STAF: A Spatio-Temporal Attention Fusion Network for Few-shot Video Classification

Rex Liu, Huanle Zhang, Hamed Pirsiavash, and Xin Liu
Department of Computer Science
University of California, Davis

Abstract

We propose STAF, a Spatio-Temporal Attention Fusion network for few-shot video classification. STAF first extracts coarse-grained spatial and temporal features of videos by applying a 3D Convolutional Neural Networks embedding network. It then fine-tunes the extracted features using self-attention and cross-attention networks. Last, STAF applies a lightweight fusion network and a nearest neighbor classifier to classify each query video. To evaluate STAF, we conduct extensive experiments on three benchmarks (UCF101, HMDB51, and Something-Something-V2). The experimental results show that STAF improves state-of-the-art accuracy by a large margin, e.g., STAF increases the five-way one-shot accuracy by 5.3% and 7.0% for UCF101 and HMDB51, respectively.

1. Introduction

Few-shot learning has received increasing attention in video classification for its potential to reduce the video annotation cost significantly [2]. In few-shot video classification, the video samples in the training and test sets are from different classes (i.e., unseen classes in the test set). To classify an unlabeled video sample (query), a few-shot video classification model compares each query with the labeled video samples in each unseen class (support set) and classifies the query to the unseen class that is most similar to the query. Few-shot video classification aims to generalize the trained model to unseen classes with only a few labeled video samples in each unseen class. Inspired by the recent work of few-shot image classification [5, 13, 20], recent few-shot video classification approaches using metric-learning-based methods achieve state-of-the-art performance [19, 29]. This paper targets metric-learning-based few-shot video classification.

A metric-learning-based few-shot video learning algorithm classifies a query based on the similarity between the features of the query and the features of each class in support set. Therefore, the core to metric-learning-based few-shot video classification is to design feature extraction and representation for the support sets and the query. Many feature embedding networks have been designed for this purpose. Perrett et al. [19] leverage attention mechanism in temporally-ordered frames from support sets to match query frames after extracting representation for each frame with pre-trained 2D Convolutional Neural Network (2D CNN). Zhang et al. [28] introduce permutation-invariant pooling and self-supervised learning tasks to enhance representations after extracting from a 3D Convolutional Neural Network (3D CNN) embedding network.

Intuitively, the model with a 3D CNN embedding network should outperform the model with a 2D CNN embedding network for video classification [11]. However, in a few-shot scenario, the previous efforts with a 2D CNN embedding network outperformed those with a 3D CNN embedding network [19, 29]. In contrast to tackling the temporal information after getting the rich spatial representation extracted from a 2D CNN embedding network [13], we note there are two challenges for a few-shot video classification model based on a 3D CNN embedding network. The first one is to generate general representations for different classes in a few-shot scenario using a 3D CNN embedding network. The second one is to locate the region of the representations from the embedding network to discriminate different classes.

In this paper, we propose our few-shot video learning algorithm Spatio-Temporal Attention Fusion network, named STAF. First, to address the challenge of generating general representations, we increase the class diversity of the pre-training database for our 3D CNN embedding network. As the model sees diverse categories during the pre-training process, the model can generate distinguishable representations in a few-shot scenario. Then, as shown in Figure 1, we apply a self-attention network and a cross-attention network to highlight the representative regions in feature maps to tackle the second challenge. The self-attention network emphasizes the regions of the feature map that are essential for representing each class and the query, while the
cross-attention network emphasizes the regions of the feature maps that enhance the discriminability between the query and the unseen classes. Then, we measure the similarity between the query and each unseen class based on the feature maps from each attention network. Last, we classify the query video by a simple but effective fusion network. We also add one multi-task training setting, i.e., global video classification task, to regularize the embedding module and further improve generalization performance. More details are given in Section 3.

Contributions We make the following contributions.

1. We propose STAF, the novel attention-based network with a 3D CNN embedding network for few-shot video classification.
2. We design a fusion mechanism to integrate self-attention and cross-attention networks, which greatly enhances the essential spatial and temporal regions of feature maps.
3. We extensively evaluate STAF using three benchmarks, i.e., UCF101 [24], HMDB51 [15], and Something-Something V2 [9]. Compared to the existing works, STAF improves state-of-the-art accuracy by a large margin. Our code is available at https://anonymous.4open.science/r/STAF-30CF.

2. Related work

Few shot learning Most existing few-shot learning algorithms can be divided into three categories: model-based methods [17, 22], optimization-based methods [7, 21], and metric-learning-based methods [19, 23, 29].

