TBAC: Transformers Based Attention Consensus for Human Activity Recognition

Santosh Kumar Yadav
AcSIR, CSIR-CEERI, Pilani
santosh.yadav@pilani.bits-pilani.ac.in

Shreyas Bhat Kera
Birla Institute of Technology and Science
f20181119@pilani.bits-pilani.ac.in

Raghurama Varma Gonela
Birla Institute of Technology and Science
f20181120@pilani.bits-pilani.ac.in

Kamlesh Tiwari
Birla Institute of Technology and Science
kamlesh.tiwari@pilani.bits-pilani.ac.in

Hari Mohan Pandey
Bournemouth University
profharimohanpandey@gmail.com

Shaik Ali Akbar
AcSIR, CSIR-CEERI, Pilani
saakbar158@gmail.com

***Abstract*—Human Activity Recognition is an important task in Computer Vision that involves the utilization of spatio-temporal features of videos to classify human actions. The temporal portion of videos contains vital information needed for accurate classification. However, common Deep Learning methods simply average the temporal features, thereby giving all frames equal importance irrespective of their relevance, which negatively impacts the accuracy of the model. To combat this adverse effect, this paper proposes a novel Transformer Based Attention Consensus (TBAC) module. The TBAC module can be used in a plug-and-play manner as an alternate to the conventional consensus methods of any existing video action recognition network. The TBAC module contains four components: (i) Query Sampling Unit, (ii) Attention Extraction Unit, (iii) Softening Unit, and (iv) Attention Consensus Unit. Our experiments demonstrate that the use of the TBAC module in place of classical consensus can improve the performance of the CNN-based action recognition models, such as Channel Separated Convolutional Network (CSN), Temporal Shift Module (TSM), and Temporal Segment Network (TSN). We also propose the Decision Consensus (DC) algorithm that utilizes multiple independent but related action recognizer models in order to improve upon the performance of most of these constituent models, using a novel fusion algorithm. Results have been obtained on two benchmark human action recognition datasets, HMDB51 and HAA500. The use of the proposed TBAC module along with Decision Consensus achieves state-of-the-art performances, with 85.23% and 83.73% classification accuracies on the two databases HMDB51 and HAA500, respectively. The code will be made publicly available.**

***Index Terms*—Video Action Recognition, Human Activity Recognition, Transformers, Temporal Attention, Consensus, Convolutional Neural Networks**

I. Introduction

Action recognition is one of the representative tasks in computer vision, which aims to classify actions performed in videos into predefined classes based on certain spatio-temporal features. It finds applications in many domains including human-robot interaction, sports analysis, video understanding, behavior analysis, *etc.* [1].

The two dominant types of information that are considered while classifying actions are spatial and temporal features extracted from videos. Although information of the spatial category can be derived from individual frames, they lack the ability to give a global understanding of the video, which is why the interrelation between contexts of the frames provided by temporal features is necessary. Video Action Recognition involves the challenges of identifying interclass similarities and intraclass variations, making it critical to utilize relevant spatial and temporal features in the most desirable way to achieve greater performance.

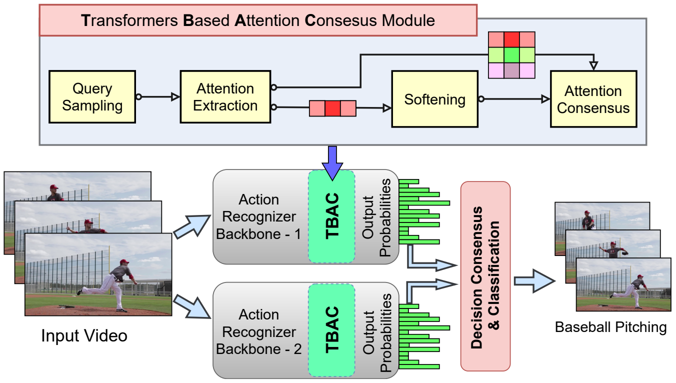
Transformers Based Attention Consesus Module
Query Sampling
Attention Extraction
Softening
Attention Consensus
Action Recognizer Backbone - 1
TBAC
Output Probabilities
Action Recognizer Backbone - 2
TBAC
Output Probabilities
Decision Consensus & Classification
Input Video
Baseball Pitching

Fig. 1. Overview of the TBAC Module with Decision Consensus. Suitable action classification networks can be plugged into our Decision Consensus (DC) function to derive improved performance.

In this paper, we propose the Transformer Based Attention Consensus (TBAC) module, a novel approach to video action recognition that supplements CNN-based architectures designed for the same task. The introduced TBAC module consists of four main components: 1) Query Sampling Unit, 2) Attention Extraction Unit, 3) Softening Unit, and 4) Attention Consensus Unit. A backbone first converts the input video to a spatio-temporal context map, from which an assortment of features is drawn by the Query Sampling Unit to serve as a query set. The attention-based integrants of the Attention Extraction Unit generate values representative of the importance of the given temporal features. The Softening Unit alters the relative sensitivity towards these attentional values to balance any acute bias. As a replacement to the standard consensus of the original network, the Attention Consensus Unit treats the softened values as relevance measures to derive a final feature representation for classification. We further propose the Decision Consensus algorithm, an innovative method to enhance accuracy using the confidence scores of employed video action recognizers.

