

# Image retrieval report

We used **Resnet18** encoder for all the tasks:

- training multilabel classification from scratch
- finetuning multilabel classification
- training a metric learning model
- training in a self-supervised manner

Additionally, we trained a **U-Net** with **Resnet18** encoder for segmentation.

## Training/validation procedures

We randomly split the labeled dataset into two parts:

- training dataset - 90%
- validation dataset - 10%

We use early stopping to prevent overfitting, however for some runs we stop earlier.

## Classifier from scratch

We replace the last FC layer to accommodate for different number of classes. We also add a **sigmoid** activation after the last layer and treat each value of activation as the probability of the corresponding apparel being present in the image. This is a common approach for multi-label classification.

All weights are initialized randomly. **Kaiming He initialization** is used.

The chosen criterion is **BCE (binary crossentropy)**.

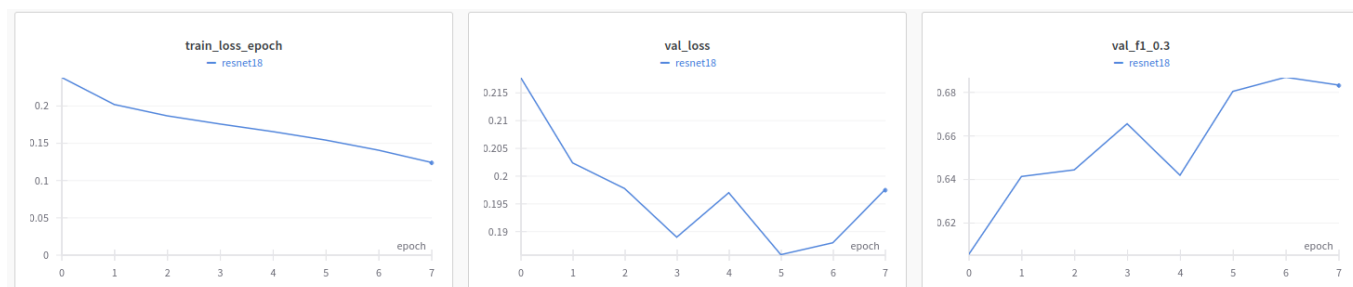
Additionally, we monitor the **example-wise F1 score** of both validation and training datasets, as well as **example-wise accuracy, precision and recall**. To select the best threshold that defines the final prediction of the model (whether the class is or is not present) we monitor the aforementioned F1-score.

We use Adam with a mini-batch size of 128. The learning rate starts from  $3e-4$  and is divided by 10 when the validation BCE plateaus. The model is trained until convergence of validation BCE.

We resize the input images to 224x224 and normalize them as in the original paper.

### *Experiment 1*

First we trained a model with no augmentations.

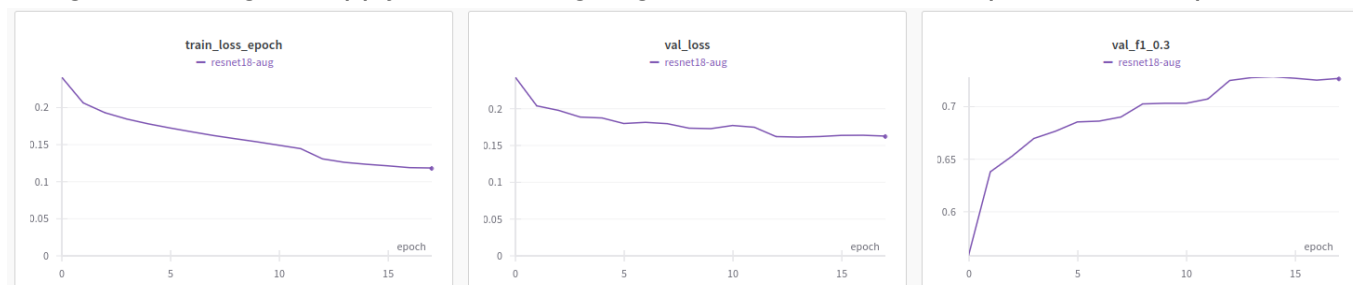


The achieved validation **F1 score** is 0.69

We noticed that 0.3 threshold is the best in terms of F1-score.

