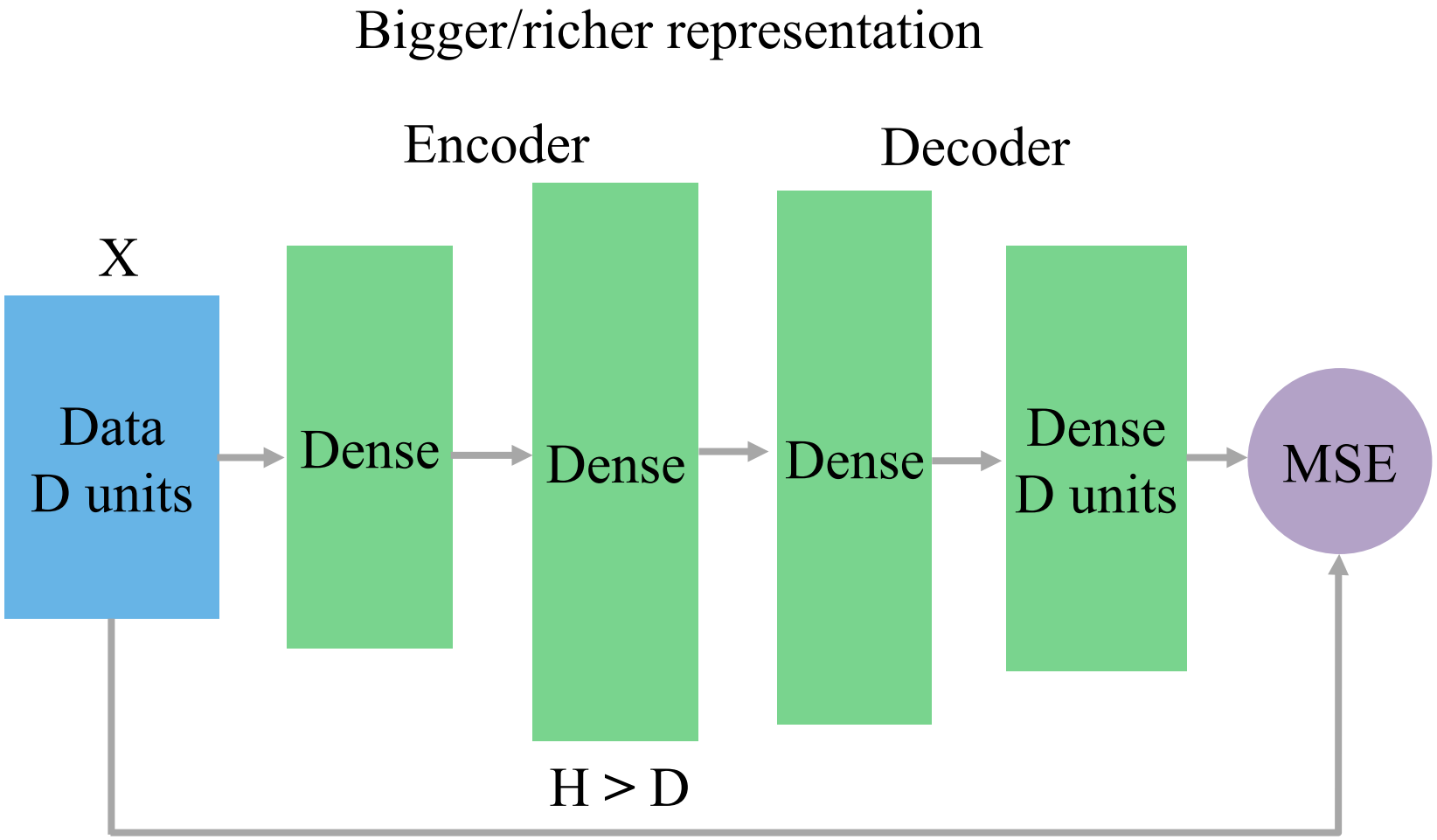
# Why do we ever need that?

if we develop a network that gets an image, encodes to some smaller format and then decodes it back, then we've made a compression algorithm!

