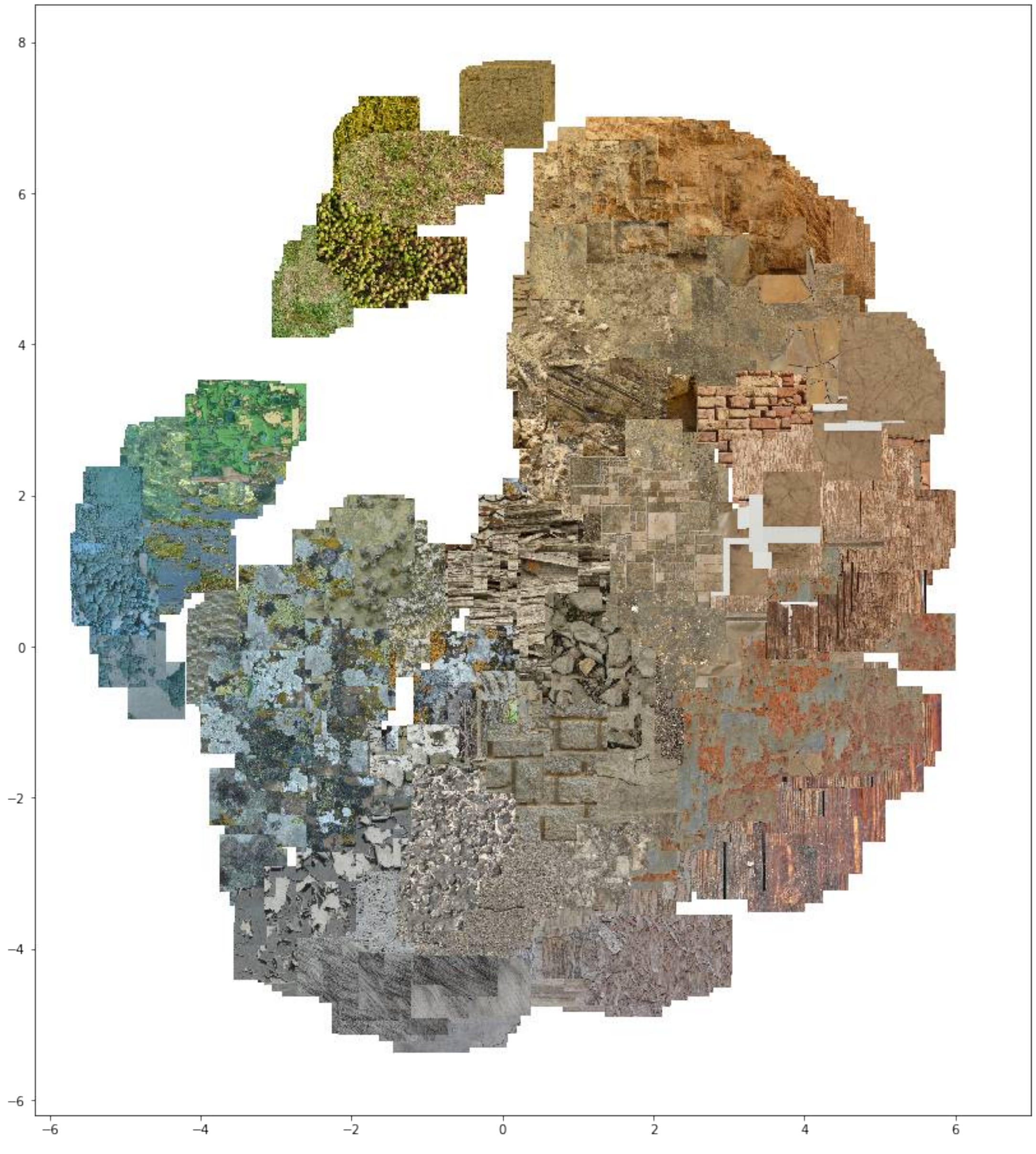
# Unsupervised Learning: Embedding Methods