## Experiment 2

To fight overfitting, we apply the following augmentations: random crop, horizontal flip.

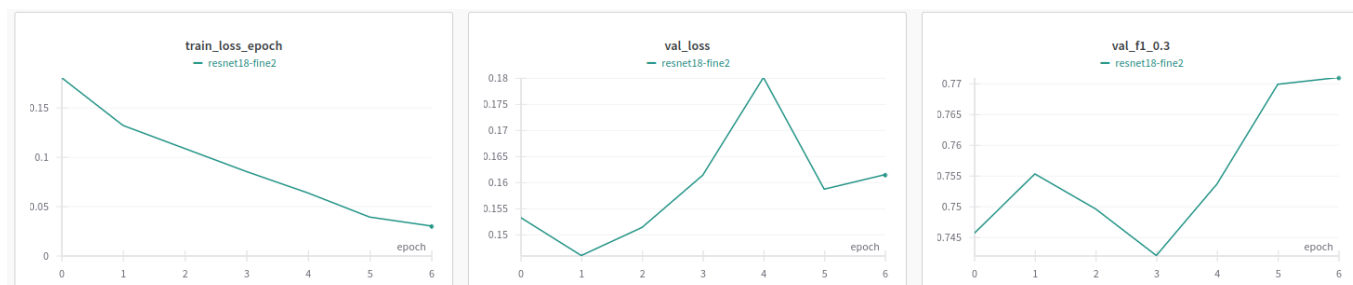


This way we managed to increase the example-wise **F1 score** to 0.73.

## Finetune classifier

The weights are from the model pretrained on ImageNet. The last FC layer is replaced and is randomly initialized.

The setup is similar to training from scratch. We apply augmentations and achieve validation **F1 score** of 0.77

