Appendix

A. End Tasks

The descriptions of all end tasks are provided below. Semantic Image-level, Structural Image-level, Semantic Pixelwise, and Structural Pixelwise tasks are shown with different colors.

• **ImageNet Cls.** [15] - This is a 1000 class natural image classification task. The dataset includes a variety of categories, such as coffee mugs, drum, and fire engine. The images are of varying resolution but they are all resized to 224×224.

• **ImageNet v2 Cls.** [50] - This is a natural image classification task with the same set of categories as ImageNet. This task has the same train set as ImageNet, but has a re-collected test set with the same distribution and categories.

• **Pets Cls.** [44] - This is a natural image classification task with images of cats and dogs. There are a total of 37 classes corresponding to breeds of cats and dogs including Persian, Chihuahua and Bengal. The images are of varying resolution but they are all resized to 224×224.

• **CalTech Cls.** [35] - This is a 101 class natural image classification task with pictures of objects such as planes, chairs, and animals. The images are of varying resolution but they are all resized to 224×224.

• **CIFAR-100 Cls.** [33] - This is a 100 class natural image classification task. The classes include apples, bottles and bicycles. The Images are of size 32×32 but they are all resized to 224×224.

• **SUN Scene Cls.** [61] - This is a 397 class scenery image classification task. The classes include scene categories such as cathedral, river, or archipelago.

• **EuroSAT Cls.** [48] - This is a 20 class dataset of satellite imagery classification. The spatial resolution corresponds to 10 meters per pixel and includes categories of different land use, such as Residential, Industrial and Highway. All images are resized to 224×224.

• **dtd Cls.** [11] - This is a 47 class dataset of textural imagery classification. Some textures include bubbly, lined and porous. All images are resized to 224×224.

• **Kinetics Action Pred.** [29] - This task consists of predicting the action that a person is performing from 6 ordered video frames. The dataset contains 400 classes including bowling and dining. The dataset for this task is 50,000 image frames captured from the Kinetics400 dataset videos at 6 frames per video. All images are resized to 224×224.

• **CLEVR Count** [28] - This is a synthetic visual question answering dataset designed to evaluate algorithmic visual reasoning. The task consists of classifying the number of objects in the image. All images are resized to 224×224.

• **THOR Num. Steps** - This is a task where the maximum number of forward steps (of 0.25 meters) that a robot in AI2-THOR [30] can take is predicted from a frame of the robot's point of view. This task is structured as classification, rather than regression, of the images from the simulation and the correct answer will always be between 0 and 4 steps inclusive (thus this task is a 5-way classification). This task is first proposed in this paper.

• **nuScenes Egomotion** - This is an egomotion prediction task from two consecutive frames of the nuScenes self driving car dataset [4]. The types of motion include forward, forward-left and forward-right motion as well as a no motion action. Both frames are resized to 224×224. This task is first proposed in this paper.

• **THOR Egomotion** - This is an egomotion prediction task from two consecutive frames in the AI2-THOR [30] simulator. The types of motion include moving forward, left and right rotation, and looking up and down. Frames are resized to 224×224. This task is first proposed in this paper.

• **Cityscapes Seg.** [13] - This is a semantic segmentation task where every pixel is labeled as one of 20 categories. The images consist of dashboard camera views of cities and roads. The task contains categories such as person, traffic light and sky (there is also a background class for pixels that do not fit into any other category). Crops of size 513×513 sampled from the full image are used during training, and evaluation is done at full resolution.

• **Pets Instance Seg.** - This is an instance segmentation task on the Pets dataset [44], where each image contains exactly one cat or dog. Each image (and its ground truth instance label) is resized to 224×224.

• **EgoHands Seg.** [3] - This is an instance segmentation task on a dataset of video frames of human hands performing various tasks. The videos are captured using a Google glass camera and are from the egocentric view of one person performing a task with another person. Each frame has at most 4 hands (the left and right hand of the person wearing the Google glass and the right and left hand of their partner) and each of these has its own associated class (there is also a background class). Crops of size 513×513 sampled from the full image are used during training, and evaluation is done at full resolution.

• **NYU Depth** [40] - This is a pixelwise depth prediction task on a dataset of natural images of building interiors obtained from videos. The images are resized to 224×224 and the output is predicted in meters.

• **THOR Depth** - This is a pixelwise depth prediction task on a dataset of synthetic images of building interiors produced by the AI2-THOR [30] simulator. The images are resized to 224×224 and the output is predicted in meters. This task is first proposed in this paper.

• **Taskonomy Depth** [62] - This is a pixelwise depth prediction task on a dataset of natural images of building interiors from a variety of building types. The images are resized to 224×224 and the output is predicted in meters. This is a

common task but the dataset split is first proposed in this paper.

• **NYU Walkable** [39] - This is a pixelwise detection task. Each pixel is labeled as walkable (floor, carpet, etc.) or non-walkable (wall, window, ceiling, etc). The dataset consists of images of interior rooms. All images are resized to 224×224.

• **KITTI Opt. Flow** [20] - This is an optical flow prediction task from two consecutive frames. The data comes from a self driving dataset. Crops of size 513×513 sampled from the full image are used during training, and evaluation is done at full resolution.

The following tasks have been adopted from VTAB [63]: Caltech Cls., CIFAR-100 Cls., dtd Cls., Pets Cls., SUN Scene Cls., EuroSAT Cls., and CLEVR Count.

B. End Task Networks

The architecture and the loss functions used for each end task have been shown in Table 1. Top-1 accuracy is the percentage of test samples labeled with the correct class, mIOU is the class wise average intersection over union between the prediction class mask and the ground truth, Negative L1 Error is the negative absolute distance between the prediction and the label averaged over all the pixels, and 1-All is 1 minus the percentage of outliers averaged over all ground truth pixels. Figures 8–12 show the details of each network. The orange box in the figures shows the frozen encoder. The output dimensions for each block are also shown. The variables h and w represent the height and width of the input image, respectively, while n represents the batch size and s represents the sequence length.

