TOKENCOMPOSE: Text-to-Image Diffusion with Token-level Supervision

Zirui Wang, Zhizhou Sha, Zheng Ding, Yilin Wang, Zhuowen Tu

Mathis Scheffler MVA 2024-2025

Latent diffusion

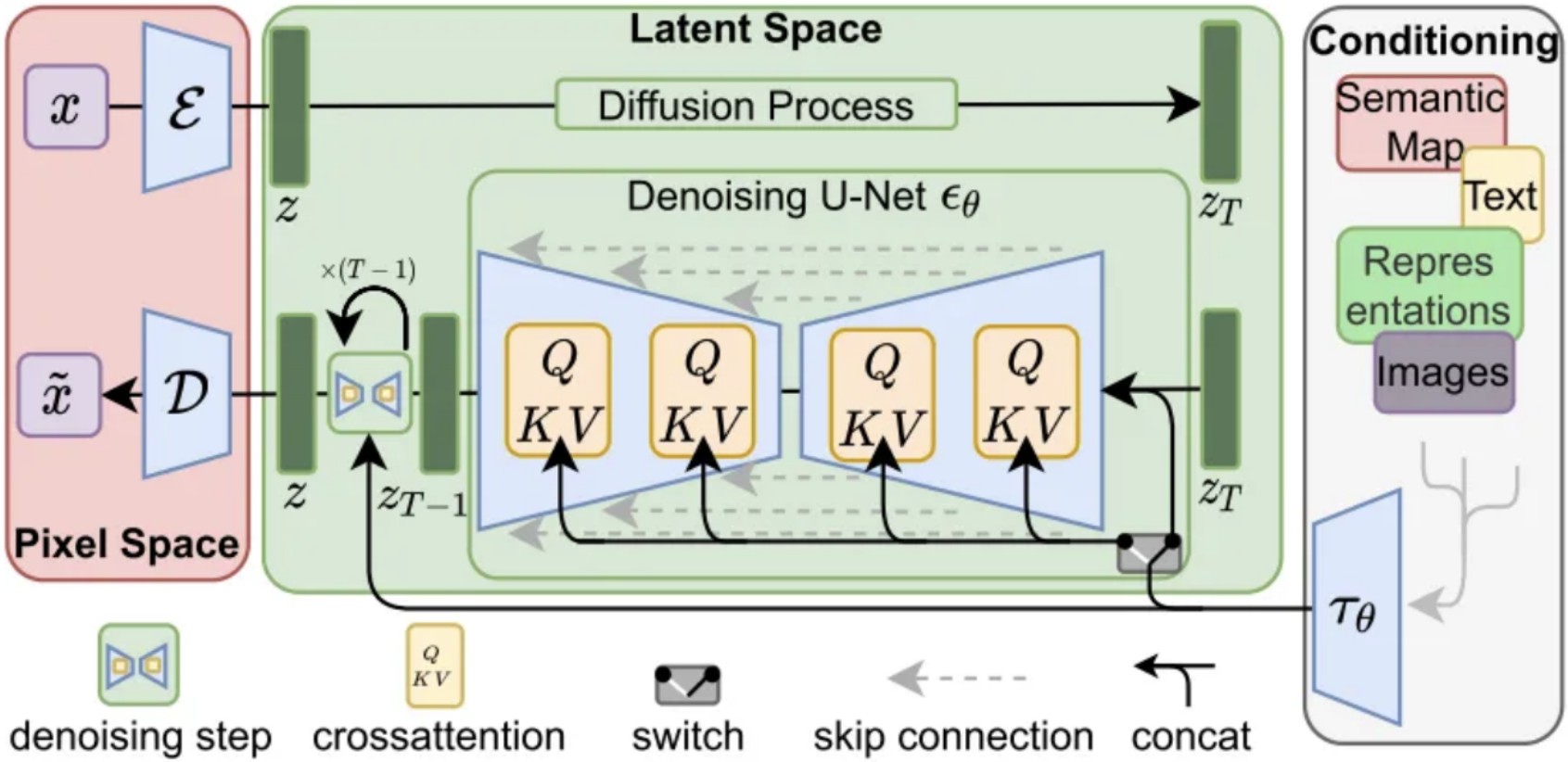


Figure: "Latent Diffusion Models: A Review"

