Self-Supervision with Dynamic Filters for Egocentric Action Recognition

Lorenzo Atzeni
Politecnico di Torino, Italy
s280131@studenti.polito.it

Abstract

Egocentric activity recognition has, in the last years, gained more interest in the research community, due to the rise in popularity of wearable devices. While this growing interest, this task is one of the most challenging in computer vision due to the finest of details needed to correctly identify small object and their manipulation. Previous work has been focused on two-stream network, but this kind of method requires training of two different architectures with a consequence increase in computational time and resources. Recently self-supervised approaches have been proposed as an alternative to two stream methods, allowing for better performance with less computational resources. A problem that is very present in egocentric action recognition is the presence the camera movements that causes the whole image to appear in motion. To solve this problem have been created more specific feature extraction algorithm that allow to eliminate camera movement, like wrap flow and IDT. While state of the art model recur to the use of particular methodologies to extract movement features like wrap flow or IDT , either for the two stream model or the self supervised task, this work explores the possibility that the RGB model is capable of reaching comparable results without the use of particular feature extraction mechanisms. This work also experiments with LSTM specific self-supervision architectures with promising results.

1. Introduction

Egocentric action recognition is one of the most challenging problems in computer vision. While there have been major progress in image-recognition, egocentric action recognition has yet to reach enough maturity to allow deployment on the market. Action recognition has two components in it, the first being the recognition of the objects that are being manipulated or that are subject to an action, and the second being the movement itself. The movement can't be extrapolated by a single image, but it is represented by the evolution over time of the action. Specifically egocentric actions have to face new and different problems than third person action recognition, primarily due to the problems related to camera movement and inter-subject differences. While in the case of third person action recognition the moving object are the ones that are of interest, this is not the case in first person action recognition, where camera movement can make hard to establish which movement are relevant to the recognition of the activity. Recent development has been made to solve this issue making use of particular feature extraction algorithm, like wrap flow and IDT, that are capable of distinguishing between camera motion and movements due to a particular action. Two main approaches have been used to make use of the feature extracted:

- Two stream model that require the training of two different models, one trained with RGB image and one trained with Wrap Flow features that typically have no correlation 1 to 1 between each other. There are different disadvantages in using such architecture, first of all probably is the increased computational resources for both the training and evaluation phase.
- Self-supervised task that tries to reproduce those particular features with the aim of forcing the model to learn spatio-temporal feature necessary for the classification task.

This works explores the possibility that the model is capable of learning such features without the need of those particular feature extraction algorithms. In particular I implement a model that performs self-supervision by learning to predict the next frame in a sequence. The architecture that I propose is composed by a Resnet34 backbone connected to a CAM that is connected to a ConvLSTM layer. The ConvLSTM layer is at the end connected to a self-supervision head and a classification layer. The Backbone and the CAM are used to find the useful features in the images and the ConvLSTM layers is capable of retaining information of previous frames to extract spatio-temporal information. The self-supervised task presented is very simple, the objective is to try to predict the next frame starting from the information obtained by the the previous frames