Our key contributions are as follows: We introduce the TBAC module, a novel concept that uses attention from a transformer-based mechanism to reach a consensus between temporal features, which can be used in a plug-and-play fashion. We use the notion of balanced attention to sustain a degree of fairness among the temporal relations. Moreover, we propose the Decision Consensus algorithm to bring the best out of individual, but related video action recognizers. We analytically assess our proposals with comprehensive ablation studies to exhibit the efficacy of each Unit as well as the overall impact of both the TBAC module and the Decision Consensus algorithm. Experimental results demonstrate that our methods outperform prior state-of-the-art action recognition networks, on both the HMDB51 and HAA500 datasets. We report accuracies of 85.23% and 83.73% on the HMDB51 and HAA500 datasets, respectively.

II. Related Works

A. Action Recognition using CNN Architectures

In recent years, Convolutional Neural Networks (CNNs) have achieved remarkable performance in almost all domains of computer vision [2]–[14]. In particular, CNNs have become the de-facto base network for almost any recent video recognition network architecture. Video Action Recognition has seen tremendous growth in recent years. DeepVideo [15] is among the earliest architectures to apply deep CNNs for video action recognition. There are two trends in video action recognition, $i.e.$ Two-Stream CNNs, and 3D-CNNs. The first trend utilizes two streams for spatial and temporal feature extraction. This trend started from the *Two-Stream Networks* [16], followed by many others like TSN [17], Fusion [18], TRN [19], $etc.$ TSN [17] samples video clips from evenly divided segments in order to effectively learn representations of the whole video. To capture information along with temporal dimension, TRN [19] and TSM [20] utilize a shift module by replacing average pooling with an interpretable relational module. However, two-stream networks generally utilize a 2D CNN, which incorporates the mechanism of average consensus among the temporal features, which may inhibit the model from effectively utilizing relevant information. The second trend uses 3D convolutional kernels throughout the whole architecture to jointly model the spatial and temporal semantics. Examples include C3D [21], I3D [22], R3D [23], Non-Local [24], SlowFast [25], R(2+1)D [26], $etc.$ Another important 3D CNN architecture, which we make use of, is Channel-Separated Convolutional Networks (CSN) wherein the interactions among channels are performed separately from the interactions among spatio-temporal features [27]. These 3D CNN architectures also utilize a mechanism that gives equal importance to all the temporal features, such as the global spatio-temporal pooling layer of CSN, which may also suffer the same limitation as average consensus.

B. Transformers in Vision

Transformer-based networks have made significant progress in tackling various problems in Deep Learning. These networks rely on the concept of trainable attention which helps identify complex dependencies between the elements of each input sequence [28]. Transformers, introduced in [29], is intended to address the challenges of sequence modeling tasks, and its success in the field of NLP has motivated researchers to pursue applications of transformers in the domain of Computer Vision. The transformer's primary concept of self-attention has been availed in many recent methodologies in diverse visual tasks, such as DETR [30] for object detection, ViT [31] and DeIT [32] for image classification, PVT [33] for dense prediction tasks, TransTrack [34] for multiple object tracking, Relaxed Transformers [35] for action proposal, and CrossTransformers [36], LTM-BERT [37], GCF-NET [38], and GTA [39] for action recognition. The remarkable ability of attention to gauge interdependencies among sequential contexts incentivizes us to use it in our method. To the best of our knowledge, the proposed method is the earliest to explicitly use attention as a means to calculate weights for deriving a temporal consensus.

III. Proposed Methodology

In this section, we delineate the architecture of action recognizers that make use of TBAC. The starting point is a pre-existing action recognizer with a head that performs average consensus on temporal features. This model serves as the base network, into which the TBAC module can be inserted, simply by slicing out the aforementioned consensus operation and loading the TBAC module in its place, in a plug-and-play fashion. A succinct, black-box overview of the model is as follows: the base network produces a spatio-temporal map from the input video, which are fed to the TBAC module to provide a temporal consensus, and finally, these values produce confidence scores for the C classes of the training data. We further detail the Decision Consensus algorithm which can also be used in a plug-and-play manner to improve performance by using the class scores of distinct but related action recognizers.

A. Base Network

The backbones have been derived from ResNet-based architectures such as CSN, TSM, and TSN. Before the operations of the backbone commence, T frames are extracted from the RGB video to be resized, normalized, and finally reshaped to a dimension of $T \times 3 \times H \times W$ (3 channels for RGB). The numerous convolutions and pooling layers transform the dimensions to $T' \times D \times \frac{H}{32} \times \frac{W}{32}$, where D is the specified channel dimension. For CSN $T' = \frac{T}{8}$, while for TSM and TSN $T' = T$. Further, for TSN and TSM, spatial average pooling is included within the backbone to allow the operations of the head to work solely on temporal features. For CSN, the temporal portion of the already existing global spatio-temporal average pooling is removed. The extracted feature map, having a dimension of $T' \times D$, is submitted to the TBAC module for temporal consensus.

