Implicitly using Human Skeleton in Self-supervised Learning: Influence on Spatio-temporal Puzzle Solving and on Video Action Recognition

Mathieu Riand[1,2], Laurent Dollé[1] and Patrick Le Callet[2]
[1]*CEA Tech Pays de la Loire, 44340 Bouguenais, France*
[2]*Equipe Image Perception et Interaction, Laboratoire des Sciences du Numérique de Nantes, Nantes University,*

Keywords: Self-supervised Learning, Siamese Network, Skeleton Keypoints, Action Recognition, Few-shot Learning.

Abstract: In this paper we studied the influence of adding skeleton data on top of human actions videos when performing self-supervised learning and action recognition. We show that adding this information without additional constraints actually hurts the accuracy of the network; we argue that the added skeleton is not considered by the network and seen as a noise masking part of the natural image. We bring first results on puzzle solving and video action recognition to support this hypothesis.

1 INTRODUCTION

Action recognition (Yamato et al., 1992) can provide various informations for a robot-related task, like giving an action plan by cutting a video demonstration in multiple sub-videos, each corresponding to a lower-level action. However, action recognition is a task that requires a consequent amount of annotations which can be laborious to produce.

Meanwhile, in the recent years, self-supervised learning (Chen et al., 2020) has emerged as a convenient solution to the global lack of annotated data; indeed, there is a large amount of available raw data that are widely collected from many sources (YouTube videos, for instance), but the number of annotations remains very small in comparison. Self-supervised learning then offers a way of reducing the number of needed annotations for action recognition, by first pretaining the models on a pretext task in which labels are automatically obtained from the data; this pretraining allows the model to grasp a representation of the data in its intermediate layers that is hopefully helpful for the target task, action recognition.

Finally, skeleton data has appeared to be very useful for action recognition (Li et al., 2017a); architectures taking sequences of body poses as input have been designed in order to solve this problem, and pretext tasks based on skeleton also have been proposed for self-supervised learning.

In this work, we tried to add the skeleton as an implicit information in the data, by adding it on top of the videos, and mesured the influence of this added signal on two tasks : puzzle solving, a self-supervised task, and action recognition, trained from scratch. In the first section, we will go over works related to action recognition and self-supervised learning. Then, we will introduce our method and share our first results. Finally, we will make some remarks on the presented results and conclude.

2 RELATED WORK

In this section, we review works covering action recognition and the use of self-supervised learning in this context. We will then focus on the specific pretext task that jigsaw puzzles represent, and finally end by discussing the different uses that have been made when it comes to human skeleton data.

2.1 Action Recognition

Action recognition (AR) is the task that consists in assigning an action label to a video clip (Jhuang et al., 2013) or a specific location of it, also called action classification.

Most approaches are applied on video datasets such as HMBD51 (Kuehne et al., 2011); the used architectures are thus of the same kind as the ones used for classical image classification, namely Convolutional Neural Networks (CNNs). (Tran et al., 2018)

Riand, M., Dollé, L. and Le Callet, P.
Implicitly using Human Skeleton in Self-supervised Learning: Influence on Spatio-temporal Puzzle Solving and on Video Action Recognition.
DOI: 10.5220/0010689500003061
In *Proceedings of the 2nd International Conference on Robotics, Computer Vision and Intelligent Systems (ROBOVIS 2021)*, pages 128-135
ISBN: 978-989-758-537-1

proposed to use 3D convolutional networks, or 3DCNNs, and showed that they were suited for performing action recognition on videos. Moreover, they built a spatiotemporal convolutional block usable in CNNs instead of classical convolutions to boost their performance on datasets such as HMBD51.

(Girdhar and Ramanan, 2017) introduced attentional pooling as an alternative pooling layer in CNNs; this was a way of linking attention concepts and action recognition, and it actually yielded great improvements over state-of-the-art AR benchmarks.