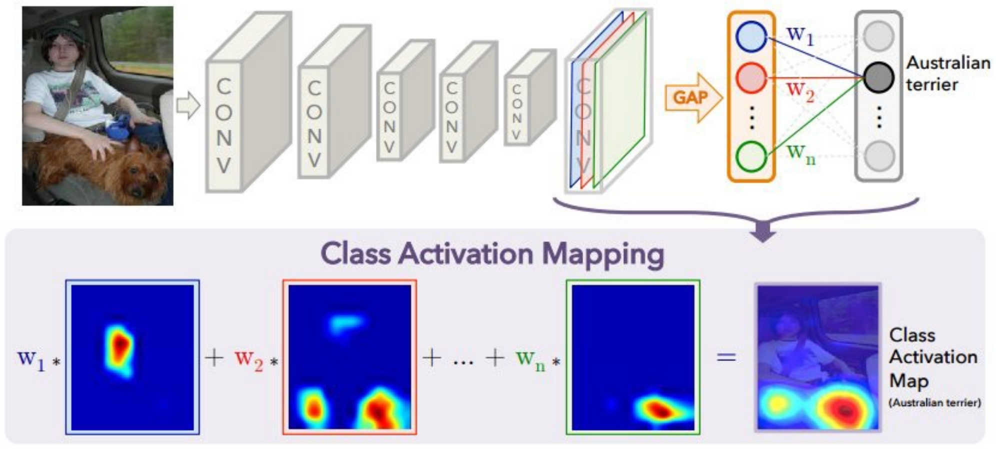
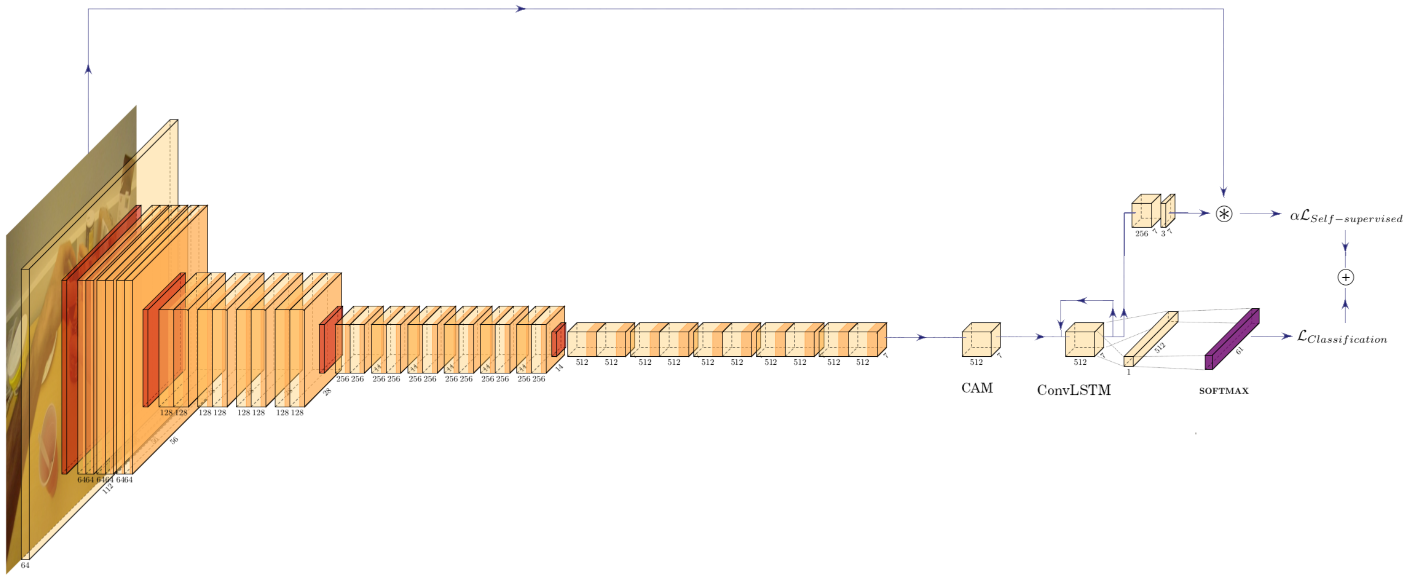


Figure 1. Class Activation Map illustration.

provided as an input to the model. This task requires both spatial and temporal features to be solved optimally, for this reason I find appropriate to incorporate it on top of the ConvLSTM layer. This choice is supported by the results obtained in future frame prediction with LSTM [1]. Inspired by [2] I decide to use dynamic filters for the self-supervised task proposed in this work. Dynamic filters are different from standard convolutional filters due to the fact that they are not a parameter of the model, but they are created as output and for this reason they can be very different from image to image. The reason to use such filters is primarily due to the fact that have been demonstrated to achieve better results with smaller model [2]. The possibility of obtaining better performance with lower resources is particularly attractive for this task since we want the model to learn useful feature without increasing too much the complexity of the network.
The rest of the paper is divided in a relate work section where is considered the work already done to solve this kind of task, followed by an architecture section where the different architectures implemented are exposed in greater details and an experiment section to compare the results obtained by the different architectures. In the end I try to establish the contribution of the different mechanism implemented in this work in an ablation study. The results are summarized in the conclusion.