C. Training Details

In this work encoders and end task networks are trained separately. Below we describe the training procedure for each.

We train the encoders by MoCov2 [9] and SwAV [5] algorithms. For the rest of the training algorithms, we use the publicly released weights for the trained models. We train every model using code publicly released by the authors and the same hyperparameters as the original implementation.

We train the end task networks by freezing the encoders and training just the end task network layers. For each task, we perform a grid search of 4 sets of optimizers and learning rates using the encoder trained with SwAV on ImageNet for 200 epochs. We then select the best performing set of hyperparameters and use them for all other runs. We also use the grid search training runs to determine the number of epochs necessary for each task to converge. We performed grid search for each individual encoder on a subset of all the tasks and found that the hyperparameters we found were the same across all encoders for almost all tasks (and where they were not the same, the performance difference was so small it could be attributed to noise), so due to computation constrains we decided to not perform a full grid search for every task and every model. In Table 2 we report the specific hyperparameters used for each end task.

D. Correlation Analysis of the End Tasks

To better understand the relationships between the end tasks chosen for this paper, we analyze the correlation between their performances using different encoders. Specifically, for every task A and every task B we compute the correlation between the performance of task A and B of all of the encoders we analyze. This shows whether good performance on one task is indicative of good performance on another.

Figures 13 and 14 show the Pearson and Spearman (rank) correlations between the end task performance of the encoders. One clear trend is that we see pockets of strong correlation within each task category. Sometimes they are well defined (Semantic Image-level or Structural Pixelwise tasks represented by red and yellow boxes in Figure 14) and sometimes they are more subtle (Semantic Pixelwise represented by the green box in Figure 14). Another trend that these figures show is that ImageNet classification performance is not a good universal metric for encoder performance (especially for pixelwise output tasks, where there is a low correlation).

E. CKA Analysis Details

Centered Kernel Alignment [31] is a method of quantifying the similarity of representations between images as they are processed through an encoder. For this study we compare how the relationship between the representations of two images change across the different blocks of the ResNet encoder. We select a balanced subset of 10,000 images from the ImageNet dataset to measure the similarity of representations, and downscale the images to 112×112 before processing them through the encoder. We then compute the CKA between the representations of every pair of images in our subset for every block of the ResNet encoder (this similarity metric has a range of 0 to 1). We find that all encoders trained with the MoCov2 algorithm have an average increase of 0.18 of the average correlation between the layers versus the encoders trained with the SwAV algorithm. This indicates that the MoCov2 encoders retain more spatial information about the images in the later layers and offers a potential hypothesis as to why MoCov2 encoders tend to outperform SwAV encoders at pixelwise output tasks.

It is important to note that this analysis was performed using only a subsample of ImageNet data. ImageNet was chosen for this analysis as it is amongst the most diverse datasets utilized in this paper, but it makes this analysis far

from entirely comprehensive. The reason for running this analysis on just this subsample was computational complexity, as evaluating the CKA on all the data available to us is computationally impractical.

F. ANOVA Tests

For this test, we consider encoders trained with the MoCov2 and SwAV algorithms on subsets of ImageNet (as discussed in the main text). We examine the relationship between encoders trained on class unbalanced versions of ImageNet and their balanced counterparts with an equivalent number of samples. We use the end task results of the following encoders in our analysis: SwAV Half ImageNet 200, SwAV Linear Unbalanced ImageNet 200, SwAV Quarter ImageNet 200, SwAV Log Unbalanced ImageNet 200, MoCov2 Half ImageNet 200, MoCov2 Linear Unbalanced ImageNet 200, MoCov2 Quarter ImageNet 200, MoCov2 Log Unbalanced ImageNet 200.

Our analysis found evidence that an encoder trained on a Log Unbalanced subset of ImageNet outperforms an encoder trained on a balanced subset of ImageNet with an equivalent number of samples. To further validate this conclusion we trained 2 additional encoders using SwAV on 2 different logarithmically unbalanced subsets of ImageNet and included them in the following test.

We fit an ANOVA model to all of the results we obtain, treating the task, training algorithm, dataset balance, dataset size and number of training steps as variables. We find that (unsurprisingly) the task, dataset size and number of training steps are statistically significant indicators of end task performance. We also find that the algorithm used to train the encoder (MoCov2 vs SwAV) is a statistically significant indicator of end task performance, with SwAV models performing better (this does not contradict our claim that SwAV is not universally better than MoCov2, as we simply have more tasks that SwAV is good at in our test battery). Finally, we do not find any statistically significant evidence that an encoder trained with the balanced ImageNet is better than the encoders trained on the discussed unbalanced variations. We do however find evidence that an encoder trained on a Log unbalanced subset of ImageNet tends to perform better than one trained on a balanced subset. Perhaps the (comparatively) larger number of samples of the same few categories is a good match for the contrastive learning algorithm, but further experiments are needed to determine the exact cause and extent of this phenomenon.

G. Variance of the Results

The main source of variance in our results is the self-supervised training of the encoder. Since each encoder requires over 500 GPU hours to be trained for 200 epochs with the MoCov2 training algorithm, and over 1000 GPU hours to be trained for 200 epochs with the SwAV training algorithm, it is impractical for us to test multiple training runs of every encoder configuration that we study in this work.

To provide some context regarding the magnitude of variations across runs, we train three encoders using SwAV on ImageNet for 200 epoch with different random seeds. All training parameters are exactly the same as those used by the SwAV authors to obtain their SwAV 200 model.

