Transformers for Egocentric Action Recognition

Group 3

Andrés Felipe Cárdenas Meza
Politecnico di Torino
LM Data Science and Engineering
Turin, Italy
s301231@studenti.polito.it

Edgar Abraham Gaytán Quezada
Politecnico di Torino
LM Data Science and Engineering
Turin, Italy
s300164@studenti.polito.it

Luca Scalenghe
Politecnico di Torino
LM Data Science and Engineering
Turin, Italy
s303452@studenti.polito.it

***Abstract*—Action recognition is a fascinating branch of computer vision. A sub-branch that has been recently getting more attention from researchers is egocentric action recognition; videos recorded from the perspective of the performer. We replicate the results of the most recent approaches for action recognition and domain adaptation and we propose the use of transformers, a model that is revolutionising *Natural Language Processing* (NLP), to implement an attention mechanism in this context. The proposed approach achieves *state-of-the-art* (SOTA) results for the domain adaptation challenge. The code can be found in https://github.com/afcarzero1/TransformerDA**

I. Introduction

The content of this paper can be summarized as follows:

- We give an overview of the approaches that have been proposed for the egocentric action recognition task.
- We propose a method to combine those approaches with the transformer architecture.

Transformers are very recent architectures used in the *NLP* field that have shown an extraordinary success at processing sequential data, like words. Given that videos are a kind of sequential data, it is natural to think that transformers can be used for this task.

A. State of The Art in Action Recognition

First, it has to be considered that the usage of egocentric videos poses a series of challenges that differ from traditional third-person videos, as stated in [9]: the fine granularity of the actions performed, the short span of said actions and the fact that many of these actions can be inside a long untrimmed video makes action recognition a difficult task.

Secondly, it is necessary to address the traditional challenges in video action recognition, the need of extracting good features from the videos. Current approaches involve 3D convolutional neural networks like I3D [2] or 2D neural networks enhanced for video understanding like TSM [10]. Due to the nature of video, it is possible to exploit its structure to extract useful information about the existing temporal relationships that will enhance the predictions done by the model. The typical approaches are to take the average of the features along the temporal dimension or to use more sophisticated methods like TRN [18].

In this paper, we will also consider the domain adaptation (DA) challenge. The idea is to use data from a source domain to train a model that is able to generalize to a target domain. The model must be able to learn features that are common between the domains. We will focus on the *unsupervised domain adaptation* (UDA) field, in which we do not have labels for the target domain during training and the label set is the same across domains, as shown in [17]. A very recent architecture to tackle this task is TA3N, based on the idea of doing spatial and temporal alignment between domains [3].

B. Transformers for egocentric action recognition

The transformer architecture was first presented in the paper "*Attention is all you need*" [15]. It has demonstrated to be an incredibly powerful architecture able to learn sequential patterns present in a language and encode *meaning* without the computational complexity of LTSMs [15]. Instead of processing data sequentially like a normal recurrent neural network it processes all at once and uses a clever idea to keep the information about the features order: positional encoding. Egocentric video processing presents an under-explored area of incredible potential for transformers, where the only drawback is the scarcity of data. This poses a serious challenge to the training of transformers because they require a huge amount of data, as we see in the big language models existing nowadays. For this reason we will be constrained to use lightweight networks.

The transformer architecture can be placed in the network as a temporal aggregator, completely substituting current approaches or as a complementary network to architectures like TRN. It is also possible to use transformers in the network for the domain adaptation task for selecting relevant features.

II. Related Work

Activity recognition is a core problem in computer vision and has been widely explored in recent research. With the rise of *Convolutional Neural Networks* (ConvNets) many works have tried to design effective deep convolutional networks for activity recognition as shown in [18]. There are two main branches that try to tackle this topic: 2D ConvNets and 3D ConvNets. 2D ConvNets have less parameters and thus are easier to train, but fail to directly model hierarchical representations of spatio-temporal data. Oppositely 3D ConvNets need many more parameters due to the additional