Loss Function for Latent Diffusion Models

The loss function used during the training of a latent diffusion model is typically defined as:

$$\mathcal{L}_{denoise} = \mathbb{E}_{q(x_0)} \left[\sum_{t=1}^{T} \gamma_t \mathbb{E}_{q(x_t|x_0)} \left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2 \right] \right] \tag{1}$$

The paper add two losses:

- $\mathcal{L}_{\text{token}}$ for each token: $\left(1 - \frac{\sum_{u \in B} \mathcal{A}_u}{\sum_u \mathcal{A}_u}\right)^2$
- $\mathcal{L}_{\text{pixel}} = -\frac{1}{L_{\tau_\theta(y)} L_{z_t}} \sum_i \sum_u \left(\mathcal{M}_{(i,u)} \log(\mathcal{A}_{(i,u)}) + (1 - \mathcal{M}_{(i,u)}) \log(1 - \mathcal{A}_{(i,u)})\right)$
- where :
  A(i, u) = cross attention at position u for token i
  M(i, u) = mask for token i at position u

Further analysis of token supervision losses

Figure: Attention maps for several tokens

Ablation study

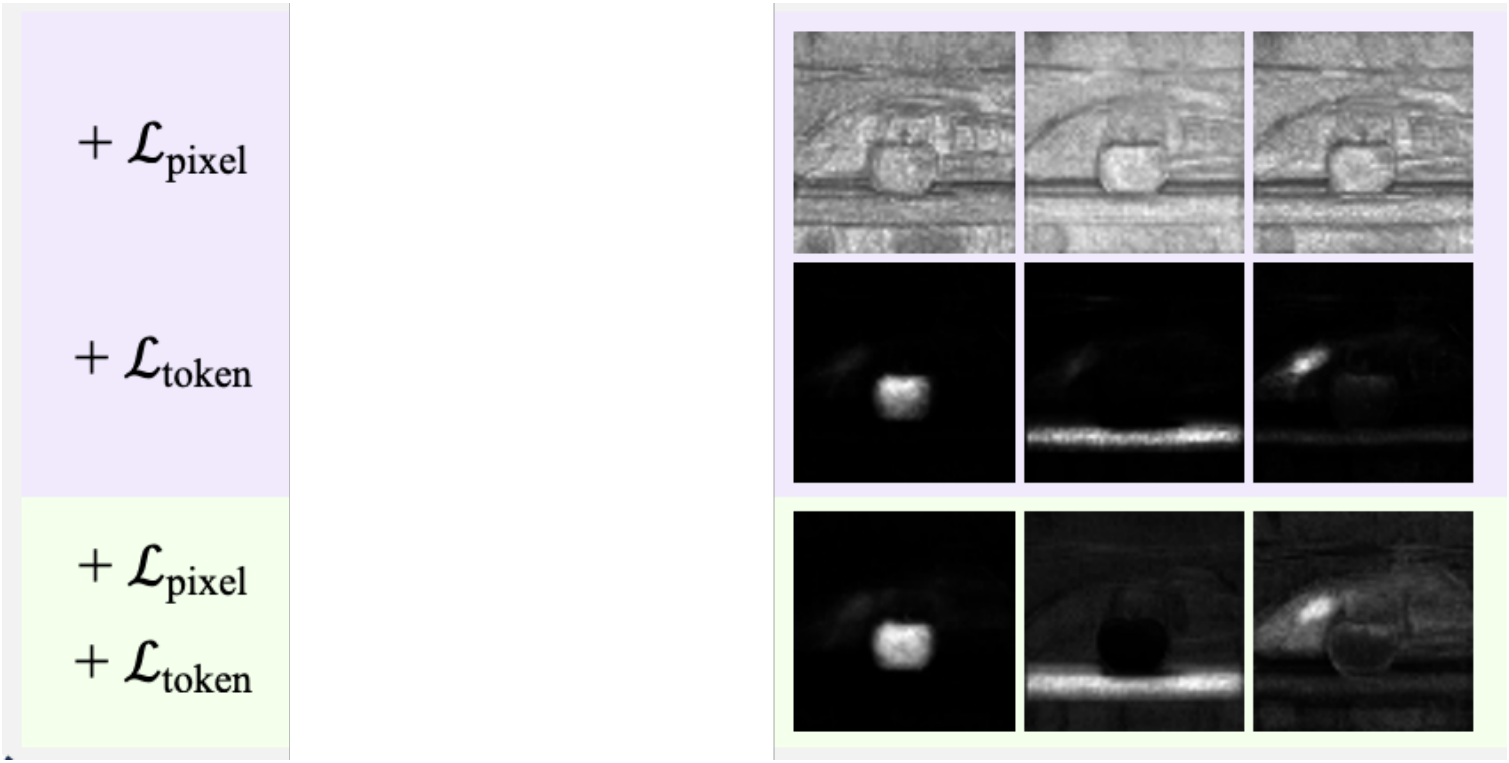


Figure: Attention maps after training with different losses

MultiGen Benchmark

- Sample 5 categories/items
- Prompt the model to generate an image with all these items
- See how many of these items are correctly present in the generated image