Our results show that, on average, the variation in the performance of the end tasks is less than 0.85% (relative difference with the average performance), which can be negligible.

H. List of Encoders

Table 3 provides a complete list of all 30 encoders that are used for our analysis.

I. Effects of MultiCrop Pre-processing

This work draws some comparisons between the MoCov2 and SwAV training pipelines and identifies some trends in the performance of encoders trained with them.

The two pipelines do not just contain a different training algorithm, but they also employ different pre-processing methods. To understand if the observed differences in end task performance are simply a result of different pre-processing we conduct an ablation study where we use the improved pre-processing methods of SwAV in conjunction with the MoCov2 training pipeline to train an encoder on ImageNet and evaluate its performance on our battery of end tasks.

We observe that the MultiCrop pre-procesing employed by SwAV is only partially responsible for the observed gap between the two training pipelines in question. Furthermore we observe that the MuliCrop pre-processing is not a universally better choice, as it seems to degrade the performance of certain Pixelwise output tasks. This result is rather expected since the MultiCrop pre-processing essentially makes the model embed a patch of the image and the entire image very similarly, thus encouraging more semantic and less structural embeddings.

Figure 15 shows that for almost all tasks the performance of the MoCov2+MultiCrop model is between that of the SwAV model and the vanilla MoCov2. From this we can hypothesize that adding MultiCrop makes the MoCov2 model behave more like model trained with SwAV when embedding images.

J. Other Encoders

One obvious axis of expansion for future work is performing this analysis on more encoders trained with different pipelines. We chose a very small subset from the cur-

rent state of the field and analyzed them very comprehensively. This meant that we would necessarily have to omit some prominent pipelines from our study. We conducted small ablations with 2 such noteworthy omissions: SimSiam [10], a siamese-style self supervised algorithm and Exemplar-v2 [66], an improved supervised training method.

Figure 16 shows that SimSiam performs very similarly to SwAV on our battery of end tasks. The distributions of the normalized end task scores of SwAV and SimSiam encoders show that SimSiam does not appear to be better and thus our analysis did not miss covering an encoder that would significantly outperform the rest.

We can also see that Exemplar-v2 does in fact outperform the vanilla supervised baseline on most end tasks, but it falls far short of the performance of certain self-supervised models like SwAV. This suggests that our findings regarding the performance of supervised vs. self supervised pipelines still hold.

Task	Category	End Task Network	Loss	Success Metric
• ImageNet Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• ImageNet v2 Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• Pets Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• CalTech Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• CIFAR-100 Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• SUN Scene Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• Eurosat Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• dtd Cls.	Semantic Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• Kinetics Action Pred.	Semantic Image-level	Multi Input Fusion Classifier	Cross Entropy	Top-1 Accuracy
• CLEVR Count	Structural Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• THOR Num. Steps	Structural Image-level	Single Layer Classifier	Cross Entropy	Top-1 Accuracy
• THOR Egomotion	Structural Image-level	Multi Input Fusion Classifier	Cross Entropy	Top-1 Accuracy
• nuScenes Egomotion	Structural Image-level	Multi Input Fusion Classifier	Cross Entropy	Top-1 Accuracy
• Cityscapes Seg.	Semantic Pixelwise	DeepLabv3+	Pixelwise Cross Entropy	mIOU
• Pets Instance Seg.	Semantic Pixelwise	DeepLabv3+	Pixelwise Cross Entropy	mIOU
• EgoHands Seg.	Semantic Pixelwise	DeepLabv3+	Pixelwise Cross Entropy	mIOU
• THOR Depth	Structural Pixelwise	U-Net	L1 Error	Negative L1 Error
• Taskonomy Depth	Structural Pixelwise	U-Net	L1 Error	Negative L1 Error
• NYU Depth	Structural Pixelwise	U-Net	L1 Error	Negative L1 Error
• NYU Walkable	Structural Pixelwise	DeepLabv3+	Pixelwise Cross Entropy	mIOU
• KITTI Opt. Flow	Structural Pixelwise	Siamese U-Net.	L1 Error	1 - All

Table 1. The network architecture, the loss and the success metric for each end task.

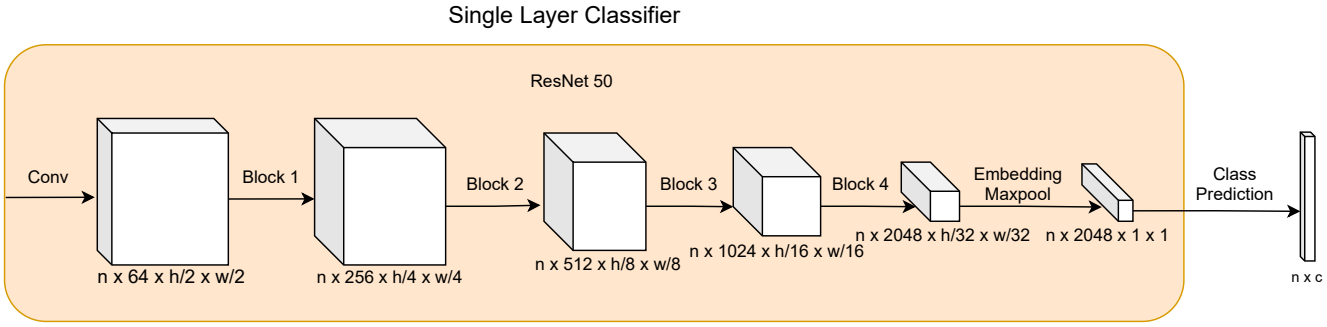

Figure 8. The Single Layer Classifier end task architecture. The orange box shows the frozen encoder.

