# Differences between 2D and 3D Scene Understanding for Human Action Recognition

Zane Durante
Stanford University
durante@stanford.edu

Yezhen Cong
Stanford University
yzcong@stanford.edu

Zhuoyi Huang
Stanford University
zhuoyih@stanford.edu

## Abstract

*In this work, we focus on comparing the differences between RGB-D and 3D scene understanding for human action recognition to see whether generating 3D point cloud sequence data for the scene will improve the performance of Video Human Action Recognition compared to using 2.5D RGB-D input. We created our own dataset splits on the NTU RGB-D dataset, so that we can simulate scenarios where we only have limited labeled data because the labels are expensive to obtain. We compared both the 2.5D and 3D methods on our few-shot data splits and conducted contrastive learning on unlabeled data using the triplet loss for RGB-D video input, and the simCLR loss for Point Cloud sequence input. The results show that raw RGB-D based methods outperform Point cloud based methods in the few shot activity classification task (accuracies of **0.804** and **0.456**, respectively). However, the point cloud methods perform a bit better than the RGB-D methods when using contrastive learning on unlabeled data. We achieve competitive performance between contrastive pre-training of RGB-D networks and point-cloud sequence networks, with accuracies of **0.486** and **0.506**, respectively. Our codes and models will be made publicly available at* `https://github.com/zanedurante/video_swin_nturgbd` *and* `https://github.com/THU17cyz/PSTNet_CS231A`.

## 1. Introduction

Human Action Recognition in complex scenes is an ubiquitous problem driven by a wide range of applications in many perceptual tasks. Scene understanding in health care settings where physicians, nurses and patients interact with each other and with a variety of medical devices is even more challenging. Two of our team members currently work in the Stanford Program in AI-Assisted Care (PAC), which is a collaboration between the Stanford AI Lab and Stanford Clinical Excellence Research Center that aims to use computer vision and machine learning to create intelligent healthcare spaces. That drives us to be interested in the differences between 2D and 3D Scene Understanding for Human Action Recognition so that we can gain insights for improving the performance of action recognition in clinical settings, since we currently have RGB-D cameras installed in hospitals. For our project, we implemented the RGB-D video and 3D point cloud sequence action recognition in different few-shot training settings, and in settings where we learn from both labeled annd unlabeled data.

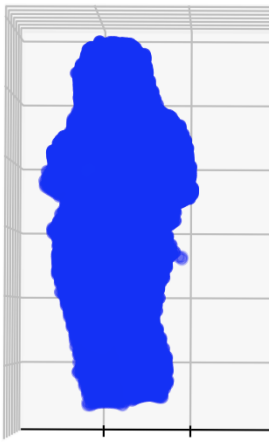
Figure 1: Sample data from NTU-RGB-D dataset. Our task is to do action recognition on the videos.

## 2. Related Work

**2D video models** In most work on 2D video recognition tasks, convolutional networks are used as the standard backbone architectures. Larger video classification datasets such as Kinetics[10] subsequently facilitated the training of spatio-temporal 3D CNNs [2, 8], which have significantly more parameters and thus require larger training

(a) RGB image

(b) Depth image



(c) Point Cloud

Figure 2: Generating Point Cloud from RGB and depth image from NTU-RGB-D dataset, the depth image is processed to be more visible.

datasets. However, recent works leverages the self-attention mechanism to broaden the limited receptive field of the convolution operator. ViViT[1], for example, present a pure-transformer based model for video classification, which extracts spatio-temporal tokens from the input video, and examines four factorized designs of spatial and temporal attention for the pre-trained ViT model. It can effectively regularize the model during training and leverage pre-trained image models to be able to train on comparatively small datasets. Video Swin Transformer[14] uses an inductive bias of locality in video Transformers, which conducted better speed-accuracy trade-off and compute self-attention globally with spatial-temporal factorization.