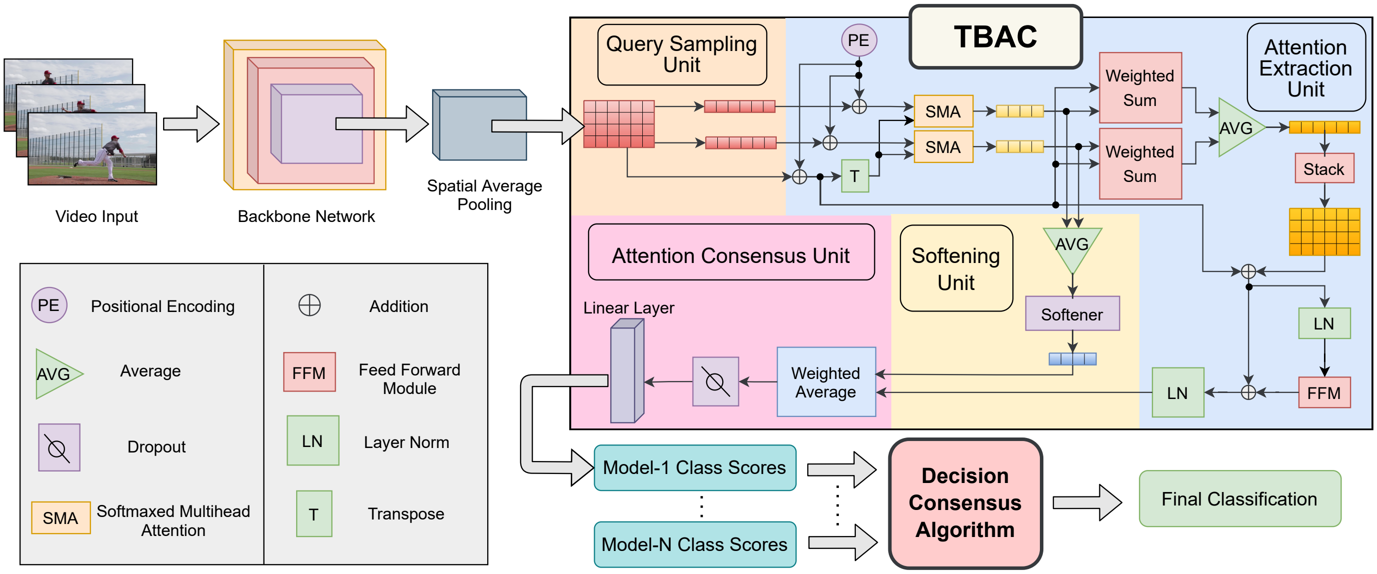

Fig. 2. Overview of the TBAC Module with Decision Consensus architecture: A CNN-based backbone extracts spatio-temporal features from an input RGB video. After spatial averaging, the Querying Sampling Unit chooses certain queries which are passed along with a key and value to the Attention Extraction Unit. This Unit makes use of transformer-based operations in order to generate attention maps and transform the temporal features. These attention maps are softened, then used as weights to arrive at a consensus between the temporal features given to the Attention Consensus Unit. Action class scores are generated from the final feature representation. The Decision Consensus algorithm obtains a consensus, based on output class probabilities of independent, but related action recognizers (with TBAC Modules inserted) in order to improve performance.

B. The TBAC Module

Broadly, the TBAC module, shown in Fig. 2, provides a consensus of temporal features by harnessing the attention mechanism of transformers, with a specific focus on balanced attention. To reach the final output vector of dimension $1 \times D$, several computations ensue which can be segregated into Units based on their sequential ordering and functionalities. Each Unit is described in the following subsections.

Query Sampling Unit: Before generating attention maps from the input features, appropriate queries must be sampled. The feature map can be visualized as T' temporally distinct vectors for a given video. A set of w vectors chosen to be the queries and their indices, $i.e.$ their temporal location within the feature map can be denoted by:

$$P = \{(p_1, ..., p_w) \in \mathbb{N}^w : 1 \leq p_i \leq T'\} \tag{1}$$

Thus, the query set is $Q = \{q_{p_1}, ..., q_{p_w}\}$. Any of the vectors can be selected as a query, however, our focus turns to the temporally-central feature (as used in [40]) as well as the temporally-starting and temporally-ending features. The motive behind this sampling of a limited number of queries is to generate moderate attention values since they would neither depend on every temporally distinct feature, nor on a singular one.

Attention Extraction Unit: This Unit aims to represent the extracted features from the action recognizer backbone in a way that reflects the requisite attention between the frames as well as outputs the attention maps themselves. Initially, incoming features are imbued with a sense of their relative position within the native video by adding positional encodings (PE), calculated in a similar manner as given in [29]. Since PE is also added to the queries, the new transformed set of queries is represented as $Q' = \{q'_{p_1}, ..., q'_{p_w}\}$, where

$$q'_{p_i} = q_{p_i} + PE(p_i) \; ; \quad 1 \leq i \leq w \tag{2}$$

The attention mechanism performed in this unit requires the selection of a key (K), value (V) pair as well. The natural choice is the convolutional feature map procured from the backbone, as it contains the temporal context of the input video. The natural choice for the key (K), value (V) pair required in the upcoming attention calculation is the feature map procured from the backbone, as it contains temporal context information. The two matrix multiplications of the scaled dot product computation of attention are bisected in order to extricate the intermediate attentional value. Finally, a Softmax operation is performed to get the resulting set $A = \{a_{p_1}, .., a_{p_w}\}$, where

$$a_{p_i} = Softmax\left(\frac{q'_{p_i} K^T}{\sqrt{D}}\right) \; ; \quad 1 \leq i \leq w \tag{3}$$