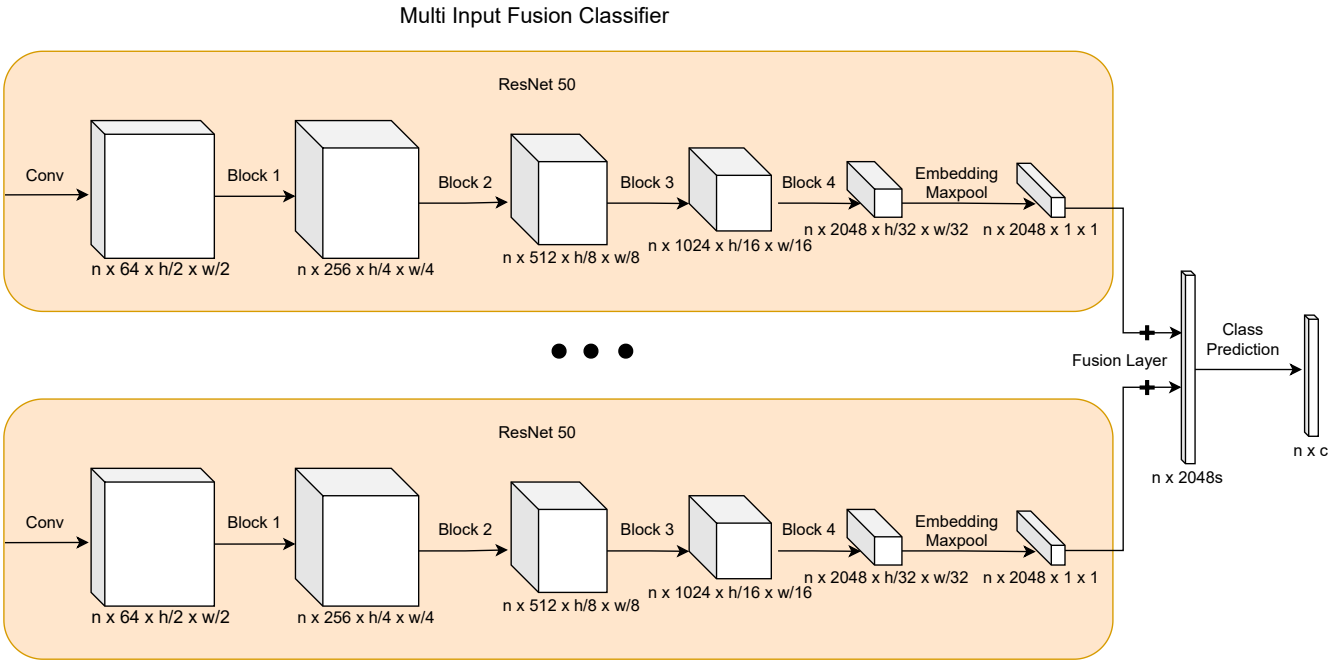

Figure 9. The Multi Input Fusion Classifier end task architecture. The orange box shows the frozen encoder.

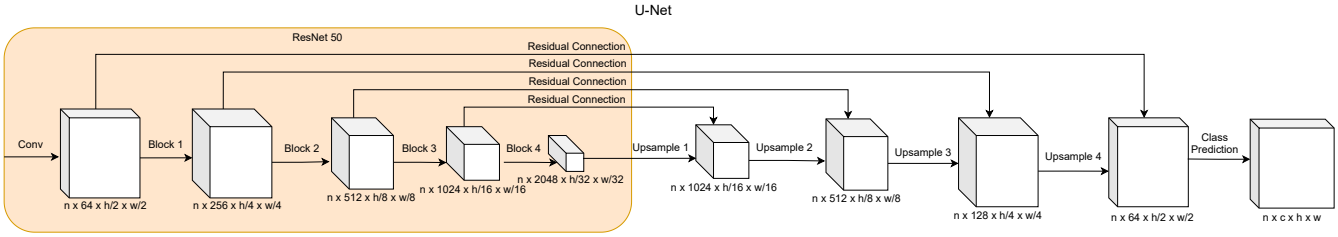

Figure 10. The U-Net end task architecture. The orange box shows the frozen encoder.