**3D point cloud models** Deep learning has been widely used in many static point cloud problems such as classification, object part segmentation, scene semantic segmentation[16, 17, 12, 25, 22], reconstruction[27, 5, 11] and object detection[4, 15], but they do not take the temporal dynamics of point clouds into account. Point cloud video modeling is a brand new task, and there are two major categories of methods. The first is performing voxelization, which means converting 3D point cloud frames into 2D bird's view voxels and then extracting features via 3D convolutions. 3DV [24] first employs a temporal rank pooling to merge point motion into a voxel set and then applies PointNet++ [17] to extract the spatio-temporal representation from the set. The second is performing directly on raw points, for example the state of the art methods such as PSTNet[7] construct the spatio-temporal hierarchy to alleviate the requirement of point tracking. P4Transformer[9] aims to avoid point tracking when capturing spatio-temporal correlation across entire point cloud videos.

**Contrastive Learning** Recently, a variety of contrastive learning frameworks have emerged that effectively make use of unlabeled data. The triplet-loss [18] was one of the first of these contrastive frameworks that uses an *anchor* data point, as well as a *positive* and *negative* point in order to learn an embedding space such that positive examples are closer to togther and negative examples are farther apart. More recently, methods such as SimCLR [3] have become more popular than using the triplet-loss and have been shown to get better results in practice. Additionally, contrastive learning has been shown to be effective when applied to video settings as well as point cloud settings [6, 13, 26]. In the point cloud setting, many works adopted a point-based contrastive learning scheme, which constructed positive pairs of points by finding correspondences between augmented copies of point clouds, instead of using augmented copies of point clouds as a whole as positive pairs. PointContrast[26] showed that this point-based contrastive learning scheme also worked well and helped improve the performance of many downstream tasks on point clouds. P4Contrast[13] took a further step by extending PointContrast to taking both 2D and 3D data as input.

It has been observed that contrastive learning often requires large batch size, long training time, and large dataset. Due to computational resource limits, we will explore the performance of contrastive learning with a smaller batch size, moderate training time, and small dataset subset.

# 3. Approaches

## 3.1. Problem Statement

We will investigate the differences in performance between RGB-D and 3D scene understanding models for human action recognition. Specifically, we are interested in understanding the performance of these models under varying amounts of labeled data, and understanding under which conditions RGB-D models outperform 3D models and vice versa. We predict an activity label given either of the two kind of sequence inputs (RGB-D and point cloud).

**Dataset:** In order to investigate differences between RGB and 3D representation models, we will be using the NTU RGB-D [20] 60 dataset. The dataset is the second largest dataset for 3D action recognition. It consists of 56K videos, with 60 action categories and 4M frames in total (around 219 GB of RGB videos and depth videos). This dataset is large enough and we are able to make splits for different settings.

**Data Pre-processing:** After generating 3D point cloud sequences, we use point cloud networks to leverage geometry information explicitly. However, point cloud networks often do not improve much with RGB information, and due to point down-sampling it is harder to learn the pattern.

**Evaluation:** We evaluate our comparison results by Action Recognition Accuracy (cross-subject and cross-setup) metrics. And, we evaluate them in a few-shot setting.

For our technical approach, we considered two different representations for activity classification: RGB-D videos and point-cloud sequences. We describe the two approaches in Section 3.3 and Section 3.4.

## 3.2. Data splits

We created our customized dataset splits which can be visualized in Figure 3. Both the RGB-D and point cloud sequence networks were trained on the **Training Set**, a subset of the cross-subject training set with 50 of the total 60 activity classes, and with 50 examples from each of the 60 activity classes. They were validated on a subset of the official cross-subject NTU-RGBD evaluation set, our **Testing Set**, that has 50 examples of each activity class. Here, cross-subject means that different actors performed the actions than in training set.

After training on this training set, we introduce the **Support Set**, that contains only 5 examples of each of the remaining 10 activities not seen in the training set. The support set is created in order to understand how the model will performed in the few-shot scenario, when only a few examples of the actions are seen during training. The model is then trained on this support set and evaluated on a separate

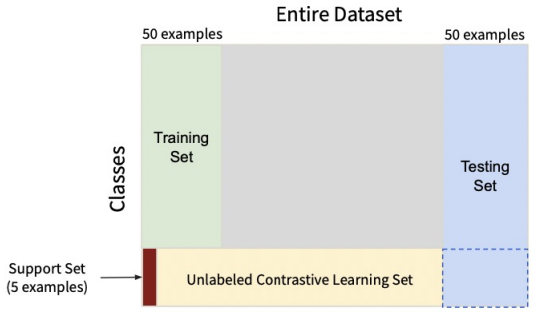



Figure 3: Our customized dataset split for NTU RGB-D dataset: Training Set, Testing Set, Support Set and Unlabeled Contrastive Learning Set.