Finally, works such as (Yan et al., 2018) used skeleton sequences instead of simple videos; this allowed to use a Graph Convolutional Network (GCN) that brougth better results for action recognition over classical methods such as CNNs.

2.2 Self-supervised Learning

Self-supervised learning is a branch of unsupervised learning that has been widely explored in recent years; it is unsupervised in the sense that the supervisory signal is automatically extracted from the data along the inputs. This allows to train a neural network on what is called *a pretext task* (or pseudo-task), which can take various forms that we will detail here; the idea behind this is to find a task that, in order to be solved, needs high-level concepts to be grasped by the intermediate layers of the network. The same network can thus be fine tuned on the target task (see Figure 1), for instance action recognition, hoping that the concepts learned during pretraining allow a better performance of the network compared to a random initialisation of its weights.

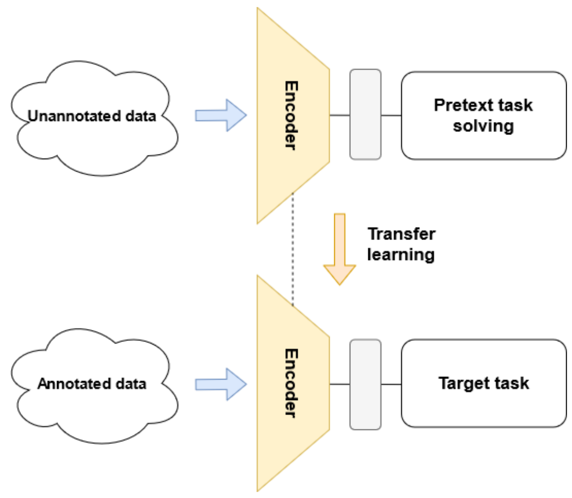


Figure 1: A classical self-supervised learning setup; an encoder is first trained on a pretext task with unannotated data, then fine tuned on the target task with annotations.

For action recognition, most datasets are built using images or videos; this means that most pretext tasks are visual tasks. One of them is colourisation, which was first introduced in (Zhang et al., 2016); the objective here is to preemptively transform a colour image into its grayscale counterpart, and train the network to output coloured pictures given the grayscale ones. Features learned through colourisation often prove useful for target tasks such as object detection.

(Vondrick et al., 2018) and (Li et al., 2019) have used colourisation in a different way on videos; a reference frame is given as input along another one in grayscale later in the video. The pretext task is still to recolour the image, but the network has a support frame to get information from. Those works have shown that this colourisation task led the network to learn trackers intrisically, especially for objects. However, this kind of tasks fails to grasp essential concepts in videos, since the temporal dimension is not taken into account.

In this spirit, works like (Sumer et al., 2017) proposed to learn jointly temporal and spatial cues; in this work, the network tries to predict if two cropped bounding boxes from a video are close to each other in time. Other temporal pretext tasks based on video frames include complete sequence ordering, where the system must output the permutation that has been done on multiple consecutive frames, as in (Lee et al., 2017), or more simply output if the given sequence is indeed ordered or not, like in (Misra et al., 2016).

(Sumer et al., 2017) rely on contrastive learning, which is another complete branch of self-supervised learning, recently brought up to date by the work of (Chen et al., 2020); in this application, random data augmentations are applied to pictures from ImageNet, like color distortion, gaussian noise, or even masking a part of the image. The network is then trained to predict if two transformed pictures come from the same original image. The principle behind contrastive learning is to build positive and negative pairs that the system must try to differentiate; in (Sumer et al., 2017), positive pairs are the ones that contain temporally close frames. Pairs can also be formed with object detection results, as in (Pirk et al., 2019); positive pairs are attracted in the representation space, while they are pushed away from the negative ones. This training naturally converges towards a disentangled representation of objects, where similar objects are close to each other.