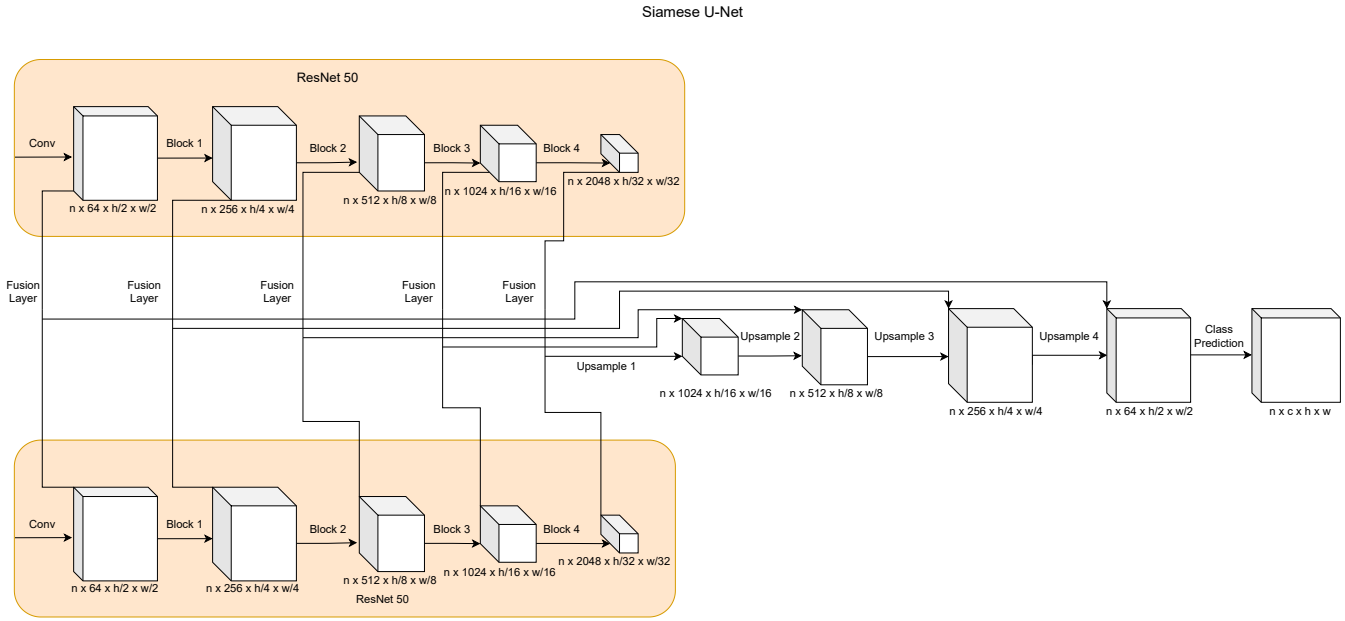

Figure 11. The Siamese U-Net end task architecture. The orange box shows the frozen encoder.

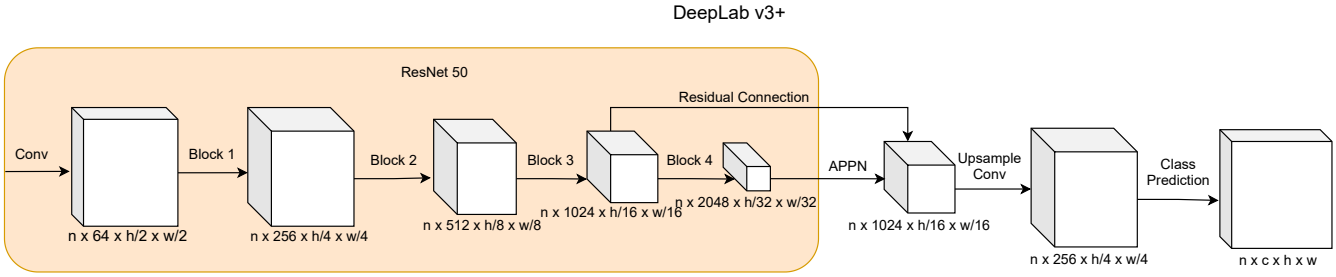

Figure 12. The DeepLabV3+ end task architecture. The orange box shows the frozen encoder.

Task	Train Set Size	Number of Train Epochs	Optimizer	Learning Rate
• ImageNet Cls.	1,281,167	100	Adam	0.0003
• Pets Cls.	3,680	500	Adam	0.0003
• CalTech Cls.	3,060	5000	Adam	0.0003
• CIFAR-100 Cls.	50,000	100	Adam	0.0003
• SUN Scene Cls.	87,003	250	Adam	0.0003
• Eurosat Cls.	21,600	200	Adam	0.0003
• dtd Cls.	3,760	100	Adam	0.0003
• Kinetics Action Pred.	50,000	100	Adam	0.0003
• CLEVR Count	70,000	100	Adam	0.0003
• THOR Num. Steps	60,000	100	Adam	0.0003
• THOR Egomotion	60,000	100	Adam	0.0003
• nuScenes Egomotion	28,000	100	Adam	0.0003
• Cityscapes Seg.	3,475	100	Adam	0.0003
• Pets Instance Seg.	3,680	100	Adam	0.0003
• EgoHands Seg.	4,800	25	Adam	0.0003
• THOR Depth	60,000	50	Adam	0.0003
• Taskonomy Depth	39,995	50	Adam	0.0003
• NYU Depth	1,159	250	Adam	0.0003
• NYU Walkable	1,159	100	Adam	0.0003
• KITTI Opt. Flow	200	250	Adam	0.0003

Table 2. Training details for each end task.

