PMLDL. Project Presentation

Anna Boronina

The idea

~~GANs for point clouds generation~~

~~VAE for point clouds generation~~

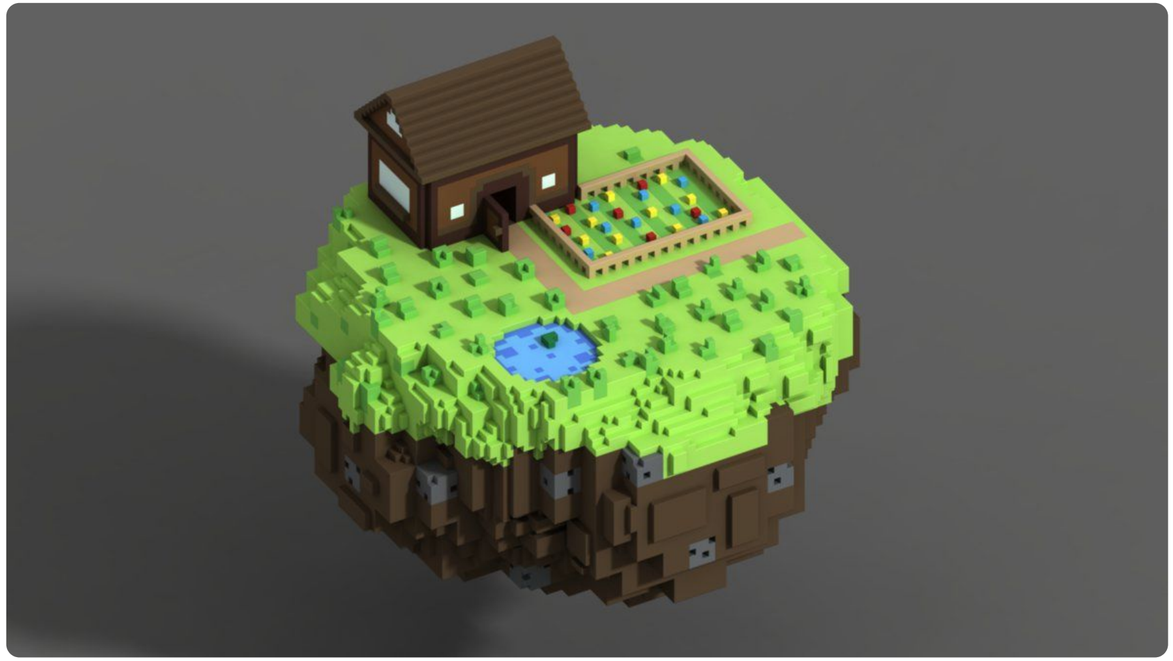
AE for point clouds

Voxels and Their Problem

- Structured
- You can apply 3D convolution

BUT

- Too large to work with
- Imagine 64x64x64 model - 262144 **voxels**

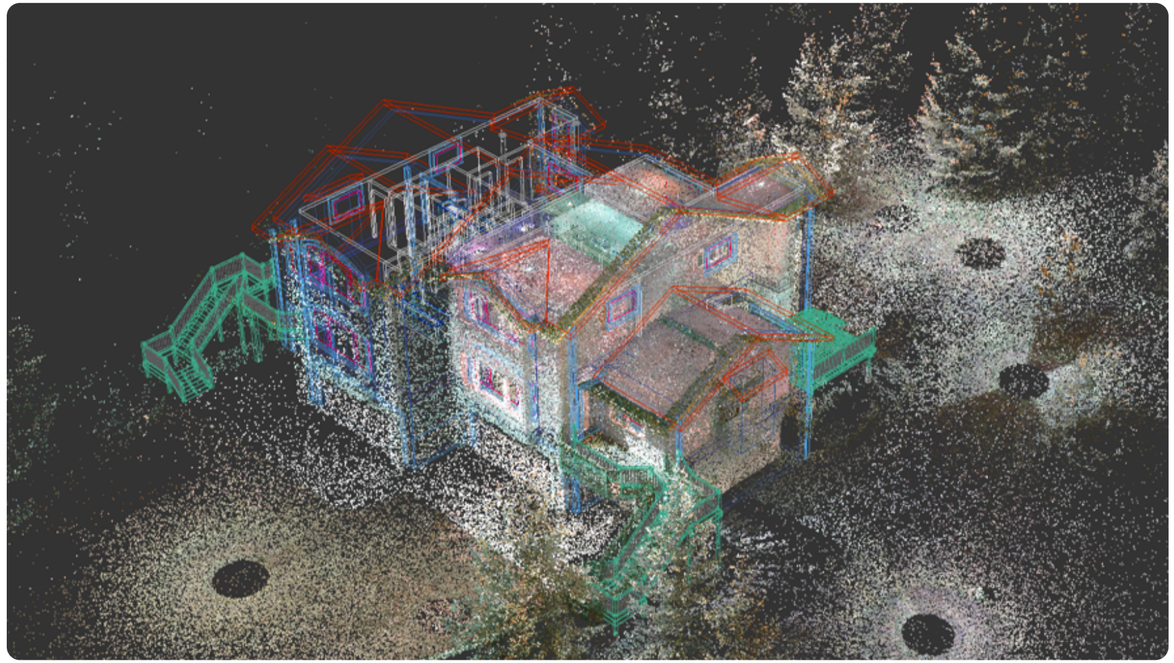


Point Clouds and Their Problem

- You can sample N points
- Lightweighted

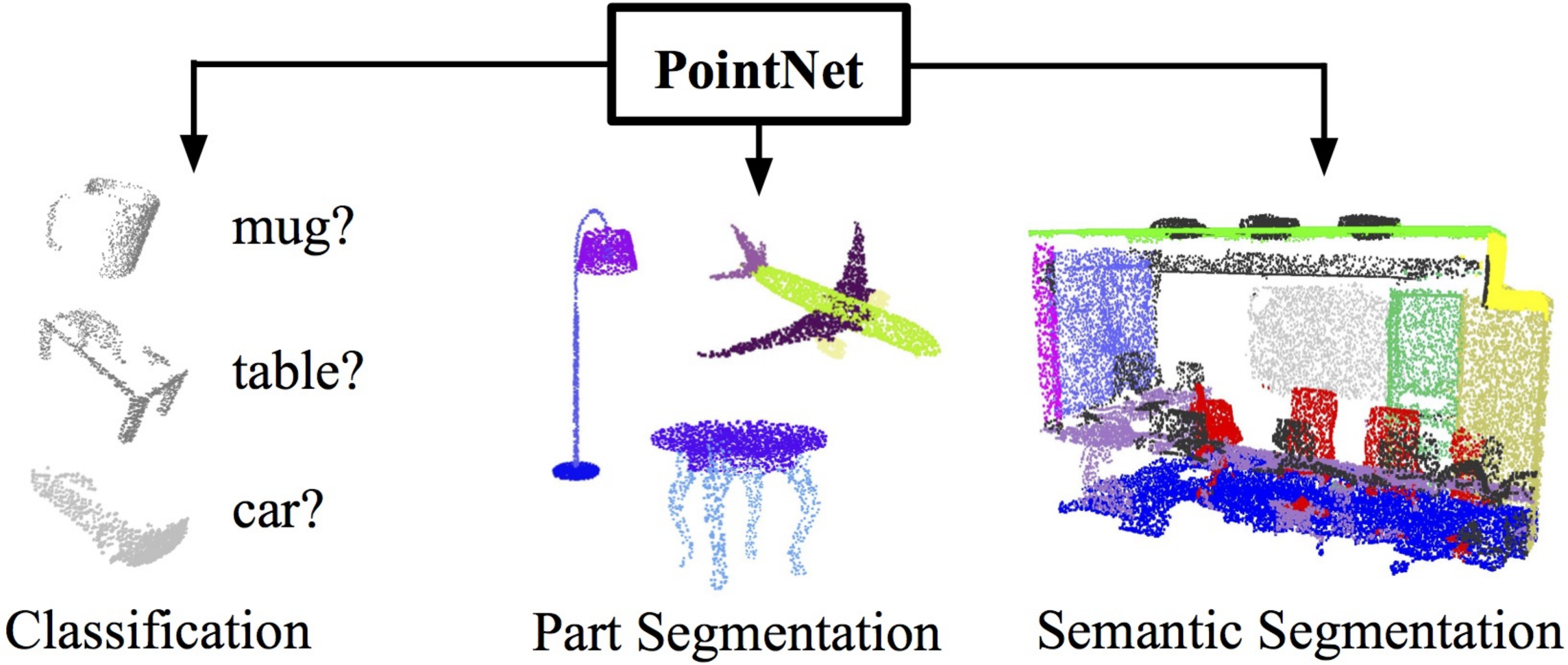
BUT

- Unstructured
- You cannot apply 3D convolution

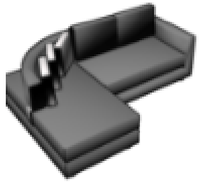


PointNet

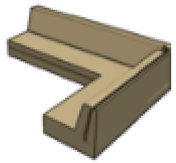
Permutation and rotation invariant!