Th subsequent step is to employ the attentional values in a weighted sum operation on the value V to provide attentional outputs $O = \{o_{p_1}, ..., o_{p_w}\}$, where

$$o_{p_i} = a_{p_i} V \; ; \quad 1 \leq i \leq w \tag{4}$$

The mean of this set, denoted by $\bar{O}$ is calculated. These evaluations are for single-head attention, but can readily be extended to incorporate multiple heads. The weighted sum operation may be adversely affected if the set A is largely imbalanced, which occurs if one temporal feature garners an inordinate amount of attention. Evidently, a residual connection between $\bar{O}$ and the original feature map (F) can alleviate this issue. Since there is a mismatch in the temporal dimension, the values of $\bar{O}$ are first stacked then the residual connection is implemented to produce the preliminary residual output R, as follows:

$$R = F + \phi(\bar{O}, T') \tag{5}$$

where, $\phi(.)$ is the stacking function, which is given $\bar{O}$, and T' (the number of times to be stacked). Thus, these modifications to the standard transformer operations help balance the attentional values. A branch incorporating a standard Feed Forward Module (FFM) and LayerNorm operations is appended as seen in Fig. 2. These culminating operations on R provide the Attention Extraction Unit output Γ. The attentional maps (A) are passed to the Softening Unit, while (Γ) is forwarded to the Attention Consensus Unit.

Softening Unit: The purpose of this Unit is to allay any possible unduly large disparity between the attention values, by abating their relative variance. The inceptive step of this unit is to reach an agreement between the w number of attention maps in A, which is done by computing their mean, denoted by $\bar{A} = \sum_{i=1}^{w} \frac{a_{p_i}}{w}$. The initial standard deviation (σ) and $\frac{1}{T'}$ *i.e.* the original mean is calculated from the T' attentional values, and using a provided value of moderated standard deviation ($\bar{\sigma}$) the softened attention map ($\bar{A}_S$) is computed as:

$$\bar{A}_S = \left(1 - \frac{\bar{\sigma}}{\sigma}\right) * \left(\frac{1}{T'}\right) + \left(\frac{\bar{\sigma}}{\sigma}\right) * (\bar{A}) \tag{6}$$

Notably, this equation maintains both the initial mean and relative priority, modifying only the standard deviation of the values. Performing this operation ensures that neither a few attentional values will overwhelmingly impact the classification nor will an initial derisory attentional value amount to a loss of temporal information that may have been fruitful to classification. Thus, this softened attention map is balanced.

Attention Consensus Unit: The common practice of classical algorithms is to simply average the temporal dimension for consensus, which may be detrimental to reaching higher performance. The averaging operation treats each of the frames' representations with equal weightage. This may cause misclassification when certain temporal features which possess more relevant information pertaining to the actual action category to be given less importance than necessary. Further, the inverse may also hold: irrelevant features are attended to with just as much importance as other features. To mitigate these adverse effects, this unit requires information about the relative relevance of the features present in Γ. This information is already present in the attention maps ($\bar{A}_S$). Unlike average consensus, these maps possess a variance that is not zero, while also not being extreme. They can be thought of as measures of relative importance between the temporal features, and thereby used as weights for a weighted average (M) taken on Γ. This operation can be expressed as $M = \bar{A}_S\Gamma$. Finally, a dropout is applied to this output, which is then fed to a Linear Layer to derive class probability scores.

C. Decision Consensus

This work also proposes the Decision Consensus module, which can complement the TBAC-inserted action recognizers. Rather than direct classification, this module uses the class probabilities obtained from the input action recognizers to compute new class probabilities. The following equations can readily be extended to n models, however, for the sake of simplicity, we assume this module extracts class probabilities from two trained recognizers, VAR_X and VAR_Y. For C classes, the output probabilities can be expressed as $X = (x_1, ..., x_C)$ and $Y = (y_1, ..., y_C)$. Taking one model (X in this case) as the primary, we first compute the apposite confidence measures (CM_X and CM_Y) as follows:

$$CM_X = \sum_{i=1}^{C} \left(x_i * x_i\right), \quad CM_Y = \sum_{i=1}^{C} \left(y_i * x_i\right) \tag{7}$$

Subsequently, these values are used in a weighted average between the original class probabilities:

$$DC(X, Y) = \frac{CM_X * X + CM_Y * Y}{CM_X + CM_Y} \tag{8}$$

The intuition behind these calculations is that each model learns a different feature space, which can be combined in an effective manner. The strategy described above is loosely based on the concept of attention, as when the models are in concord, the Decision Consensus module outputs the class they agreed upon, while when they fail to reach a consensus, the model with a greater confidence score is paid more attention. We expect that using this technique of giving proportional relevance can surpass both the constituent models' performances and the performance when simply averaging the probabilities. The complete algorithm is included in Algorithm 1.