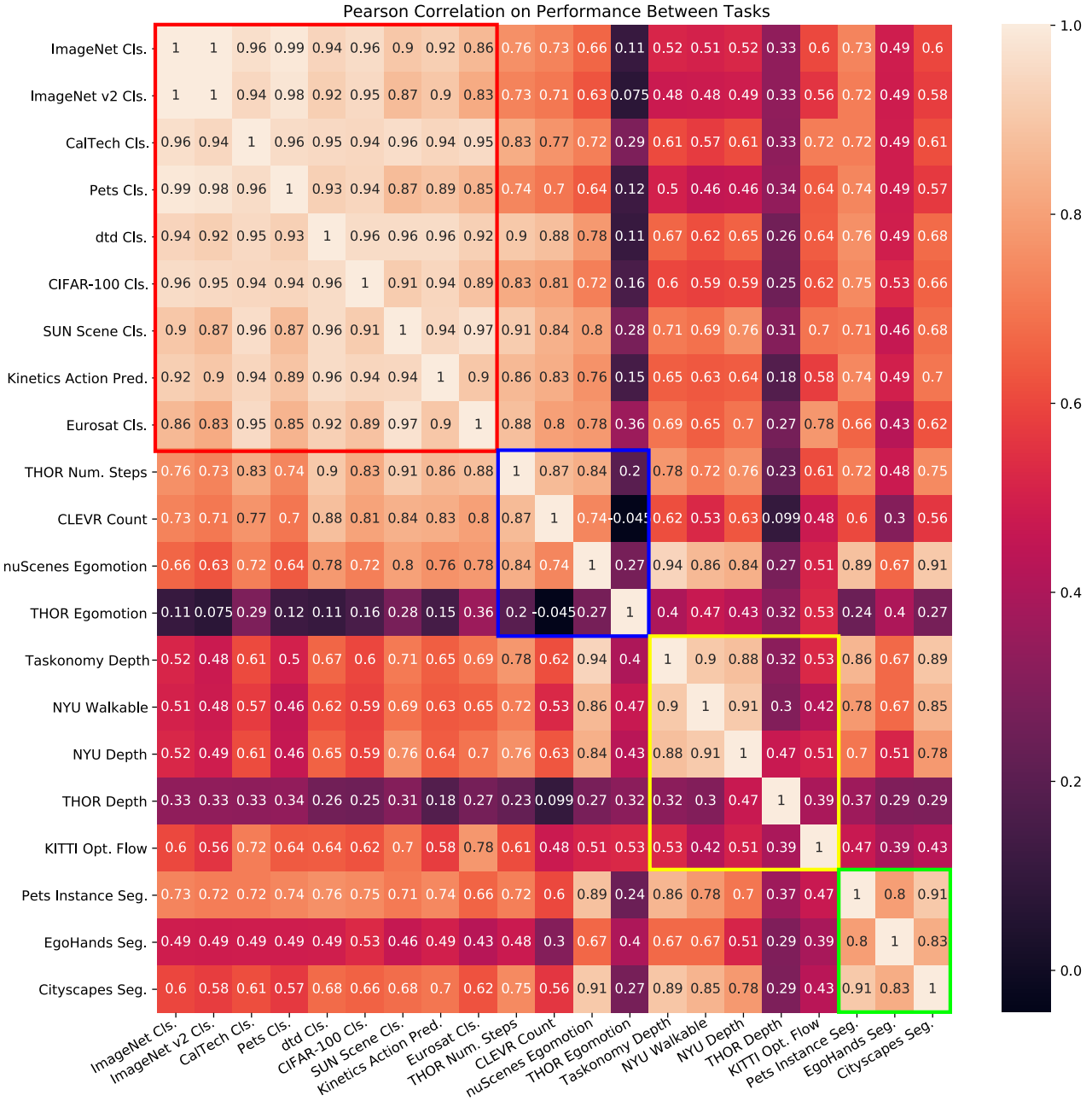

Figure 13. **Pearson Correlation** between the scores of all end tasks obtained using every encoder we study in this paper. Within category correlations for Semantic Image-level, Structural Image-level, Structural Pixelwise, and Semantic Pixelwise tasks are highlighted by red, blue, yellow and green boxes, respectively.

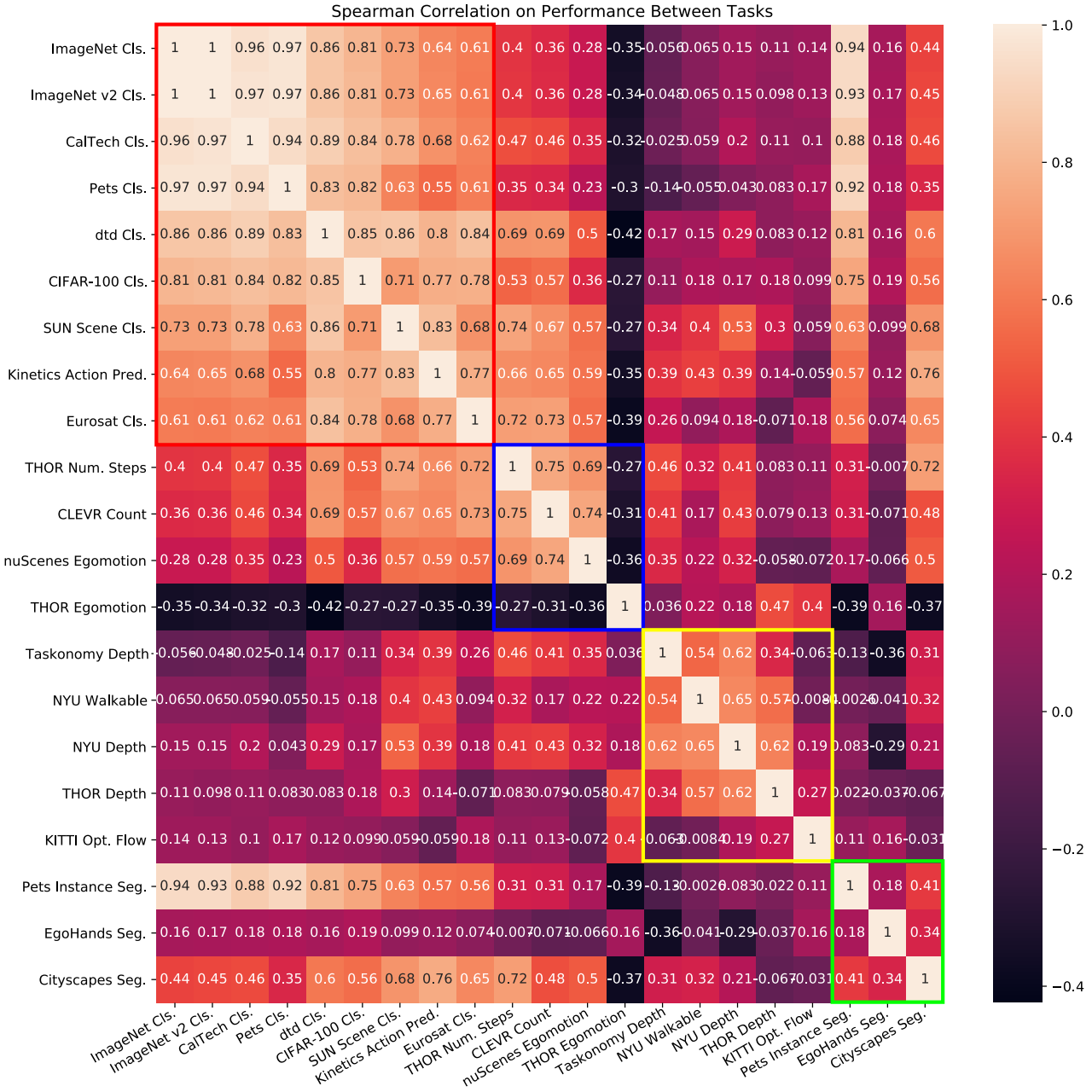

Figure 14. **Spearman Correlation** between the scores of all end tasks obtained using every encoder we study in this paper. Within category correlations for Semantic Image-level, Structural Image-level, Structural Pixelwise, and Semantic Pixelwise tasks are highlighted by red, blue, yellow and green boxes, respectively.