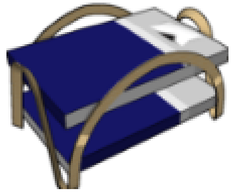


The dataset


couch

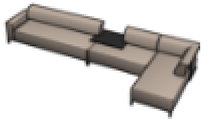

hammock


sofa

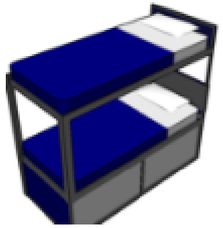

berth


L-shaped couch

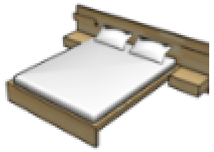

couch


sofa

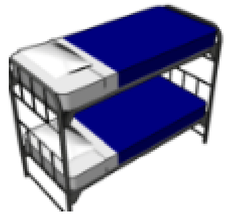

bunk bed

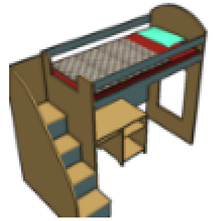

bunk bed


berth

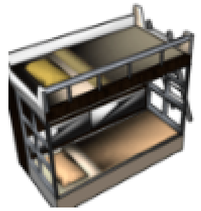

king size beds


L-shaped couch


bunk bed

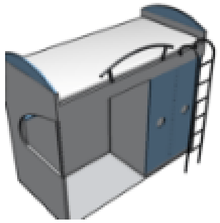

bunk

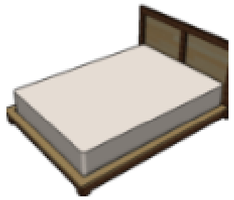

sofa

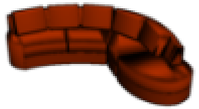

berth

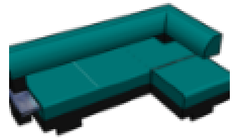

L-shaped couch


hospital bed


berth


headboard beds


couch

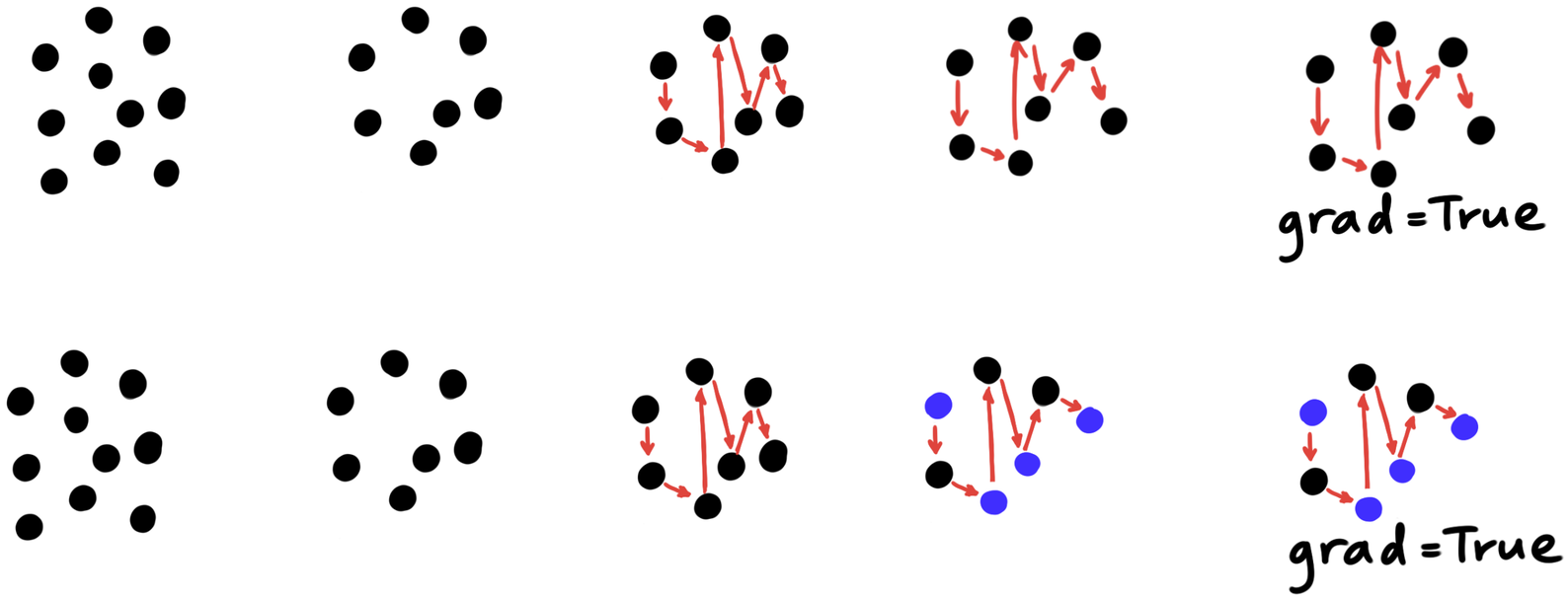

couch


bunk bed

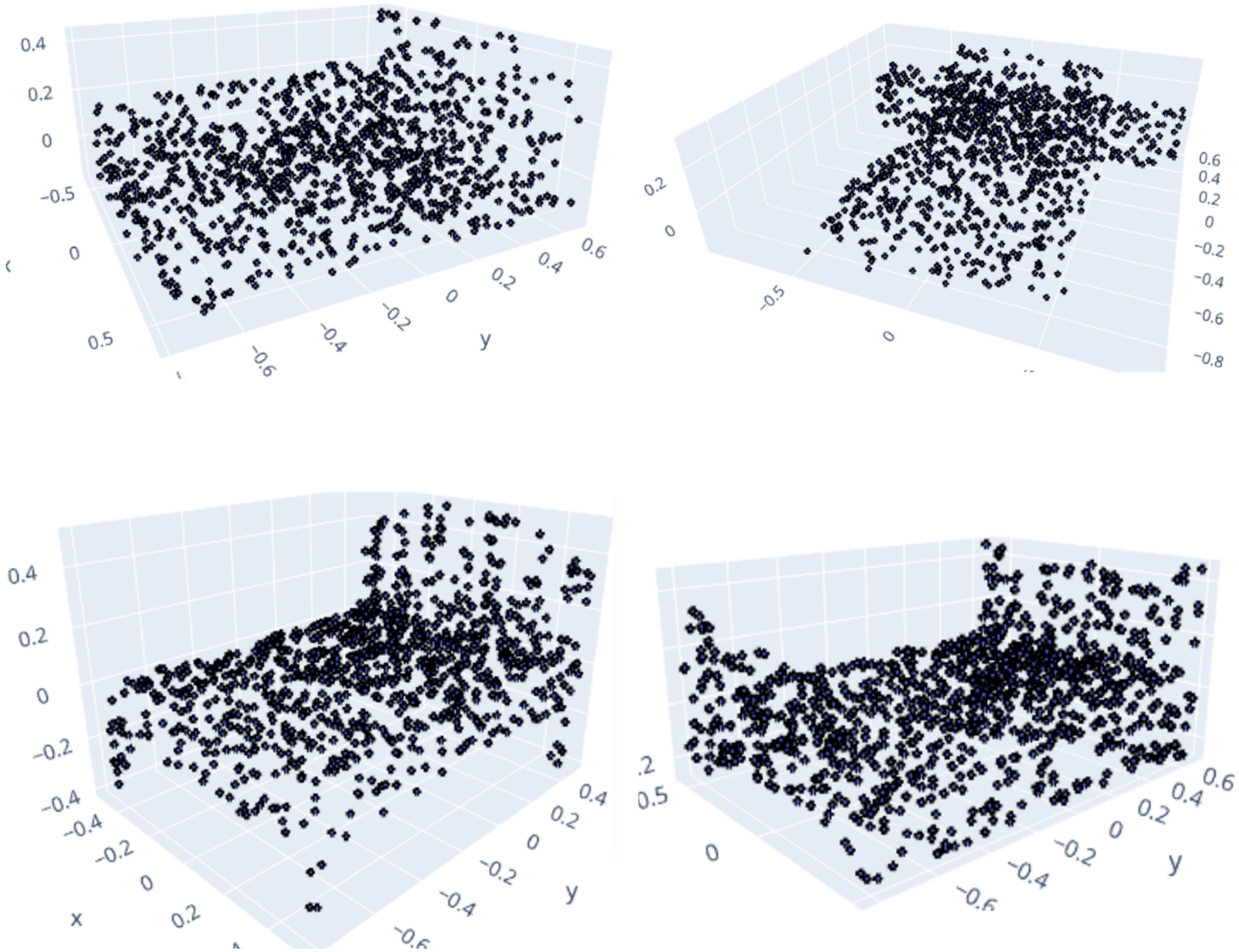

hospital bed

Data Augmentation

1. `PointSampler`
2. `ToSorted`
3. `Normalize`
4. `RandomNoise`
5. `ToTensor` (obviously)