Pretext tasks are sometimes directly applied to intermediate results obtained on the data, like object proposals in (Pirk et al., 2019), or human body keypoints detection; (Wang et al., 2019) proposes to use statistics related to motion and appearance in videos, for instance the position of the region with the largest motion. Those statistics are extracted from videos, and then used as labels; the network must predict in which area of the video we can find any of those labels. This way, a representation of the data sensitive to movement and appearance is learned.

In (Lin et al., 2020) several pretext tasks based on human skeleton detection results are proposed, and solved at the same time; for example, given a temporal sequence of skeleton poses, the network is asked to predict the end of the sequence while seeing only the beginning.

It is noteworthy that while a vast majority of works related to self-supervised learning focuses on designing new pretext tasks or new ways to combine them, some works like (Kolesnikov et al., 2019) focus on the model used to perform the pretext task, and on the influence of this model on the quality of the representation learned. They notably showed that better results on a pretext task do not necessarily translate to a better representation of the data neither to better results on the target task.

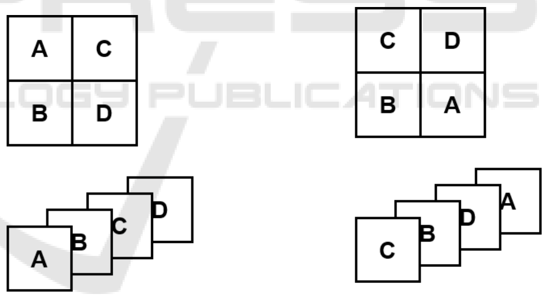
Finally, (Su et al., 2020) has shown that self-supervision is very beneficial to few-shot learning, especially when the pretext task is very complex, and that using more unlabelled data for pretraining is useful only if they come from the same domain as the ones used for the few-shot task. This is partly why in this work we used the same dataset for the self-supervised task that the one on which we want to perform action recognition.

2.3 Jigsaw Puzzles as a Pretext Task

The pretext task used in this work is the solving of jigsaw puzzles; this task consists in dividing an image in several same-sized patches, and in shuffling them. When solving spatial puzzles, humans use their knowledge about how objects look, where they are generally located, etc; when solving temporal puzzles, they can use informations related to movements, such as their direction; by teaching a network to solve puzzles, it is expected that it learns such useful concepts in the process. In our case, the network must predict which permutations have been done on the video patches; this brings the problem back to a classification problem, where the system must output a class corresponding to the way the puzzle pieces have been shuffled (for instance, for 4 pieces, 4! = 24 permutations are possible, thus we will have 24 possible classes). The shuffling can be done spatially, or temporally if applied to a video, which is equivalent to reordering a video sequence as in (Lee et al., 2017).

Early works like (Noroozi and Favaro, 2016) tend to use still images to solve spatial puzzles; the convolutional neural network (CNN) trained on puzzle solving is then repurposed to solve an object detection task, for instance. But with the recent growing interest for video data and action recognition, the focus switched on puzzles applied to image sequences, called *spatio-temporal puzzles*. Different approaches are defended when it comes to spatio-temporal jigsaw puzzles; on one hand, some works such as (Ahsan et al., 2018) define them as a unique task, where both spatial and temporal shuffling are done simultaneously. More precisely, 3 frames of the video are taken as inputs, and each one is cut in 4 patches; the simplest method would be to simply shuffle those 12 patches, and train a CNN to reorder them back, but it is a very hard and underconstrained task. In their work, the patches are first shuffled in their own frame, and then the frames are shuffled together; we still have a spatio-temporal shuffling, but the puzzle solving is made simpler thanks to those additional constraints.

On the other hand, works like (Kim et al., 2018) create separately temporal and spatial puzzles, but they feed them to the same network. Permutation classes are shared across both types of puzzles (see Figure 2), so the network is forced to discover and use features that are useful to both sides of the wanted spatiotemporal representation.



(a) Original sample. (b) Permuted sample.

Figure 2: Shared classes between spatial (top) and temporal (bottom) puzzles. Permutations are equivalent between the two types of puzzles as presented here : top-left in spatial is equivalent to first in temporal, bottom-left to second, etc.