Encoder Name	Method	Dataset	Dataset Size	Number of Epochs	Trained by us
SwAV ImageNet 800	SwAV	ImageNet	1.3M	800	No
SwAV ImageNet 200	SwAV	ImageNet	1.3M	200	No
SwAV ImageNet 100	SwAV	ImageNet	1.3M	100	Yes
SwAV ImageNet 50	SwAV	ImageNet	1.3M	50	Yes
SwAV Half ImageNet 200	SwAV	ImageNet-½	0.5M	200	Yes
SwAV Half ImageNet 100	SwAV	ImageNet-½	0.5M	100	Yes
SwAV Quarter ImageNet 200	SwAV	ImageNet-¼	0.25M	200	Yes
SwAV Linear Unbalanced ImageNet 200	SwAV	ImageNet-½-Lin	0.5M	200	Yes
SwAV Linear Unbalanced ImageNet 100	SwAV	ImageNet-½-Lin	0.5M	100	Yes
SwAV Log Unbalanced ImageNet 200	SwAV	ImageNet-¼-Log	0.25M	200	Yes
SwAV Places 200	SwAV	Places	1.3M	200	Yes
SwAV Kinetics 200	SwAV	Kinetics	1.3M	200	Yes
SwAV Taskonomy 200	SwAV	Taskonomy	1.3M	200	Yes
SwAV Combination 200	SwAV	Combination	1.3M	200	Yes
MoCov2 ImageNet 800	MoCov2	ImageNet	1.3M	800	No
MoCov2 ImageNet 200	MoCov2	ImageNet	1.3M	200	No
MoCov2 ImageNet 100	MoCov2	ImageNet	1.3M	100	Yes
MoCov2 ImageNet 50	MoCov2	ImageNet	1.3M	50	Yes
MoCov2 Half ImageNet 200	MoCov2	ImageNet-½	0.5M	200	Yes
MoCov2 Half ImageNet 100	MoCov2	ImageNet-½	0.5M	100	Yes
MoCov2 Quarter ImageNet 200	MoCov2	ImageNet-¼	0.25M	200	Yes
MoCov2 Linear Unbalanced ImageNet 200	MoCov2	ImageNet-½-Lin	0.5M	200	Yes
MoCov2 Linear Unbalanced ImageNet 100	MoCov2	ImageNet-½-Lin	0.5M	100	Yes
MoCov2 Log Unbalanced ImageNet 200	MoCov2	ImageNet-¼-Log	0.25M	200	Yes
MoCov2 Places 200	MoCov2	Places	1.3M	200	Yes
MoCov2 Kinetics 200	MoCov2	Kinetics	1.3M	200	Yes
MoCov2 Taskonomy 200	MoCov2	Taskonomy	1.3M	200	Yes
MoCov2 Combination 200	MoCov2	Combination	1.3M	200	Yes
MoCov1 ImageNet 200	MoCov1	ImageNet	1.3M	200	No
PIRL ImageNet 800	PIRL	ImageNet	1.3M	800	No

Table 3. The complete list of all 30 encoders used for the study.