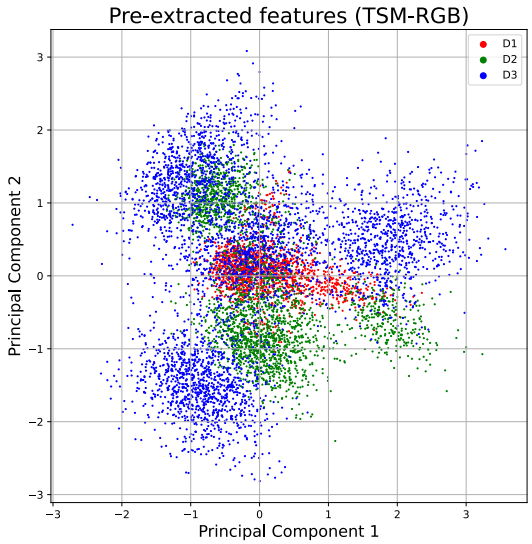

Fig. 1. **Visualization of data from different domains** Distribution of the pre-extracted features extracted by TSM using the RGB flow projected into $\mathbb{R}^2$ using *PCA*. Data from different domains distribute differently creating eye-distinguishable clusters.

kernel dimensions. The main issue with 3D ConvNets is that 3D convolutions on sequences of dense frames are computationally expensive, due to redundancy of consecutive frames. Furthermore, the number of frames that can be fed to the models is usually limited (20-30) making difficult to learn long-term temporal relations among frames [18].

A. Architectures

1) I3D: The authors of [2] propose an approach for video action recognition by creating **Two-Stream Inflated 3D ConvNet (I3D)**, where they take regular 2D ConvNets, specifically a batch-normalized version of Inception-v1 [7] as backbone, previously trained and optimized for the ImageNet challenge, as well as a variant of the C3D architecture [14] to *inflate* it and create a 3D convolutional neural network. In this way any 2D ConvNet pre-trained for the ImageNet challenge can be *inflated* to achieve a robust 3D ConvNet without having to spend the time needed to train a 3D backbone from scratch.

2) TSM: **Temporal Shift Module** is a technique implemented in [10] that builds a module capable of shifting part of the feature maps among the channels used along the temporal dimension. It has online and offline versions, the latter being the most useful for us. This is not a naive and straightforward approach that just moves features along channels. The authors carefully choose the technique for minimizing the data movement and increase the model capacity, to maximize accuracy and minimize computational costs and latency. With the proposed technique it is possible to take any off-the-self 2D ConvNet architecture, where a TSM is added at each convolutional layer inside a residual branch to preserve both the frame activation and the spatial features learning capabilities of the backbone, and transform it into a pseudo-3D model with temporal and spatial handling capabilities at no extra computational cost, mainly because data movement is negligible due to the lower proportion of channels shifts and because residual fusion for the features does not involve additional operations since it is all inside the layers of the network.

3) TRN: **Temporal Relation Network** is an approach used in [18] to attach a temporal relation reasoning module to ConvNets to achieve better results through the modelling of temporal relations between groups of frames. The addition of this module improves the performance of two-stream networks and 3D ConvNets, such as I3D [2] and C3D [14]. It is important to notice that this relations are achieved originally using the RGB channel only, rendering the attachment of this module easy and lightweight in terms of computational time. The authors of [18] introduce temporal relation reasoning as a critical part for activity recognition, forming the building blocks for describing the steps of an event. Classic ConvNets are not able to model such relations.

B. Domain Adaptation

Domain adaptation (DA) has been studied in recent years to address the problem of domain shift: the generalization from a specific domain on which the model has been trained to a different one. We focus on two specific techniques: TA3N and MCC. We have to keep in mind that videos suffer of both spatial and temporal domain discrepancy as shown also by us in the figure 1.

1) TA3N: The authors of [3] noted that similarly to the DA problems in image classification models were not able to generalize to data from different domains because of the above described discrepancy, so they implemented unsupervised DA methods using domain classifiers, whose goal is to discriminate the data origin domain and train the network to learn general features instead of specific-domain ones. The architecture achieves both spatial and temporal alignment. Not all the features are equally important to align, so they further introduced a focus mechanism to give more importance to the temporal features with larger domain discrepancy. The authors discovered that aligning temporal features is more important than aligning spatial features.