IV. Experimental Results

All the experimental results mentioned in this section were obtained by evaluating the final model and various ablations on the HAA500 and the HMDB51 datasets. We obtained the base models from the MMAction2 repository [41]. The ResNet architectures used for both the TSN and TSM derived backbones had a depth of 50, while that of the CSN derived backbone had a depth of 152. For the TBAC-applied action recognizers using a TSN or TSM base, the value of the temporal resolution (T) was kept at 8, while for TBAC-applied CSN based networks, an initial resolution of either 32 or 48 was used, which was subsequently downsampled by the temporal poolings of the component layers to produce a temporal dimension of 4 and 6, respectively, as inputs to the TBAC module. The CSN backbone made use of the interaction-reduced setting of its component blocks, as described in [27]. The feature dimensionality we used was $D = 2048$. The 2 layer MLP used an intermediate representation with feature dimension $D' = 4 * D$. We also augmented the data with random flips and crops, which tends to improve accuracy as seen in [40].

```
Input: Class scores of Primary Model ($X$) and Class
       scores of N Secondary Models ($(Y_1, ..., Y_N)$)
Output: Class scores after Decision Consensus (MC)
Function ModelConsesus($(X, Y_1, Y_2, ...Y_N)$):
    $C \longleftarrow X.length$
    $CM_X \longleftarrow 0$
    for $i$=$1$ to $N$ do
        $CM_{Y_i} \longleftarrow 0$
    end
    $MC[1..C] \longleftarrow 0$
    for i = 1 to C do
        $CM_X = CM_X + (X[i] * X[i])$
        for j = 1 to N do
            $CM_{Y_j} = CM_{Y_j} + (Y_j[i] * X[i])$
        end
    end
    for i = 1 to C do
        $MC[i] = \frac{CM_X * X[i] + CM_{Y_1} * Y_1[i] + ... + CM_{Y_N} * Y_N[i]}{CM_X + CM_{Y_1} + ... + CM_{Y_N}}$
    end
    return $MC$
END
```

Algorithm 1: Decision Consensus Algorithm

The models were trained on 2 8GB Quadro P4000 GPUs. The optimizer chosen was SGD and the dropout ratios were set to 0.4 for TSN derived models and 0.5 for TSM and CSN derived models. The loss used was Cross-Entropy Loss.

A. Dataset Details

The two datasets considered are representative of a broad range of HAR tasks, since HMDB51 contains generic class labels, while HAA500 involves temporally sensitive classes with high inter-class similarity.

HMDB51: [42] The Human Motion Data Base dataset is a large collection of videos containing general action classes. The dataset is composed of 6,849 video clips from 51 action categories with each class having a minimum of 101 clips. The evaluation is done by taking the average over three different training/testing with 70 clips for training and 30 clips for testing per class.

HAA500: [43] The Human-centric Atomic Action dataset is a fine-grained, manually annotated dataset used for Action Recognition. It contains 10,000 videos of 500 classes of different actions. Each class has 20 videos that can be split into train, test, and validation sets in the ratio 16:3:1.

B. Ablation Studies

We perform our experiments with various models (primarily using the base models of TSN, TSM, and CSN). Note that the TBAC-applied models involved in the ensuing experiments are initialized with pre-trained weights; TSN and TSM derived models with Kinetics-400 [22] weights and CSN derived models with IG65M [44] weights.

TABLE I
EFFECT OF ADDING TBAC MODULES TO TSM, TSN, AND CSN FOR HAA500 [43].

Model	Pre-trained	TBAC	Top-1	Top-3
TSM	Kinetcs-400	×	52.60%	74.00%
TSM	Kinetcs-400	✓	55.60%	75.33%
TSN	Kinetcs-400	×	56.10%	77.86%
TSN	Kinetcs-400	✓	61.47%	83.27%
CSN (32f)	IG65M	×	80.13%	94.00%
CSN (32f)	IG65M	✓	81.93%	95.53%

TABLE II
COMPARISON OF TSN ON HAA500 [43] WITH AND WITHOUT CERTAIN TBAC UNITS.

Model	Attention Extraction Unit	Attention Consensus Unit	Softening Unit	Top-1 Accuracy
TSN	×	×	×	56.10%
	✓	×	×	57.00%
	✓	✓	×	60.33%
	✓	✓	✓	61.47%

Baselines.: The TSN-derived networks differ in two properties from the results reported in [43]: firstly, they make use of pre-trained weights, and secondly, they make use of the TBAC module. The first step, therefore, would be to set suitable baselines against which fair comparisons can be made. We first train a TSN model initialized with Kinetics-400 weights, which is devoid of any TBAC-related components. Many action recognition frameworks make use of optical flow as an input stream during training, nevertheless, for this portion of the discussion, we solely concentrate on RGB video inputs. Using weights pre-trained on Kinetics-400, the accuracy of TSN increases by 0.77% (from 55.33% to 56.10%). Further, since we also make use of the TSM model when inserting the TBAC unit, we train a baseline TSM model with its pre-trained weights, resulting in an accuracy of 52.60%. Similarly, for CSN (32 frames), baseline accuracies of 80.13% and 82.92% are obtained for the HAA500 and HMDB51 datasets, respectively.