# Why use unsupervised learning?

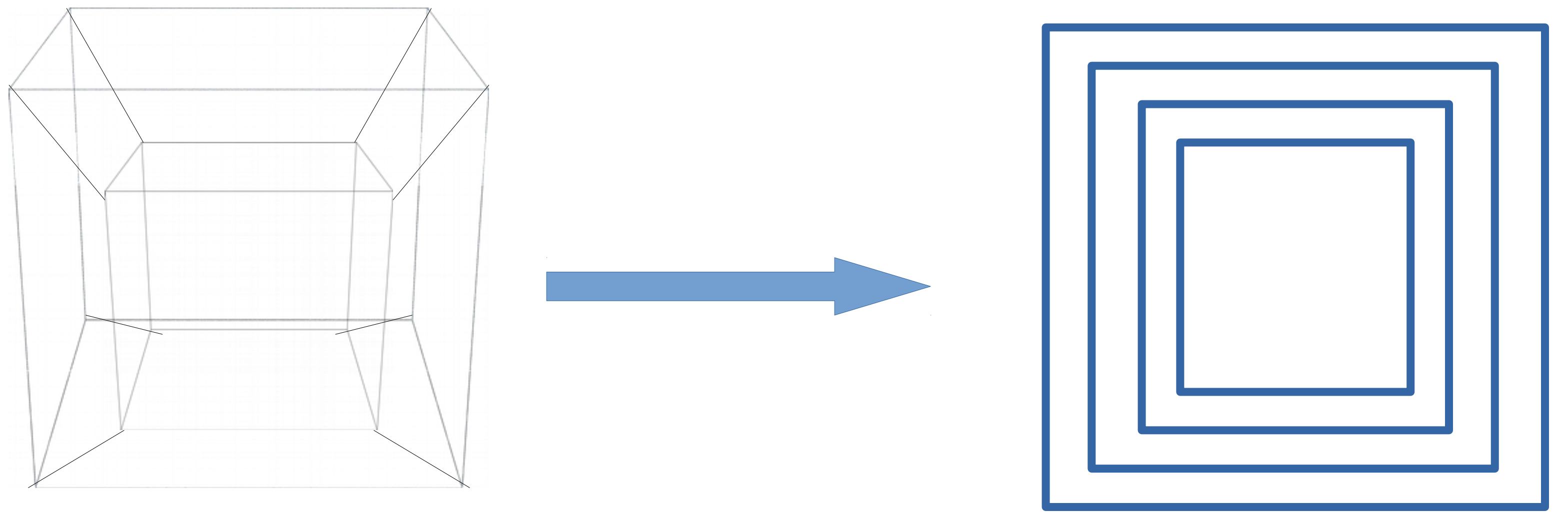
In many (industry) problems, raw data is cheap but labeling is expensive – pretraining

In scientific applications, unsupervised learning may be more useful for exploratory analysis

- Finding structure in the data that motivates subsequent explanatory work.

# Embedding

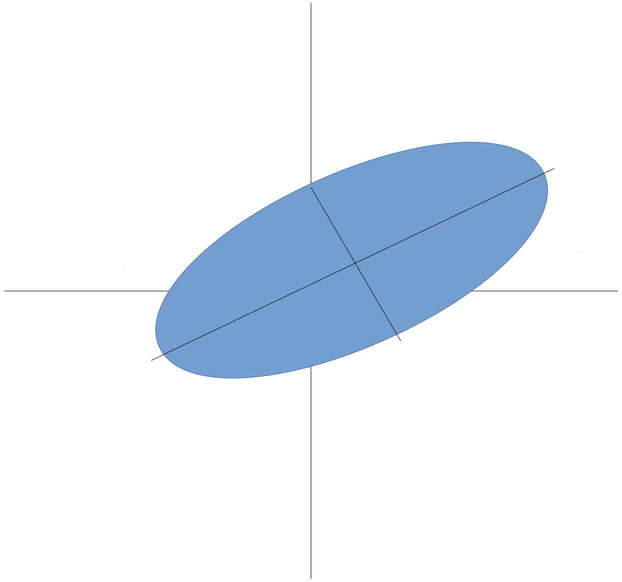
Basic idea: take high-dimensional, complicated data and project it down to make relationships in the data easier to understand.