- 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



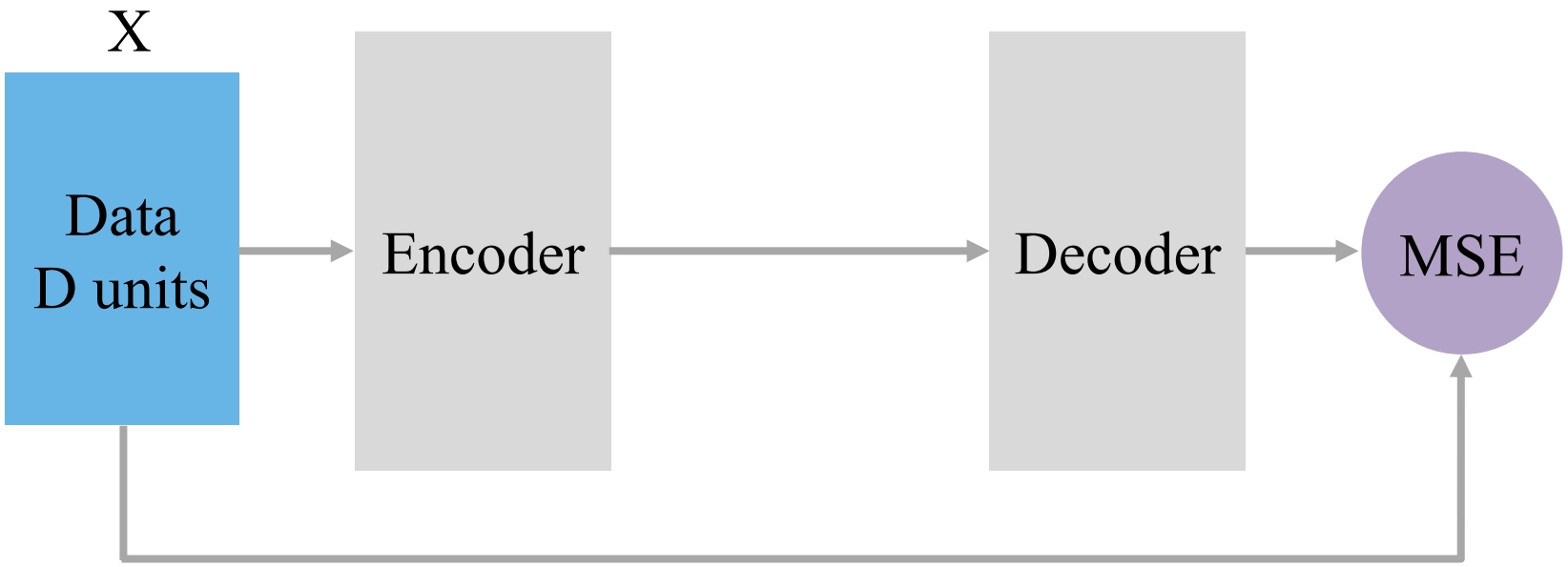

Something's wrong with this guy. **Ideas?**

# Expanding autoencoder

We don't want to learn the identity function, because no 'encoding' and 'decoding' will actually take place. The result would be useless if used in any sort of task.

Naive approach will learn identity function!

Gotta regularize!