2) MCC: Some works [11] [1] [12] have shown that it is possible to use other types of domain adaptation and combine them together. In this work, we will focus on the MCC loss [8]. The authors consider that typical domain adaptation approaches lack of versatility, so specifically tailored to some scenarios and may under-perform if used in other context. They propose a **VDA (Versatile Domain Adaptation)** method as a solution to handle many DA scenarios without modification. By adding a loss term it is possible to make the classifier less prone to errors in the classification of misleading classes. The authors introduce **class confusion** as the tendency of a classifier to confuse predictions between the correct and ambiguous classes (to misclassify a car as a truck, for example)

and they state that less class confusion leads to better transfer gains for all the domain adaptation scenarios.

C. Transformers

The work done in [15] demonstrates the power of the transformer architecture, that can be summarized in 2 modules: encoders and decoders. We will focus on the encoder module that is the one used in this work. The core of this architecture is the self-attention mechanism that takes N vectors, corresponding to a same input (for example vectors associated to words in a phrase), and computes 3 different representations of it: key, query and value. The query part is aggregated through a dot product with all the other input N key representations, and via a softmax function, attention scores are obtained. Those scores are then multiplied with each of the N value representations. Lastly, all these multiplications will be added to produce a single output vector.

Transformers have proven to be useful in egocentric action recognition [9]. Kazakos *et al.* used transformers to analyze a sliding window of actions to model temporal context. The self-attention mechanism of the transformer models the relationships between actions to enhance the predictions.

III. Technical approach

We reproduce the results of the above described convolutional neural networks I3D and TSM. Secondly, use as temporal aggregators Average Pooling and TRN. Finally, we explore the domain adaptation task reproducing the results of the TA3N network.

Then we address the following questions : *Is there any way of improving the current approaches by using transformers?* As we will see it is possible to introduce transformers and use them as temporal aggregators or as feature encoders for the domain adaptation task.

A. Feature Extraction

All of our networks will use CNNs to extract the features. In particular, we decided to use I3D and TSM from [2] and [10]. The convolutional networks extract clip-level and frame-level features. We make use of pre-trained I3D and TSM networks and the subsequent architectures are placed on top of them. Both architectures make use of RGB and Optical Flow features. Regarding the I3D network we replicate the work done in [2] by:

1) "Inflating" a 2D ConvNet by adding a new temporal dimension to the filters and pooling kernels, which typically are *N x N*, to make them *N x N x N*.
2) Is possible to use a two-stream approach for getting a boost performance from the natural recurrence that the optical flow processing has to offer.

Regarding the TSM part of the work: a 2D ConvNet called ResNet-50 [6] is used as backbone. This module performs a *temporal partial shift*, only a portion of channels are shifted for achieving maximum temporal fusion efficiency.

[1]Adapted from the original TA3N figure

As a temporal aggregator we make use of the TRN network [16]. Its working principle is the following:

1) Definition of the temporal relations achieved via 2 MLP layers that fuse discrete ordered frames N-wise, being N the number of frames to be fused.
2) Usage of multi-scale temporal relations. Thus, a combination of a defined number of N-frames relations.

Since it extracts temporal relations between frames it is used in combination with TSM; that extracts features at a frame-level. Frames are uniformly sampled.

On the other hand, for I3D we average pool over the features extracted for each one of the clips.

Features have dimension $n_f \times n_c$ where n_f is the features dimension and n_c is the number of frames (TSM) or clips (I3D). Calling $X \in \mathbb{R}^{n_f \times n_c}$ the matrix with the pre-extracted features for a given action, the network with TRN is represented by the equation 1 while the average pooling one by the equation 2. TRN temporal aggregator is $G_{tf}^{trn} : \mathbb{R}^{n_f \times n_c} \rightarrow \mathbb{R}^{n_r}$ and $AvgPool : \mathbb{R}^{n_f \times n_c} \rightarrow \mathbb{R}^{n_f}$ is the typical average pooling function over n_c . Those are followed by G_y, a feed-forward network that produces the logits for the verb classification of the action. It becomes $G_y^{trn} : \mathbb{R}^{n_r} \rightarrow \mathbb{R}^{n_{cls}}$ when TRN is used , where n_r is the number of frame-relation features. For average pooling it is $G_y^{avg} : \mathbb{R}^{n_f} \rightarrow \mathbb{R}^{n_{cls}}$. The function G_y in both cases produces n_{cls} logits for the verb classes.