2. Related work

In recent years deep networks are being the major architectures used in third-person a and first-person action recognition. Many of this networks rely on CNN with stacked frames as inputs, others exploit the use of LSTM or convolutional LSTM (ConvLSTM) to learn time-dependent features. ConvLSTM appears to be particularly suited for videos and time-dependent visual data and has been demonstrated to outperform LSTM in this set of tasks [3]. Two stream approaches with the use of both RGB images and temporal flow have demonstrated themselves suited for egocentric and third person action recognition [4]. Two stream models do require to train two different networks that are necessary also in evaluation time with consequences on the time requirement for both training and evaluation. Two stream approaches with the use of temporal flow have been successfully applied, but recent approaches may indicate that such temporal features can be learned from the RGB images with the use of self-supervised learning. In paper [5] we can observe that abandoning the two stream approach for a self-supervision approach actually provides an increase of performance sustaining the thesis that RGB models can learn spatio-temporal feature if ”forced” by a carefully chosen self-supervised task. The task used in this case was a segmentation task having as target feature the IDT. IDT together with wrap flow are two approaches that are specific for egocentric action recognition, since they are capable of extracting features that represents the movements independently to the camera motion. The Ego-rnn also provides a new approach related to trying to exploit attention mechanism to find the most informative parts in the images. In their approach they make use of CAM (class activation maps) to identify the spatial features that are more relevant for the classification. Further information on the extraction of spatial attention through the use of CAM can be found on section 3.1. Many self-supervised task have been used in recent years especially in video-related deep learning model where the amount of data at hand can easily become a limiting factor. In paper [1] Nitish et al. have presented self-supervised tasks specific to architectures provided with LSTM layers. Those self-supervised task include the ability of predict the next frame, multiple



CAM
ConvLSTM
SOFTMAX
$\alpha\mathcal{L}_{Self-supervised}$
$\mathcal{L}_{Classification}$

Figure 2. Architecture of the model with dynamic filters used in the self-supervised head.

frames in the future and to reproduce the initial sequence in a backward way. The use of this self-supervised task has provided state of the art results in different benchmarks datasets related to action recognition. In paper [2] Dynamic filter networks have been proposed to improve the performance obtained by standard LSTM task in future frames prediction with lower computational cost compared to the standard version. This kind of self-supervised task has inspird also dynamonet [6] that,using a CNN with stacked images as an input, obtained state of the art results in different dataset for third person action recognition. Taking inspiration from Dynamic filter networks and self-supervision task for LSTM layers, I propose an architecture based on convLSTM layers and Dynamic filters for fisrt-person action recognition.

3. Architecture overview

The goal is to create a model capable of learning spatio-temporal representation of the image in order to improve egocentric action recognition. I discuss three different architecture in this section. The first one is the RGB Ego-rnn as proposed in the original paper [4], the other two are two multi-task models that aim to simultaneously solve the self-supervised task and the classification problem. The two multi-task models are based on the same RGB Ego-rnn basic structure but differ in both the self-supervised task chosen an the architecture for the self-supervised head. The RGB Ego-rrn architecture is composed by a ResNet-34 followed by a spatial attention layer and then a convLSTM layer. The spatial attention layers is discussed in section 3.1, and its aim is to force the model to focus on the most important part of the picture, avoiding to capture useless information about the other item that are present in the background but are not relevant to the action classification task. As shown in figure 3 the spatial attention mechanism seems to be able to identify very well the which part of the image are relevant to the classification of the task. The first self-supervised task is the one proposed in [5] and as in the paper the output of the Resnet-34 is fed into the MS head. The self-supervision task uses IDT as a target to force the model to identify the areas that are subject to the movement. IDT have the particular characteristic to be segmentation mask that are capable of segmenting the movement of the video, ignoring camera movements. This makes this MS task particularly suited for first person action recognition. Incorporating this self-supervised task makes the model a multi-task model that solves jointly two different task. The first being the classification task that we are actually interested in solving, and the second the MS task that consist in reducing the classification loss between the IDT target and the output of the MS head. The argument behind this particular self-supervised task is to force the CNN to encode motion as well as spatial clues, allowing the convLSTM to extract more meaningful global video representation. The loss of the self-supervised task and the classification task is summed. In order to balance the effect of the self-supervised task on the backward pass, I introduce α, while I leave the weight of the classification task without any parameter.