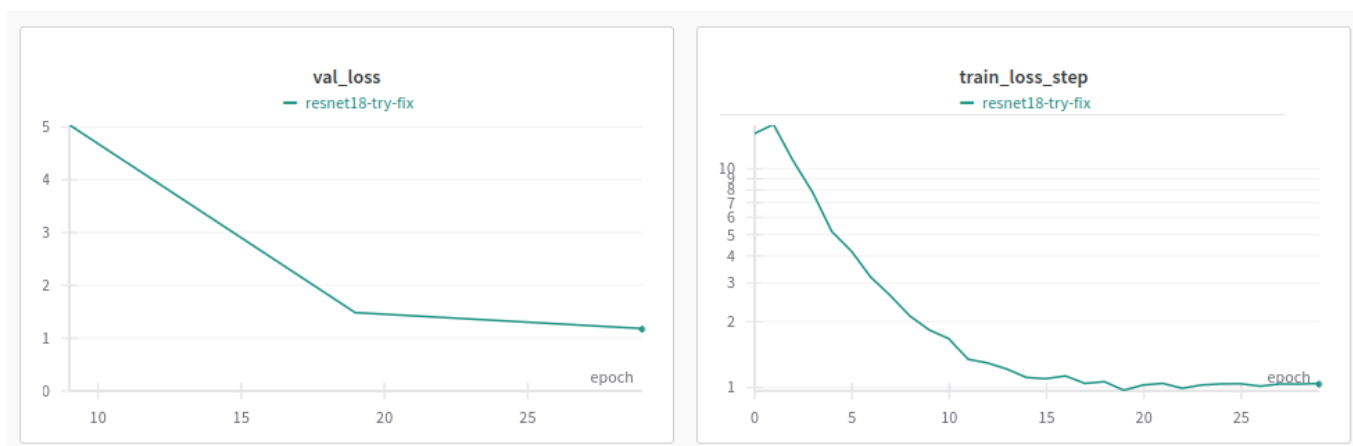


## Metric learning

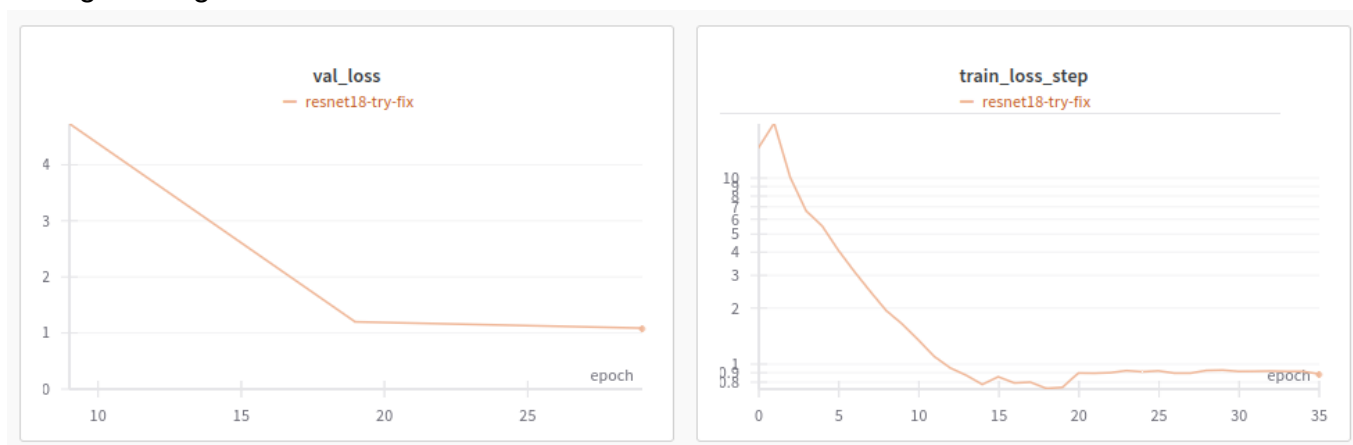
We replace the classification head with 128-wise FC layer. We start with weights pretrained on ImageNet.

We minimize the **triplet loss** with margin 1.

We couldn't train the model much because of **triplet loss collapse** - a common problem with triplet loss:



However, we noticed that the training is better if we start from small margin and increase it during training:



## Self-supervised

We augment the images by rotating them. Then we predict the angle of rotation.

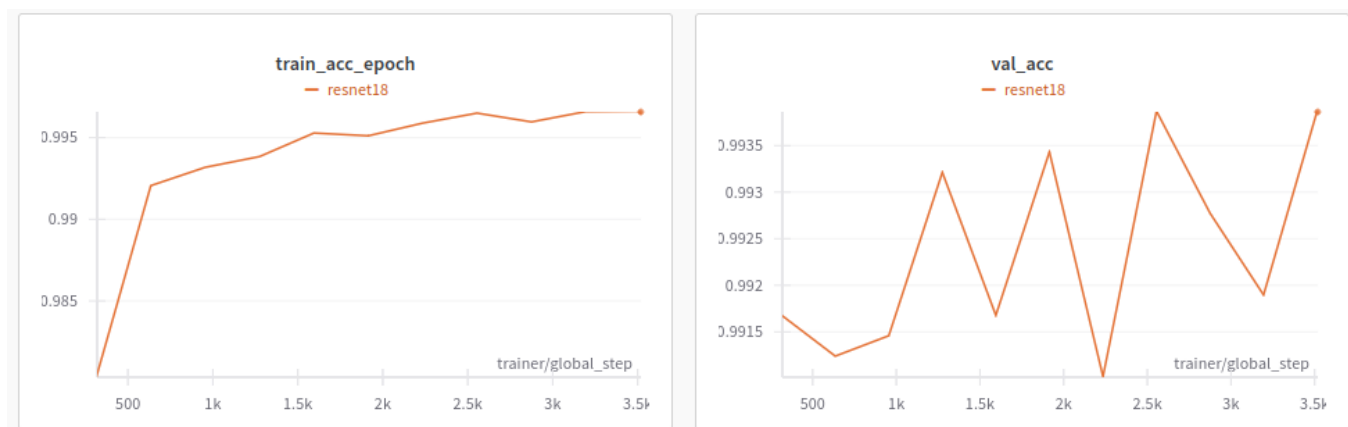
We replace the last classification layer with 4-way linear layer followed by softmax.

We start with weights pretrained on ImageNet.

We minimize the **cross-entropy criterion**.

All other details are similar to training from scratch.

Predicting angle of rotation is an easy task for the model. The following figure contains train accuracy progression through time (left) and validation accuracy through time (right).



The achieved validation accuracy is 0.99. In fact, the model performance is already high immediately after the first epoch.