The model-based method aims to rapidly update parameters on a few samples by designing the model architecture that directly establishes the mapping function of query data and unseen classes. Munkhdalai and Hong [17] proposed Meta Network with a rapid generalization ability due to the fast parameterization by learning meta-level knowledge and shifting inductive biases across different tasks. The optimization-based method is designed to accomplish a few-shot learning task by adjusting the optimization methods when considering the model with the gradient descent method challenging to converge in the few-shot scenario. Finn et al. [7] proposed MAML to learn the initialization parameters of the model to maximize the performance of the new task after several gradient steps.

Metric-learning-based method measures the distance between the representation of support samples and query samples and classifies them with the aid of the nearest neighbor to keep similar classes close and dissimilar classes far away. Particularly, Prototypical Network [23] is based on the idea that each class has a Prototypical representation which is the mean value of support set in embedding space. The few-shot learning problem then becomes the nearest neighbor in the embedding space. Therefore, to better obtain the prototype representation in the embedding space, the model needs to see as many classes as possible in each training episode to make the distance between different classes larger. Metric-learning-based methods are more promising than other two methods in few-shot video classification since the previous work with metric-learning-based achieved better performance [19, 29]. Our work is one of the metric-learning-based methods, we not only let the model see as many classes as possible in the pre-training process, but also we highlight the spatio-temporal features that need attention for each class while increasing the differences from other classes.

Few shot video classification The first module of most metric-learning based methods for a few-shot video classification model is an embedding network that extracts features from each video. The two most commonly used approaches for embedding network are 2D CNN [2, 19, 29, 31–33] and 3D CNN [1, 6, 28]. After using 2D CNN to extract features from each video frame, Zhu and Yang [31, 32] introduce a memory network structure to learn optimal representation in a larger video representation space. Instead of creating a memory structure to memorize long-term information for video representation, more recent work with 2D CNN embedding networks focus on temporal alignment exploration between query video and support set. Cao et al. [2] align the frames between the query video and support video by temporal ordering information. Zhang et al. [29] leverage an implicit temporal alignment for each pair of the video after the spatial context module and the channel context module. Perrett et al. [19] achieves state-of-the-art on 5-way 5-shot video learning by computing the distance of temporal-relational representations between query video and support video. In comparison, the features extracted from the 3D CNN embedding network already contain temporal information. Therefore, recent work focus on generating general spatio-temporal video representations for unseen classes. Dwivedi et al. [6] leverage GAN to generate the spatio-temporal video representations for the prototype of the unseen classes. Zhang et al. [28] introduce permutation-invariant pooling and self-supervised learning tasks to enhance representations whereas Bishay et al. [1] using segment-based attention and deep metric learning. Compared with previous work, we adopt a 3D embedding network and use a combination spatio-temporal attention mechanism and multi-tasks learning to generate general spatio-temporal features for unseen classes.

Attention-based learning Attention mechanism enhances the learning ability of long-range dependencies in the network to highlight the critical regions of visual representations [25]. These critical regions are beneficial for discriminating the differences between different classes. Therefore, the recent work with attention mechanisms achieve state-of-the-art accuracy for few-shot learning tasks [13, 19]. Hou et al. [13] leverages the cross attention mechanism to extract
Figure 1. Illustration of the Spatio-Temporal Attention Fusion (STAF) on a 2-way 2-shot video classification. First, we extract spatio-temporal features with a pre-trained 3D CNN embedding network for each video. Then, we compute a prototypical representation ($R_{Sc}$) for each class in the support set, which is the mean of all the representations of each class. After that, we use the self-attention module to highlight spatio-temporal features for each query and support class representation and compute the similarity score of each pair of query representation and support class representation using cosine distance. In parallel, we use the cross-attention module to highlight the spatio-temporal correlation features for each pair of query representation and support class representation, and compute the similarity score using the cosine distance. The cross-attention representations of each class in the support set are fed into a global video classifier as a multi-task training set. And the fusion results of similarity scores from the self-attention module and cross-attention module are fed into the nearest neighbor classifier. Details are in Section 3.

discriminative representations for few-shot image classification. While for few-shot video classification, Perrett et al. [19] apply a cross transformer with a multi-head attention mechanism for the representation of each frame to locate the representative frames for similarity computation. These works adopt 2D CNN to extract the features. However, our work leverage 3D CNN to extract features and uses an attention fusion network to highlight the spatio-temporal features, which help increase the inter-class distance and decrease the intra-class distance.

3. STAF: Spatio-Temporal Attention Fusion Network

3.1. Problem definition

The few-shot video classification problem aims to classify one unannotated query video into one of several annotated categories set, which we call “support set”. Each category has only a few video instances in this support set, and the model does not see these categories during the training process. Our paper focuses on C-way K-shot video classification, where C denotes the number of categories in the support set and K represents the number of video instances for each category in the support set. We follow the same episodic training with the previous study [2,19,28,31,32] that randomly select C classes with K video clips for support set. Then we select one query video from these C classes, which is different from the K video clips in the support set. For each C-way K-shot episode, the support set contains C classes, and each class has K video clips.