- Give you 5 scores: MG_k is the proportion of runs where k out of 5 items are correctly present

An overview of my work

- What I did :
  - Reproduce the paper's training pipeline. Train a model and evaluate it.
  - Implemented my own dataset manager with generator to be able to run the code with very little RAM. (Lazy evaluation)
  - I modified the paper's attention controller class to work with mps and float16
- What I tried :
  - Having batchsize greater than 1
- What I implemented but did not run:
  - A clip method to better align image and text
- What I did not do :
  - I used the paper's implementation for the MultiGen evaluation
  - I used the paper's token supervision losses implementation

The Clip allignement

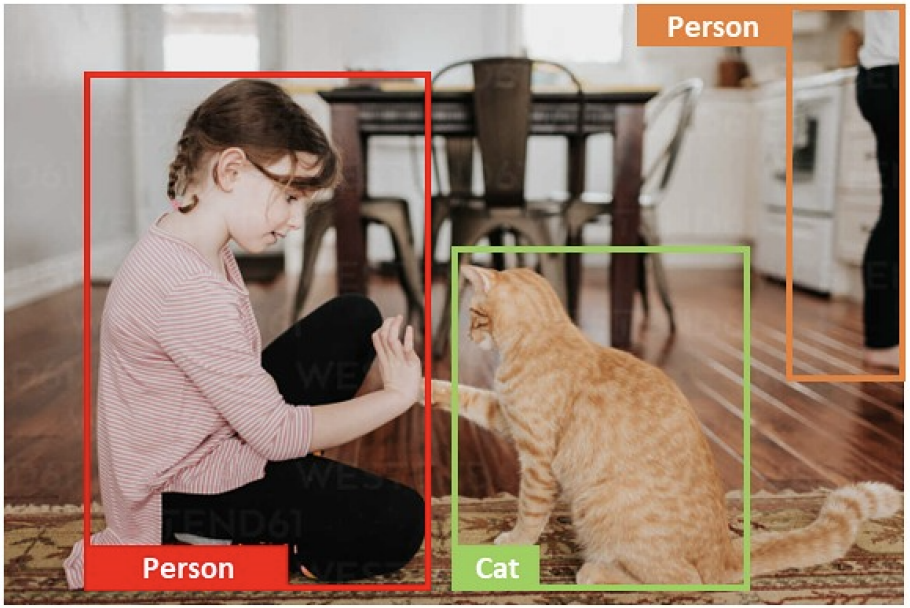


Figure: Each item is associated with a box

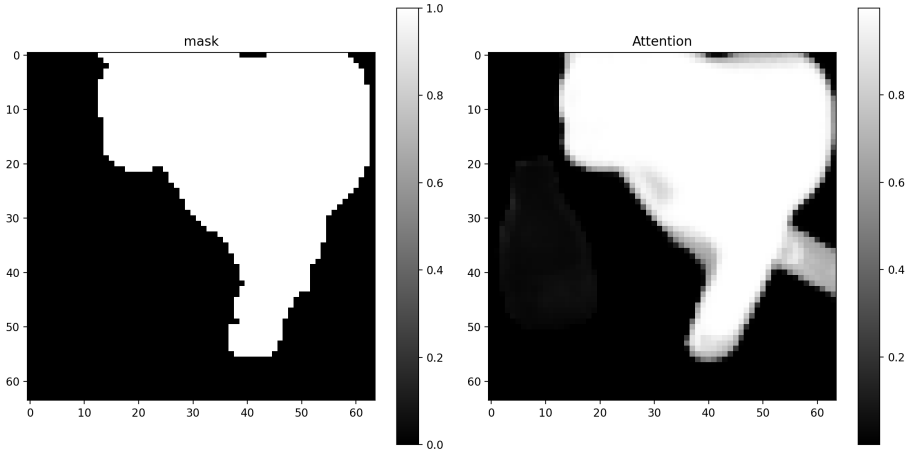
We extract representation of item boxes with ROI pooling and do a clip loss with token embedding and Roi embedding

$$\mathcal{L}_{\text{CLIP}} = -\log\left(\frac{\exp(\text{sim}(I_i, T_i)/\tau)}{\sum_{j=1}^{N}\exp(\text{sim}(I_i, T_j)/\tau)}\right) \tag{2}$$