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

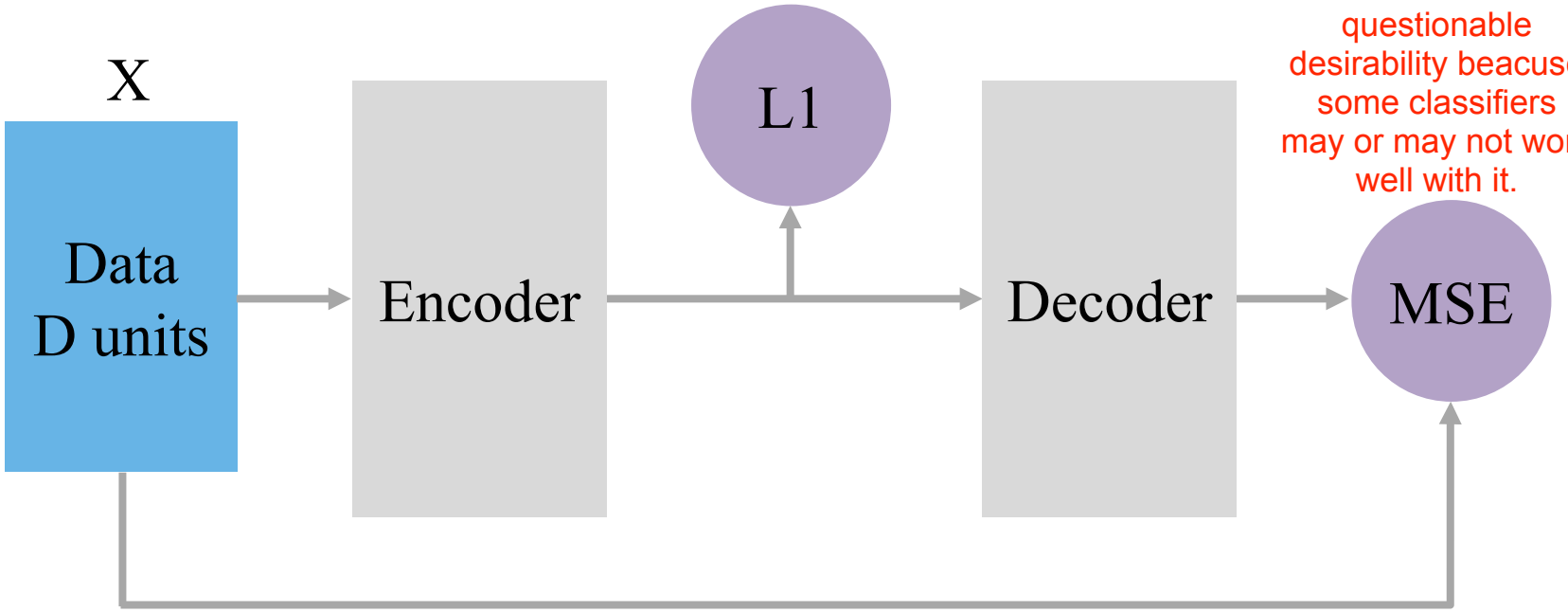
# Sparse autoencoder

we make the activations smaller.

Naive approach will learn identity function!

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

for many objects, most of the features will be 0. This is of questionable desirability beacuse some classifiers may or may not work well with it.