# Principle Component Analysis

Simplest form of embedding is Principle Component Analysis.

Find independent directions of variation in the data, look at only the few biggest components.



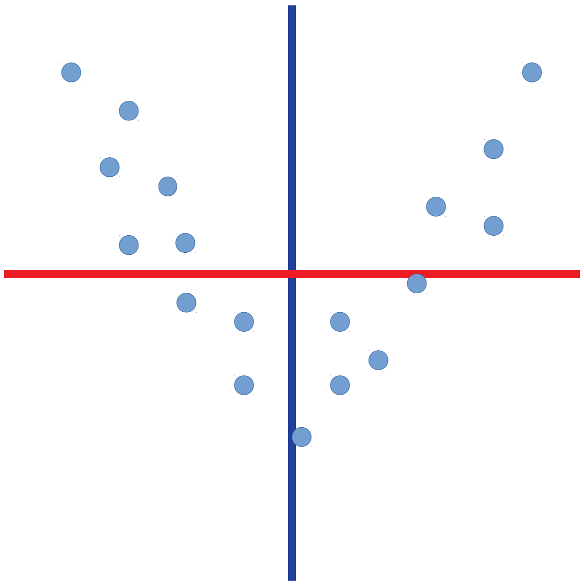
# Principle Component Analysis

In PCA, transform the data in such a way that we can approximately reconstruct it with fewer dimensions

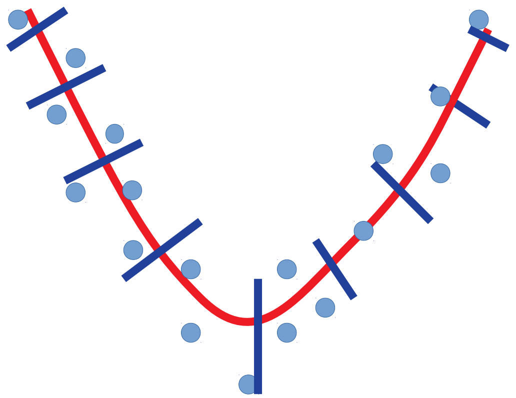
- PCA is a compression algorithm!

More generally, could use any kind of approximately invertible non-linear transform.