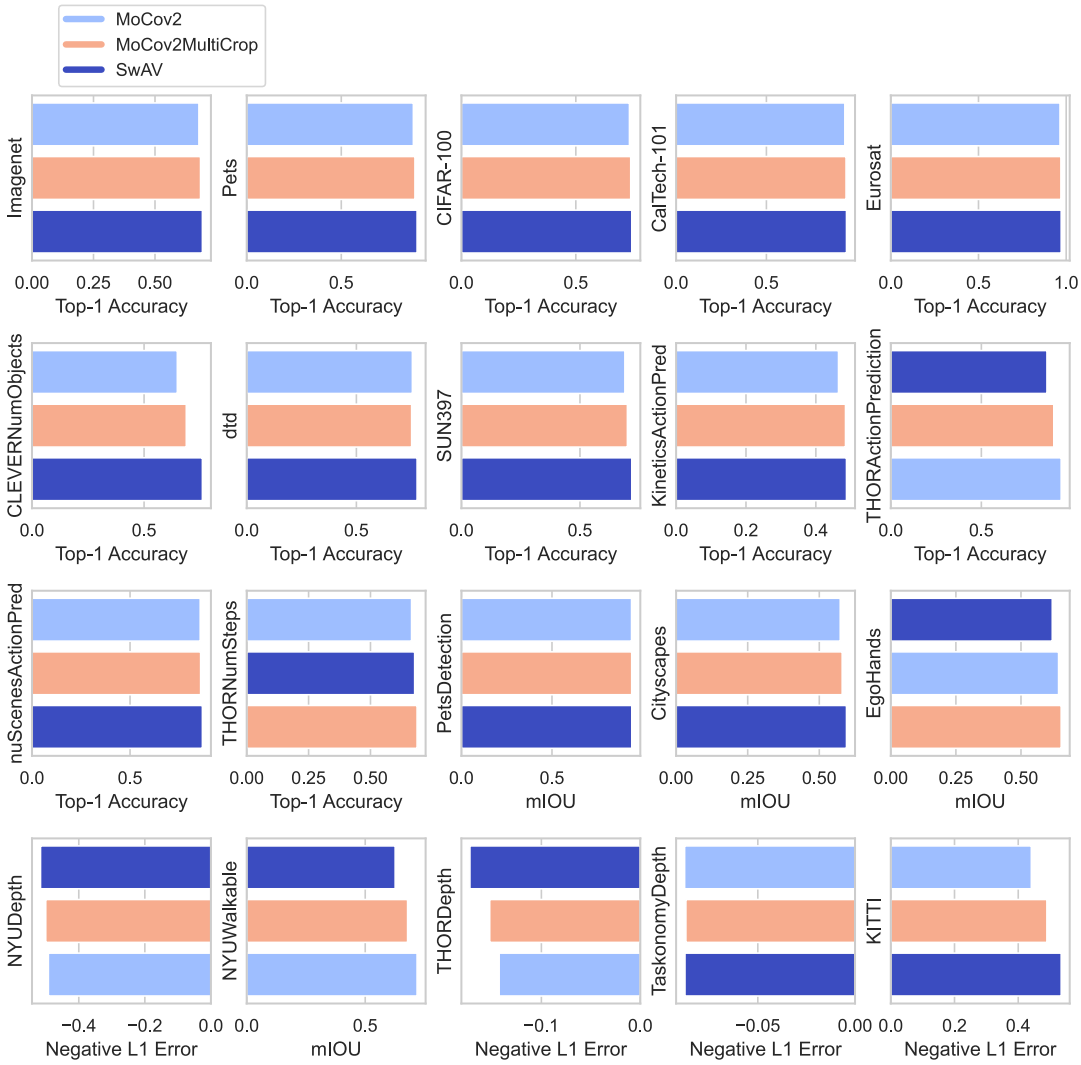

Figure 15. End Task performance of encoders trained on ImageNet using MoCov2, SwAV and MoCov2 with the MultiCrop pre-processing step from the SwAV paper. Different end tasks have different measures of performance but a higher number always indicates better performance. The bars represent the performance of different encoders and are sorted from least to most performant on each end task.

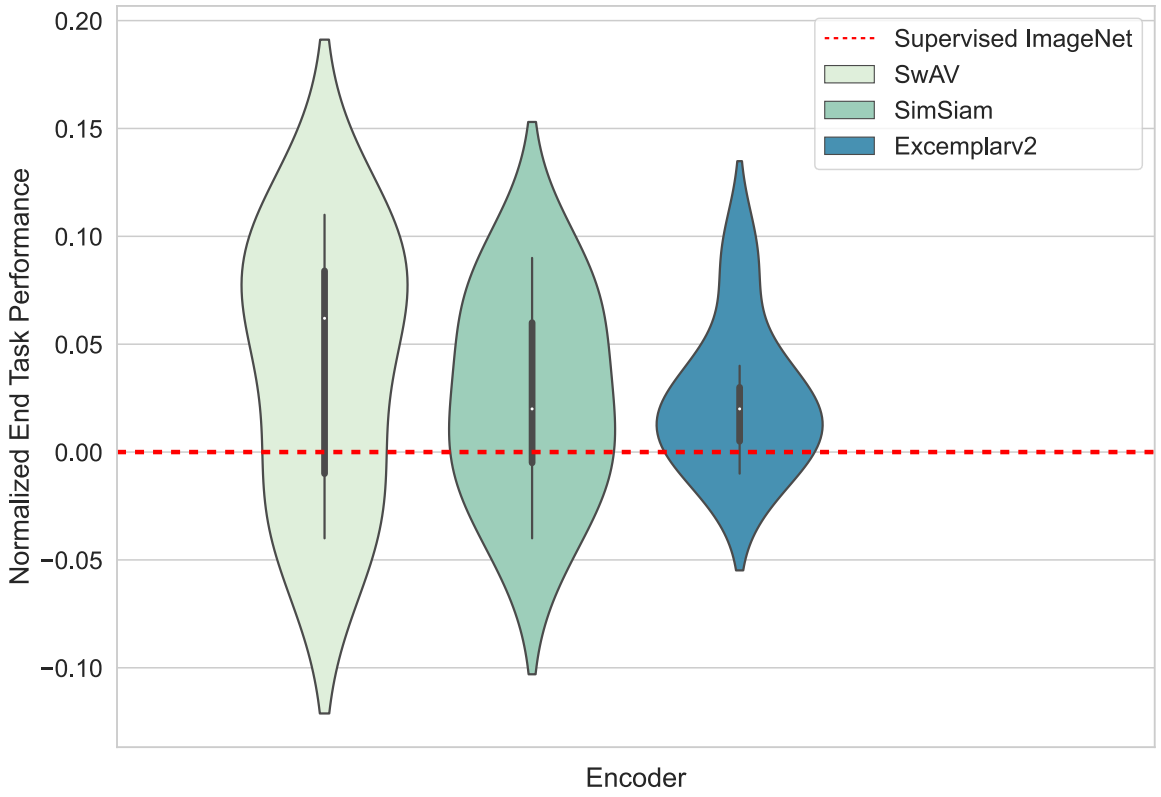

Figure 16. Distribution of normalized performances for the SwAV, SimSiam and Exemplar-v2 encoders trained on ImageNet for 200 epochs. The performances are normalized by first subtracting the performance of the vanilla supervised ImageNet encoder and then dividing by the standard deviation of all the performances for the task. Positive values show superior performance to the vanilla supervised ImageNet encoder, and the negative values show otherwise. A larger width means more performance values fall in that range. The plot shows that SimSiam tends to perform similarly to SwAV. Furthermore the plot shows that Exemplar-v2 performs better than the vanilla baseline, but worse than both SwAV and SimSiam, reinforcing our claims about the outperformance of self-supervised models.