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{Self-supervised} + \mathcal{L}_{Classification} \quad (1)$$

The second self-supervised task is inspired by [6] and takes directly as an input the output of the convLSTM. The task doesn't use any particular feature extracted by the RGB image but it is trained to reproduce the next frame prediction. The advantage of using the output of the convLSTM as input of the self-supervised task is that the feature extracted from the output of the convLSTM contains information not only the current frame but also from the frame processed before. The argument about the usefulness of this task is similar to the one made for the MS task, with the difference that we force both the CNN and convLSTM to incorporate information useful to solve the self-supervised task, retaining them useful for solving also the classification task that is the real objective of the network. The self-supervised task used is the one represented in chapter 3.3 and It is used to create dynamic filters to use in the convolution with the input image. The output of the convolution is used to create the next frame of the image. The loss used for this task is the L2 loss. As for the previous self-supervision task I use the equation (1) as the loss of the whole network. The network is represented by figure 2.

3.1. Class activation map and spatial attention

In this work I use class activation map and spatial attention as propose in [4] The CAM is obtained by multiplying the weights of the winning class by the corresponding activation of the last convolutional layer as shown in figure 1 Specifically let $f_l(i)$ be the activation of the last convolutional layer at spatial location i and channel l, and w_l be the weights corresponding to the winning class for unit l. Then the CAM $M(i)$ can be represented as

$$M(i) = \sum_l w_l f_l(i) \quad (2)$$

To compute spatial attention we perform a series of elementwise operation. We perform a Softmax between the different values in the spatial dimension and then we perform a elementwise multiplication with the activation of the last convolutional layer as shown below

$$f_{SA}(i) = f(i) \frac{e^{M(i)}}{\sum_{i'} e^{M(i')}} \quad (3)$$

To provide more clarity I wish to report the case of the Resnet-34. We identify the activation of the last convolutional layer as $f(i) \in \mathbb{R}^{512 \times 7 \times 7}$ and the weights corresponding to the winning class as $w(i) \in \mathbb{R}^{512 \times 1 \times 1}$. To obtain the CAM $M(i)$ use equation (2) and we obtain $M(i) \in \mathbb{R}^{1 \times 7 \times 7}$. The spatial activation is obtained finally by a elementwise operation as reported in equation (3) the softmax operation is broadcasted over the channel dimension.

3.2. MS task

In this paper I compare the architecture presented here with a specific MS task called IDT. The main idea of IDT is to create a segmentation mask of the movement that are independent from camera motion and last at least 10 frames. The architecture used for MS head can be described by a single convolutional layer followed by a fully connected layers and an activation function. Since I experiment with both a regression and a classification version of this task I have used different loss and activation function for the two different version. For the classification task I have applied a softmax function to each pixel of the output followed by a per-pixel cross entropy as the loss function, while for the regression task I have used a simple sigmoid followed by a MSE loss function. The MS head, differently from the self-supervised head proposed in this paper, uses as input the output coming directly from the Resnet-34 pretrained with Imagenet. In the end the loss to minimize can be expressed as

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{Self-supervised} + \mathcal{L}_{Classification} \quad (4)$$

To specify further the motion segmentation head, the convolutional layer is a 1×1 convolution that reduces the number of channel of the Resnet-34 from 512 to 100. The fully connected layer that follows the convolutional one produces an output of $2 \times 7 \times 7$ for the classification, and $1 \times 7 \times 7$ for the regression task.

3.3. Dynamic Motion Filters

The use of Dynamic filter for the self-supervision task is inspired by the paper [6]. the Self-supervision head is placed on the output of each ConvLSTM, capturing the information of the current and preceding images. The methodology used with LSTM is inspired by [1]. A dynamic filter is similar to a normal convolutional filter, with the difference that the filter is not a model parameter and as a consequence it changes dynamically based on the input image. The dimension on the filter $\Theta \in \mathbb{R}^{s \times s \times t}$ is dependent on the input, precisely t is the number of dimension of the image, in our case we deal with RGB images, with 3 channels, so we have $t = 3$. The dimension of s of the filter is chosen to be equal to 7. We use the dynamic filter to synthesize the future frame starting from the previous one by performing a convolution between the dynamic filter and the starting image. The output of the ConvLSTM is used as input of the Self-supervised head that produces the dynamic filters. The self-supervision head is composed by two convolution layers, the first performs a 3x3 convolution maintaining the same number of filters of the convLSTM,