PCA axes



Non-linear representation

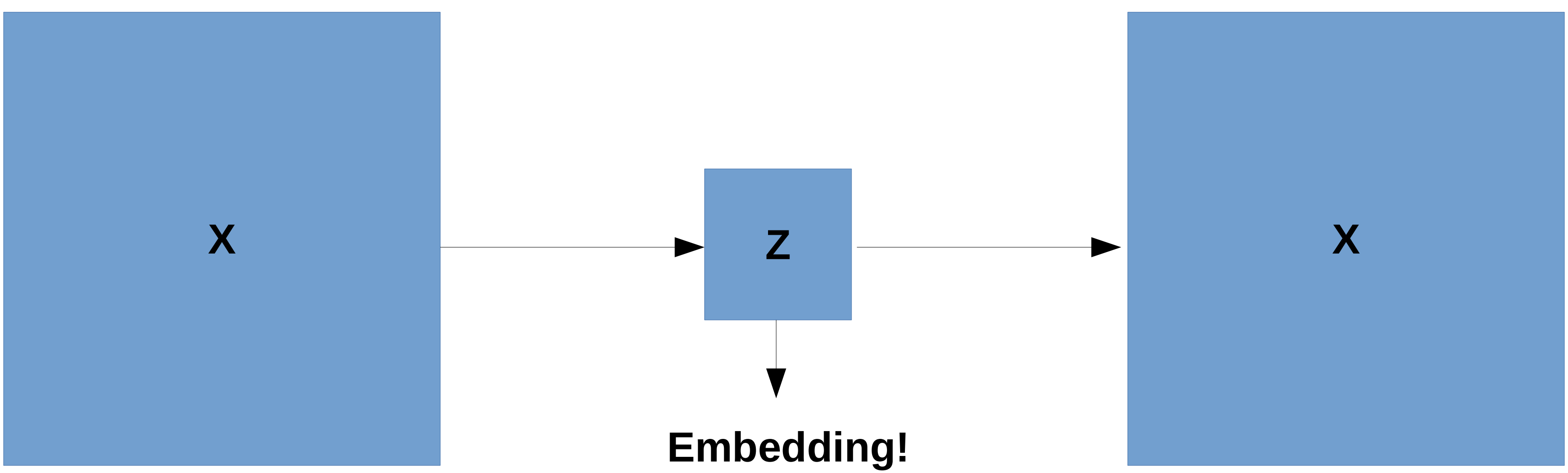


# Auto-encoders

Reconstruction task: Given **x**, predict **x**.

Trivial if no other constraints. p(x|x) = 1

But if we force it to use a lower-dimensional intermediate **latent** variable, can require compression:

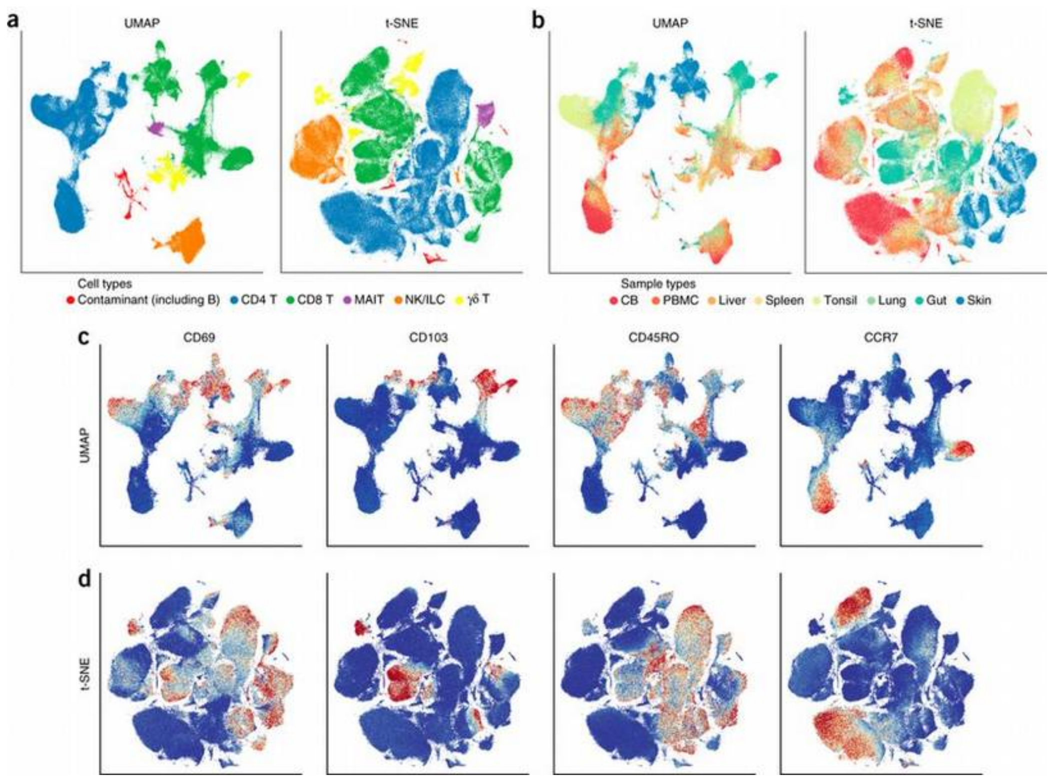


# Non-invertible embeddings

Rather than reconstructing the data, simply say that we want similar data-points to be embedded near each-other.