We use $S^c_k = \{f^c_{k,1}, f^c_{k,2}, \ldots, f^c_{k,n}\}$ denotes the $k^{th}$ video clip of class $c$, where $c$ belongs to $C$ and $k$ belongs to $K$, $f^c_{k,i}$ denotes the $i^{th}$ extracted frame from the video and $n$ denotes the total number of frames extracted from the video. For the query video, we use $S_q = \{f_1, f_2, \ldots, f_n\}$, where $f_i$ denotes the $i^{th}$ frame extracted from the query video and $n$ denotes the total number of frames extracted from the query video. The final goal is to predict $S_q$ to one of the classes.

3.2. The STAF Model

The design principle of our STAF model is to highlight the critical spatio-temporal region to minimize the intra-class distance while maximizing the inter-class distance between the query video and support set. To tackle the challenge of only having few samples for the unseen class, we first extract spatio-temporal features using a pre-trained 3D CNN. Then, we use the attention fusion module to further highlight the critical spatio-temporal region for metric learning. In parallel, we use a global classification task to regularize the embedding network. Next, we analyze each module in our STAF model, which is described in Figure 1.

Embedding module In our STAF model, the goal of the embedding module $f_{\psi}$ is to learn the spatio-temporal representations for each video. We evenly extract frames from each video, and we use $n$ to denote the total number of frames extracted from each video. We use the 3D CNN backbone as our spatio-temporal embedding module.
Given a frame sequence extracted from the video \( S_v = \{f_1, f_2, \ldots, f_n\} \), and \( R_v \in \mathbb{R}^{C_\ell \times T' \times H' \times W'} \) denotes the representation learned from the embedding model:

\[
R_v = f_\varphi(S_v).
\]  

(1)

For video clip in the support set \( S_k \), we use \( R_{S_k} \) to denote the representation learned from the embedding module. We use \( R_{S_k} \) to denote the representation of the class \( c \), which is the mean of all the representations of video clips for class \( c \) in the support set following prototypical network [23]. And since we have only one query video in the few-shot learning task, we use \( R_{S_q} \) to denote the representation for the query video clip. After we get the representations for the support set and query video, we go through two separate attention modules in parallel, i.e., the self-attention module and the cross-attention module.

**Self-attention module** Our goal of the self-attention module is to highlight the critical information in the representation of each class. As shown in Figure 2, we first reshape each representation to \( R'_{v} \in \mathbb{R}^{L \times L} \), where \( L (L = T' \times H' \times W') \) is the number of spatio-temporal positions on each feature cubic map. After that, for each class in the support, \( R_{S_k} \) becomes \( R'_{S_k} \), i.e., \([R'_1, \ldots, R'_i, \ldots, R'_C]\), where \( R'_i \) denotes the feature vectors at the \( i \)th spatio-temporal position in the \( R'_{S_k} \). For each query video, \( R_{S_q} \) becomes \( R'_{S_q} \), i.e., \([R'_1, \ldots, R'_i, \ldots, R'_C]\), where \( R'_i \) denotes the feature vectors at the \( i \)th spatio-temporal position in the \( R'_{S_q} \). Then we compute the self-attention map for each representation as:

\[
M^{self} = (R'_v)^\top R'_v,
\]

(2)

where \( M^{self} \in \mathbb{R}^{L \times L} \) that denotes the self-attention map for each video, where \( M'^{self}_i \) denotes the self-attention at \( i \)th spatio-temporal position in feature map. Then we apply convolutional operation with a kernel \( d \), i.e., \( d \in \mathbb{R}^L \), to fuse each position self-relation vector into an attention scalar, which is \( \mathbb{R}^{T' \times H' \times W'} \). Then we leverage a softmax function to draw self-attention for each \( i \)th position:

\[
A'^{self}_i = \frac{\exp((d \top M^{self}_i)/\tau)}{\sum_{j=1}^{L} \exp((d \top M^{self}_j)/\tau)},
\]

(3)

where \( \tau \) is the temperature hyperparameter to amplify the variance and \( A'^{self}_i \) denotes the \( i \)th position of self-attention map \( A'^{self} \), i.e., \( A'^{self} \in \mathbb{R}^{T' \times H' \times W'} \).