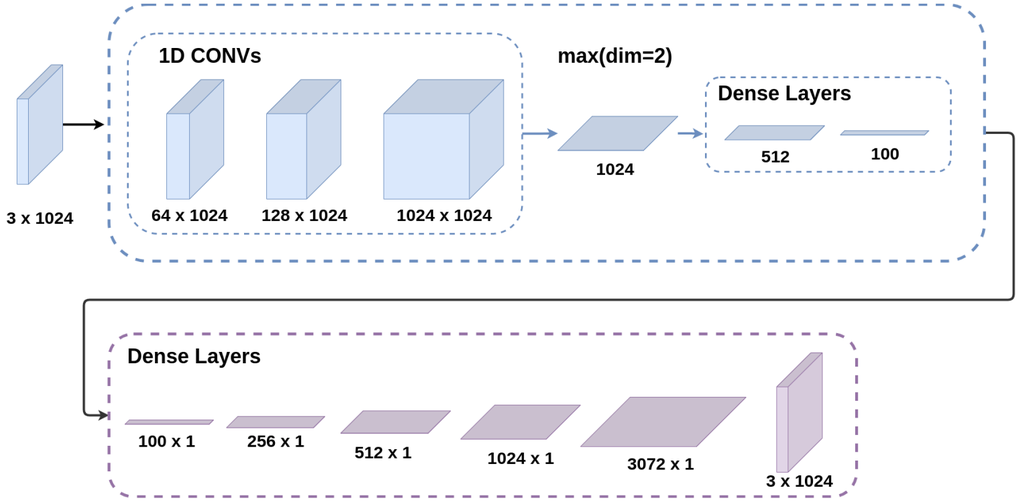


After Augmentation

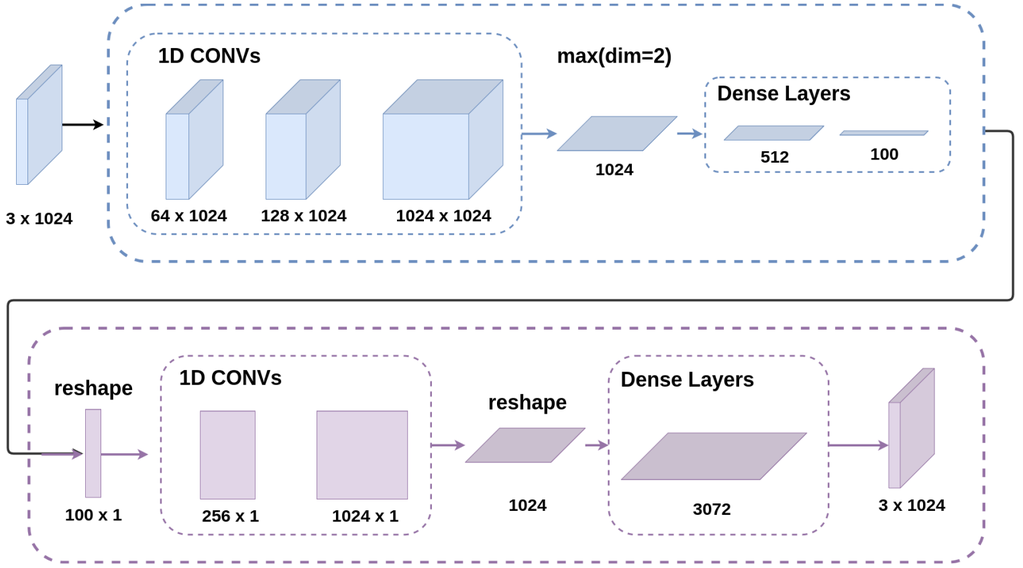


Architectures

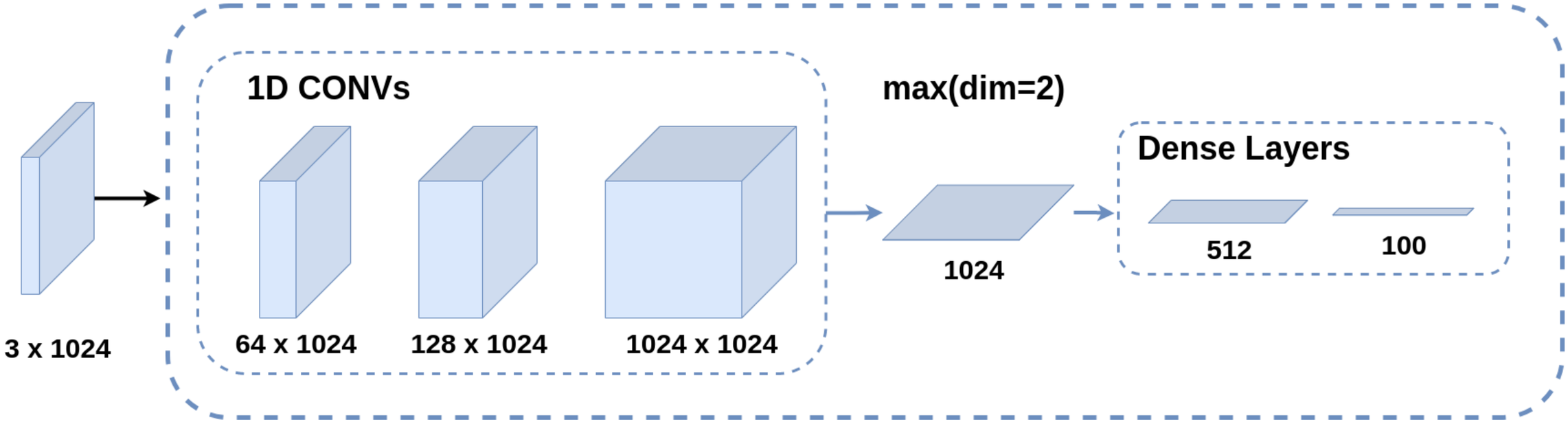
Encoder + Dense Decoder



Encoder + Convolutional Decoder



Encoder


1D CONVs
max(dim=2)
Dense Layers
3 x 1024
64 x 1024
128 x 1024
1024 x 1024
1024
512
100

Intermediate Recap :)

