Ablation Studies on Boundary-Matching Network for Temporal Action Localization

Ben Sukboontip
Carnegie Mellon University
Pittsburgh, PA
bsukboon@cs.cmu.edu

Jeffery Cao
Carnegie Mellon University
Pittsburgh, PA
jeffery2@cs.cmu.edu

Prasoon Varshney
Carnegie Mellon University
Pittsburgh, PA
pvarshne@cs.cmu.edu

Su Park
Carnegie Mellon University
Pittsburgh, PA
suminpar@cs.cmu.edu*

Abstract

Temporal Action Localization (TAL) aims to automate the detection of activities within a video stream and output the start and end timestamps. The Boundary Matching Network [3] (BMN) introduced by Lin et. al. is a landmark paper in this domain that achieves state of the art performance on the two large-scale datasets, ActivityNet1.3 [1] and Thumos14 [2]. In this work, we used the two-stream network [5]'s pre-trained feature embeddings for videos in the ActivityNet dataset to implement the baseline Boundary Matching Network (BMN) for TAL. On reproducing the baseline, we noticed severe overfitting and set out to address it. We applied several different data augmentation and ensembling techniques to achieve a superior generalization performance. Our contributions [2] include ablation studies using temporal shift modules, adding global information using Squeeze and Excite modules, ensembling baseline with models trained on reversed video features and labels. **We are able to achieve a 0.9 point AUC improvement** over our reproduction of the baseline BMN model. **We further contribute by conducting a detailed error analysis** of the dataset and provide examples where our model performs well but the ground truth is mislabeled.

1 Introduction

The amount of video content available on the Internet has grown rapidly in the past decade. Automatically identifying when a specific action starts and ends within a video clip is of immense practical value when applied to tasks like elderly care or surveillance footage analysis, hence has become a research discipline of its own. This is a challenging task for untrimmed video content of various lengths as they can contain multiple action instances and background scenes in one video and be extremely unconstrained in terms of space and time.

Long, untrimmed video content can be processed in two steps, temporal action localization and action classification, which can occur either sequentially or simultaneously (independently of one another). Temporal action localization identifies the relevant portions of a video by proposing the start and end timestamps using a probabilistic model, whereas action classification entails understanding the types

*Equal contributions. Authors listed alphabetically.

[2]Our code is available at https://github.com/11785-Group-9/BMN-Boundary-Matching-Network

of actions being performed in those relevant portions. While impressive progress has been made on both fronts, we focus on the first and the more difficult task of temporal action localization. We aim to improve upon the state-of-the-art (Lin et. al. (2019)), which had utilized the Boundary-Matching Network (BMN) that evaluates confidence scores of densely distributed proposals then combines them into a single confidence map to generate proposals with precise boundaries.

The rest of this report is structured as follows: In Section 2, we detail a chronological literature review of how this topic has been studied before and after the BMN network was proposed and our motivation for ultimately selecting BMN as our baseline model. We present a structural overview of the model and its training objectives in Section 3 and illustrate our experiment setup, evaluation methods, and results of successfully replicating the baseline model. Finally, in Section 5, we propose several methods to improve upon the state-of-the-art that we plan to try out in the latter half of this project. These include incorporating a combination of local and global context, implementing temporal perturbations such as shifts to increase model robustness, and using an ensemble of models trained on original and reversed video features.

2 Literature Review

For the TAL task, an overwhelming majority of methods prior to BMN, including the Single-Stream Temporal Action Proposals (SST) [6] and Deep Action Proposals (DAPs) [7], focused on a "top-down" approach which generates proposals by sliding multi-scale temporal windows in regular intervals then evaluating confidence scores of the generated windows. Due to the very nature of sliding windows of fixed lengths, these top-down approaches were found to have neither temporally precise nor flexible boundaries.

More recent research adopts a "bottom-up" approach that tackles the issue by first locating the temporal boundaries using confidence scores at each frame in the video and then smartly combining them as candidate proposals and evaluating an overall confidence score for each candidate proposal. Boundary-Sensitive Network (BSN) proposed by Lin et. al. (2018) [4] was the first to adopt such a bottom-up approach in an effort to utilize temporally local features. In BSN, proposals are defined as probabilistic intervals within the untrimmed video that contain actions with the local probabilities of each point lending to candidate start points and end points for building out a proposal. The network then globally matches these candidate starting and ending points to generate proposals.