cross-subject evaluation set containing 50 examples of the action classes of the support set. We use significant data augmentations (flipping, cropping, sampling) to effectively expand the size of the support set. Furthermore, we created a de-labeled subset of the dataset that contains the same 10 classes as the support set with the rest of the samples not used for training (6653 videos), named the **Unlabeled Contrastive Learning Set**.

## 3.3. RGB-D Videos

We process both the depth and RGB streams of the video independently via a depth network and a RGB network and use late fusion (averaging of output logits) for prediction on a given video. For both networks, we use a Video-Swin Transformer [14] backbone (pre-trained on Kinetics-400 [10]). We evaluate the efficacy of keeping these pre-trained weights frozen in Figure 5. In order to use pre-trained weights for the depth network, the depth channel was copied three times in order to match the input size of the RGB network. We keep the original hyperparameters of the Kinetics-400 training set, but decrease the learning rate by a factor of 10.

## 3.4. Point Cloud Sequences

**Generating Point Cloud Sequences** For the dataset NTU RGB-D 60, there's no given camera parameters (intrinsic or extrinsic matrix), so following other's work we used a fixed focal length $f$ where $f = 280$, and utilizing the following formulation to generate point cloud sequences from depth map,

$$\begin{aligned} x_{3D} &= \frac{1}{f}((x_{2D} - \frac{W}{2}) * depth) \\ y_{3D} &= \frac{1}{f}((y_{2D} - \frac{H}{2}) * depth) \\ z_{3D} &= depth \end{aligned} \tag{1}$$

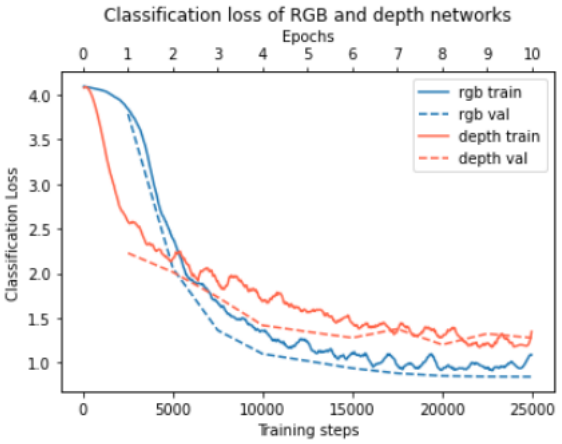



Figure 4: Performance of the depth and RGB networks on the training set.

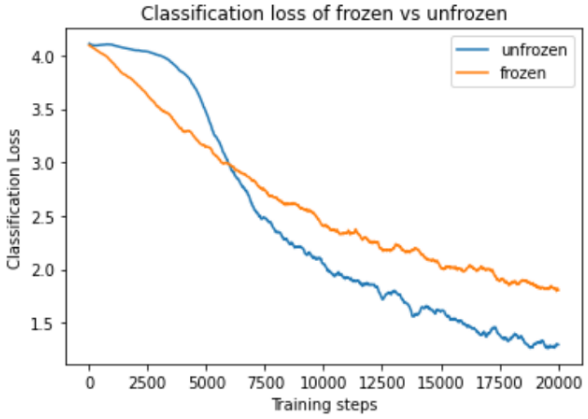



Figure 5: Effects of frozen backbone weights during training. After around 5000 steps, the unfrozen model begins to outperform the frozen model.

In other words, the projection matrix is

$$\begin{bmatrix} f & 0 & \frac{W}{2} & 0 \\ 0 & f & \frac{H}{2} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Where $x_{2D}$, $y_{2D}$ are the pixel position in 2D image, $H$, $W$ are the height and width of the 2D iamge, $z$ is the depth of pixel, and $x_{3D}$, $y_{3D}$, $z_{3D}$ are the corresponding voxel position in point cloud. Figure below showed an example of our generated Point Cloud from the depth data of a frame in a video in our dataset.