$$\hat{y} = G_y^{trn}(G_{tf}^{trn}(X)) \tag{1}$$

$$\hat{y} = G_y^{avg}(AvgPool(X)) \tag{2}$$

B. Transformer Temporal Aggregator

Here we see an opportunity for inserting the transformer as temporal aggregator. Transformers are used to model relationships between frames. We explore the possibility to use the standalone transformer combined with average pooling as seen in equation 3, additionally we will use it combined with TRN. The transformer is represented by $T_{tf} : \mathbb{R}^{n_f \times n_c} \rightarrow \mathbb{R}^{n_f \times n_c}$.

$$\hat{y} = G_y^{avg}(AvgPool(T_{tf}(X))) \tag{3}$$

This idea will be exploited in the DA challenge.

C. TA3N

Feature alignment is implemented using adversarial training so a set of domain classifiers are introduced. The goal is to learn feature extractors so that those classifiers do not distinguish between different domains. Therefore, the network must learn features that do not depend on the domain. The TA3N network is implemented and tested in our context. We consider only features extracted using the TSM network from RGB flow. The network is described by the equation 4. A fully connected layer $G_{sf} : \mathbb{R}^{n_f \times n_c} \rightarrow \mathbb{R}^{n_h \times n_c}$, with the goal of doing the spatial alignment is added before the temporal aggregator. The subsequent layer G_{tf}^{trn} is modified accordingly to new dimension of the features n_h. The loss function is modified to include the loss relative to the domain classifiers

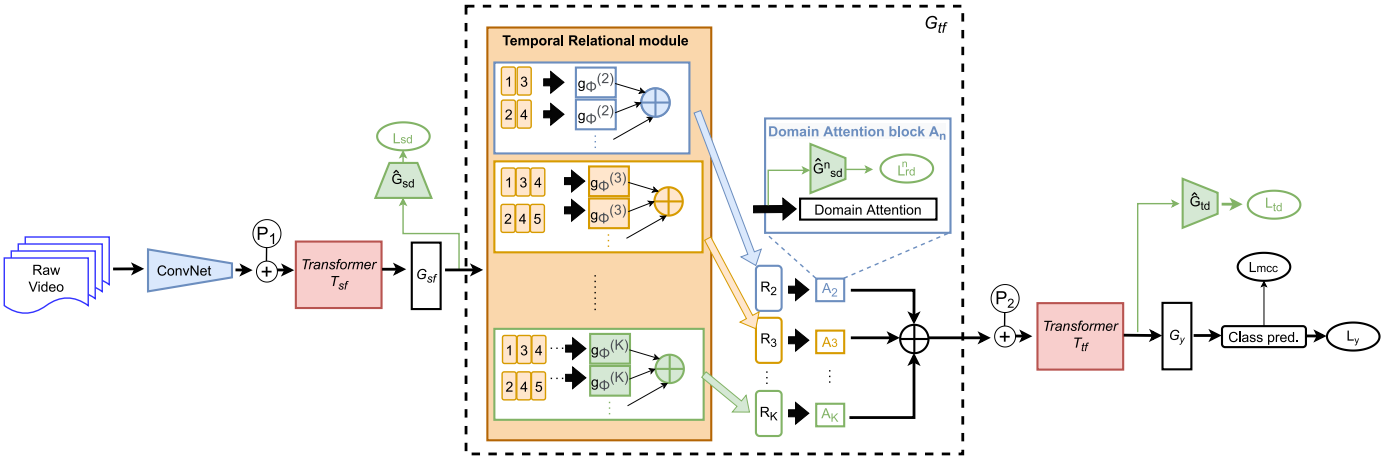

Fig. 2. **TA3N with Transformer** [1]

This figure shows how transformers are introduced in the TA3N network. The neural network G_{sf} for extracting the spatial features is preceded by a transformer T_{sf} that has to learn to distinguish common and specific domain features. Likewise, a transformer T_{tf} is introduced in the output of the TRN network. The MCC loss is also represented in the figure.