Performance of the TBAC module.: Having set suitable baselines, we can directly compare the networks with and without the use of the TBAC module. We use the TSM, TSN and CSN derived architectures to exemplify the performance of the TBAC module on the HAA500 dataset. Further, we use the CSN derived architecture to illustrate the same for the HMDB51 dataset. The results can be seen in TABLE I.

The effect of adding the TBAC module, without any other runtime modifications, provides substantial improvements; an increase of 3%, 5.37%, and 1.8% in Top-1 accuracy for TSM, TSN, and CSN respectively. Using the TBAC module for CSN on the HMDB51 dataset improves the performance from 82.92% to 83.73%.

Impact of TBAC Units.: Having seen the overall effect of adding the TBAC module, we can evaluate the impact of the units within the TBAC module by comparing certain settings of the module, with and without selected units. The

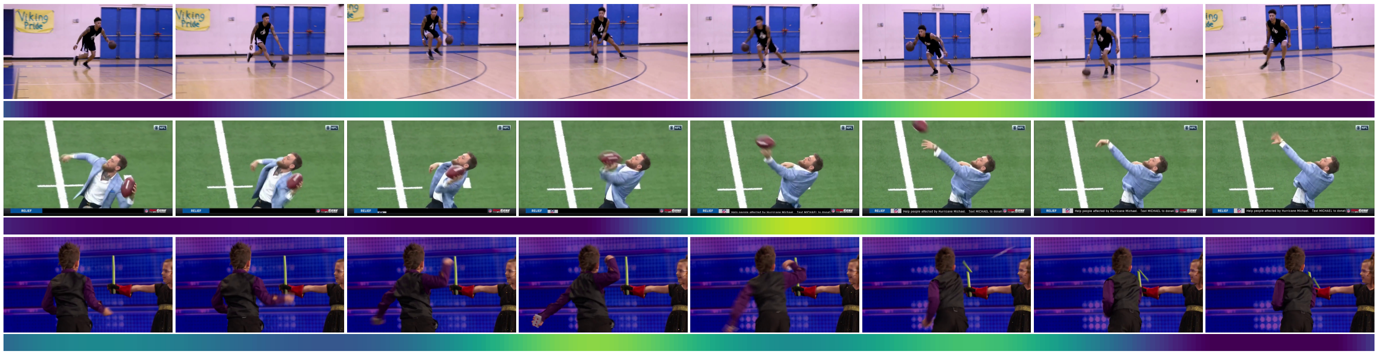

Fig. 3. Example visualizations for the TBAC-TSN model on the HAA500 dataset where higher attention values denote a greater relevance for that frame during attention consensus.

three settings (apart from the previous baseline) for this experiment are chosen in a step-by-step manner. The first setting comprises a TSN base plus the use of only the Attention Extraction Unit. This works by forwarding the $1 \times D$ vector - computed after the weighted sum operation in the unit - directly to the final Linear Layer for classification. The second setting is the TSN base plus both the Transformer and Attention Consensus Units (since the presence of the Attention Consensus Unit necessitates the existence of the Attention Extraction Unit). The third setting is the TSN base with the Transformer, Attention Consensus, and Softening Units. All settings make use of Kinetics-400 [22] pre-trained weights and RGB input modalities. The results of these settings can be seen in TABLE II.

These results show that the combination of TBAC's units works best in order to achieve higher accuracies. Furthermore, all subsequent additions of TBAC units generate an incremental accuracy improvement. The addition of the Attention Extraction Unit increases accuracy by 0.9%, while the greatest boost of 3.33% is obtained when adding our novel Attention Consensus Unit. The Softening Unit too improves the accuracy by 1.14%. Visualization of the attention heatmaps generated by the Attention Extraction Unit is given in Fig. 3.

To analyze the choices that can be made for the parameters of the Query Sampling Unit, we used the TSN-derived TBAC model, starting from Kinetics-400 pre-trained weights. There were two parameters to be considered for this Unit, one was the value of w, and the other was choosing values for p_1 to p_w. For sampling a combination of queries from a set of size T', a total of $2^{T'} - 1$ different choices can be made. Thus for the sake of brevity, we considered three choices that semantically encompassed the range of possible options. Firstly, we took $w = 1$ and $p_1 = \frac{T'}{2}$, i.e. only the middle temporal index was chosen as query. Secondly, we took $w = T'$ with $p_1 = 1, p_2 = 2, \ldots, p_w = w$, thus sampling every temporal index possible as queries. Thirdly, we chose a more balanced version with $w = 3$ and $p_1 = 1, p_2 = \frac{T'}{2}$ and $p_3 = T'$, thereby sampling the first, middle and ending temporal indices as queries. The results can be seen in Table III.

TABLE III
Effect of different choices for selective querying.

Model	w	Values of p	Top-1 Accuracy
TBAC-TSN	1	$T'/2$	58.93%
	T’	1,2,...,T'	60.60%
	3	$1, T'/2, T'$	61.47%

TABLE IV
Outcome of Decision Consensus (DC) on TBAC-TSN and TBAC-TSM trained on HAA500 [43].

Model	Top-1	Top-3
TBAC-TSM	55.60%	75.33%
TBAC-TSN	61.47%	83.27%
DC (TBAC-TSN, TBAC-TSM)	64.47%	84.93%

TABLE V
Outcome of Decision Consensus on Splits of HMDB51 using TBAC-CSN variants.