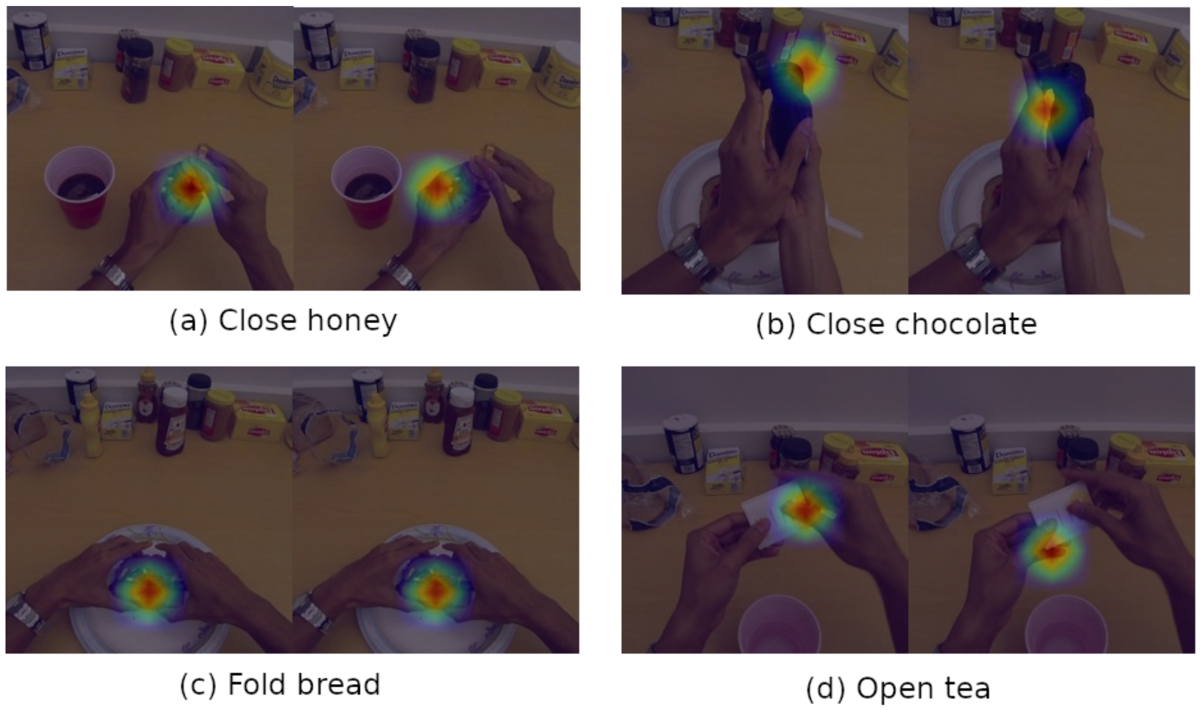


Figure 3. Comparison between CAM with and without supervision. Images with Supervision on the right.

Configurations	Accuracy 7 frames	Accuracy 16 frames
ConvLSTM	50.0	46.5
ConvLSTM-attention	50.0	62.9
Temporal-warp flow*	50.0	50.0
two-stream (joint train)*	63.8	69.8

Table 1. Different Ego-rnn architectures results. *The flow network uses only 5 stacked frames

while the second convolution is characterized by a 1x1 convolution with the aim to reduce the number of filters to the ones required for the input image. The number of filters corresponds to $t = 3$ as stated before.

4. Experiments

Starting from the RGB Ego-rnn architecture, I perform experiments with different architectures and different number of frames. The first experiment is done on the RGB Ego-rnn that we call ConvLSTM-attention and we observe the effects of changing the number of frames on the performances of the network. From Table 1 it appears clear that using 16 frames allows for better performances if compared with the model with only 7 frames. The model with 16 frames shows a gain in accuracy of 12.9%.
Removing the CAM from the network, we can observe significantly worse performance, especially for the 16 frames version that goes from 62.9% accuracy to 46.5% as shown in Table 1 on the ConvLSTM entry.
Observing the performance of the network with only temporal-wrapflow we can observe poor performances, symptom of the fact that the model has problems with predicting the actions without the RGB image. Using the two-steam approach we can avoid this problem with better accuracies overall, 63.8% for the convLSTM-attention with 7 frames and 69.8% for the convLSTM-attention with 16 frames. For both the architectures the two-stream approach provides better accuracy. For more information about how this approach is implemented refer to chapter 4.2.
To investigate the effects of self-supervision tasks, we implement the MS task as explained in section 3, and 3.2. We observe that the regression and the classification version of this task have perform in similarly as shown in Table 3, leading up to the choice of the regression version to perform multiple training trials obtaining mean accuracy of 67.81% as shown in table 4 under the entry convLSTM-IDT. This multiple training trials have been performed on the best performing hyper-parameters bolded in table 3. The per-