5, where N_S and N_T are the number of source examples and target examples. The full architecture is shown in figure 2.

$$\hat{y} = G_y^{trn}(G_{tf}^{trn}(G_{sf}(X))) \tag{4}$$

$$\mathcal{L} = \frac{1}{N_S}\sum_{i=1}^{N_S}\mathcal{L}_y^i - \frac{1}{N_{S\cup T}}\sum_{i=1}^{N_{S\cup T}}\left(\lambda_S\mathcal{L}_{sd}^i + \lambda_t\mathcal{L}_{td}^i + \lambda_r\mathcal{L}_{rd}^i\right) \tag{5}$$

D. Frame attention - TA3N

The TA3N network implements fully connected layers for the transformation of the frame-level features.*Is there a clever way to extract the features so that the spatial domain alignment is more effective?*

With the aim of answering this question we introduced a transformer for encoding the frame-level features. The transformer will learn to focus the attention of the network on those features that both domains share leading to an improvement in the performance of the network. The transformer is particularly well suited to process the features that represent sequential data, but it is necessary to add a positional encoding to keep this information; it is represented by the matrix $P \in R^{n_f \times n_c}$, having the weights for absolute positional encoding. We use the *Fourier* weights as in the original paper [15]. The position of this transformer T_{sf} can be observed in 2. The resulting transformation applied on the input features is the one in equation 6.

$$\hat{y} = G_y^{trn}(G_{tf}^{trn}(G_{sf}(T_{sf}(X + P_1)))) \tag{6}$$

Transformers provide a natural way of fusing different modalities as shown in [9]. This idea is left for further exploration in future work.

E. Frame and Relation attention - TA3N

As in the previous section we can observe that TA3N implements a fully connected layer for doing the predictions G_y from the temporal features. Therefore, it is natural to think that the network can benefit from using a transformer for further elaboration of the features extracted by the temporal aggregator (TRN) as in 7 or use it as temporal aggregator as previously proposed, shown in 8. The transformer will learn to encode only those features common to both domains.

Furthermore, both attention mechanisms can be combined into a single network that learns simultaneously to focus its attention on relevant spatial and temporal features or used separately, as it will be discussed below.

$$\hat{y} = G_y^{trn}(T_{tf}(G_{tf}^{trn}(G_{sf}(T_{sf}(X + P_1))) + P_2)) \tag{7}$$

$$\hat{y} = G_y^{avg}(AvgPool(T_{tf}(G_{sf}(T_{sf}(X + P_1))) + P_2)) \tag{8}$$

F. Frame and Relation attention - TA3N boosted

Since *MCC* is a technique that can be combined with other DA methods, we introduced it as a final booster to the network to achieve a better performance. In particular the extra term in the loss is shown in equation 9. It exploits the error matrices of the labels obtained for the target domain. First, those predictions are re-calibrated to represent probabilities $\hat{Y_{ij}} = \frac{exp(Z_{ij}/T)}{\sum_{j'=1}^{|C|} exp(Z_{ij'}/T)}$ and then the class correlation matrix is obtained as $\boldsymbol{C}_{jj'} = \hat{\boldsymbol{y}}_{\cdot j}^T \boldsymbol{W} \hat{\boldsymbol{y}}_{\cdot j'}$, where $\boldsymbol{W}$ is a diagonal matrix quantifying the importance of each example. Finally the obtained matrix is normalized by classes and used for computing the MCC loss.

$$L_{MCC}(\hat{Y_t}) = \frac{1}{|\mathcal{C}|}\sum_{j=1}^{|\mathcal{C}|}\sum_{j'\neq j}^{|\mathcal{C}|}|\tilde{C}_{jj'}| \tag{9}$$