Instead of assigning equal weight to every position, we add a meta-learner to learn the kernel \( d \) dynamically to pay attention to the critical positions in the feature cubic map. First, we leverage row-wise global average pooling for \( M^{self} \) to get an averaged vector \( \overline{M}^{self} \), which \( \overline{M}^{self} \in \mathbb{R}^L \). Then we use a meta-learner to learn the kernel \( d \) dynamically:

\[
d = f_\gamma(\sigma(f_\delta(\overline{M}^{self}))),
\]

(4)

where \( f_\delta : \mathbb{R}^L \rightarrow \mathbb{R}^l \) and \( f_\gamma : \mathbb{R}^l \rightarrow \mathbb{R}^{L} \), i.e., \( l \) denotes the scaled dimension and \( \sigma \) represents the ReLU function [18].

After we get the self-attention cubic map \( A'^{self} \), we leverage a residual attention mechanism to weigh each element of the original map \( R_v \) with \( 1 + A'^{self} \) to get the self-attention representation \( R_v^{self} \) for each class:

\[
R_v^{self} = R_v (1 + A'^{self}),
\]

(5)

where \( R_v^{self} \in \mathbb{R}^{C_\ell \times T' \times H' \times W'} \).

**Cross-attention module** While the self-attention module highlights the critical spatio-temporal region in the representation itself, the cross-attention module focuses on the correlation between the query video and the support set. As shown in Figure 3, we follow the same steps with the self-attention module to reshape each representation to \( R_v \in \mathbb{R}^{C_\ell \times L} \).

After that, we compute the correlation map for each pair of the query video and the support class prototype. For example, for the pair of the query video \( R_{S_q} \) and support class \( \bar{c} \), i.e., \( R_{S_k} \), we compute the correlation map for query video \( M^{cross}_{S_k \rightarrow S_q} \) between the query video and support class:

\[
M^{cross}_{S_k \rightarrow S_q} = (R'_{S_k})^\top R'_{S_q}.
\]

(6)

Then for the support class \( c \), the correlation map \( M^{cross}_{S_k \rightarrow S_q} \) between the query video and support class:

\[
M^{cross}_{S_k \rightarrow S_q} = (R'_{S_k})^\top R'_{S_q}.
\]

(7)

After getting the correlation map for query video and support class in each pair, we go through the same steps with the self-attention module, which are shown in the Figure 3, to get the cross-attention representation for query video and support class in each pair, i.e., \( R_{S_k}^{cross} \) and \( R_{S_q}^{cross} \).

**Attention fusion module** After we get the self-attention and cross-attention representation from the two attention modules, first, we compute the probability of predicting \( S_q \) as the class \( k \) using self-attention representation:

\[
P_{self}(y = k|S_q) = \frac{\exp(-D_{cos}(R_v^{self}, R_k^{self}))}{\sum_{j=1}^{C} \exp(-D_{cos}(R_v^{self}, R_j^{self}))},
\]

(8)

where \( D_{cos} \) denotes the cosine distance and \( P_{self}(y = k|S_q) \) denotes the probability of predicting \( S_q \) as the class \( k \in \{1, 2, \ldots, C\} \) using self-attention representations. Then we compute the probability of predicting \( S_q \) as the class \( k \) using cross-attention representation:

\[
P_{cross}(y = k|S_q) = \frac{\exp(-D_{cos}(R_{S_k \rightarrow S_q}^{cross}, R_k^{cross}))}{\sum_{j=1}^{C} \exp(-D_{cos}(R_{S_k \rightarrow S_q}^{cross}, R_j^{cross}))},
\]

(9)

where \( P_{cross}(y = k|S_q) \) denotes the probability of predicting \( S_q \) as the class \( k \in \{1, 2, \ldots, C\} \) using cross-attention module.
To take advantage of the discriminative information from two attention mechanisms, we leverage the attention fusion module with the nearest neighbor classifier:

\[
P(y = k | S_q) = \frac{1}{2} [P_{cross}(y = k | S_q) + P_{cross}(y = k | S_q)],
\]

where \( P(y = k | S_q) \) denotes the final probability of predicting \( S_q \) as the class \( k \in \{1, 2, ..., C\} \).

Multi-task training

To reduce the risk of overfitting in the training dataset and generate a general representation for unseen class, we train our STAF model in a multi-task setting to regularize the embedding network. We combine the nearest neighbor classifier and the global video classifier.

After the attention fusion module computes the probability of predicting query video to one of the classes in the support set, we use a negative log-probability as the loss function of the nearest neighbor classifier based on the actual class label:

\[
L_1 = - \sum_{k=1}^{C} \log P(y = k | S_q).
\]

Since the representations after the cross-attention module contain highlighting regions related to the query video, we choose these representations to predict the global class in the whole training dataset. The total class number in the training dataset is \( Z \). We feed these cross-attention representations to a fully connected layer and a softmax layer to get the probability of predicting the global class, i.e., \( P(y = z | S_c) \) where \( z \in \{1, 2, ..., Z\} \). Then we define the loss function of the global video classifier as:

\[
L_2 = - \sum_{z=1}^{Z} \log P(y = z | S_c).
\]

Finally, the loss function of our STAF model is defined as:

\[
L = L_1 + \lambda L_2,
\]

where we use \( \lambda \) to weigh the impact of different classification tasks.