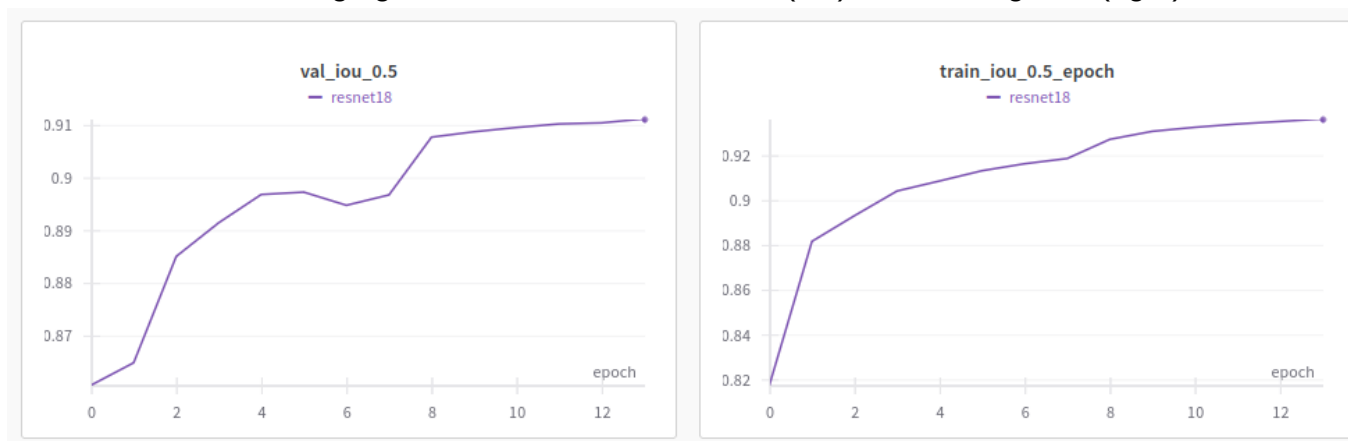
## Segmentation

We train a **U-Net** with **Resnet18** encoder. The encoder is initialized with weights trained on ImageNet. The output of the model is a one-channel feature map with per-pixel probability of being apparel. The model is trained using Dice loss.

We resize all images and masks to the size of (224, 224) and normalize images using mean and standard deviation calculated on ImageNet.

Other details are similar to training from scratch.

















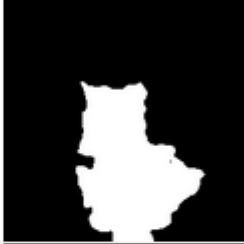



We use example-wise IOU (Intersection Over Union) as a metric. We achieve 0.91 IOU on validation. The following figure contains validation IOU (left) and training IOU (right):



Looks like further improvements are still possible.

## Qualitative evaluation

We display several prediction for validation dataset. Left - images, middle - GT, right - model prediction.

images	GT	pred
		
		
		
		
		
		
		

The model predicts masks of reasonably good quality. It can fail to predict masks for small details, e.g. hats, glasses.