**Few-shot Training** After generating the point cloud sequences, we used PSTNet[7] (Point Spatio-Temporal Convolution on Point Cloud Sequences) to extract features of point cloud sequences in a hierarchical manner for 3D action recognition. And we used the same training and testing splits described above in the RGB-D videos to conduct different experiments.

### 3.5. Contrastive Learning Framework

In order to effectively leverage unlabeled data, we "delabel" a large portion of the dataset (6653 examples total) from the same classes of the support set. Thus, for each activity class in the support set, we train on 5 labeled examples and approximately 660 unlabeled examples, and we evaluate on 50 labeled examples.

For RGB-D video contrastive learning, we adopt a framework similar to [19] and use a triplet-loss [18].

For point cloud sequence contrastive learning, we adopt a framework similar to SimCLR [3]. However, SimCLR is used on single image datasets, so it leverages data augmentation to construct positive pairs. We directly construct positive pairs by grouping the images of the same setting but taken under different cameras. Since our estimated camera matrix does not give an aligned set of point clouds, we believe the data augmentation is no longer necessary. Every batch of $2k$ images consists of $k$ distinct positive pairs of images. This gives a similar batch composition to SimCLR. We also adopt the NT-Xent loss from SimCLR, but use different learning schedules tailored to the PSTNet method.

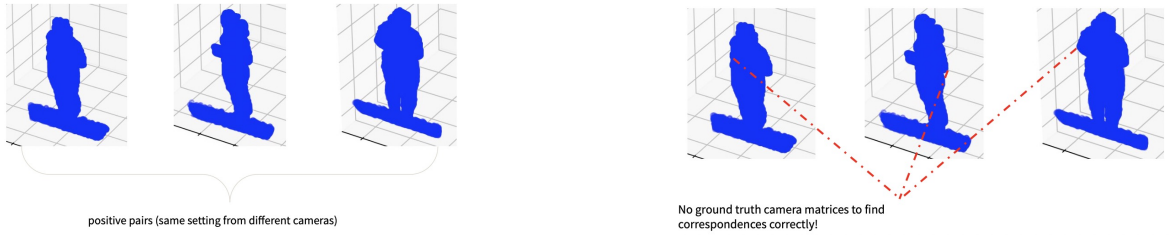
For the NTU-RGB-D dataset, we decided between the two contrastive learning positive pair generation schemes shown in Figure 6. The first is a view-based approach, which means we consider the point cloud sequence of the same setting taken from different cameras as positive pairs. The other is point-based, where we find correspondences of points across different views as positive pairs. Noisy methods to estimate camera matrices may not yield good results for this since we do not have the ground truth camera matrices, and it is hard to obtain correct point correspondences. We were also aware of the success of point-based contrastive learning methods on many downstream computer vision tasks. However, we chose to experiment with the view-based contrastive learning scheme based on SimCLR in this work.

## 4. Experiments

### 4.1. Supervised Learning

#### 4.1.1 Frozen vs unfrozen RGB Backbone

Figure 5 shows the training plots of using a frozen model backbone compared to an unfrozen backbone while training on the training set.




(a) View-based approach

(b) Point-based approach

Figure 6: Two possible contrastive learning approach for 3D point cloud sequence data.

#### 4.1.2 RGB and Depth Network Performance on Training set

Figure 4 shows the training plots of the RGB and depth networks using the training set and the 50 example evaluation set. We importantly note that the models could have continued training, and the loss would have likely decreased. Moving forward we plan to train longer and will likely achieve higher accuracy. The RGB, depth, and fusion networks achieve accuracies of 0.676, 0.599, and 0.728 respectively on the 50-class evaluation set.