4. Evaluation

4.1. Experimental Setup

Datasets. We use the merged video dataset with Kinetics-700 [3], Moment-in-time [16], and START-action [27] to pre-train our 3D CNN embedding network. We compare STAF with existing works on UCF101 [24], HMDB51 [15], and Something-Something V2 (SSv2) [9]. In particular, SSv2 is more challenging because it focuses on actions related to temporal relationships such as ‘pretending to take something from somewhere’ versus ‘take something from somewhere’ [30]. There are two few-shot splits for SSv2 proposed by CMN [31] and OTAM [2], containing 64, 12, and 24 classes as the training, validation, and testing set. We use SSv2-part and SSv2-all denote the split from CMN [31] and the split from OTAM [2]. The difference between these two splits is the number of video samples in each class. For SSv2-part, Zhu and Yang [31] randomly selects 100 samples for each class, whereas for SSv2-all, Cao et al. [2] uses all the samples in the original SSv2. We evaluate our methods in these two splits. Additionally, we also follow the split in ARN [28] for HMDB51 and UCF101.

Evaluation and baseline. Following the same evaluation process with TRX [19], we evaluate the 5-way 1-shot and 5-way 5-shot video classification task and report the average accuracy over 10,000 randomly selected episodes from the testing set. We compare our results with nine state-of-the-art algorithms, i.e., CMN++ [31], CMN-J [32], OTAM [2], FEAT [26], PAL [33], TRX [19], ITANet [29], ProtoGAN [6], ARN [28]. Seven of the nine use 2D CNN, and the other two use 3D CNN.

Experimental Configuration. We evenly sample 16 frames from each video sample and then resize each frame to \( 256 \times 256 \). Afterward, we randomly flip each frame horizontally and crop the center region of \( 224 \times 224 \) to augment the training data. For testing data, we only crop the center with the same size without the horizontal flipping. Then we use a 3D ResNet-50 [11] with the weights pre-trained on the combined dataset with Kinetics-700 [3], Moments in Time [16], and Start Action [27] as our embedding network. After finetuning in the validation dataset, We set 0.025 as the temperature hyperparameter (\( \tau \) in Eq 3) and set 6 as our
meta-learner scaled dimension($l$ is the scaled dimension of $f_c$ in Eq 4), and set 2 as our loss weight hyperparameter($\lambda$ in Eq 13). We train our model for 128,000 episodes in either NVIDIA RTX A5000 GPU (except for the larger SSv2-all, we train our model for 256,000 episodes). We optimize our STAF model with SGD, in which the learning rate is 0.01. We adopt the batch-size of 128, 64, 32, 32 for UCF101, HMDB51, SSv2-part, and SSv2-all, respectively.

4.2. Comparison with State-of-the-arts

Table 1 tabulates the overall 5-way 1-shot and 5-way 5-shot performance compared with existing methods on UCF101, HMDB51, and two splits of SSv2. Our approach outperforms existing algorithms on these datasets for both 5-way 1-shot and 5-shot video classification, except the 5-way 5-shot video classification on the SSv2-part dataset [31]. We can categorize these comparative methods into two groups based on their embedding networks, i.e., 2D CNN embedding network and 3D CNN embedding network. From the Table 1, we can observe that in the previous work, the algorithms with 2D CNN embedding networks outperform those with 3D CNN embedding networks.

With the knowledge learned from ImageNet [4], the works with 2D CNN embedding networks focus on aligning the support video with the query video using temporal information. ITANet [29] performed best for 5-way 1-shot learning on the SSv2-all dataset after adding position encoding for temporal alignment. TRX [19] achieves state-of-the-art performance for 5-way 5-shot learning due to leveraging the temporal information from different frames from different videos in the support set. Therefore a more accurate and robust temporal alignment strategy improves performance with 2D CNN embedding networks.

In contrast, our model takes full advantage of the spatio-temporal knowledge learned from the 3D CNN model to distinguish the query video from the videos in the support set. In addition, different from architecture modification in ProtoGAN [6] and ARN [28], our model highlights the critical region in the spatio-temporal space to improve performance further. Finally, our model defeats previous work on HMDB51 and UCF101 by 5.3% and 7.0%, respectively, for 5-way 1-shot video classification. Due to the small number of samples under one class on the SSv2-part, our model achieves a new state-of-the-art result on the SSv2-all database for the 5-way 5-shot task.