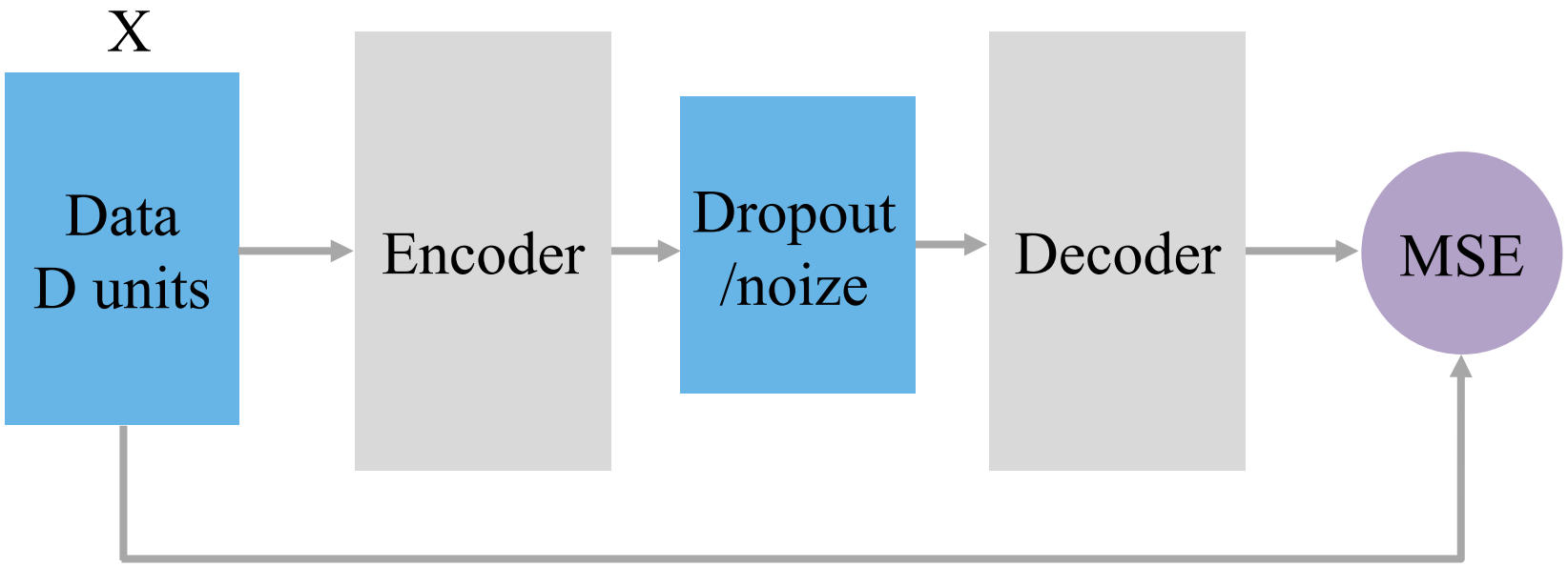



$$L = \|X - Dec(Enc(X))\| + \sum_i |Enc_i(X)|$$

# Redundant autoencoder

Naive approach will learn identity function!