1. We saw the encoder

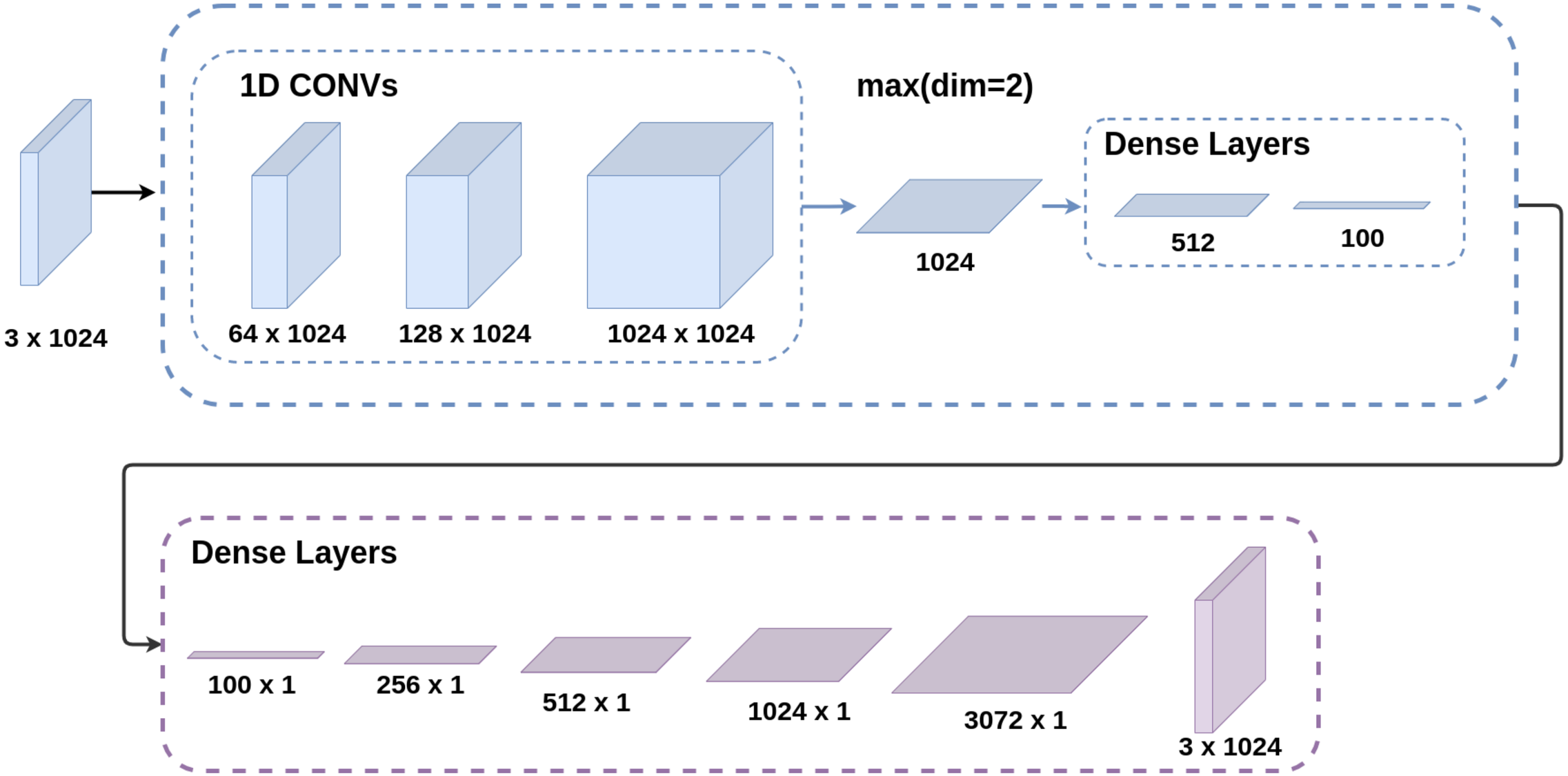
2. Two augmentations

    - fully static
    - static + dynamic

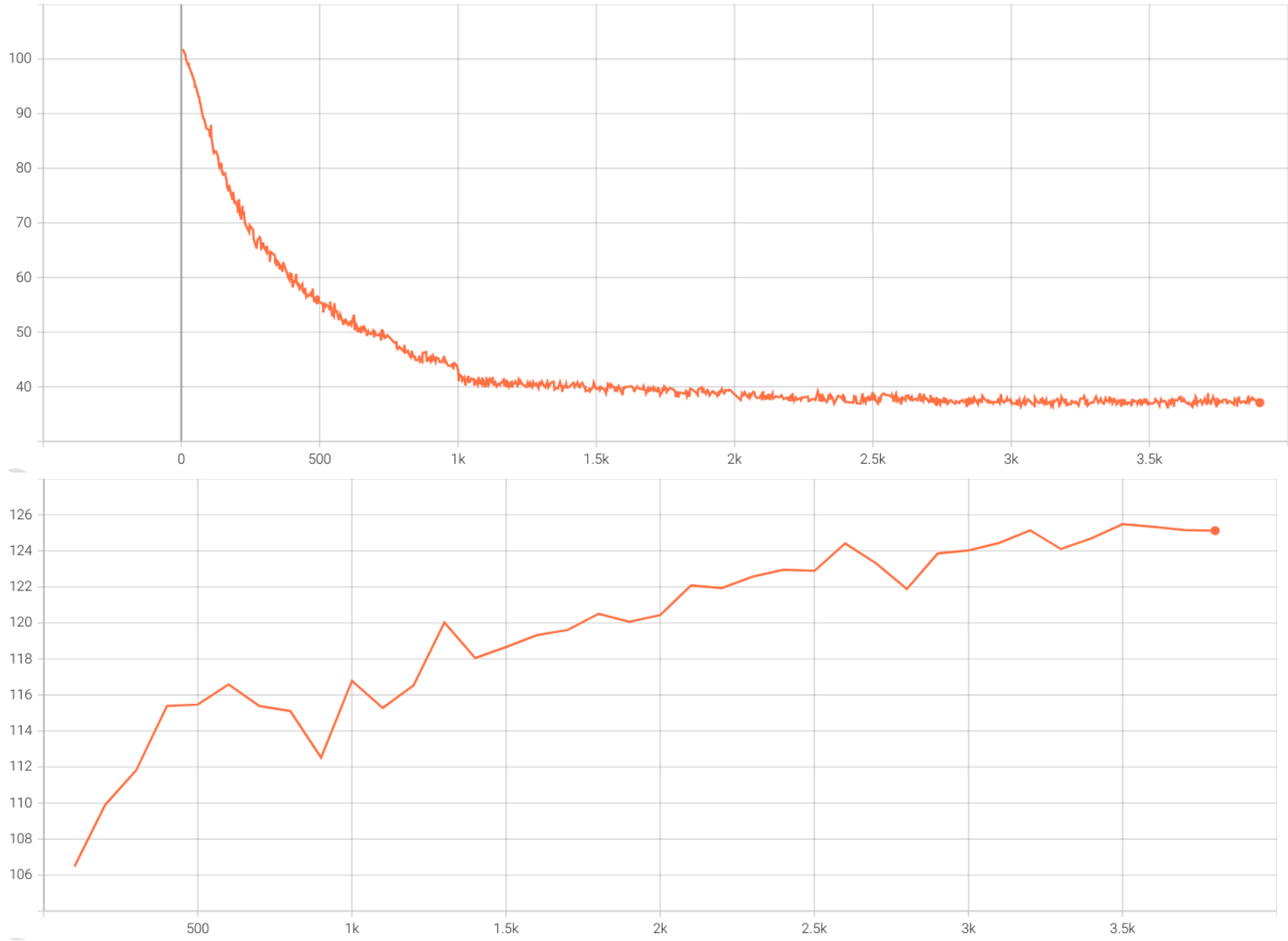
3. Two architectures

    - dense decoder
    - convolutional decoder

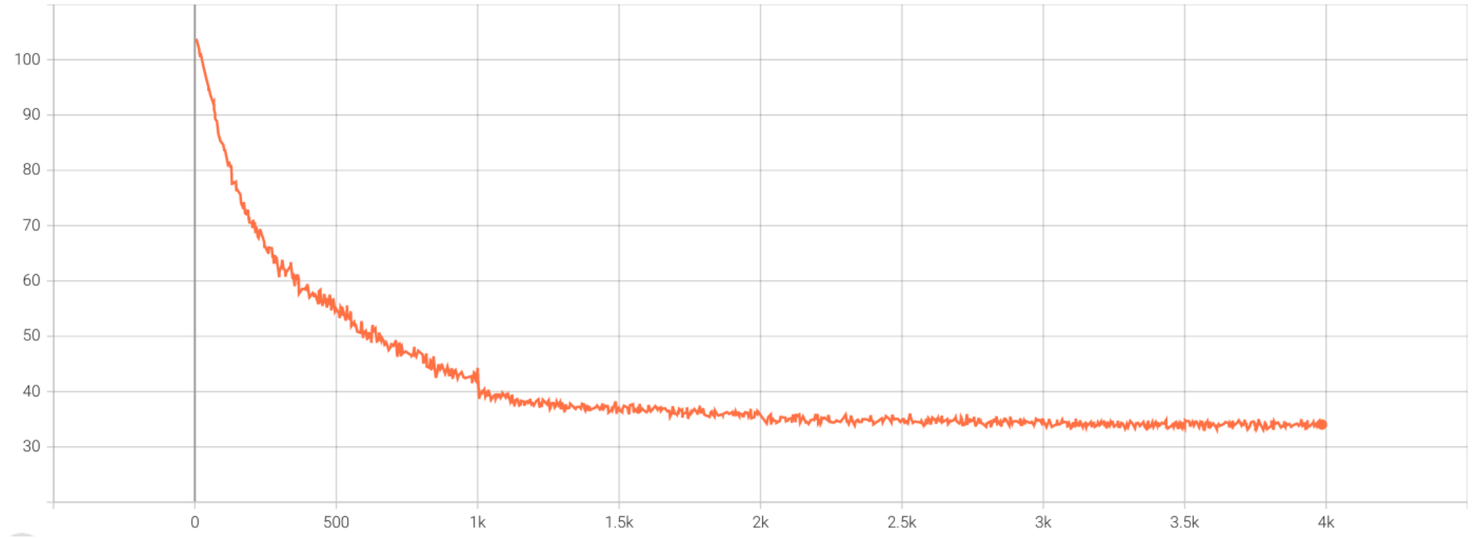
Dense Decoder

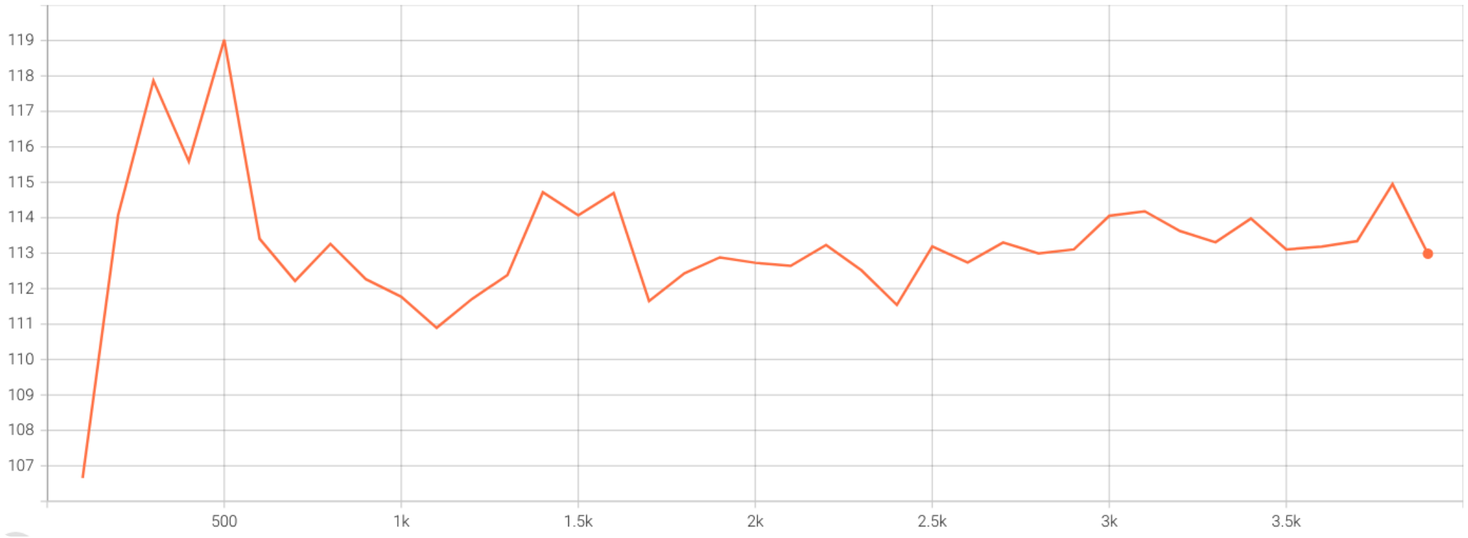