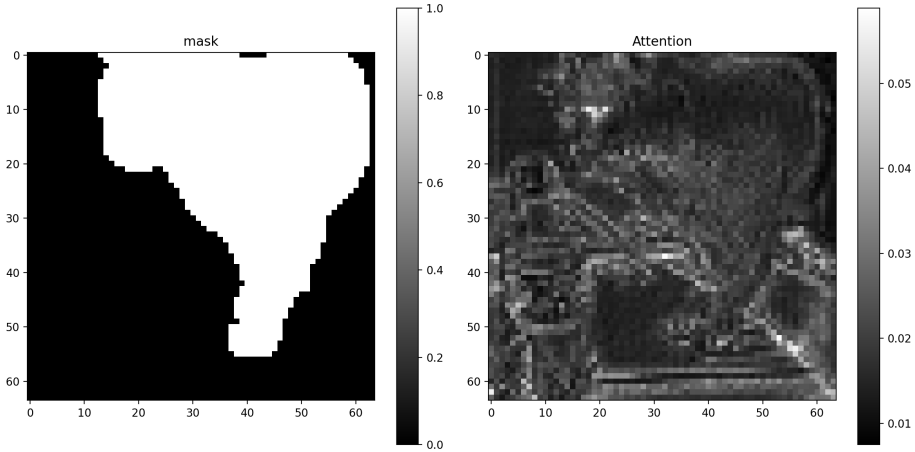
Results

Figure: Image test

Cat

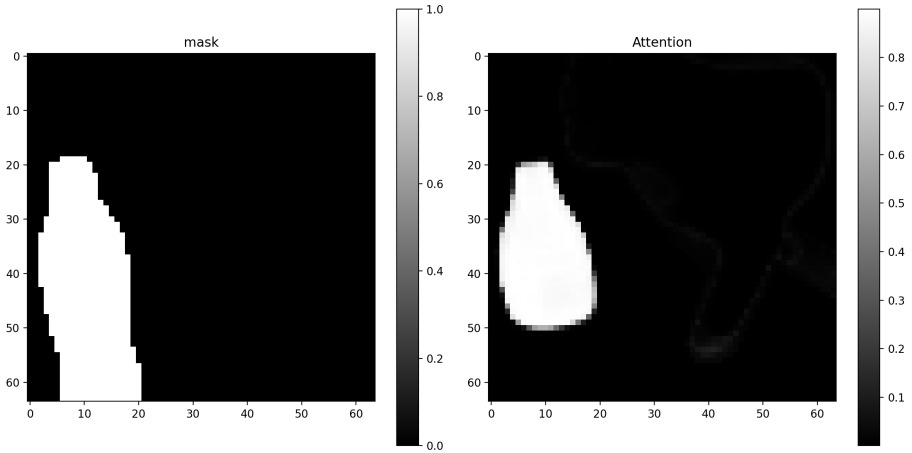


figureMy model (mask=left, attention=right)