Idea 2: noize/dropout, redundant code

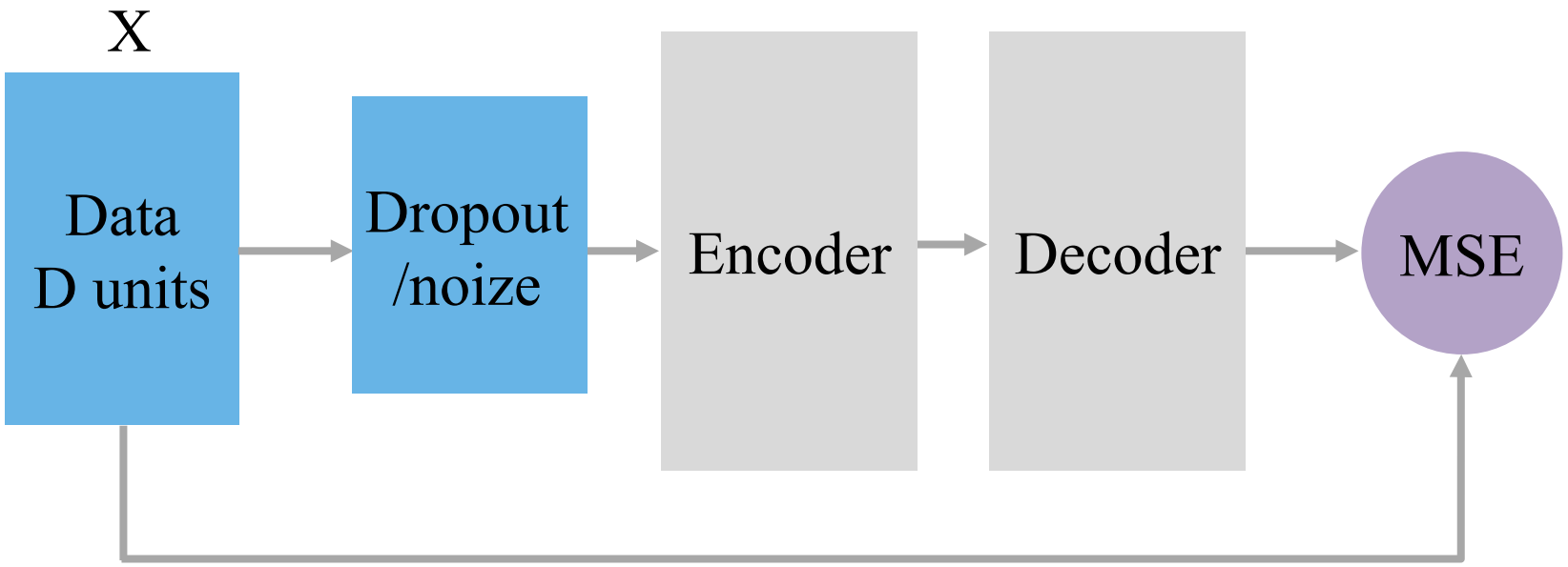



$$L = || X - Dec(Enc(Noise(X)) ||$$

# Denoizing autoencoder

Naive approach will learn identity function!

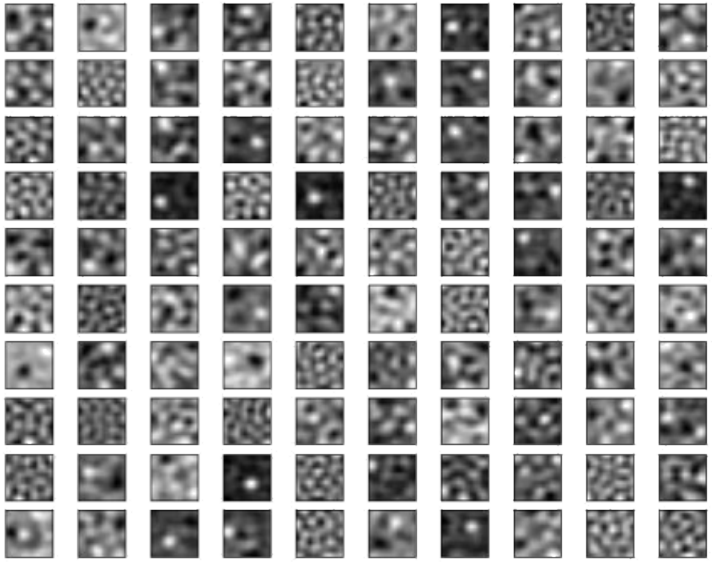
Idea 3: distort input, learn to fix distorsion