Dense Decoder. Static augmentation



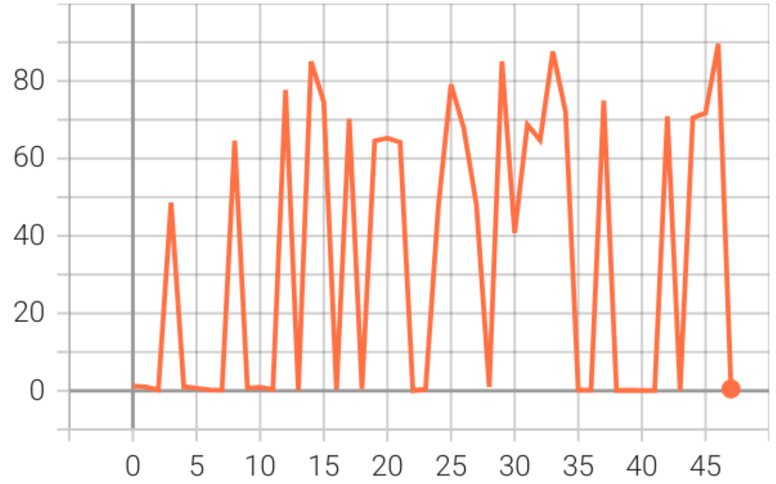
Dense Decoder. Dynamic augmentation

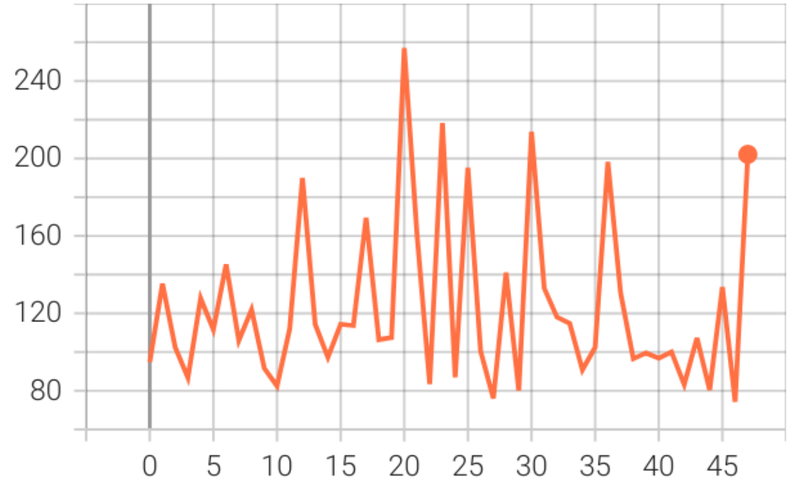




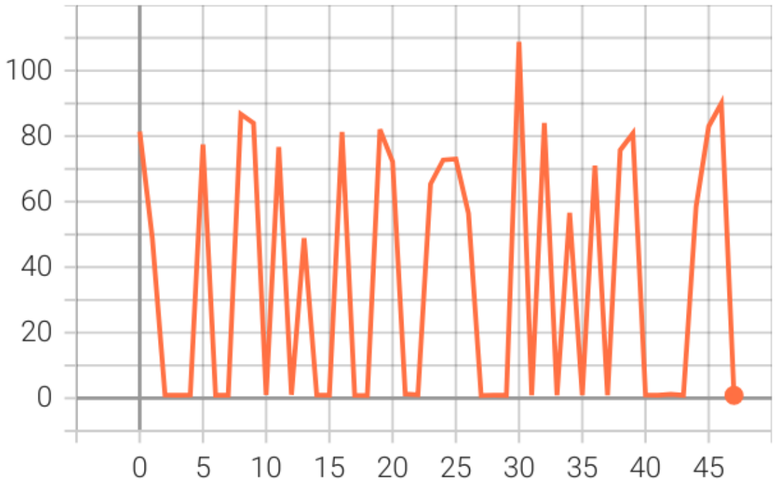
Dense Decoder. Comparison

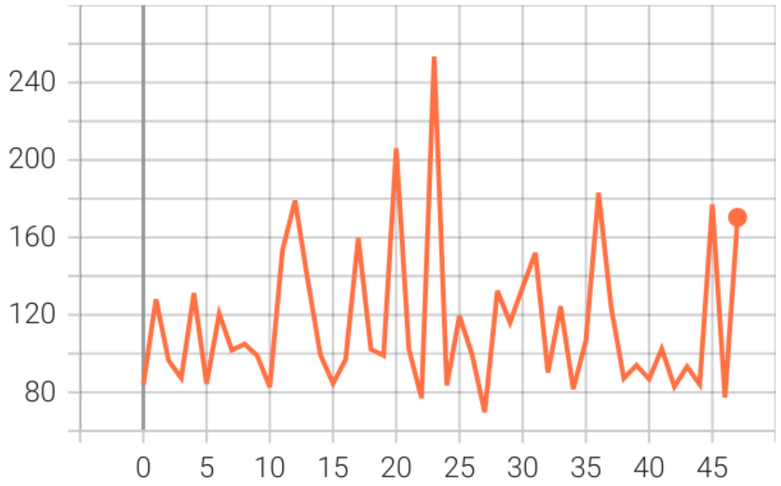
Static Augmentation: train and valid batches