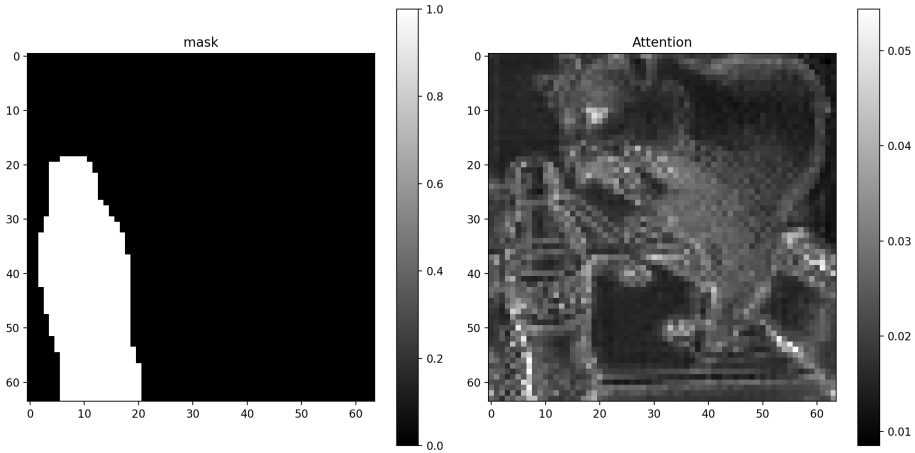


figureInitial model (mask=left, attention=right)

Bottle



figureMy model (mask=left, attention=right)



figureInitial model (mask=left, attention=right)