However, BSN proved to have several faults: (1) proposal feature construction and confidence evaluation procedures are conducted for each proposal resulting in computational inefficiency; (2) the proposal features constructed in BSN are too simple to capture sufficient temporal context; (3) it does not present a unified framework but is instead distributed across multiple stages. In order to overcome these drawbacks, Lin et. al. (2019) introduced the Boundary-Matching Network (BMN) [3] that evaluates confidence for all proposals simultaneously with richer temporal context. Within a BMN, each proposal is a matching pair of its starting and ending boundaries that are combined into a two-dimensional confidence map to represent the proposals with temporal duration and contiguous starting boundaries.

2.1 Baseline Selection

Very recently, Proposal Relation Networks (PRN), Wang et. al. (2021) [9] achieved state-of-the-art on the ActivityNet dataset by building upon BMN. This was accomplished in two steps: first, the authors introduced minor temporal perturbations in the features to make the model more robust to temporal noise, then they designed a self-attention mechanism to capture dependencies between action proposals and combined them to generate proposals with higher confidence.

Another recent research direction, adopted by Bagchi et. al. (2021) in AVFusion [12], is to incorporate features based on the audio modality to improve prediction power with minimal modifications to the base architecture handling video-based spatio-temporal features. While this method achieves state-of-the-art on a much smaller Thumos'14 dataset, PRN beats it by a margin for the larger and the more interesting ActivityNet-1.3 dataset.

Overall, PRN and AVFusion both largely borrow their base architecture and setup from BMN and contributed a few improvements through techniques including data augmentation and attention mechanisms. For this project, we are interested in understanding and building out the base model

architecture then designing our own simulation and ablation experiments. Thus, we selected the BMN as our baseline model.

3 Baseline Model

3.1 Feature Extraction

To extract the features of each video, we used a two-stream network proposed by Simonyan et. al. [5] to generate a fixed size vector for each interval. The two-stream network is popularly used [16; 17] in many video analysis models because of its architecture's relevance to spatial and temporal information present in videos. Specifically, the two-stream network uses two different networks to detect and construct features based on spatial and temporal information separately and then fuses the output of the two networks to produce features that encode this geo-temporal information.

When extracting the features, each video will have a corresponding feature embedding (F) of size $C \times T$ where C is the size of the embedding vector for each frame, which is 400, and T is the number of frames extracted from the video, which is 100. The two-stream network divides every video into 100 different frames that are spaced equally apart so that every video will have the same amount of frames being extracted. In this work, we load the features from the official repository to avoid the long computation time of the feature extraction process.

The training labels for the model consist of three different variables: the confidence score (G_C), the start match score (G_S), and the end match score (G_E). G_C is a $T \times T$ matrix where the rows represent the starting frame and the columns represent the ending frames. Therefore, the confidence score matrix could represent every possible interval within the video. The values of the matrix are computed using the Intersection over Union (IoU) between the corresponding interval and the interval from the ground-truth as shown in equation 1 where A is the indexed interval and B is the ground-truth interval. Similarly, the G_S and G_E are vectors of length T where each value represents the likelihood of that point being a start or end of an action respectively. To calculate the overlap, the start and end actions of the labels are converted from specific frames to intervals of frames such that overlaps could be calculated. The overlaps are consequently calculated using equation 2 where A is the indexed interval and B is the ground-truth interval.

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

$$Overlap(A, B) = \frac{|A \cap B|}{|A|} \tag{2}$$

3.2 Model Overview

As mentioned in Section 2.1, we follow Boundary Matching Network (BMN) as our baseline. The BMN model takes in the extracted features from the two-stream network as discussed in the section above and generates proposals with precise temporal boundaries. The model also generates confidence scores corresponding to each proposal that evaluates how probable each proposal is. Using these two outputs, we are able to rank and retrieve the most probable start and end times for a particular action.

The BMN model comprises of three separate modules, namely the Base Module, the Temporal Evaluation Module, and the Proposal Evaluation Module. As visualized in Figure 2, the Base Module first encodes the input feature sequence, and the output feature sequence is then used as inputs to both the Temporal Evaluation and Proposal Evaluation Module.

3.2.1 Base Module

The Base Module consists of two one-dimensional convolutional layers with ReLU activations. This module serves as the backbone feature extractor of the network. It outputs a shared feature sequence that is used for both the Temporal Evaluation Module and the Proposal Evaluation Module.