# Image retrieval

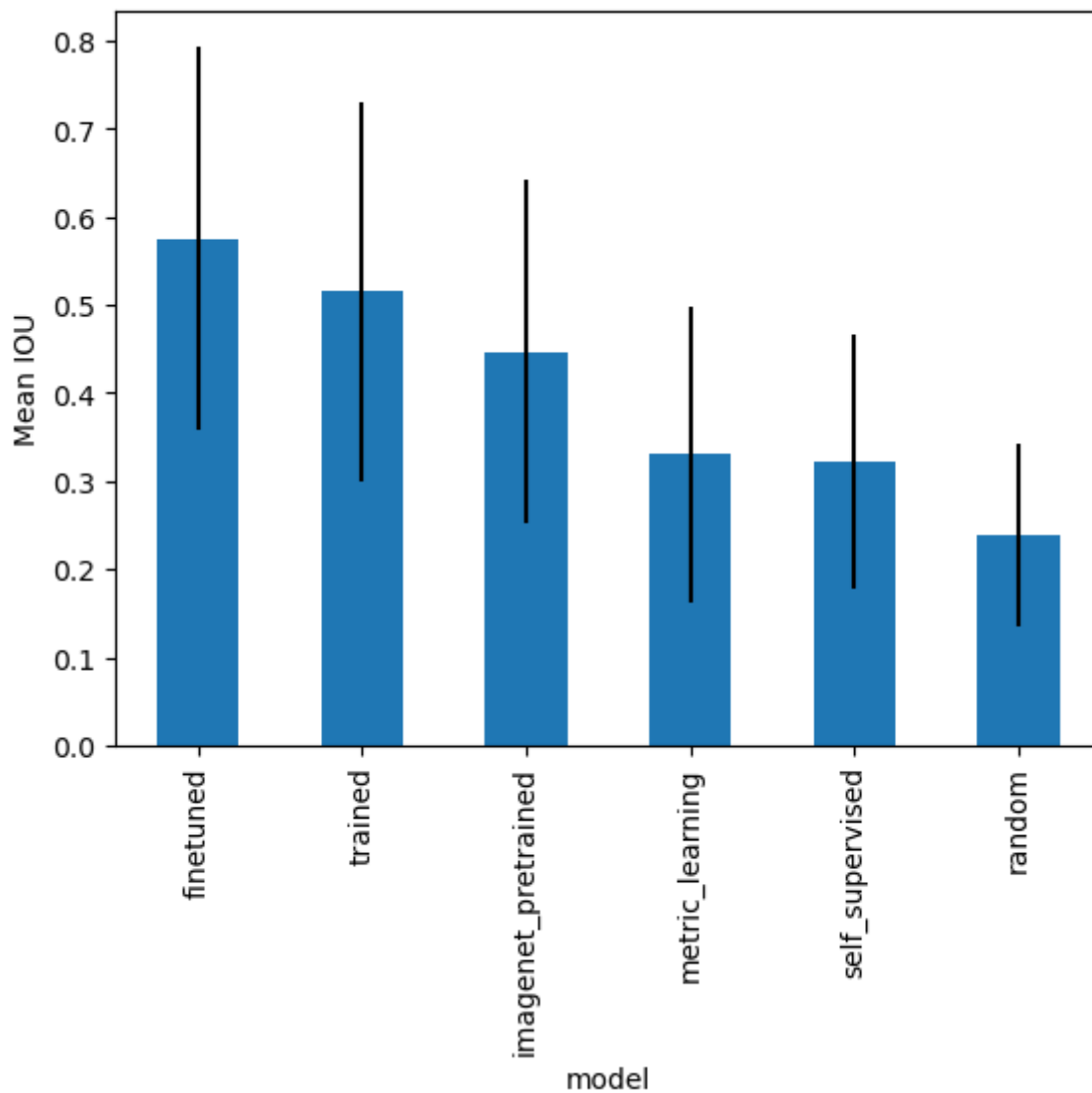
We use the vector search from **FAISS** to find the closest vectors by L2 norm.

## *Metrics*

In order to compare retrieval capabilities of different models, we proceed as follows:

1. Take random 500 query images from the validation dataset.
2. For each model:
  1. Create an index of embeddings for each image from the train dataset
  2. Search 5 closest vectors for each query image and calculate IOU for each pair of query and retrieved image. The IOU between the query and retrieved image is calculated as a ratio between the number of overlapping labels and the number of unique labels in both images.
  3. Average the IOUs for each of the query images
  4. Calculate **the mean and standard deviation** of all average IOUs across all query images.

The following figure depicts the means and standard deviations of IOUs for each of the model. Additionally, for comparison reasons we evaluate the random retrieval.