t-SNE

UMAP



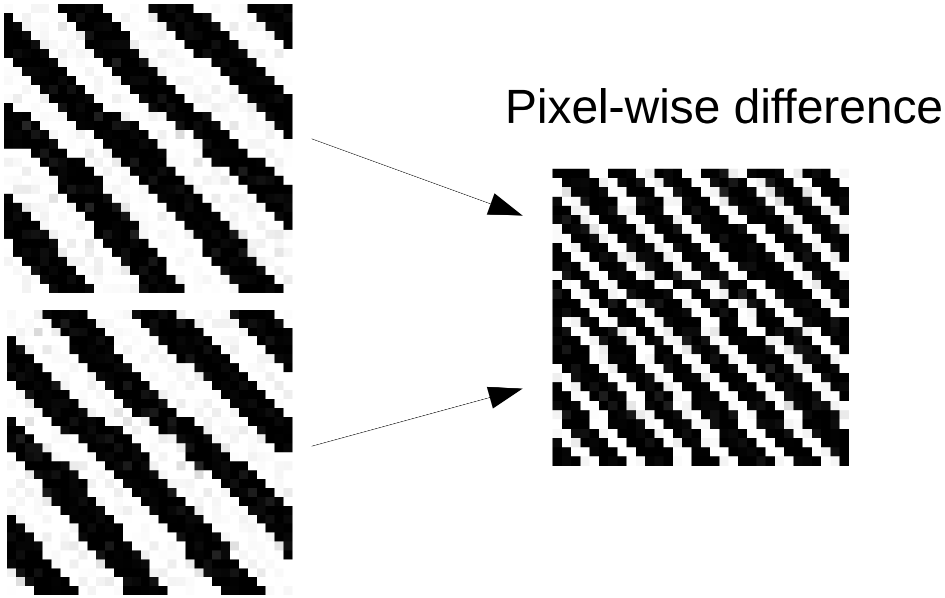

Becht et al, “Dimensionality reduction for visualizing single-cell data using UMAP”
https://www.nature.com/articles/nbt.4314

# Auto-encoders - problem

Auto-encoders try to compress the data, but with reference to the original representation.

But this may not correspond to physically meaningful senses of similarity.

Example: slightly offset textures are very dissimilar in pixel space, but are still ‘the same thing’.



# Similarity-based embedding

If we can provide an external reference for what **similarity** means, can use that to guide the geometry of the embedding.

Optimize the embedding function such that points that are more similar (in the externally provided sense) are closer to each-other in the embedding space.

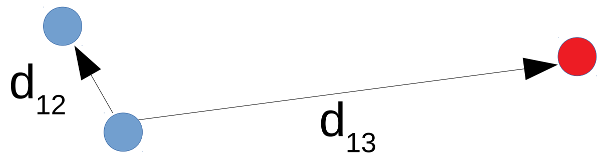
# Triplet Embedding

Pick two examples which are similar ($x_1$, $x_2$) and one which is different $x_3$

Embed examples using the same function (neural network):

$$x_1, x_2, x_3 \rightarrow z_1, z_2, z_3$$

Compare the $z_1$-$z_2$ distance to the $z_1$-$z_3$ distance.



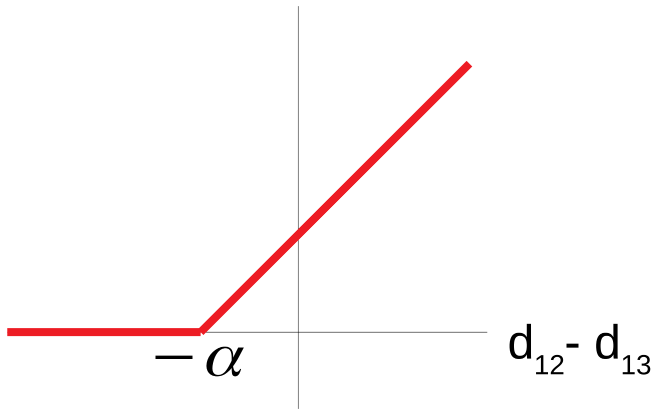

**Ask $d_{12}$ to be less than $d_{13}$**

# Triplet Embedding Loss

Potential problem: if we just ask E[$d_{13}$ – $d_{12}$] to be large, we can do this by having a few points run off to infinity.

Fix: If the difference is bigger than some cutoff value, we won't care about making it even bigger:

$$L_{triplet} = max\left(d_{12} - d_{13} + \alpha, 0\right)$$




(Hinge loss)

# Implementing Triplet Embedding with Convolutional Neural Networks

Start with standard classification architecture.