4.3. Ablation study

We have two motivations in proposing our model. The first one is to illustrate the importance of the performance of supervised video classification and class diversity in the pre-training process for learning spatio-temporal knowledge by using a 3D CNN embedding network in a few-shot learning scenario. The other is to increase the weight of the critical spatio-temporal region to increase the inter-class distance and decrease the intra-class distance by the attention fusion mechanism. We perform detailed ablation studies to confirm these two hypotheses and show each module’s impact. Specifically, we evaluate different components as follows: 4.3.1 Pre-training process, 4.3.2 Different layer’s representation of pre-trained 3D CNN models, 4.3.3. Multi-task learning setting, 4.3.4 Attention fusion mechanism, 4.3.5 Meta-learner in attention network, 4.3.6 Residual structure in the attention network.

4.3.1 Pre-training process

We introduce a baseline for comparison, i.e., STAF-Scratch, training from scratch for 3D ResNet-50. Then we compare this baseline with the other five models trained from the combinations of three different large-scale video datasets, i.e., Kinetics-700 [3], Moment-in-time [16], and START-action [27].

Table 2 shows the comparison results on all five few-shot dataset splits. First, by comparing the STAFF-Scratch with the other five pre-trained STAF, we observe that the pre-training process is critical for a few-shot video learning approach with a 3D CNN embedding network. Notably, we obtain a significant difference, i.e., nearly 13% on SSv2-all, for a 5-way 1-shot setting.

Second, both the performance of supervised video classification and data diversity are important in the pre-training process for the few-shot video learning scenario. These two factors ultimately determine the performance of our model. For simplicity, $K$, $M$, and $S$ denote Kinetics-700 [3], Moment-in-time [16], and START-action [27], respectively. Hara et al. [11] show that 3D ResNet-50 pre-trained on $M$ dataset performs better than the model pre-trained in $M+S$ dataset over 6.2% and 3D ResNet-50 pre-trained on $K+M$ dataset performs better than the model pre-trained in $K+M+S$ dataset over 2% for top-1 video level accuracy on HMDB51. However, different from the supervised video classification, Table 2 shows that merging different large-scale video datasets can counteract poor classification performance and even give a more general video representation. For example, although STAF-{$M$} still outperforms STAF-{$M+S$} on HMDB51, the gap between the two models is negligible. And STAF-{$K+M+S$} model performs slightly better than STAF-{$K+M$} in the few-shot video learning task on HMDB51. Although STAF-{$K+M$} still achieves the best results on SSv2-all, the gap is tiny. We see similar patterns in other databases such as UCF101 and SSv2-part. We argue that after merging these three large-scale datasets in our experiments, the total number of classes in the $K+M+S$ is 1,149, which increases the
Table 1. Comparison on 5-way 1-shot and 5-shot benchmarks of UCF101, HMDB51, SSv2-part, and SSv2-all. The best performance is highlighted.

| Method       | # of Classes | UCF101  | HMDB51  | SSv2-part | SSv2-all |
|--------------|--------------|---------|---------|-----------|----------|
| STAF-Scratch | -            | 54.7    | 37.1    | 27.0      | 34.7     |
| STAF         | -            | 339     | 84.7    | 62.2      | 35.8     |
| STAF-(M)     | 439          | 83.4    | 62.0    | 34.5      | 45.4     |
| STAF-(M+S)   | 700          | 88.6    | 65.2    | 37.7      | 49.3     |
| STAF-(K)     | 1039         | 90.3    | 67.3    | 39.9      | 50.9     |
| STAF-(K+M+S) | 1139         | 90.6    | 67.9    | 39.9      | 50.3     |

Table 2. Comparison results with different combination of video dataset in the pre-training process for the 5-way 1-shot video classification.

| Method       | UCF101  | SSv2-all |
|--------------|---------|----------|
| STAF-(layer2) | 60.0    | 28.4     |
| STAF-(layer3) | 70.2    | 42.6     |
| STAF-(layer4) | **90.6** | **50.3** |

Table 3. Comparison results with different layer’s representation of 3D ResNet-50 embedding network for 5-way 1-shot video classification. We use layer2, layer3, layer4 (notation from [11]) to denote the representation from each layer’s representation.

representation performance from different layers on the target task after transfer when dealing with transfer learning. If the target task is not highly relevant to the pre-train task, the top layer may contain specific features for the pre-train task, thus lacking general features. Goyal et al. [8] show that when the jigsaw task is used as a pre-train task, the higher layer of ResNet does not perform well when dealing with image classification tasks. Although the task in our pre-training process was video classification and the target task was a matrix learning task in a few-shot scenario, we also need to evaluate whether the higher layer would generate a more general representation. Table 3 shows that layer4’s representation can achieve the best performance, which means the video classification task is highly relevant to our nearest neighbor video classification in a few-shot learning scenario. It also implies that a global video classification task can help produce more general representation in a few-shot video classification scenario.