Some images generated

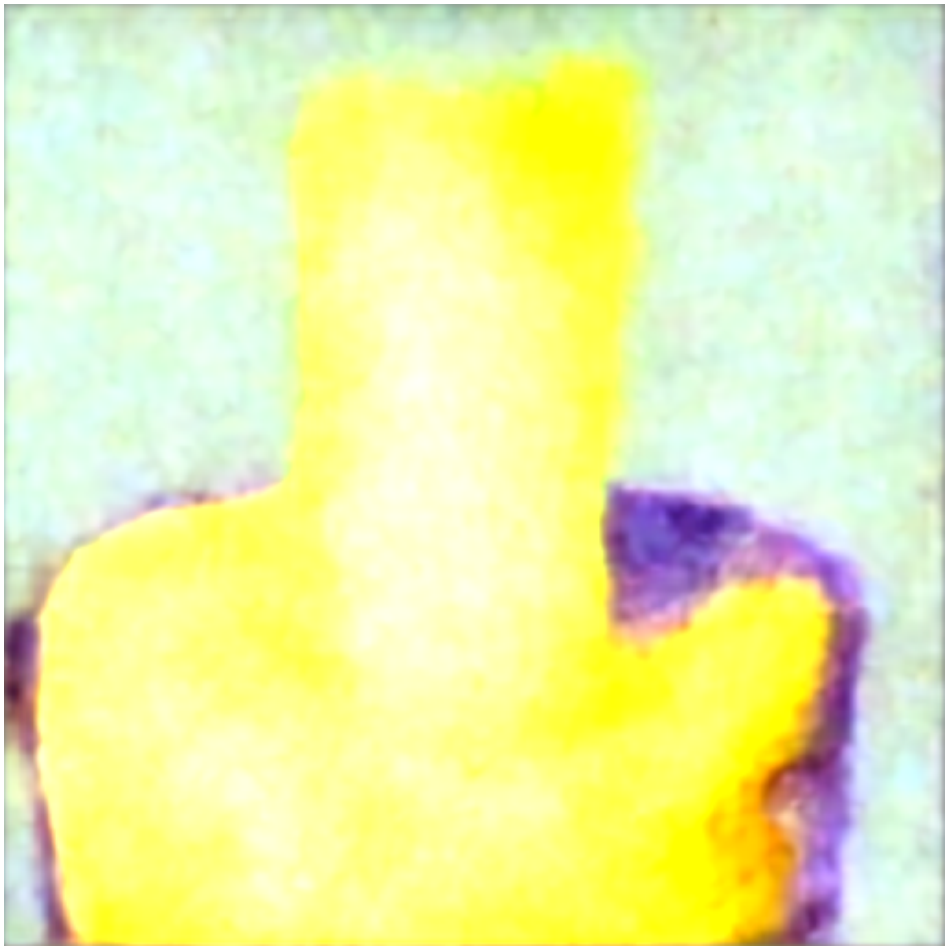


Figure: Image generated by my model

Some images generated

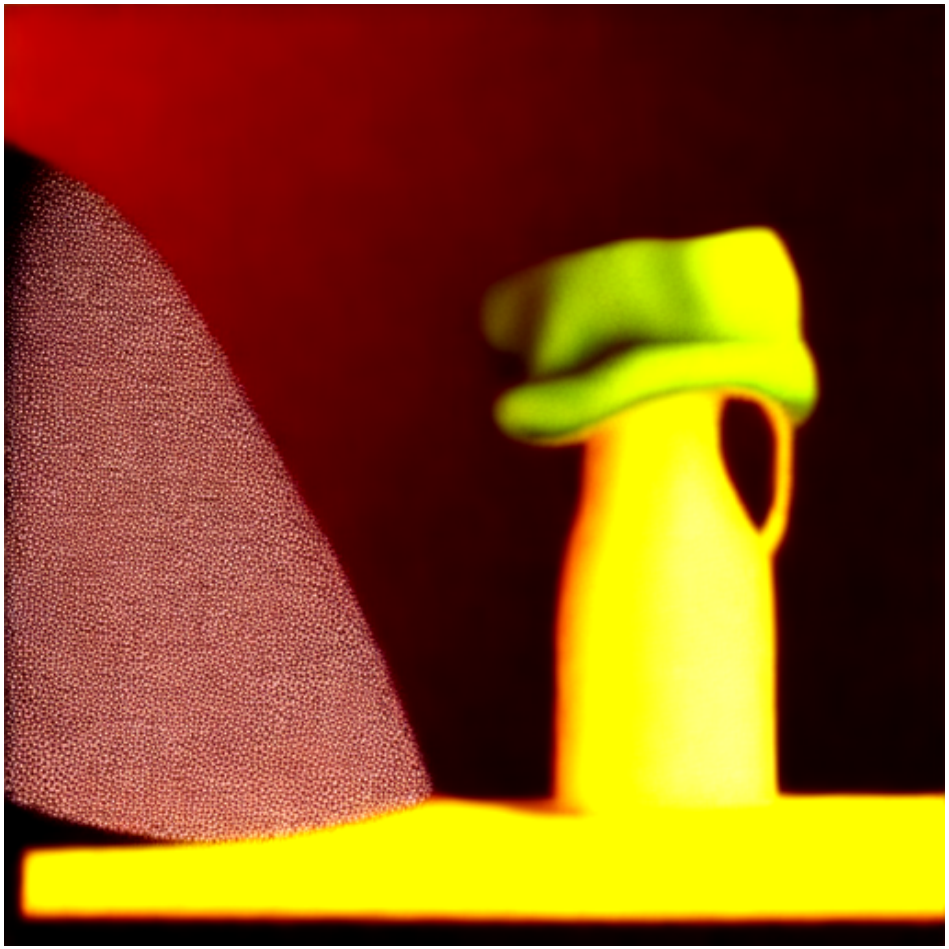


Figure: Image generated by base model

Possible explanation

1. The weight on my token supervision loss is too big.

Possible explaination

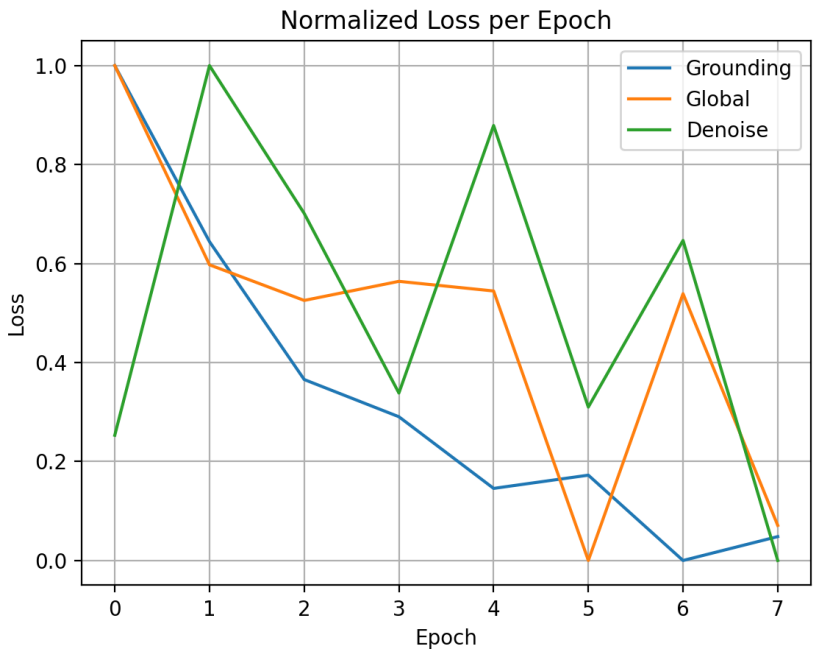


Figure: Min max scaled loss per epoch

Thank you for your attention!

Questions?