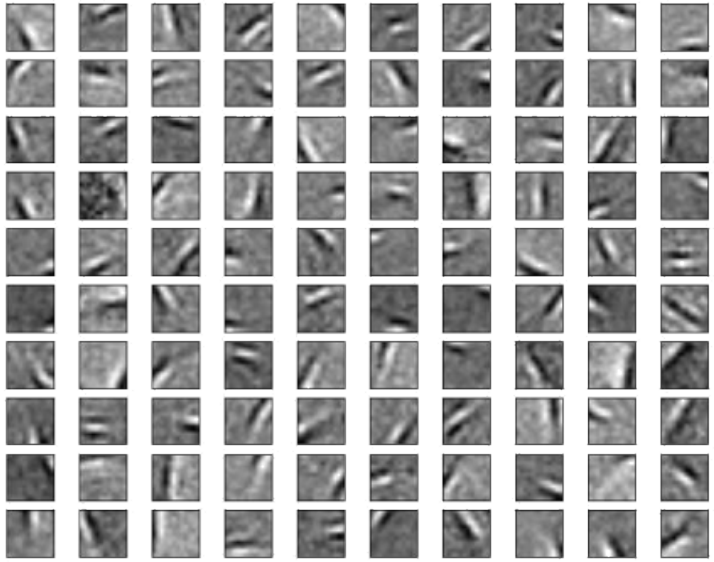

$$L = || X - Dec(Enc(Noise(X)) ||$$

# Sparse Vs Denoizing

Filter weights, 12x12 patches



Sparse AE



Denoizing AE

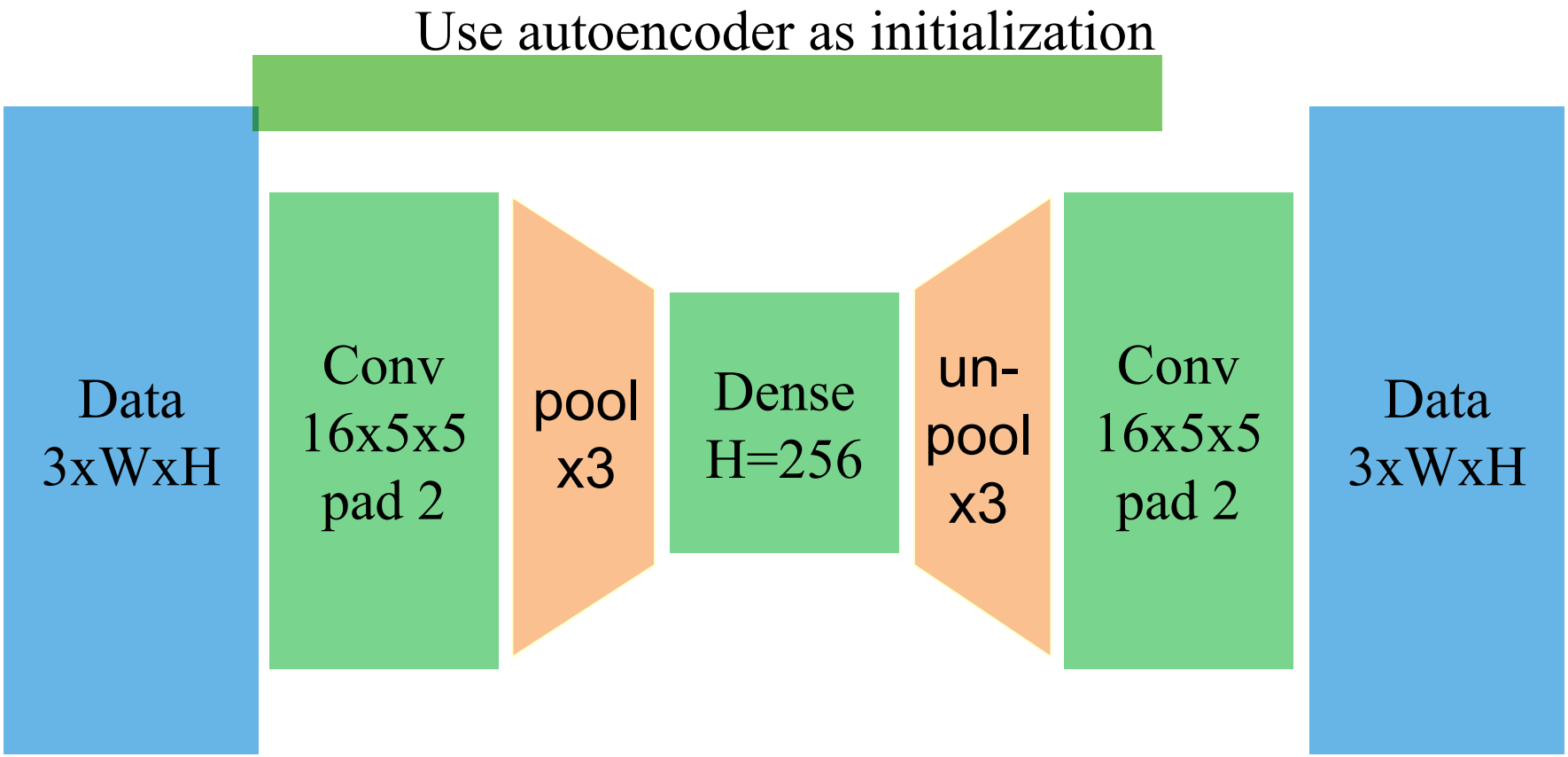
These images are actually clueless :)

# 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

we're going to focus on pretraining now.