lr	lr super	lr resnet	α	step	accuracy
1e-4	1e-4	4e-4	1e-2	30 80	68.1
1e-4	**1e-4**	**4e-4**	**1e-3**	**30 80**	**73.2**
1e-4	5e-5	2e-4	1e-2	30 80	67.2
1e-4	4e-4	4e-4	1e-3	30 80	68.9
3e-4	1e-4	4e-4	1e-3	30 80	70.6
5e-5	1e-4	4e-4	1e-3	30 80	63.7
1e-5	1e-5	2e-5	1	30 80	37.0
2e-5	1e-5	4e-5	5e-1	30 80	52.5

Table 2. Grid search performed on the model with the dynamic filters self-supervised task.

lr	lr super	lr resnet	α	step	loss	accuracy
2e-4	2e-4	2e-4	1	45 90	Cross entropy	65.5
2e-4	1e-4	2e-4	1	45 90	Cross entropy	68.9
2e-4	5e-5	4e-4	0.1	45 90	MSE	72.4
2e-4	1e-4	4e-4	1	45 95	Cross entropy	70.6
2e-4	1e-4	4e-4	0.5	45 90	Cross entropy	71.5
2e-4	1e-4	4e-4	0.1	45 90	Cross entropy	70.7
2e-4	2e-4	4e-4	0.5	45 90	MSE	63.3
2e-4	1e-4	4e-4	0.1	45 90	MSE	68.1
2e-4	**1e-4**	**4e-4**	**0.5**	**45 90**	**MSE**	**73.2**
2e-4	2e-4	4e-4	0.1	45 90	MSE	64.6

Table 3. Grid search performed on the model with the MS task.

formance increase compared to the ConvLSTM-attention is significant with a 4.9% increase.
The self-supervised task with dynamic filters has been implemented on the second stage of the Ego-rnn and also this task has provided improvement compared to the model with convLSTM-attention. The grid search performed on this architecture is reported on tab 2. The results are comparable to the one obtained with MS task and the best results provide a mean accuracy of 68.38% over multiple trials. The value of alpha implemented in this task appear to be quite different from the ones used for the MS task, in fact values of alpha above 0.1 preform very poorly with $\alpha = 1e-3$ providing the best performances. We can observe in figure 3 the attention map for both the supervised and unsupervised model with IDT. The activation map seems to be pretty consistent between the two methods, behaving in both ways pretty similar and pretty accurately.

4.1. Ablation studies

In this section we study the effect of dynamic filters compared with the MS task. We also study the effect of using the output feature of the convLSTM layer as input to the self-supervision tasks in comparison to using the output of the CNN. I performed four different experiments:

- Dynamic filters self-supervised task applied to the output of the convLSTM (Dynamic-net)
- Motion segmentation task applied to the output of the convLSTM (IDT-net)
- dynamic filters self-supervised task applied to the output of the CNN (convLSTM-dynamic)
- Motion segmentation task applied to the output of the CNN (convLSTM-IDT)

We can observe from table 4 that in both cases the implementation of the self-supervision task to the output of the convLSTM has increased the performance. Around 2.5% increase in the case of the IDT self-supervision and around 2% increase in case of dynamic filters. This provides an indication that both models benefit from the extra information provided by the convLSTM to the self-supervised task. The IDT task can use the information provided by preceding frames providing a better quality embedding to the clas-