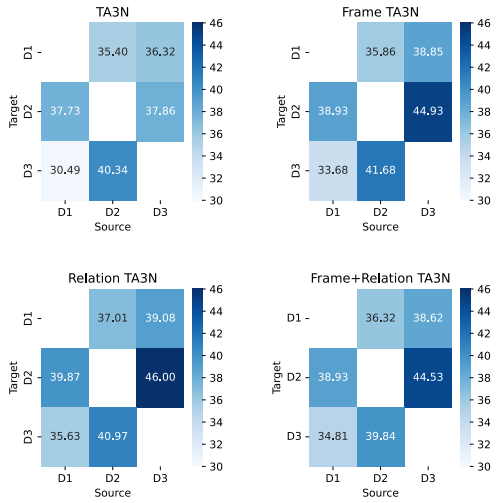

Fig. 3. **Classifier results on different domains**
The results correspond to the full network with all domain classifiers and with transformer as temporal aggregator when compatible with the architecture, average pooling otherwise. . On the x-axis we have the source domain and in y-axis the target domain.

Network	Accuracy RGB (%)	Accuracy Optical Flow (%)	Accuracy RGB+Flow(%)
I3D	53.7	57.77	59.76
TSM	70.06	69.71	75.43
I3D + AveragePool	62.57	66.57	x
TSM + TSN	71.98	68.86	x

TABLE I
END-TO-END AND TEMPORAL AGGREGATION RESULTS

Results of the experiments with I3D and TSM networks standalone and combined with a temporal aggregator.

IV. EXPERIMENTS

The dataset used in our experiments is the EPIC-KITHCENS dataset [4] which is a large scale egocentric-video benchmark produced in 2018. It is composed by 32 participants of 10 different nationalities in 4 different cities that recorded their cooking sessions. It has 55 hours of videos, with 11.5M frames labelled for a total of 39.6K action segments. In our experiments we focused on the videos recorded by 3 participants (P01, P08 and P22). A graphical representation of the dataset is shown in figure 1. It represents the extracted features from the RGB flow using the TSM network . Notice the difference between the features of different participants.

A. End-to-end pre-trained architectures

The first experiment we run had the goal to reproduce the results obtained by the convolutional neural networks trained end-to-end. We used the weights of a pre-trained network and tested RGB, optical flow and both flows combined. Two kinds of frame sampling were implemented: dense and uniform. The first one takes n consecutive frames of a clip while the second technique takes those frames uniformly sampled along the clip. For I3D we used dense sampling while for TSM we used uniform sampling. Results are shown in the table I.

Combined flows show higher overall accuracy and TSM clearly obtains better results. This result is consistent with the two-stream hypothesis in [13] which is clearly explained in [5]. In fact, it has been a largely adopted strategy to train architectures that performed very effectively in image recognition tasks. Using the additional information that comes with the nature of videos, motion mapped through motion vectors, it is possible to give further discriminative information to the model.

Network	Modality	Hidden sizes	Dropout	Weight Decay
I3D + AveragePool	RGB	(512),**(512,64)**	0,**0.25**,0.5	**0**,1e-6,1e-5
	Flow	**(512)**,(512,64)	0,0.25,**0.5**	0,**1e-6**,1e-5
TSM + TRN	RGB	x	**0.1**,0.6,0.7	0,**1e-5**,1e-6
	Flow	x	0.1,**0.6**,0.7	0,**1e-5**,1e-6

TABLE II
HYPER-PARAMETERS TESTED

Hyper-parameters used for grid search on architectures with temporal aggregators. In bold configurations with best performance.

B. Temporal Aggregation : TRN, Average Pooling

We tested both architectures using RGB and Optical flow and performed a grid search to find the best configuration of hyper-parameters on each case. Configurations are reported in II. We used a value of learning rate of $lr = 0.01$. We trained the networks for 600 epochs but observed that they saturated at around 80 epochs.

The best results for TRN and Average Pooling are reported in the table I in the two last rows. There is an improvement obtained by introducing the inductive bias that exploits the temporal structure of data, specially for I3D. We briefly tested transformer as temporal aggregator in this context as in equation 3 obtaining results that were around 3% lower. We believe this is due to the lack of training examples and we leave its deep exploration for future work. We observed that for the transformer it was necessary to decrease the learning rate to $lr = 0.001$.