# Pretraining

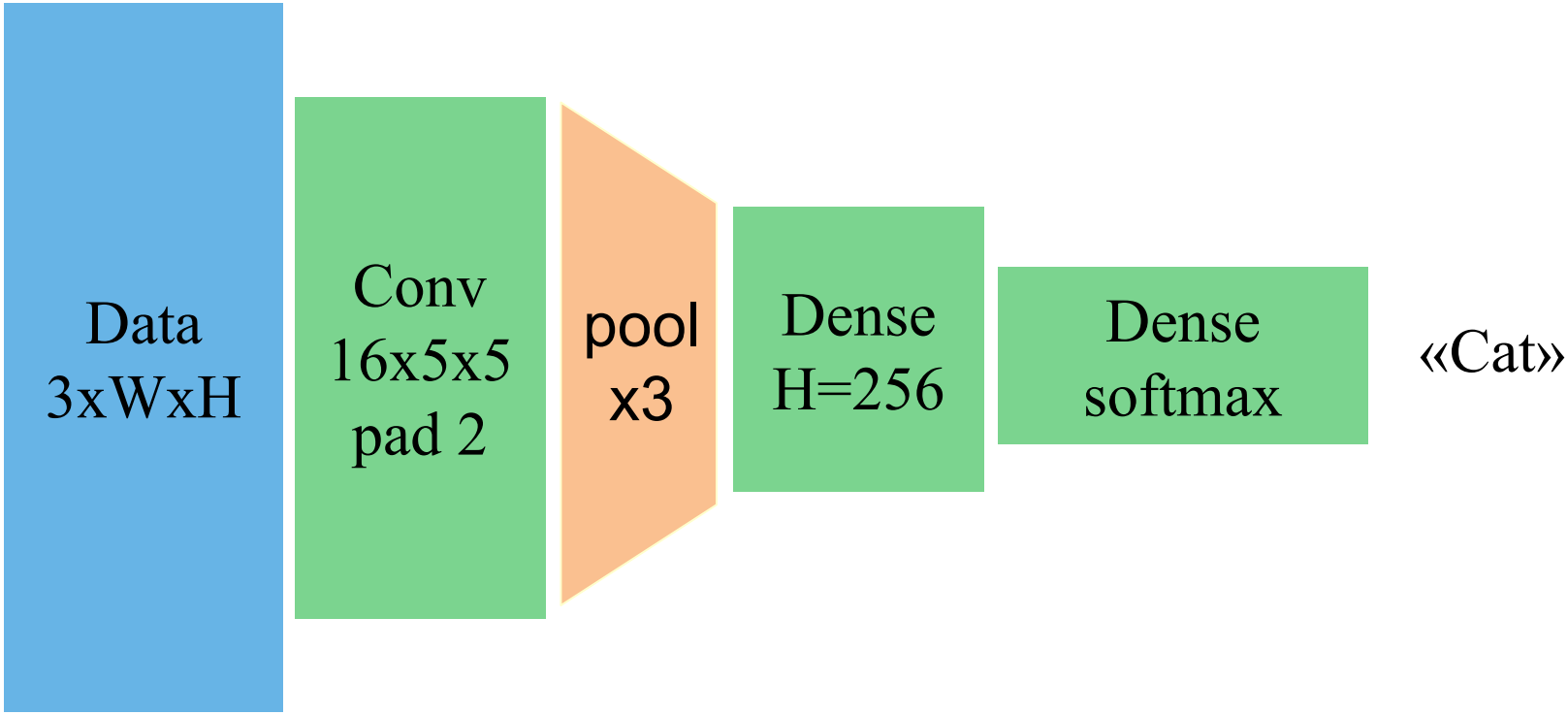



autoencoder for initialization is useful if we don't have a lot of data.
We learn the encodings and thus the 'high level' representations of the image quickly by using the first half ('encoding' part)..
The other alternative would be to use a pretrained model, or hand crafted features (even worse)

From the above autoencoder, we do (..next page..)

# 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

if yiour domain is  near to a 'popular problem'. Otherwise,

Unsupervised pretraining (autoencoder)

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