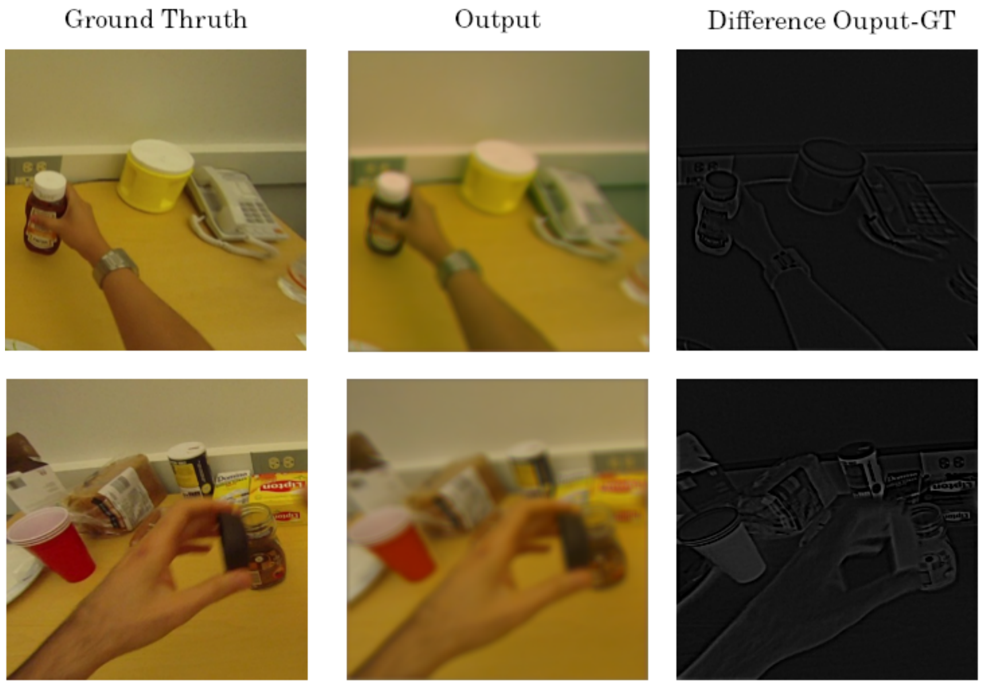


Figure 4. Input output of the self-supervised task with dynamic filters and difference between GT and output.

model	accuracy	increase RGB
IDT-net	70.39	7.4
Dynamic-net	68.38	5.4
convLSTM-dynamic	66.37	3.4
convLSTM-IDT	67.81	4.9

Table 4. Results obtained with different configurations. The entry "increase RGB" refers to the increase in performance with respect to the convLSTM-attention

sification layers of the network. Provided with frames preceding, the self-supervision task has the information that allows it to capture the temporal information. As an example, in the case the subject has two object in his hands, one of the two is being hold still, it is not possible to have enough information to distinguish which one is moving only by a single frame.

The dynamic filters task is a task that aims to capture information similar to the optical flow. Not only is expected to learn information about which part of the image is moving, but also to predict the direction of the movement. This kind of information is highly dependent on the preceding frames so It was expected to provide better accuracy when provided with the convLSTM output. While all the self-supervision task have provided increments in performance with respect to the network without them, the IDT self-supervised task seems to perform generally better than the one with the dynamic filters. This result is not unexpected, the IDT is an elaborate feature that is extracted specifically for first person action recognition. Investigating the reasons behind this result, I have plotted the output of the dynamic self-supervised task as shown in figure 4. In the figure we can observe the input, the output and the difference between the output and the input. The results don't seems to provide any evidence that the self-supervised task is grasping the movement of the hands and the object manipulated. My interpretation is that since all the images are moving due to camera movements, the self-supervised task sees all the items in the image as moving without providing special attention to the ones that are actually object of the first person movement. One of the targets of this work was to study whether it was possible to extract the same or better quality embedding without the use of task specific feature extraction mechanisms. Unfortunately I have not provided any proof to sustain this hypothesis. Nevertheless the performance obtained are significantly better than the performance obtained with RGB Ego-rnn without self-supervised task as shown in table 4.