#### 4.1.3 RGB and Depth Network Performance on Support Set

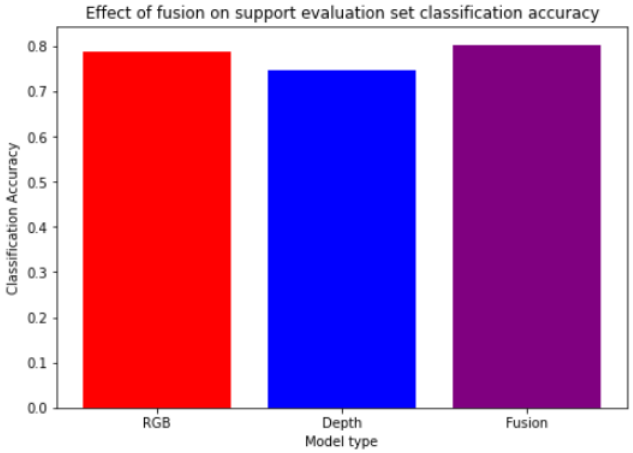



Figure 7: Evaluation performance on the 10-class few-shot evaluation set. Performance increases when the RGB and depth modalities are fused. The RGB, depth, and fusion models achieved 0.788, 0.756, and 0.804 accuracy respectively.

Training on the support set is much noisier than the training set, due to the small number of labels and the model quickly over-fitting the data. However, one key advantage is that there are only 10 classes on the support set. Thus,

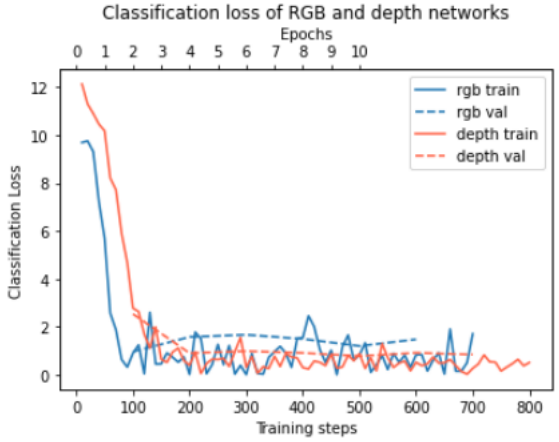



Figure 8: Training performance of the depth and RGB networks on the 10-class few-shot training set and the cross-subject support evaluation set.

although there are fewer examples than in the training set, there are also fewer classes (making the classification problem easier). Figure 8 shows the performance of the depth and RGB models on the support set.

Notably, the RGB-D networks achieve higher classification accuracy on the support set (likely due to the smaller number of classes). The RGB, depth, and fusion networks achieve accuracies of 0.788, 0.756, and **0.804** respectively on the 10-class support evaluation set.

#### 4.1.4 Point Cloud Few-shot Performance

Table 1 shows the experiment results of PSTNet on support validation set under different few-shot training settings. Finetuning only last fully connected layer yields a better result, suggesting that this may alleviate over-fitting on support train set.

| Method | Training Settings | Finetune only last fc layer? | Top 1 Acc |
|---|---|---|---|
| PSTNet[7] | Only training on support train set | | 18.95 |
| | First training on train set | | 38.10 |
| | First training on train set | ✓ | **45.36** |

Table 1: Few-shot action Recognition Performance on support valid set under different settings. Finetune only last fc layer means that when training on support train set, we freeze all weights except for the weights of the last fc layer.

| Method | Training Settings | Finetune only last fc layer? | Top 1 Acc |
|---|---|---|---|
| PSTNet[7] | Only training on support train set | | 18.95 |
| | First pre-training (CL) on unlabeled set | | **50.60** |
| | First pre-training (CL) on unlabeled set | ✓ | 27.82 |
| | First pre-training (CL) on unlabeled set, then training on train set | | 38.51 |
| | First pre-training (CL) on unlabeled set, then training on train set | ✓ | 42.94 |

Table 2: Action Recognition Performance on support valid set under different contrastive learning. Finetune only last fc layer means that when training on support train set, we freeze all weights except for the weights of the last fc layer.

## 4.2. Contrastive Learning

### 4.2.1 RGB-D Contrastive Learning

To determine the efficacy of contrastive learning for improving RGB-D feature extraction, we initialize the network with the weights learned from the training set. Afterwards, we further train the network on the contrastive learning set (6653 samples) using a triplet loss.