4.3.2 Different layer’s representations

After pre-training the 3D CNN embedding network, we extract representations from different layers of the networks. Our experiments use 3D ResNet-50 as our embedding network, so we extract representations from the last layer of every residual step. To simplify, we use conv1, layer1, layer2, layer3, layer4 (notation from [11]) to denote the representation from each stage. We need to evaluate

4.3.3 Multi-task learning setting

Inspired by the results of different layer representations, we add a global video classification task in the multi-task learning setting. Table 4 shows the comparison results in which we fixed other hyperparameters but without global video classification task in the STAF-No-Global model. From the results, we can see that there is an improvement by adding a global classification task, which demonstrates the
Table 5. Comparison results with three variants of STAF for 5-way 1-shot video classification.

| Method         | UCF101 | SSv2-all |
|----------------|--------|----------|
| STAF-Neighbor  | 82.7   | 43.2     |
| STAF-Self      | 90.3   | 49.4     |
| STAF-Cross     | 90.5   | 49.2     |
| STAF           | 90.6   | 50.3     |

Table 6. Comparison results between STAF-NoML-Mean and STAF for 5-way 1-shot video classification. In STAF-NoML-Mean, we leverage a fixed kernel for convolutional operation.

| Method         | UCF101 | SSv2-all |
|----------------|--------|----------|
| STAF-NoML-Mean | 89.9   | 49.3     |
| STAF           | 90.6   | 50.3     |

Table 7. Comparison results between STAF-NoRes and STAF for 5-way 1-shot video classification. In STAF-NoRes, there is no residual design in both the self-attention module and cross-attention module.

| Method         | UCF101 | SSv2-all |
|----------------|--------|----------|
| STAF-NoRes     | 88.9   | 49.2     |
| STAF           | 90.6   | 50.3     |

4.3.4 Attention fusion mechanism

To explore the effectiveness of the attention fusion mechanism, we introduce three comparison models, i.e., STAF-Neighbor, STAF-Self, STAF-Cross. In STAF-Neighbor, representations learned from the embedding network are fed into the nearest neighbor classifier and a global video classifier directly without our attention mechanisms. For STAF-Self and STAF-Cross, before being fed into two classifiers, the representations go through the self-attention and cross-attention mechanism, respectively. Table 5 shows the comparison results. STAF-Neighbor can achieve a better performance than some previous works, demonstrating the effectiveness of the pre-training process for 3D CNN embedding network in a few-shot learning scenario. Compared with STAF-Neighbor, after adding the attention mechanism, all three other models have a performance improvement, demonstrating that representations after the embedding network have some spatio-temporal features related to the non-target action region. The cross-attention mechanism in STAF-Cross aid in highlighting the spatio-temporal features associated with the target action region among the query video and support set. The STAF-Self’s self-attention module helps highlight the spatio-temporal features related to the action in each video itself. Therefore, combining two different attention modules can take advantage of each module to further extract more discriminative spatio-temporal representations. The results in Table 5 demonstrate our argument. We also provide three positive cases from SSv2-all dataset to demonstrate the effect of fusion mechanism in the appendix.

4.3.5 Meta-learner

We evaluate the influence of meat-learner in our STAF by developing a model without the meta-learner, i.e., STAF-NoML-Mean. In STAF-NoML-Mean, we use the average pooling on each relation map($M_{\text{cross}}^{\text{eq}}$ in Eq 6 and $M_{\text{cross}}^{S\leftarrow S}$ in Eq 7) as the kernel to compute the attention map in each self-attention module and cross-attention module. As we can see from Table 6, our STAF with meta-learner outperform STAF-NoML-Mean, which means the meta-learner dynamically generates the kernel to summarize the local features in each relation and correlation map.

4.3.6 Residual structure in the attention network

To verify the effectiveness of residual structure in the attention network, we create a baseline model, i.e., STAF-NoRes, in which we remove the residual design in both the self-attention module and cross-attention module. The result in Table 7 shows that our STAF outperforms the STAF-NoRes, which demonstrates the residual structure is beneficial for few-shot video classification because it helps to remain the similar representation for the videos from the same classes and call attention to the minor differences for videos from the different classes.