4.2. Implementation details

The Ego-rnn has been used with ResNet-34 pre-trained on imagenet as backbone, with a convLSTM module with 512 hidden units for temporal encoding. The network has been trained following the parameter used in the original paper [4]. As in paper [4] the RGB network is trained in two stages. In the first stage, the network is trained for 200 epochs with an initial learning rate of 10^{-3} and the learning rate is decayed by a factor of 0.1 after 25, 75 and 150 epochs. For the first stage the whole CNN is freezed. In the second stage, the network is trained for 150 epochs with a learning rate of 10^{-4} and is decayed after 25 and 75 epochs by a factor of 0.1. In the second stage all the layers down to the 4th convolutional block of the Resnet are trained. The temporal network is trained, as in paper [4], for 750 epochs with a batch size of 32 using stochastic gradient descent algorithm with a momentum of 0.9. The learning rate is fixed as 10^{-2} initially and is reduced by a factor of 0.5 after 150, 300 and 500 epochs. A five stacked optical flow images is used as the input to the network and during evaluation, we average the scores of 5 such stacks, uniformly sampled in time, to obtain the final classification scores. For the joint train approach we fine-tune all the layers down to the 4th convolutional block of both networks for 250 epochs with a learning rate of 10^{-2}, which is reduced by a factor of 0.99 after each epoch.
Due to the better performance obtained with 16 frames, with respect to the ones obtained with only 7 frames, both the MS and dynamic feature task are trained with 16 frames. The MS task has been implemented on the second stage of the Ego-rnn and with 32 batch size and the numEpochs 100 decayRate 0.1. The dynamic feature head has been implemented also in the second stage of the Ego-rnn with 32 batch size and numEpochs 100, decay rate 0.2. The hyper-parameters not included here are shown in table 3 for the MS task and in table 2 for the dynamic feature one. A grid search for the IDT-net and for the convLSTM-dynamic has been performed and the results are shown in appendix A, table 5 and 6 respectively. The results shown in table 4 are obtained by performing three trials with the best performing hyper-parameters for all the different architecture.

5. Conclusions

In this work I experiment with existing architectures and I discuss different self-supervision task and their effect on ego action recognition. I obtained better embeddings by using the convLSTM feature as input of both the self-supervised tasks providing evidence that the results obtained in third person action recognition still holds for the first person tasks. In all the cases explored here the self-supervised task has provided improved performance when compared with convLSTM-attention. The self-supervised task using convLSTM output as input of the MS head has provided better performance overall, even better than the joint train network. I also tried to investigate the necessity of using features specifics to first person action recognition, unfortunately I didn't managed to provide evidence of the possibility to do so without effecting performance. Said so, there are many different self-supervised task that have shown interesting results in others video related problems and may actually be suited for third person action recognition. As a bonus, the backbone of both the self-supervised task is extremely suited for changes in the architecture, and further improvements may be obtained experimenting with different architectures and how they interact with the self-supervised tasks.

6. References

[1] Nitish Srivastava, Elman Mansimov, Ruslan Salakhutdinov. "Unsupervised Learning of Video Representations using LSTMs".

[2] Bert De Brabandere, Xu Jia, Tinne Tuytelaars, Luc Van Gool. "Dynamic Filter Networks".

[3] Xingjian Shi , Zhourong Chen , Hao Wang , Hao Wang, Wai-kin Wong, Wang-chun Woo. "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting".

[4] Swathikiran Sudhakaran, Oswald Lanz. "Attention is All We Need: Nailing Down Object-centric Attention for Egocentric Activity Recognition".

[5] Mirco Planamente, Andrea Bottino, Barbara Caputo. "Joint Encoding of Appearance and Motion Features with Self-supervisionfor First Person Action Recognition".

[6] Ali Diba, Vivek Sharma, Luc Van Gool, Rainer Stiefelhagen. "DynamoNet: Dynamic Action and Motion Network".

Appendix A

lr	lr super	lr resnet	α	step	loss	accuracy
2e-4	1e-5	2e-4	0.5	45 90	MSE	72.41
2e-4	1e-5	2e-4	0.1	45 90	MSE	65.51
2e-4	1e-4	2e-4	1	45 90	MSE	70.68
2e-4	5e-5	2e-4	0.5	45 95	MSE	75.86
2e-4	5e-5	4e-4	0.1	45 90	MSE	71.5

Table 5. Grid search performed on the IDT-net.

lr	lr super	lr resnet	α	step	accuracy
2e-4	1e-4	2e-4	0.01	45 90	70.68
2e-4	1e-4	4e-4	0.01	45 90	69.82
2e-4	1e-4	2e-4	1	45 90	64.65
2e-4	1e-5	2e-4	1	45 95	62.93
2e-4	1e-5	2e-4	0.1	45 90	63.79
2e-4	5e-5	4e-4	0.1	45 90	67.24

Table 6. Grid search performed on the convLSTM-dynamic.