2.4 Skeleton Data Usage

In this last part, we will cover a few works that used skeleton data as their main input.

The most simple way to feed a skeleton in a network is to simply consider it as numerical data (Li et al., 2017b), namely the coordinates of each keypoint, that can then be processed by a network to predict action labels, for instance. (Lin et al., 2020) also used raw skeleton sequences as inputs for self-supervised learning.

Other works such as (Shi et al., 2019) and (Yan et al., 2018) take advantage of the fact that skeletons are naturally graphs. In (Shi et al., 2019), motion information is also encoded in a similar graph structure; both graphs are fed in a two-stream Graph Convolutional Neural Network (GCNN). In (Yan et al., 2018), the graph has two kinds of edges : intra-body edges link body joints together, and inter-frame edges link similar joints across time. This way a compact graph representing the whole skeleton sequence is obtained and can be processed by a Spatial Temporal GCNN.

Finally, as in (Li et al., 2017a) or (Wu et al., 2019a), the whole skeleton video sequence can be converted to one single RGB image; this image can then be processed by traditional CNNs to perform action recognition on them. For instance, each row of the image can represent one frame, each column a joint of the skeleton, and RGB values are for the XYZ coordinates; this 2D representation can also be learned.

3 METHOD

3.1 Spatio-temporal Puzzles

Our work draws its main inspiration from (Kim et al., 2018); from a set of video demonstrations performed by different actors, we extract spatial and temporal puzzles composed of 4 pieces. We generate a number of puzzles proportional to the duration of each video. When created, a jigsaw puzzle can either be temporal or spatial, each with a 50% probability. In our work, a puzzle piece is an image, but we also ran some experiments on videos; in both cases, the first step in the puzzle generation process is to pick a reference frame that serves as a starting point in time for puzzles. Then, the video is spatially cut in 4 corners of the same size, and is also temporally cut every 2 seconds, as presented in Figure 3; this forms what (Kim et al., 2018) called *space-time cuboids*. Finally, we apply a random permutation among the 24 available to those cuboids.

Puzzles are fed in a 4-headed siamese-like network (see Figure 7), where each head is a CNN/3DCNN that extracts features from the images/videos, respectively; as in a siamese network, weights are shared between all 4 heads. The obtained outputs are then concatenated, and a final fully connected layer allows the network to make a prediction regarding the class of the puzzle.

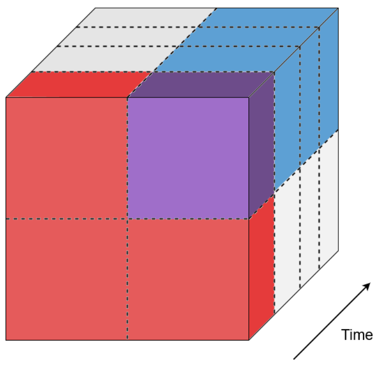


Figure 3: Extracting puzzles from videos; in our method, videos are cut both spatially and temporally in cuboids. Then we can chose between creating a spatial puzzle with the red cuboids, or a temporal puzzle with the blue ones.

The main difference of this work compared to other puzzle-based methods is that we added the results of a body keypoint detector applied to every frame of the videos directly on the image (see Figure 5 and Figure 6); more precisely, we draw the human skeleton on top of each video frame when an actor is present. This is what we called *implicitly* adding skeleton information on the image.

3.2 Additional Processing

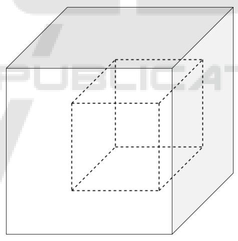


Figure 4: Subsampling of a puzzle piece : given a video clip, we select a spatial crop of the video, and only keep half of its duration.