In order to evaluate our feature extraction networks for RGB-D videos, we use a combination of qualitative and quantitative techniques. Qualitatively, we visualize our RGB feature extractors (after fine-tuning on the training set, before contrastive learning and after contrastive learning) using t-SNE [23] in figures 9 and 10. Quantitatively, we calculate downstream classification performance via fine-tuning and by using prototypical networks for few-shot classification [21]. We show the performance of both prototypical networks and downstream fine-tuning in table 3.

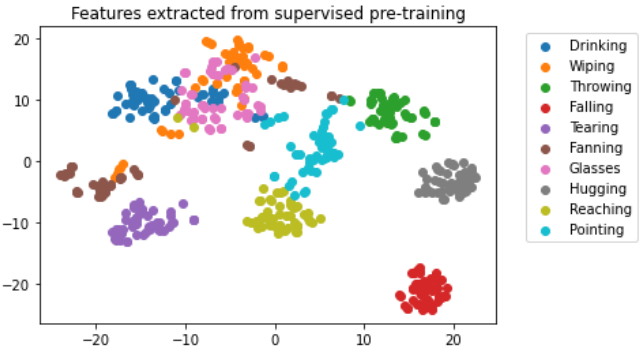



Figure 9: Here we visualize the features extracted from the supervised pre-trained model (pre-trained on 50 examples of 50 activity classes) using t-SNE [23]. We visualize the feature-space of the remaining 10 unseen classes below, and show promising separation between the classes that leads to good downstream classification (acc@1=0.788).

### 4.2.2 Contrastive learning on Point Clouds

Figure 11 gives a glance of the training process for contrastive learning on unlabelled data split. The evaluation results on support dataset split of the different training settings for point cloud sequence action recognition are shown in table 2. We can see that contrastive learning alone or training on the train set alone benefits the performance on support set. However, including both components do not bring further performance boost but instead hurts the performance. We think this is because a better representation obtained by contrastive learning and training on the train set undesirably leads to easier over-fitting.

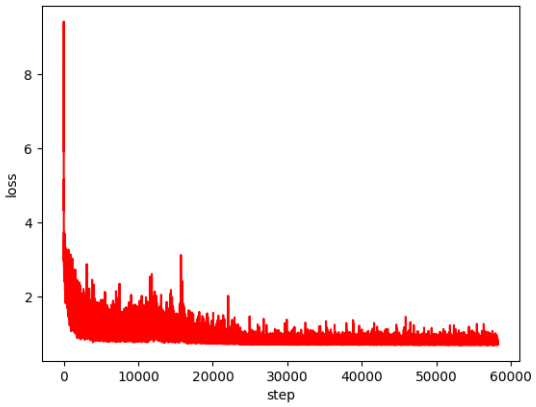



Figure 11: Loss curve of the contrastive learning process on point clouds

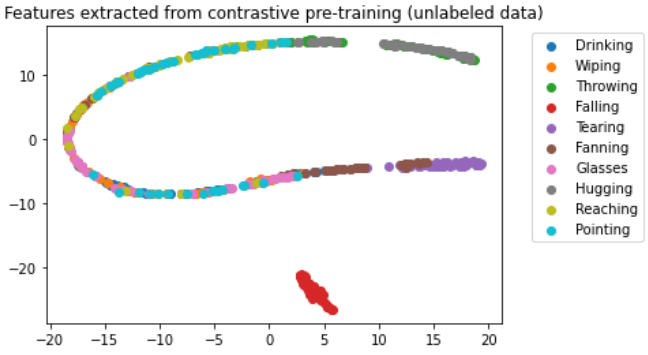



Figure 10: Here we visualize the features extracted from the contrastive self-supervised model (trained on thousands of unlabeled examples of the 10 activity classes in the support set) using t-SNE [23]. We show poor separation, as most of the classes lying on the same manifold with an exception for the "falling down" class. This visualization supports our findings of poor classification accuracy when using 2D contrastive learning.

Prototypical Network Classification accuracy

| | SL | SL-FT | CL | CL-FT |
|---|---|---|---|---|
| RGB | 0.744 | 0.788 | 0.204 | 0.480 |
| Depth | 0.712 | 0.756 | 0.196 | 0.356 |
| Fusion | 0.746 | 0.804 | 0.204 | 0.486 |

Table 3: **SL** indicates supervised pre-training on labeled data of other classes, whereas **CL** indicates contrastive pre-training on unlabeled data of the same classes. Results after supervised fine-tuning on the support set are indicated by **FT**. Empirically, these values are close to the final results for supervised pre-training and farther from the final results for contrastive learning, indicating the greater need for extensive fine-tuning after contrastive pre-training.