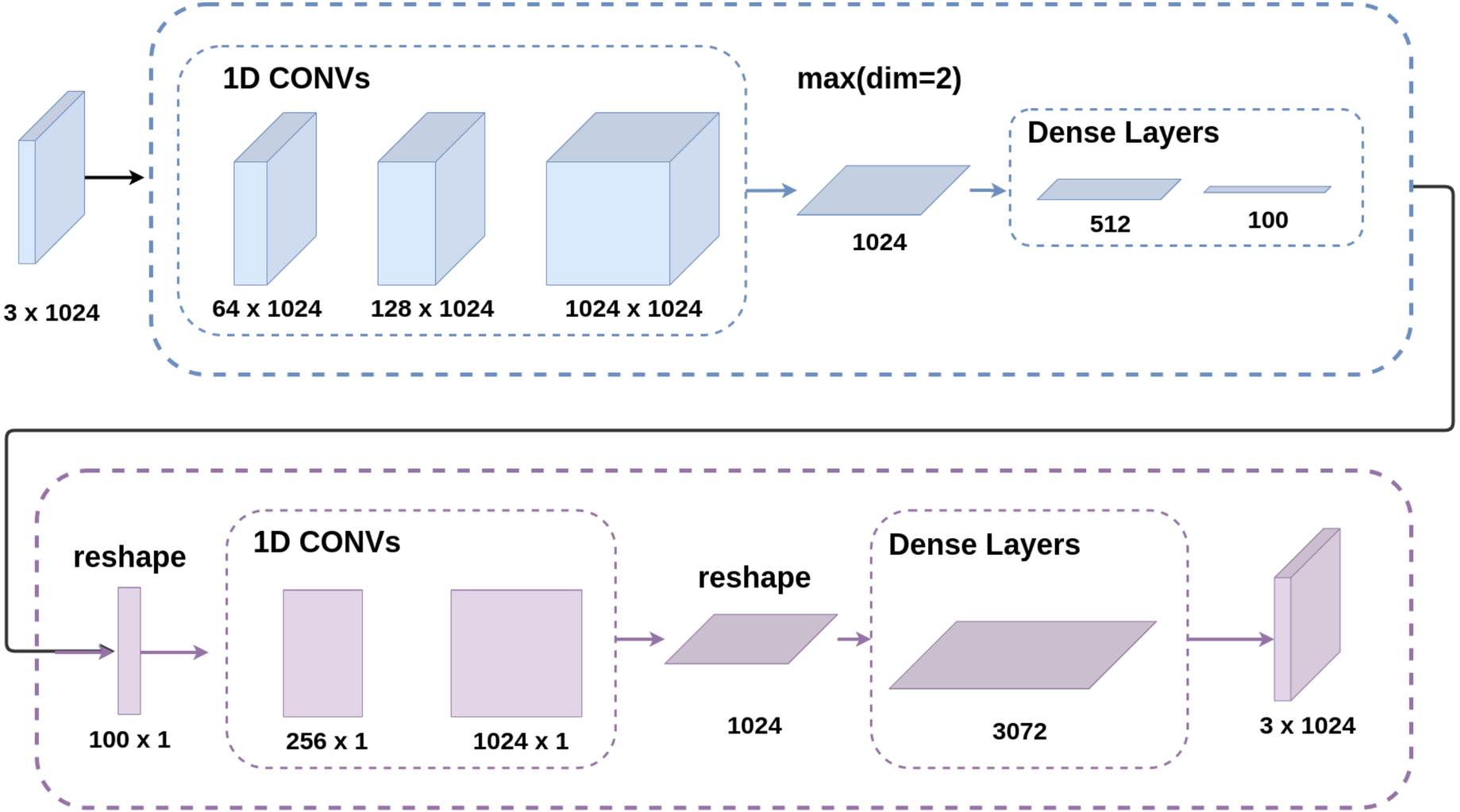


Dynamic Augmentation: train and valid batches

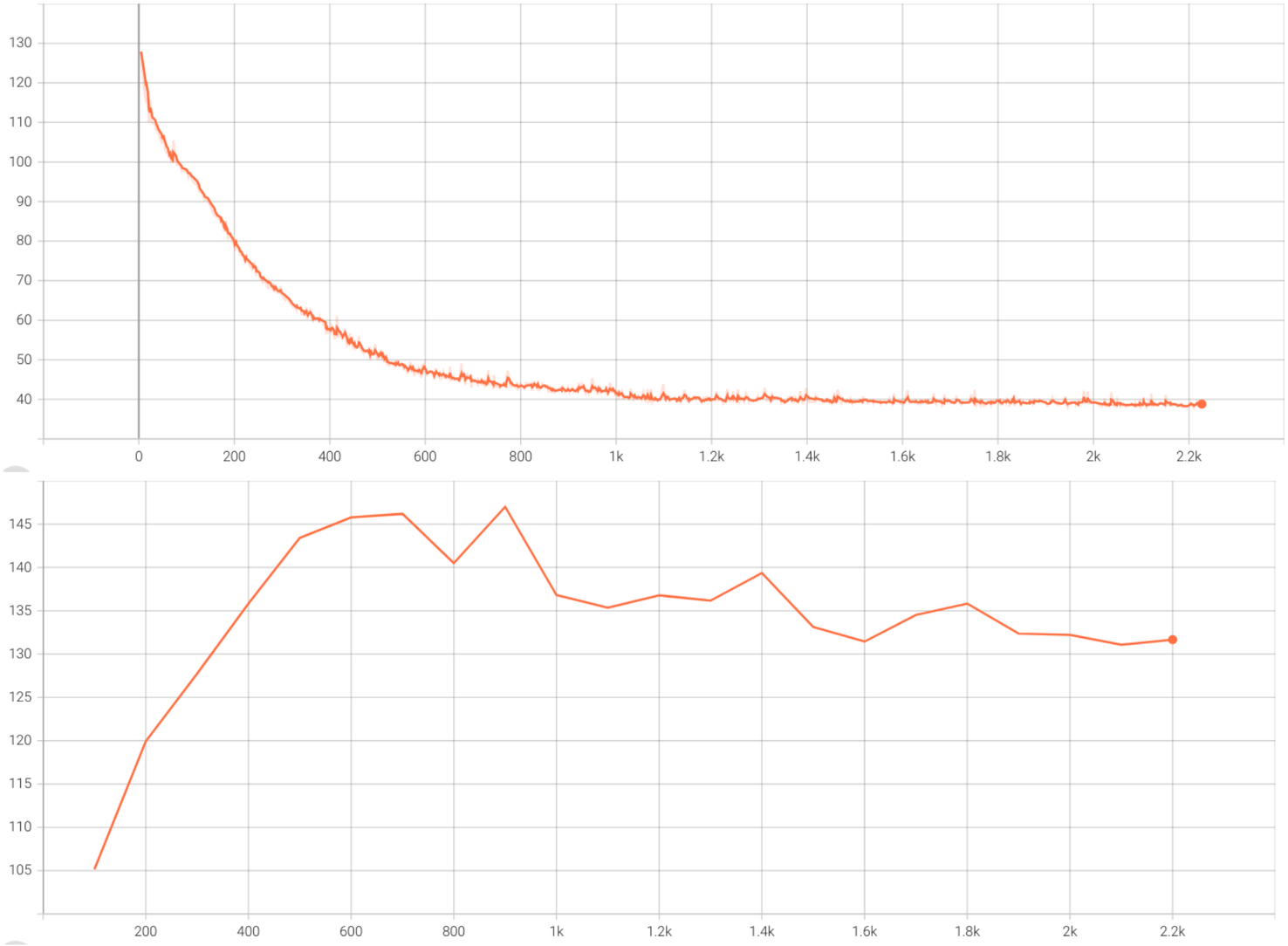




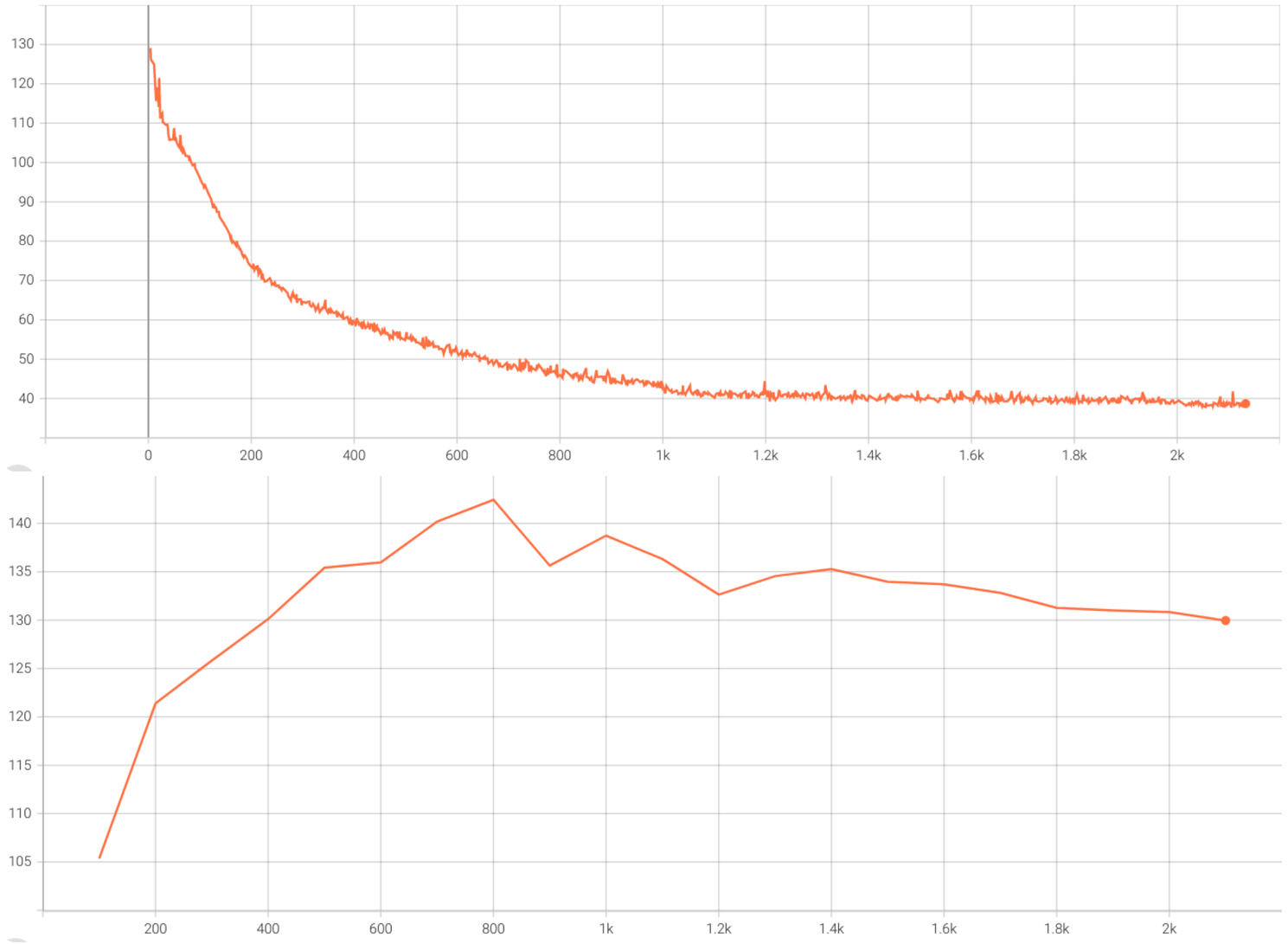
Convolutional Decoder



Convolutional Decoder. Static augmentation

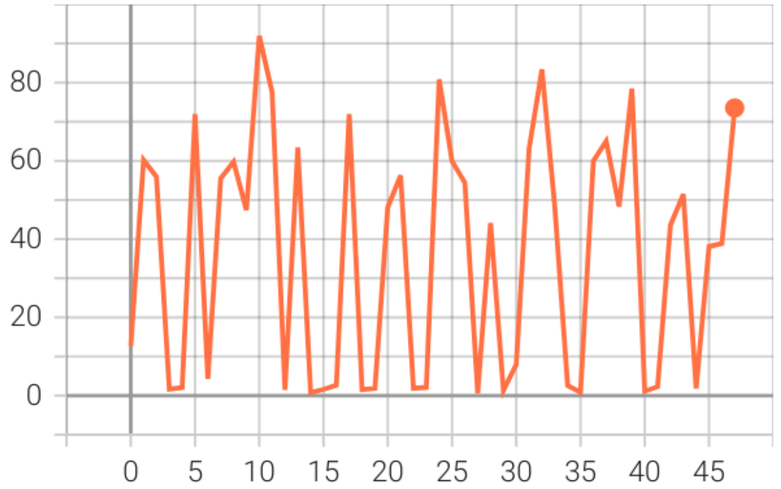


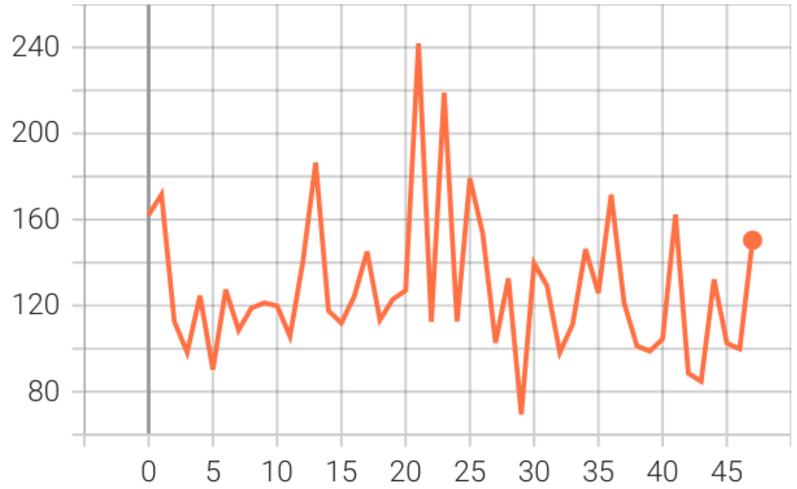
Convolutional Decoder. Dynamic augmentation



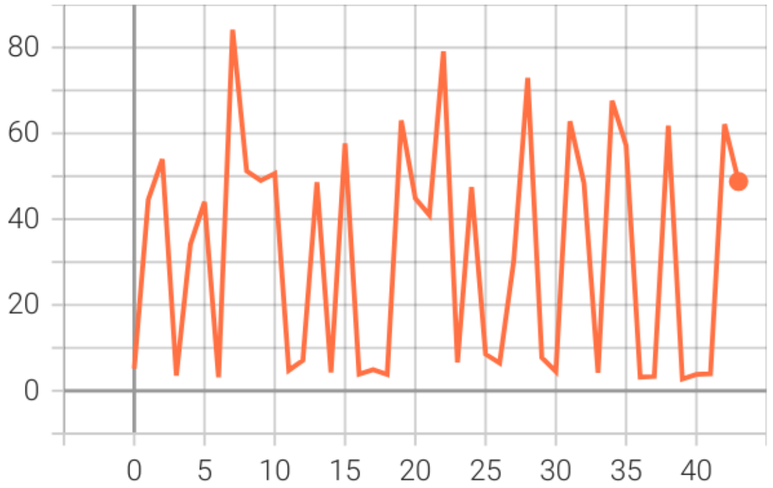
Convolutional Decoder. Comparison