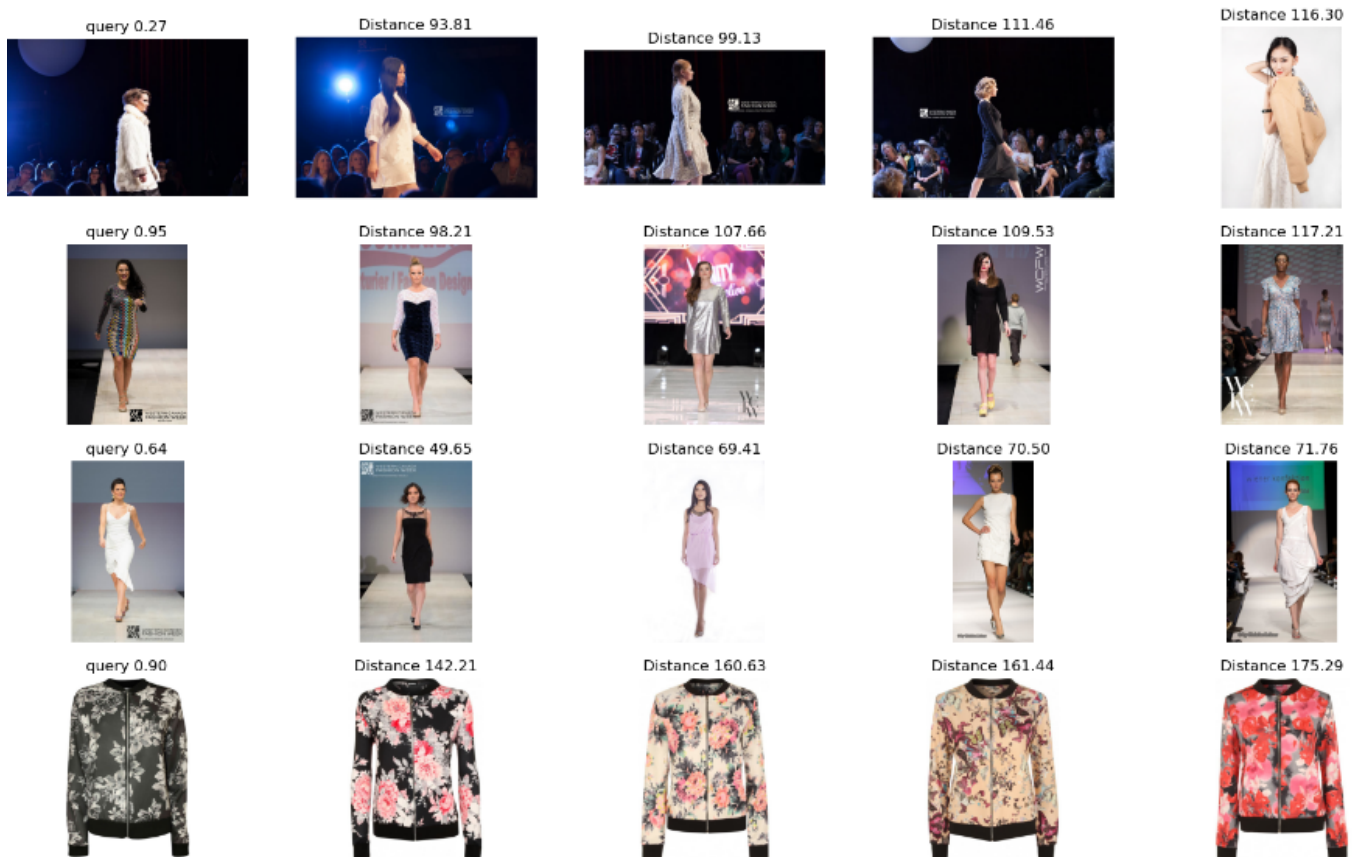
We observe that all models perform better than random retrieval. The models sorted by the quality of retrieval in descending order:

3. Finetuned model
4. Trained from scratch
5. ImageNet-pretrained model
6. Metric learning (triplet loss) model
7. Self-supervised model

### *Qualitative analysis*

We display the retrieval results for several query images using finetuned model (each row corresponds to different retrieval result):

# finetuned



As we can see, all retrieval results have similar background. This can be due to the fact that all models partially base their decision on the background. Thus the produced embeddings are not the representations of the apparel only, but also of the background.

## Incorporating segmentation into image retrieval

We zero out portion of an image that doesn't contain the apparel and calculate the embeddings based on the masked image. This method improved the retrieval quality (in terms of IOU) for some models and decreased for other models. In particular:

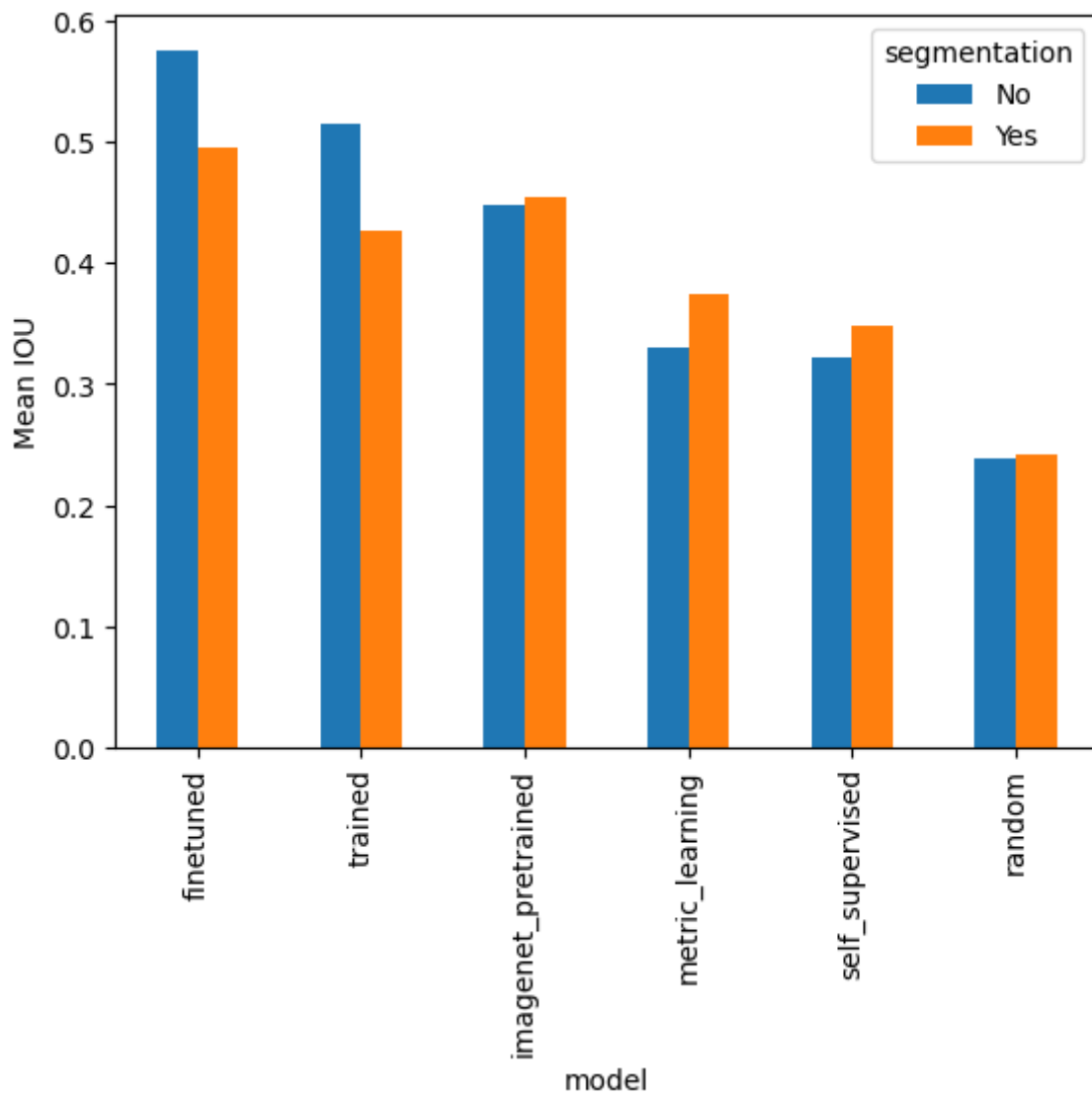
- segmentation **worsened** the retrieval using **finetuned model**
- segmentation **worsened** the retrieval using the model **trained from scratch**
- segmentation **improved** the retrieval of a **Imagenet-pretrained** encoder by a little bit.
- segmentation **improved** the performance of **metric learning model**
- segmentation **improved** the performance of **self-supervised model**

All of the above statements are based on the statistically significant data.