As in most self-supervised learning works, we applied some transformations to the data in order to keep the network from learning using low-level cues without any semantical meaning. First, as used in (Lee et al., 2017), we performed channel replication on the images; this method consists in randomly chosing one of the color channels and copying its values to the other 2 channels (we can also perform channel splitting, where we only keep one channel). Classical grayscale images depend on all 3 colors, contrarily to the ones obtained through channel replication; as shown in (Lee et al., 2017), using gray images makes the task more challenging for the network, forcing it to discover features that are unrelated to color, and improving its results on action recognition.

As in (Kim et al., 2018), we also randomly subsample all of our puzzle pieces, both in time and space (Figure 4); this causes the cuboids to not be aligned in both dimensions, which in turn keeps the network from learning by just aligning edges of objects or shapes, which holds very little interest as features learned through self-supervised learning.

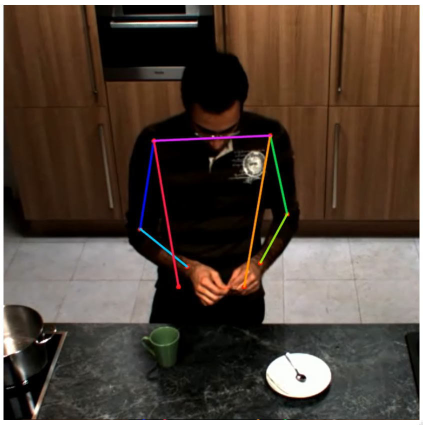


Figure 5: Human skeleton keypoints and edges added on top of a MPII Cooking 2 Dataset video frame.

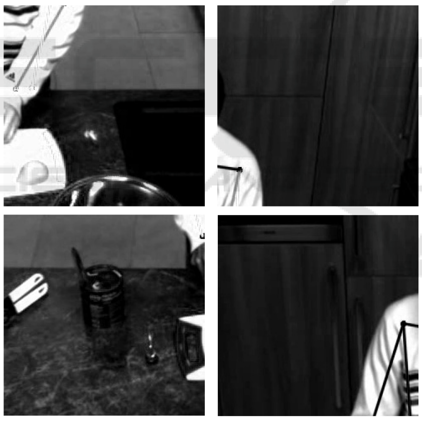


Figure 6: An example of a spatial puzzle used in our self-supervised learning experiments; both actor's shoulders are on the right pieces; top-left piece contains actor's left arm.

3.3 Transfering Knowledge

Our target task is video action recognition, which in our case could also be named *video label prediction*; usually, when performing transfer learning from a self-supervised pretrained network, as in (Kim et al., 2018), only one of the 4 heads is kept to act as an encoder for the target task, and the final layers are replaced to match the new dimension of the output. This means that only one video clip is fed into the network in order to predict the action that is performed in this precise timelapse.

We take a different approach to this *video label prediction* problem; rather than predicting the action performed in the video clip, we want to be able to predict the global action performed in the whole video. In order to do this, each video in the dataset is separated in four parts of equal size, and an image (or a video clip) is randomly extracted from each of those parts. This allows us to get 4 clips that are representative of the whole video; we can repeat this operation any number of time to generate several quadruplets for one video, which is our way of generating annotated data for action recognition at no cost. The 4 clips can then be fed in the same network that was used for spatio-temporal puzzle solving, with the last layer being changed to fit the number of action classes in the dataset.

4 EXPERIMENTS

4.1 Dataset

For our experiments, we used the MPII (Max Planck Institute for Informatics) Cooking 2 Dataset (Rohrbach et al., 2015); it consists of 273 videos of actors performing recipes in a kitchen. 59 different recipes are available, such as ”pouring a beer” or ”preparing a salad”; given the limited number of videos, each recipe is performed between 1 and 11 times by a different actor. This was one of the reason we chose this dataset; it allows us to set ourselves in a context of few-shot learning, where very few examples are available for each action, leading to the quadruplets generation we discussed before. Also the level of the annotations in the MPII Cooking 2 Dataset was perfectly suited for an action recognition task in a robotic context : the annotations consist in high-level manipulation tasks, not in atomic actions (”grasp”, ”drop”...) nor high-level human actions that involves the whole human body (”playing basketball”, ”dancing”...).