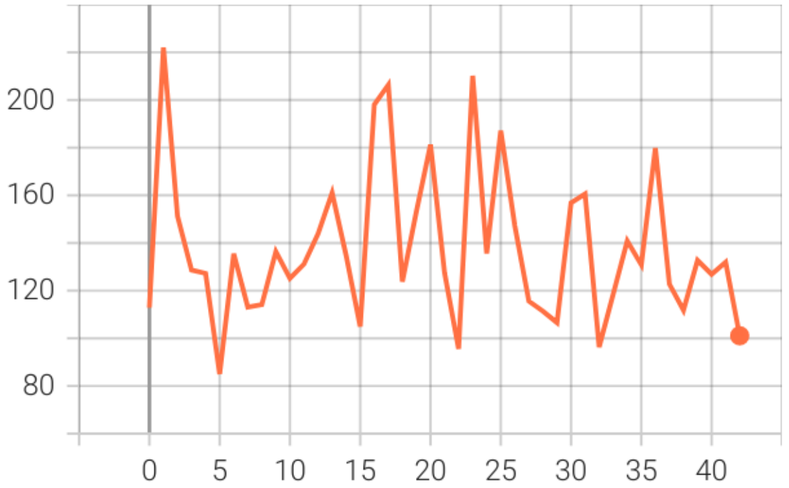
Static Augmentation: train and valid batches





Dynamic Augmentation: train and valid batches



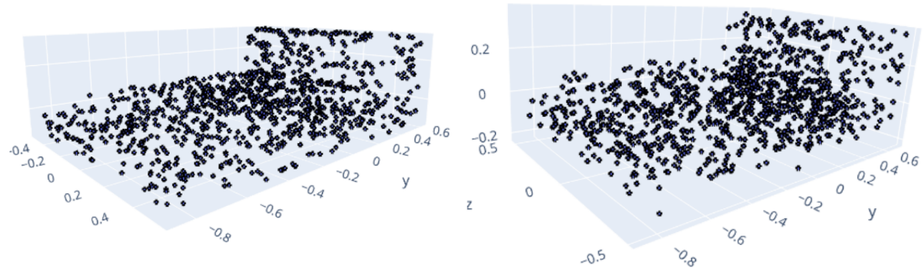


Main points

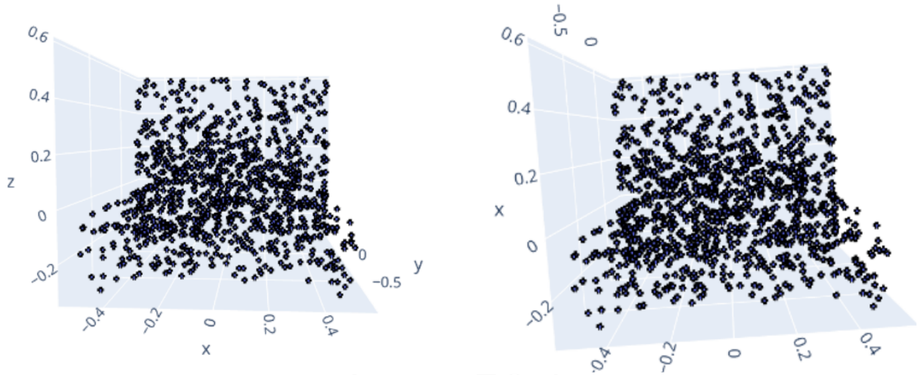
1. Purely static augmentation is evil (especially when there is no dropout)
2. Use 1D convolution to increase the number of dimensions
3. Dense and Convolutional Ecnoder have nearly the same performance on dynamically augmented data