Make final layer linear rather than SoftMax

# of output neurons = embedding dimension

Form batches composed of sets of three images, two ‘similar’, one ‘different’. Pass all three through the network.

Compute $L_{triplet}$ and back-propagate.

# Interpreting Embeddings

What does it mean if two points are near each-other?

What do shapes in the embedding mean?

Are embedding structures reproducible?

Often no firm answers to these questions – exploratory, rather than explanatory.

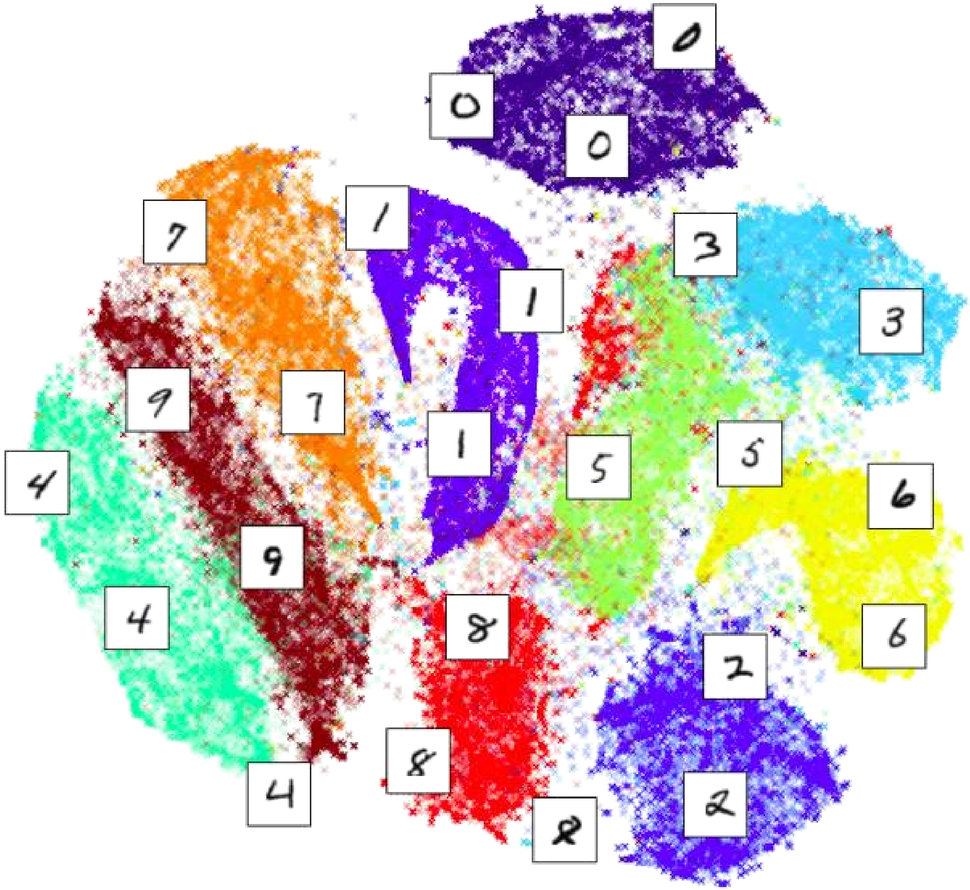
# Interpreting Embeddings

If you have labels, one way to test embeddings is:

- Train a classifier on the embedding coordinates and compare with a classifier on the original data.

If accuracies are similar: you successfully captured the relevant information.

Perform direct analysis (inspecting correlations) on the embedding coordinates.




https://nlml.github.io/in-raw-numpy/in-raw-numpy-t-sne/

# Interpreting Triplet Embeddings

In triplet embeddings, the distance in the embedding space can tell you about **ambiguities** in the data.

Points that **should** be far apart but are not (based on your similarity source) may indicate the existence of equivalency classes not captured by your sense of similarity.

Example: if similarity = geographical distance, then far away points that map to the same **z** may correspond to recurring terrain types.