Split	32 frames	48 frames	Average Consensus	Decision Consensus
Split-1	83.46%	84.84%	84.90%	85.10%
Split-2	84.84%	85.62%	85.95%	86.47%
Split-3	82.88%	83.40%	83.99%	84.12%
Average	83.73%	84.62%	84.95%	85.23%

TABLE VI
Comparison with the state-of-the-art results on the **HMDB51** [42] dataset. **Blue** represents the previous state-of-the-art. **Red** denotes the best results.

Model	Pre-trained	Top-1 Accuracy
ResNeXt101 [8]	Kinetics-400	81.78%
ResNeXt101 BERT [37]	Kinetics-400	83.55%
R(2+1)D BERT (32f) [37]	IG65M	83.99%
R(2+1)D BERT (64f) [37]	IG65M	**85.10%**
CSN (32f)	IG65M	82.92%
TBAC-CSN (32f) (ours)	IG65M	83.73%
TBAC-CSN (48f) (ours)	IG65M	84.62%
DC-TBAC-CSN (32f, 48f) (ours)	**IG65M**	**85.23%**

Ablation studies for different settings of the Query Sampling Unit are included in the Appendix.

TABLE VII
COMPARISON WITH THE STATE-OF-THE-ART RESULTS ON THE **HAA500** [43] DATASET. **BLUE** REPRESENTS THE PREVIOUS STATE-OF-THE-ART. **RED** DENOTES THE BEST RESULTS.

Model	Pre-trained	Top-1 Accuracy	Top-3 Accuracy
TSM [20]	Kinetics-400	52.60%	74.00%
TBAC-TSM(ours)	Kinetcs-400	55.60%	75.33%
TSN [17]	×	55.33%	75.00%
TSN [17] (Flow)	×	49.13%	66.60%
TSN [17] (Two-Stream)	×	**64.40%**	**80.13%**
TSN	Kinetics-400	56.10%	77.86%
TBAC-TSN(ours)	Kinetcs-400	61.47%	83.27%
DC (TBAC-TSN,TBAC-TSM)(ours)	Kinetcs-400	64.47%	84.93%
Semi-supervised Few-Shot [45]	×	**80.68%**	**-**
CSN (32f)	IG65M	80.13%	94%
TBAC-CSN (32f) (ours)	IG65M	81.93%	95.53%
TBAC-CSN (48f) (ours)	IG65M	82.40%	94.87%
DC-TBAC-CSN (32f, 48f) (ours)	**IG65M**	**83.73%**	**95.73%**

Impact of Decision Consensus.: The second proposal of this paper is a methodology aimed at enhancing the overall performance of models by extrapolating a better class probability distribution from the constituent models used. Taking the case of two models, namely TBAC-TSN and TBAC-TSM trained on the HAA500 dataset, we can see that the disparity between the Top-1 and Top-3 accuracies is large - around 20% difference for both. This empirically shows that in many cases, the correct class's probability may be within the top few probabilities, just not with the necessary confidence that it would need for a correct prediction. This is where the Decision Consensus mechanism can help, by allowing two distinct models to interact and arrive at a consensus, based on their respective class scores. Results for Decision Consensus between TBAC-TSN and TBAC-TSM can be found in TABLE IV.

Decision Consensus thus provides a boost of 8.87% and 3% over the Top-1 accuracies of TBAC-TSM and TBAC-TSN, respectively. We also find that a Decision Consensus can be found between two of the same base models, but with varying input frame rates (T). The intuition behind using this approach can be loosely derived from the concept introduced in [25], where differing input frame rates can cause the same model to learn differing semantic information. To demonstrate this notion, we test the Decision Consensus mechanism on the TBAC-CSN model variants trained on the HMDB51 dataset. To exhibit the consistency that this function has on improving accuracy, all 3 splits of the HMDB51 dataset as well as the average of these values, before and after Decision Consensus are evaluated. Further, the two temporal resolutions taken are $T = 32$ and $T = 48$. We had stated earlier that we expect the performance of Decision Consensus to exceed that of naive averaging, so we include a column for average consensus. As shown in TABLE V, for every split, the Decision Consensus increases the overall accuracy, by 1.5% for the 32 frame setting and 0.61% for the 48 frame setting. The Decision Consensus accuracy obtained from the two constituent TBAC-CSN models is consistently better than that of simple average consensus, by a margin of 0.28%. Thus, we confirm our initial hypothesis, which stipulated that the Decision Consensus function's weighted average mechanism is superior to a simple average.

C. *Comparison with State-of-the-Art*

Here, we compare the results of our proposed TBAC-inserted models against relevant state-of-the-art methods. Although all TBAC inserted models improve results, TBAC-CSN models with IG65M pre-trained weights generally provide the highest accuracies. We show the results on the two benchmark datasets mentioned earlier: HMDB51 in TABLE VI, and HAA500 in TABLE VII.

For the HMDB51 dataset, the Decision Consensus on TBAC-CSN (32 and 48 frames) reaches a new peak accuracy of 85.23%, which is higher than the previous state-of-the-art by a margin of 0.13%. The component models of TBAC-CSN (32f) and TBAC-CSN (48f) too obtain comparable performance with the prior state-of-the-art.