5. Conclusion

This paper proposes Spatio-Temporal Attention Fusion network(STAF) for a few-shot video classification. STAF takes advantage of the knowledge learned from the merged large-scale datasets and uses self-attention and cross-attention to highlight the spatio-temporal features to increase the inter-class distance and decrease the intra-class distance. STAF achieves the state-of-the-art performance of 5-way 1-shot and 5-shot video classification on UCF101, HMDB51, and two splits of the SSv2 dataset for few-shot video classification. Through the detailed ablation study, STAF also shows the benefits of the pre-training process, fusion network and multi-task training setting.
References

[1] Mina Bishay, Georgios Zoumpourlis, and Ioannis Patras. Tam: Temporal attentive relation network for few-shot and zero-shot action recognition. In BMVC. 2019. 2
[2] K. Cao, J. Ji, Z. Cao, C. Chang, and J. Niebles. Few-shot video classification via temporal alignment. In CVPR, pages 10615–10624, 2020. 1, 2, 3, 5, 7
[3] Joao Carreira, Eric Noland, Chloé Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset. ArXiv, 2019. 5, 6
[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR, 2009. 6, 7
[5] Carl Doersch, Ankush Gupta, and Andrew Zisserman. Cross transformers: spatially-aware few-shot transfer. In NeurIPS, 2021. 1
[6] Sai Dwivedi, Vikram Gupta, Rahul Mitra, Shuaib Ahmed, Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. Temporal relation reasoning in videos. In NeurIPS, 2019. 2, 5, 6, 7
[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the 34th International Conference on Machine Learning, pages 1126–1135, 2017. 2
[8] Priya Goyal, Dhruv Kumar Mahajan, Abhinav Gupta, and Josh Tenenbaum. A short note on the kinetics-700 human action dataset. ArXiv, 2019. 5, 6
[9] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurai, Ingo Bax, and Roland Memisevic. The “something something” video dataset for learning and evaluating visual common sense. In ICCV, pages 5843–5851, 10 2017. 2, 5
[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, andIllia Polosukhin. Attention is all you need. In NeurIPS, 2020. 1
[11] Adam Santoro, Sergey Bartunov, Matthew M. Botvinick, Daan Wierstra, and Timothy P. Lillicrap. One-shot learning with memory-augmented neural networks. In Proceedings of the International Conference on Machine Learning, 2016. 2
[12] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurai, Ingo Bax, and Roland Memisevic. The “something something” video dataset for learning and evaluating visual common sense. In ICCV, pages 5843–5851, 10 2017. 2, 5
[13] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. CoRR, 2017. 2, 4
[14] Anirudh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml. In ICLR, 2020. 1
[15] Sachin Ravi and H. Larochelle. Optimization as a model for few-shot learning. In ICLR, 2017. 2
[16] Kenso Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In ICCV Workshops, pages 3154–3160, 10 2017. 7
[17] Ronghang Zhang, Li Zhang, Xiaojuan Qi, Hongdong Li, Philip Torr, and Pietro Perona. Few-shot action recognition with permutation-invariant attention. In ECCV, 2020. 1, 2, 3, 5, 6, 7
[18] Xiatian Zhu, Antoine Toisoul, Juan-Manuel Perez-Rua, Li Zhang, Brais Martinez, and Tao Xiang. Few-shot action recognition with prototype-centered attentive learning. ArXiv, 2021. 2, 5, 7
[19] Tsendsuren Munkhdalai and Hong Yu. Meta networks. Proceedings of machine learning research, 70, 03 2017. 2
[20] Harry Cao, J. Ji, Z. Cao, C. Chang, and J. Niebles. Few-shot video classification via temporal alignment. In CVPR, pages 10615–10624, 2020. 1, 2, 3, 5, 7
[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, andIllia Polosukhin. Attention is all you need. In NeurIPS, 2020. 1
[22] Adam Santoro, Sergey Bartunov, Matthew M. Botvinick, Daan Wierstra, and Timothy P. Lillicrap. One-shot learning with memory-augmented neural networks. In Proceedings of the International Conference on Machine Learning, 2016. 2
[23] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. CoRR, 2017. 2, 4
[24] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. In CVPR, 2012. 2, 5
[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In NeurIPS, volume 30, 2017. 2
[26] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In CVPR, 2020. 5, 7
[27] Kenso Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In ICCV Workshops, pages 3154–3160, 10 2017. 7
[28] Kenso Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In CVPR, pages 6546–6555, 2018. 1, 5, 6, 7
[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, pages 770–778, 2016. 7
[30] Ruijing Hou, Hong Chang, Bingpeng Ma, Shiguang Shan, and XiLin Chen. Cross attention network for few-shot classification. In NeurIPS, 2019. 1, 2
[31] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In CVPR, 2014. 7
[32] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In CVPR, 2020. 5, 7
[33] Wei Liu, Hang Su,拖bi Zhang, and Antonino Torralba. Temporal relational reasoning in videos. In ECCV, 2018. 5