C. Domain Adaptation

Because of the computational cost of performing experiments using the TA3N associated networks we did not perform a grid search for finding the best hyper-parameter configuration. However, we did a test to understand the importance of each of the components of the network: we tested separately the effect of the domain classifiers. Guided by the previous experiment we decided to use a different learning rate for the transformers present on the network. For all the other components the value of learning rate used was $lr_1 = 0.001$ while for the transformer it was $lr_2 = 0.0001$. An important observation we did during the experimentation is that using a unique learning rate or other values leads to significantly lower results. The transformer used had a single encoder layer with 2 heads. The dropout used for the transformer was 0.1 while for all the other components its value is 0.8. For the *GRLs*

Components	Aggregation Method	TA3N	Frame Self-Attention	Relational Self-Attention	Frame + Relational Self-Attention	Frame + Relational + MCC
Source	Average Pooling	36.05	36.05	37.30	38.43	**38.74***
	TRN Pooling	36.34	36.34	**38.61**	37.46	37.47*
Grd	Average Pooling	36.05	38.86	39.13	**39.23**	37.81
	TRN Pooling	35.92	39.32	**39.41**	38.44	38.02
Gsd	Average Pooling	36.05	38.86	39.13	**39.23**	37.81
	TRN Pooling	36.34	38.57	**39.23**	38.27	37.59
Gtd	Average Pooling	36.05	**38.33**	37.81	38.10	37.99
	TRN Pooling	36.75	38.73	**39.45**	38.43	38.69
All Gd	Average Pooling	36.29	38.99	**39.76**	38.84	38.07
	TRN Pooling	36.63	**38.87**	38.86	38.63	38.57
All Gd + Domain Attention	Average Pooling	37.22	**39.32**	39.18	38.78	37.47
	TRN Pooling	37.47	38.66	39.28	38.60	**39.04**

TABLE III
RESULTS DOMAIN ADAPTATION TASK

The table reports the results of the experiments done using the different networks for the domain adaptation task. The values reported above are the average results obtained by a network over the 6 possible shifts in our data (training in one kitchen and testing on other two kitchens). For each network the importance of the 3 domain classifiers is explored and two different temporal aggregators are used: TRN and Average Pooling. In blue it is reported the absolute maximum and in bold the best result for each row. Results marked with * cannot be considered as source only since the MCC loss effectively uses the target.

(Gradient reversal Layer) we set a hyper-parameter $\beta = 1$ for all the domain classifiers. We briefly tested different values of β obtaining significantly lower results. For the MCC loss we used a temperature value of $T = 2$ and weighting value $\mu = 1$. In all architectures G_{sf} and G_y are single hidden-layer neural networks. Transformers T_{sf} and T_{tf} have two attention heads and one single encoder layer.

Results are reported in the table III. As expected the introduction of the adversarial task is beneficial, and the peak is obtained in almost all cases when the three classifiers are used simultaneously. For the TA3N network without any modification the best result is obtained using TRN as temporal aggregator with the domain attention mechanism discussed in [3].

An important observation is that certain domain shifts are particularly challenging for the networks. We observed that the most difficult shift is when D1 is used as source set and D3 as target set. This can be partially explained observing in 1 that the overlap between *regions* of D1 and D3 is almost coincident with D1 so D3 can generalize better to D1 than the opposite.

The introduction of transformers in the network proved to be beneficial. There is an improvement with respect to the original TA3N in every case; being it around 2%. Also in the case where no adversarial classifier is introduced the predictions improve. We must take into account that we are not training the network end-to-end but we are using only pre-extracted features.

The absolute maximum in performance is obtained using all the three domain classifiers with the transformer as temporal aggregator. This result coincides with the results of [3] : it is more important to do temporal alignment than spatial alignment. In fact, the network that focuses on temporal alignment obtains the best results. We think think the full architecture does not obtain the best results because in the one with only T_{tf} there is still a domain alignment performed by the fully connected layer G_{sf} (without T_{sf}) in the spatial domain so it benefits from both mechanisms without having excessive complexity. Indeed, networks with only one transformer seem able to generalize better.