## 5. Conclusion

Overall, we see competitive performance between contrastive pre-training of RGB-D networks and point-cloud sequence networks, achieving accuracies of 0.486 and 0.506. Some of this disparity may be due to the fact that the point-cloud sequence network uses SimCLR [3], whereas the RGB-D network uses the triplet-loss [18].

For our RGB-D models, we found that using contrastive learning leads to worse feature extractors than supervised pre-training, both in terms of the separation of the embedding space (as observed by the t-SNE visualizations in figures 9 and 10) and downstream classification accuracy.

However, the greatest support and training set performance was achieved by using the Video-Swin RGB-D model pre-trained on Kinetics-400. These pre-trained weights provide a unique advantage for activity recognition that cannot be used by the point-cloud network. It is ultimately for these reasons that we achieve best performance for both the regular and few-shot activity classification when using 2D RGB-D late fusion networks.

In the future, we could use SimCLR instead of triplet loss for RGB-D video method.

## References


[1] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lucic, and C. Schmid. Vivit: A video vision transformer. *CoRR*, abs/2103.15691, 2021.

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. *CoRR*, abs/1705.07750, 2017.

[3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[4] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. *CoRR*, abs/1611.07759, 2016.

[5] A. Dai, A. X. Chang, M. Savva, M. Halber, T. A. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *CoRR*, abs/1702.04405, 2017.

[6] I. Dave, R. Gupta, M. N. Rizve, and M. Shah. Tclr: Temporal contrastive learning for video representation. *arXiv preprint arXiv:2101.07974*, 2021.

[7] H. Fan, X. Yu, Y. Ding, Y. Yang, and M. Kankanhalli. PSTNet: Point spatio-temporal convolution on point cloud sequences. In *International Conference on Learning Representations*, 2021.

[8] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal residual networks for video action recognition. *CoRR*, abs/1611.02155, 2016.

[9] Z. Hang, Y. Wang, and S. Huang. P4 transformer: Towards unified programming for the data plane of software defined network. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 544–551, Los Alamitos, CA, USA, jul 2021. IEEE Computer Society.

[10] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[11] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. Pugan: A point cloud upsampling adversarial network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[12] Y. Li, R. Bu, M. Sun, and B. Chen. Pointcnn. *CoRR*, abs/1801.07791, 2018.

[13] Y. Liu, L. Yi, S. Zhang, Q. Fan, T. Funkhouser, and H. Dong. P4contrast: Contrastive learning with pairs of point-pixel pairs for rgb-d scene understanding. *arXiv preprint arXiv:2012.13089*, 2020.

[14] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu. Video swin transformer. *CoRR*, abs/2106.13230, 2021.

[15] C. R. Qi, O. Litany, K. He, and L. J. Guibas. Deep hough voting for 3d object detection in point clouds. *CoRR*, abs/1904.09664, 2019.

[16] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016.

[17] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *CoRR*, abs/1706.02413, 2017.

[18] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[19] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.

[20] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1010–1019, 2016.

[21] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[22] H. Thomas, C. R. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. *CoRR*, abs/1904.08889, 2019.

[23] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[24] Y. Wang, Y. Xiao, F. Xiong, W. Jiang, Z. Cao, J. T. Zhou, and J. Yuan. 3dv: 3d dynamic voxel for action recognition in depth video. *CoRR*, abs/2005.05501, 2020.

[25] W. Wu, Z. Qi, and F. Li. Pointconv: Deep convolutional networks on 3d point clouds. *CoRR*, abs/1811.07246, 2018.

[26] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *European conference on computer vision*, pages 574–591. Springer, 2020.

[27] L. Yu, X. Li, C. Fu, D. Cohen-Or, and P. Heng. Ec-net: an edge-aware point set consolidation network. *CoRR*, abs/1807.06010, 2018.