Regarding the augmentation of this dataset with human skeleton data, we used the PyTorch implementation of Keypoint R-CNN (Wu et al., 2019b), available in the *torchvision* package, in order to perform skeleton keypoints detection on each video frame, as shown in Figure 5. Then the keypoints and skeleton edges were drawn on top of the videos using OpenCV (Bradski, 2000); videos were cropped around the center of the skeleton in a 640*640 pixels window. When creating the puzzle pieces, each cuboid was then a 320*320*120 video (the temporal dimension is mesured in frames); after subsampling, each puzzle piece was reduced to a 240*240*60 area, namely

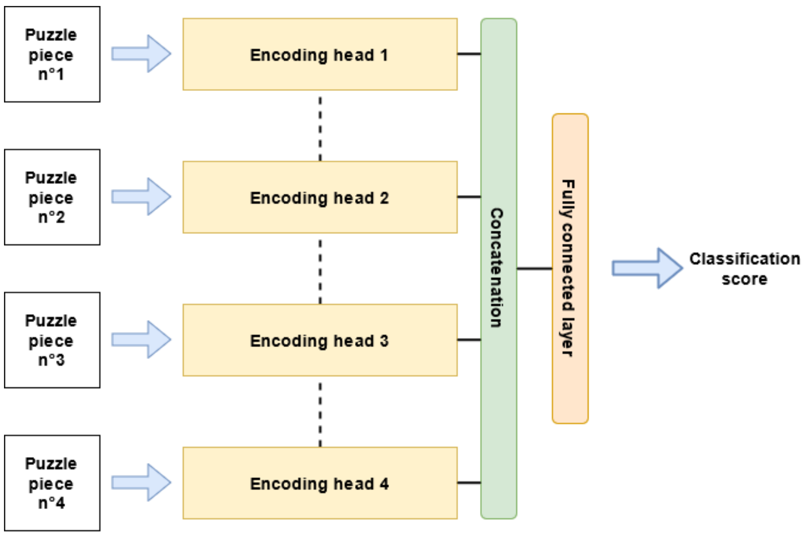


Figure 7: The typical architecture used in this work; each head takes a puzzle piece as input and extracts features from it. All four heads' outputs are then concatenated, and used by a fully connected layer to make a prediction regarding the class of the puzzle.

2 seconds long video clips, and channel replication was applied, resulting in puzzles such as the one presented in Figure 6. We created around 13000 puzzles both with and without skeleton from the MPII Cooking 2 Dataset; the 2 sets of puzzles were identical, the only difference being the presence of the skeleton on the image for the first set.

4.2 Models

As mentionned, we used a siamese-like architecture (Figure 7) to process spatio-temporal puzzles. In the case where images were used, each head was composed of 6 2D convolutional layers (with respectively 32, 64, 128, 256, 512 and 512 filters); kernel size was (3,3) and padding was set to 1. Each layer was followed by a LeakyReLu activation function, and a max pooling operation : from layers 1 to 4, filter size was (2,2), and it was (3,3) for convolutional layers 5 and 6. After concatenation of the outputs of the 4 heads and the fully connected layer, a softmax layer was applied to output classification scores. The model was implemented using PyTorch.

4.3 Results

All trainings reported here have been run for 500 epochs.

We first present the mean accuracy obtained on the testing set for the spatio-temporal puzzle solving task, reported in Table 1, for different learning rate values.

As we can see, there is no major improvement in terms of accuracy from adding the skeleton; for high

Table 1: Final mean accuracy on test set for spatio-temporal puzzle solving (lr = learning rate).