For the HAA500 dataset, the Top-1 accuracy is increased by a margin of 19.33% using Decision Consensus on TBAC-CSN (32f, 48f). Unless specified, models use RGB inputs. Although we display the result for the proposed model in [45], we do not consider it for comparison as it makes use of a train/test split of 310/156 classes, which differs from the split originally provided by the authors of HAA500 [43]. We use all 500 classes for training as well as testing. The component models themselves top the prior state-of-the-art, taken from [43] itself; TBAC-CSN (32f) and TBAC-CSN (48f) exceed the benchmark accuracy by 17.53%, and 18%, respectively. Thus, our models perform well on a coarse-grained atomic action dataset like HMDB51, as well as on a fine-grained atomic action dataset like HAA500.

V. CONCLUSION

This paper proposed the Transformer Based Attention Consensus (TBAC) module. This module can be used in a plug-and-play fashion with existing CNN architectures for action recognition. The extensive experimental analysis showed that introducing the TBAC module improved the recognition performance of CNN action recognition architectures. Additionally, we proposed the Decision Consensus (DC) algorithm that boosts performance by generating new apposite class probabilities based on the confidence scores of similar action recognizers. DC can also be used in a plug-and-play manner. The proposed model has outperformed the state-of-the-art on the HMDB51 and HAA500 datasets, providing accuracies of 85.23% and 83.73% respectively using DC-TBAC-CSN (32f, 48f). The extensive ablation studies quantitatively demonstrate the effectiveness of our proposed TBAC module and Decision Consensus Algorithm.

REFERENCES

[1] Y. Zhu, X. Li, C. Liu, M. Zolfaghari, Y. Xiong, C. Wu, Z. Zhang, J. Tighe, R. Manmatha, and M. Li, "A comprehensive study of deep video action recognition," *arXiv preprint arXiv:2012.06567*, 2020.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[3] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[4] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[5] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 633–641.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[11] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[12] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[13] A. Kirillov, R. Girshick, K. He, and P. Dollár, "Panoptic feature pyramid networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6399–6408.

[14] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, and L. Wang, "Tea: Temporal excitation and aggregation for action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 909–918.

[15] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[16] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *arXiv preprint arXiv:1406.2199*, 2014.

[17] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks for action recognition in videos," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 11, pp. 2740–2755, 2018.

[18] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1933–1941.

[19] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 803–818.

[20] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7083–7093.

[21] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[22] J. Carreira and A. Zisserman, "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.

[23] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6546–6555.

[24] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803.

[25] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6202–6211.

[26] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6450–6459.

[27] D. Tran, H. Wang, L. Torresani, and M. Feiszli, "Video classification with channel-separated convolutional networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5552–5561.

[28] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser *et al.*, "Rethinking attention with performers," *arXiv preprint arXiv:2009.14794*, 2020.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[30] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*. Springer, 2020, pp. 213–229.

[31] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[32] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," *arXiv preprint arXiv:2012.12877*, 2020.

[33] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," *arXiv preprint arXiv:2102.12122*, 2021.

[34] P. Sun, Y. Jiang, R. Zhang, E. Xie, J. Cao, X. Hu, T. Kong, Z. Yuan, C. Wang, and P. Luo, "Transtrack: Multiple-object tracking with transformer," *arXiv preprint arXiv:2012.15460*, 2020.

[35] J. Tan, J. Tang, L. Wang, and G. Wu, "Relaxed transformer decoders for direct action proposal generation," *arXiv preprint arXiv:2102.01894*, 2021.

[36] T. Perrett, A. Masullo, T. Burghardt, M. Mirmehdi, and D. Damen, "Temporal-relational crosstransformers for few-shot action recognition," *arXiv preprint arXiv:2101.06184*, 2021.

[37] M. E. Kalfaoglu, S. Kalkan, and A. A. Alatan, "Late temporal modeling in 3d cnn architectures with bert for action recognition," in *European Conference on Computer Vision*. Springer, 2020, pp. 731–747.

[38] J. Hsiao, J. Chen, and C. Ho, "Gcf-net: Gated clip fusion network for video action recognition," in *European Conference on Computer Vision*. Springer, 2020, pp. 699–713.

[39] B. He, X. Yang, Z. Wu, H. Chen, S.-N. Lim, and A. Shrivastava, "Gta: Global temporal attention for video action understanding," *British Machine Vision Conference*, 2021.

[40] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, "Video action transformer network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 244–253.

[41] M. Contributors, "Openmmlab's next generation video understanding toolbox and benchmark," https://github.com/open-mmlab/mmaction2, 2020.

[42] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *2011 International conference on computer vision*. IEEE, 2011, pp. 2556–2563.

[43] J. Chung, C.-h. Wuu, H.-r. Yang, Y.-W. Tai, and C.-K. Tang, "Haa500: Human-centric atomic action dataset with curated videos," *arXiv preprint arXiv:2009.05224*, 2020.

[44] D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 046–12 055.

[45] X. Ni, S. Song, Y.-W. Tai, and C.-K. Tang, "Semi-supervised few-shot atomic action recognition," *arXiv preprint arXiv:2011.08410*, 2020.