The *MCC* loss is beneficial only in particular cases and we attribute it to the lack of hyper-parameter tuning because of our computational resources. We can see that when no adversarial task is introduced it helps the network in the DA task.

V. CONCLUSIONS

To sum up, egocentric action recognition is a field with still a lot of research to be done. In this work we presented in an organized way the current approaches used for tackling this challenge: convolutional neural networks, temporal aggregators and domain adaptation methods. Subsequently, we presented a promising improvement using a very recent architecture: transformers. As shown by our results transformers can successfully improve the results of existing architectures, particularly in the domain adaptation field. The performance of the network that uses transformers is superior to the original network in every test we did. It proved to be a good solution for doing the spatial and the temporal alignment and further research should be done on integrating other modalities. Although we did not observe an improvement using MCC loss we think with further exploration it can be successfully combined. One question remains open on *how to use deeper transformers without incurring into excessive complexity?*. In fact, a major drawback to be solved is the fact that those networks require a lot of training data; which is not available. Acquiring more data is very expensive since it has to be collected and labeled. As a consequence, more research has to be done on how to make transformers work in scenarios where huge quantities of training data are not available and in particular in the egocentric action recognition field.

REFERENCES

[1] CAO, Z., MA, L., LONG, M., AND WANG, J. Partial adversarial domain adaptation. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 135–150.

[2] CARREIRA, J., AND ZISSERMAN, A. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 6299–6308.

[3] CHEN, M.-H., KIRA, Z., ALREGIB, G., YOO, J., CHEN, R., AND ZHENG, J. Temporal attentive alignment for large-scale video domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 6321–6330.

[4] DAMEN, D., DOUGHTY, H., FARINELLA, G. M., FIDLER, S., FURNARI, A., KAZAKOS, E., MOLTISANTI, D., MUNRO, J., PERRETT, T., PRICE, W., ET AL. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 720–736.

[5] GOODALE, M. A., AND MILNER, A. D. Separate visual pathways for perception and action. *Trends in neurosciences 15*, 1 (1992), 20–25.

[6] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.

[7] IOFFE, S., AND SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (2015), PMLR, pp. 448–456.

[8] JIN, Y., WANG, X., LONG, M., AND WANG, J. Minimum class confusion for versatile domain adaptation. In *European Conference on Computer Vision* (2020), Springer, pp. 464–480.

[9] KAZAKOS, E., HUH, J., NAGRANI, A., ZISSERMAN, A., AND DAMEN, D. With a little help from my temporal context: Multimodal egocentric action recognition. *arXiv preprint arXiv:2111.01024* (2021).

[10] LIN, J., GAN, C., AND HAN, S. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 7083–7093.

[11] PENG, X., HUANG, Z., SUN, X., AND SAENKO, K. Domain agnostic learning with disentangled representations. In *International Conference on Machine Learning* (2019), PMLR, pp. 5102–5112.

[12] PENG, X., HUANG, Z., SUN, X., AND SAENKO, K. Domain agnostic learning with disentangled representations. In *International Conference on Machine Learning* (2019), PMLR, pp. 5102–5112.

[13] SIMONYAN, K., AND ZISSERMAN, A. Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems 27* (2014).

[14] TRAN, D., BOURDEV, L., FERGUS, R., TORRESANI, L., AND PALURI, M. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 4489–4497.

[15] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, Ł., AND POLOSUKHIN, I. Attention is all you need. *Advances in neural information processing systems 30* (2017).

[16] WANG, L., XIONG, Y., WANG, Z., QIAO, Y., LIN, D., TANG, X., AND GOOL, L. V. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision* (2016), Springer, pp. 20–36.

[17] XU, R., LI, G., YANG, J., AND LIN, L. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 1426–1435.

[18] ZHOU, B., ANDONIAN, A., OLIVA, A., AND TORRALBA, A. Temporal relational reasoning in videos. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 803–818.