	With skeleton	Without skeleton
lr = 0.1	0.292	0.325
lr = 0.02	0.300	0.308
lr = 0.01	0.323	0.324
lr = 0.001	0.314	0.311

learning rates the network actually reaches worse performances when the skeleton is present. It can also be noted that the reached accuracy is quite low in both cases. To see where the network was struggling the most, we ran additional experiments with a 0.01 learning rate either only on temporal puzzles or on spatial puzzles (see Table 2).

Table 2: Final mean accuracy on test set for temporal and spatial puzzle solving.

	With skeleton	Without skeleton
Temporal puzzles	0.037	0.049
Spatial puzzles	0.765	0.780

It appears clearly that the temporal puzzles were reducing the overall accuracy of the network that was trained to solve both types of puzzles, since networks trained only on temporally shuffled data achieve very poor performances, comparable to random predictions. Meanwhile, spatial puzzles are easily solved, and the models achieve a good accuracy considering the difficulty of the task. However, in both cases the gap between skeleton and non-skeleton puzzles still holds.

Finally, we also trained the same type of architecture (4-headed siamese network) to solve the action recognition problem from scratch, namely without a self-supervised pretraining. Results over 3 different trainings are reported in Table 3.

Table 3: Final accuracies on test set for the action recognition task trained from scratch.

	With skeleton	Without skeleton
Training 1	0.651	0.692
Training 2	0.577	0.644
Training 3	0.650	0.657

Even trained from scratch, both networks manage to predict the right labels consistantly, especially given the high number of classes (59), and the difficulty of predicting a video label from only 4 representative still images. Once again we can notice the same gap in accuracy when adding the skeleton.

5 DISCUSSION

In the previous section, we have seen that the proposed architecture fails to solve temporal, and thus spatio-temporal, puzzles; nevertheless, the network performs well on both spatial puzzle solving and video action recognition. We also observed that adding skeleton data directly on top of images without explicitly specifying it to the network does not yield improved results, and even actually hurts its performance on all tasks. This implies that the network does not consider the skeleton, and that it only acts as a noise over the natural image behind it. This hypothesis makes sense since the added skeleton can actually be considered as a mask placed in front of the videos, which hides part of the information, hence the deterioration in accuracy.

We still have to run some experiments on transfering knowledge learned in a self-supervised way onto an action recognition task to see if this difference in accuracy still holds, but the initial results presented here allow us to conclude that simply adding the skeleton on images is not enough for it to be grasped by the network.

To fix this issue, we will constrain more the puzzles over the skeleton data; namely, instead of simply adding it on top of the video frames, we could find ways to include this information such that the network will tend to look at it. We can think of this as skeleton as an additional supervisory signal for learning from images, such as in (Alwassel et al., 2020) where audio was used to supervise video learning and vice versa.

We will also run experiments without channel replication applied to the videos; while it forces the network to grasp features unrelated to color, this preprocessing also destroys part of the information that is contained in natural images.

Another line of research we will explore consists in using the skeleton as an input in itself, on which we will also perform self-supervised tasks. We could then fuse results from both types of inputs to improve the performance of action recognition from human-related videos.

6 CONCLUSION

In this work, we tried to add the human skeleton as an implicit signal in images and videos, and performed both puzzle solving in a self-supervised way and action recognition from scratch on those ”augmented” images. We noticed an overall decrease in performance when adding the skeleton that we attributed to a fail from the network to grasp it as an information; without any further constraints, this additional pixels are considered as noise by the network. For future works, we plan on constraining more our architecture based on skeleton poses, whether it is by adding it in a different way to the image or by giving it as a secondary input to our network.

ACKNOWLEDGEMENTS

This work was supported by Conseil Régional des Pays de la Loire. We would also like to thank Yassine El Manyari and Laurent Frering from CEA Tech Pays de la Loire for their discussions which have been of great help in this work.

REFERENCES

Ahsan, U., Madhok, R., and Essa, I. (2018). Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition.