### Metrics

The following figure contains the mean IOU calculated for both segmented and original images:



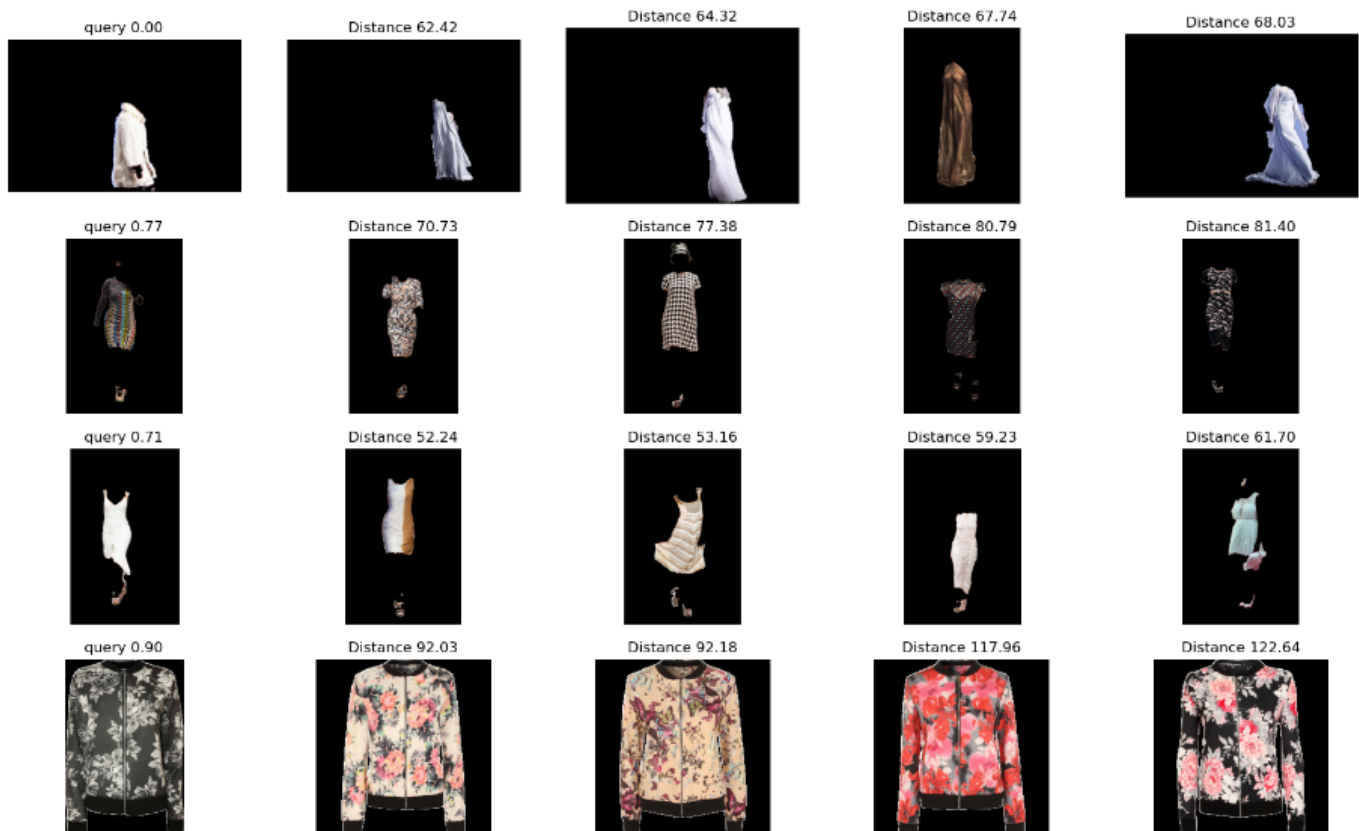


### *Visualizing retrievals*

We display the retrieval results for several query images using finetuned model (each row

corresponds to different retrieval result):

## finetuned



## Conclusions

- Training from scratch takes much longer. Still, in our case, finetuning outperforms training from scratch.
- Segmentation model performs good enough, but still can be improved.
- All models gave better performance than simple random retrieval. The best model in terms of retrieval capability is the finetuned one.
- Segmentation didn't improve the retrieval quality of the best model. This might be due to the fact the model wasn't trained on masked images. Thus it can fail to generalize to them.

## Further improvements

- We want to retrain/finetune the models on the masked images.
- We probably can increase the performance on validation just by applying more augmentations.
- Due to the imposed time limitations, we stopped training some models earlier. We might improve the result just by continuing training.

- Triplet loss model observed **loss collapse**. We can facilitate training by starting from weights from multilabel classification model.