Results. Train Set

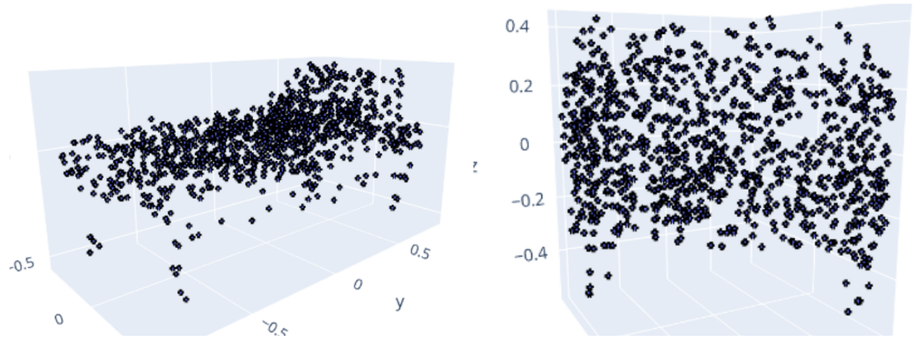
Loss: 2.4



Loss: 4.3

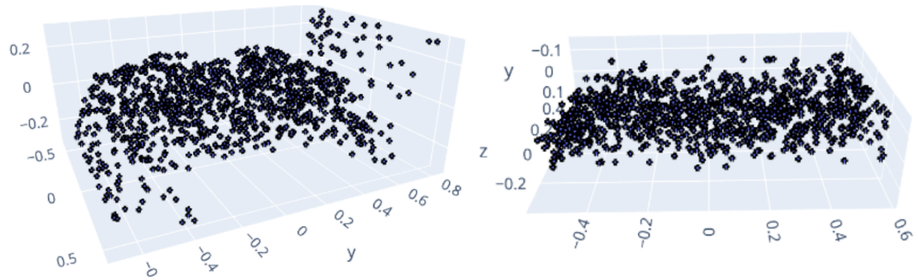


Loss: 52.1

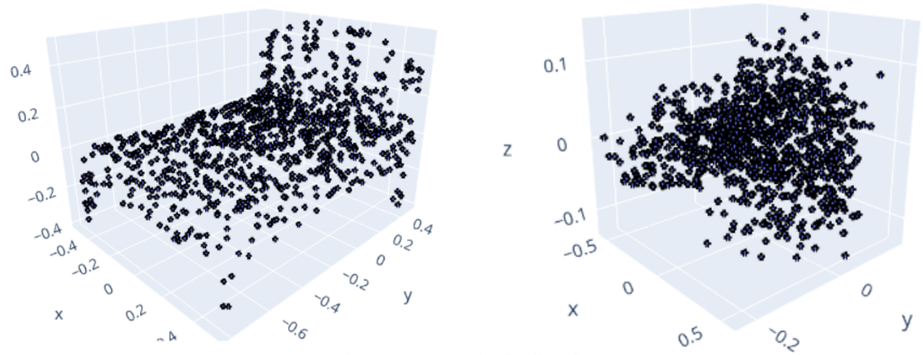


Results. Valid Set

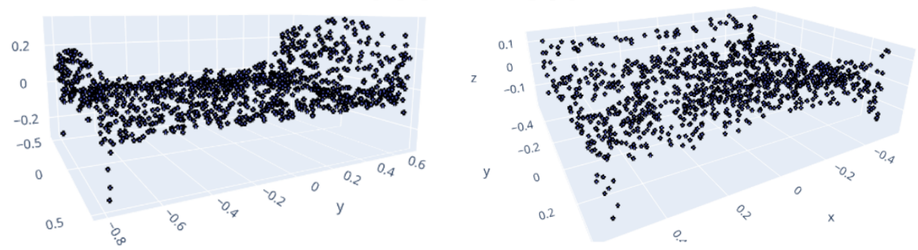
Loss: 80.1



Loss: 84.9



Loss: 128.0



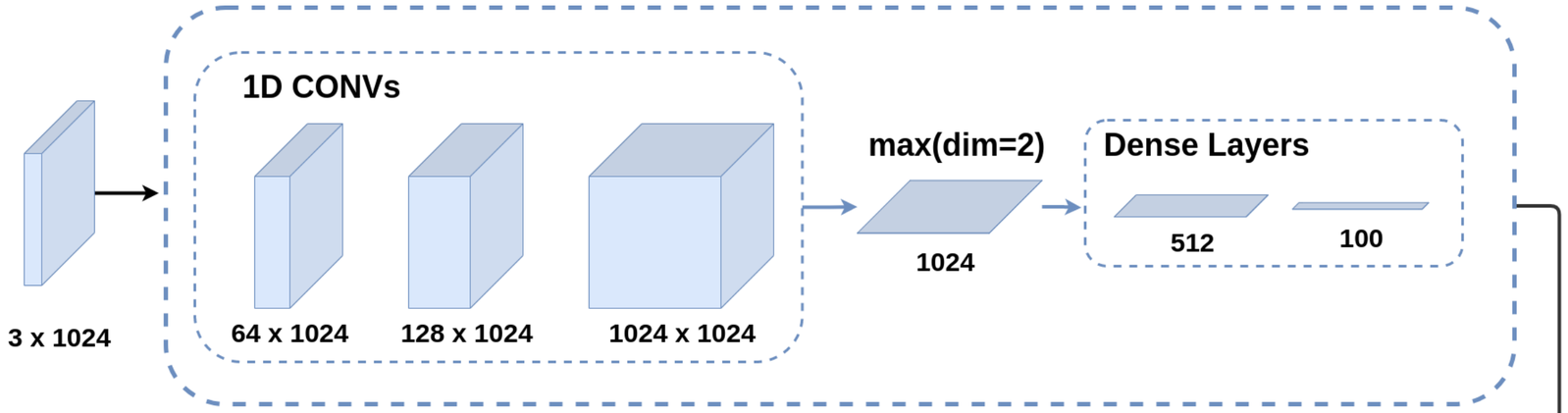
Loss Function: Chamfer Distance

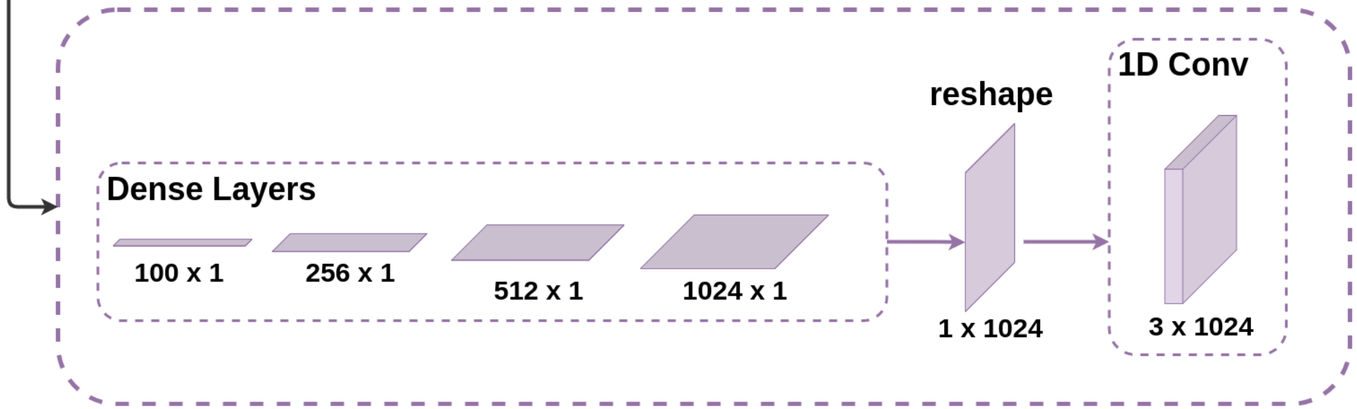
$$CD(U,V) = \Sigma_{v \in V} ||v - u||_2^2 + \Sigma_{u \in U} ||u - v||_2^2$$

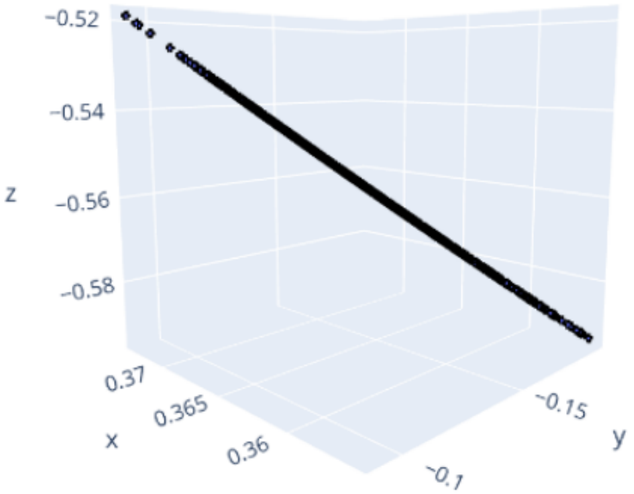
A discovery

YAY: use 1D convolution to increase the number of points

NAY: do not use it to increase the number of dimensions







Possible Further Work

1. Train for longer
2. Graph Neural Network
3. Variational Autoencoder
4. Change size of latent space
5. Use this architecture to train a GAN

Variational Autoencoder