Alwassel, H., Mahajan, D., Korbar, B., Torresani, L., Ghanem, B., and Tran, D. (2020). Self-supervised learning by cross-modal audio-video clustering. In *NeurIPS 2020, 34th Conference on Neural Information Processing Systems*.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *ICML 2020, 37th International Conference on Machine Learning*.

Girdhar, R. and Ramanan, D. (2017). Attentional pooling for action recognition. In *NeurIPS 2017, 31st Conference on Neural Information Processing Systems*.

Jhuang, H., Gall, J., Zuffi, S., Schmid, C., and Black, M. J. (2013). Towards understanding action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 3192–3199.

Kim, D., Cho, D., and Kweon, I. S. (2018). Self-supervised video representation learning with space-time cubic puzzles. In *AAAI 2019, 33rd AAAI Conference on Artificial Intelligence*.

Kolesnikov, A., Zhai, X., and Beyer, L. (2019). Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., and Serre, T. (2011). Hmdb: A large video database for human motion recognition. In *2011 International Conference on Computer Vision*, pages 2556–2563.

Lee, H.-Y., Huang, J.-B., Singh, M., and Yang, M.-H. (2017). Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Li, B., Dai, Y., Cheng, X., Chen, H., Lin, Y., and He, M. (2017a). Skeleton based action recognition using translation-scale invariant image mapping and multi-scale deep cnn. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 601–604. IEEE.

Li, C., Zhong, Q., Xie, D., and Pu, S. (2017b). Skeleton-based action recognition with convolutional neural networks. In *2017 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, pages 597–600.

Li, X., Liu, S., Mello, S. D., Wang, X., Kautz, J., and Yang, M.-H. (2019). Joint-task self-supervised learning for temporal correspondence. In *NeurIPS 2019, 33rd Conference on Neural Information Processing Systems*.

Lin, L., Song, S., Yang, W., and Liu, J. (2020). Ms2l : Multi-task self-supervised learning for skeleton based action recognition. *Proceedings of the 28th ACM International Conference on Multimedia*.

Misra, I., Zitnick, C. L., and Hebert, M. (2016). Shuffle and learn: Unsupervised learning using temporal order verification. In *ECCV 2016, 12th European Conference on Computer Vision*.

Noroozi, M. and Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV 2016, 12th European Conference on Computer Vision*.

Pirk, S., Khansari, M., Bai, Y., Lynch, C., and Sermanet, P. (2019). Online object representations with contrastive learning.

Rohrbach, M., Rohrbach, A., Regneri, M., Amin, S., Andriluka, M., Pinkal, M., and Schiele, B. (2015). Recognizing fine-grained and composite activities using hand-centric features and script data. *International Journal of Computer Vision*, pages 1–28.

Shi, L., Zhang, Y., Cheng, J., and Lu, H. (2019). Skeleton-based action recognition with directed graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7912–7921.

Su, J.-C., Maji, S., and Hariharan, B. (2020). When does self-supervision improve few-shot learning? In *ECCV 2020, 16th European Conference on Computer Vision*.

Sumer, O., Dencker, T., and Ommer, B. (2017). Self-supervised learning of pose embeddings from spatiotemporal relations in videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., and Paluri, M. (2018). A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., and Murphy, K. (2018). Tracking emerges by colorizing videos. In *ECCV 2018, 14th European Conference on Computer Vision*.

Wang, J., Jiao, J., Bao, L., He, S., Liu, Y., and Liu, W. (2019). Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, J., Li, Y., Wang, L., Wang, K., Li, R., and Zhou, T. (2019a). Skeleton based temporal action detection with yolo. In *Journal of Physics: Conference Series*, volume 1237, page 022087. IOP Publishing.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. (2019b). Detectron2.

Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden markov model. In *CVPR*, volume 92, pages 379–385.

Yan, S., Xiong, Y., and Lin, D. (2018). Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Zhang, R., Isola, P., and Efros, A. A. (2016). Colorful image colorization. In *ECCV 2016, 12th European Conference on Computer Vision*.