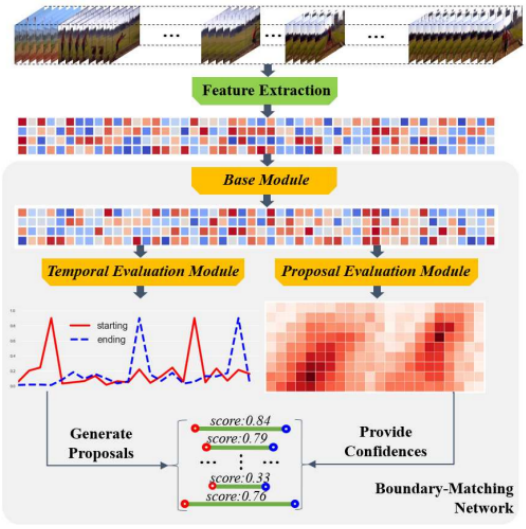

Figure 1: BMN-Framework Lin et. al. [3]

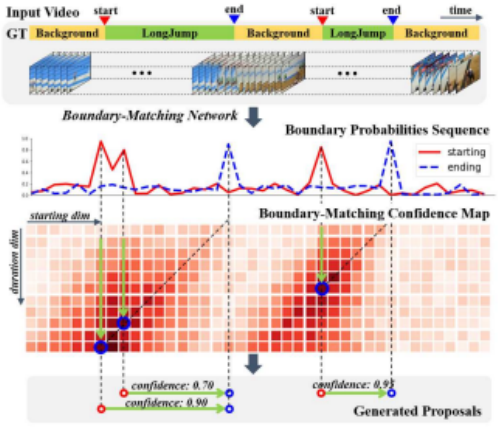

Figure 2: Proposal Generation Lin et. al. [3]

3.2.2 Temporal Evaluation Module

The purpose of the Temporal Evaluation Module (TEM) is to evaluate the probabilities that a particular temporal location is the start or end time of an action for all possible temporal locations in the video. This is done by passing the output feature sequence from the Base Module through two 1-Dimensional convolutional layers and then applying the sigmoid-activation function. This will result in a set of two probability sequences that quantifies how likely a particular temporal location is a start time or end time respectively.

These probability sequences will be used to generate possible proposals. As visualized above, we consider start and end times with high probabilities and match them to obtain a possible proposed duration for the action in the video. A Soft-Non-maximum Suppression (Soft-NMS) algorithm is also used to remove unlikely proposals to consider fewer proposals.

3.2.3 Proposal Evaluation Module

The Proposal Evaluation Module (PEM) generates a Boundary-Matching confidence map that provides confidence scores for all possible proposals.

First, the PEM applies the Boundary-Matching (BM) layer that transforms the 1-dimensional input feature sequence from the Base Module into a 2-dimensional feature map. This is done by uniformly

sampling across the temporal dimension in the input sequence. According to the paper, the generated feature map will contain rich features and temporal contexts for every proposal.

Next, the PEM applies a set of convolutional layers to further extract features and the sigmoid activation is applied to generate the BMN confidence scores. More specifically, two types of maps are generated M_{CC} and M_{CR} as they are required for the binary classification and regression loss function respectively during training. Using the confidence scores, we can rank the proposals generated from TEM as explained in the section above and retrieve the most probable start and end times that contain an action.

3.3 Training Objective

$$L_{Total} = L_{TEM} + \lambda_1 \cdot L_{PEM} + \lambda_2 \cdot L_2(\theta) \tag{3}$$

$$L_{TEM} = L_{bl}(P_S, G_S) + L_{bl}(P_E, G_E) \tag{4}$$

$$L_{PEM} = L_{bl}(M_{CC}, G_C) + L_R(M_{CR}, G_C) \tag{5}$$

The loss of the BMN model is formulated as a multi-task loss function and can be broken down into three portions: loss contributions from TEM, loss contributions from PEM, and regularization. The λ terms quantify the weightage of the contributions of each portion and set to 1 and 0.0001 for λ_1 and λ_2 respectively.

The regularization term is taken as the L2 regularization which takes the sum of the squared values of the model's parameters. TEM's loss sums the binary logistic loss (L_{bl}) for both the generated start probabilities (P_S) and end probabilities (P_E), where the binary logistic loss is defined in equation 6, where l_w is the window length and $b_i = sign(g_i - 0.5)$. g_i is the confidence score assigned to a particular proposal at time i. Moreover $l^+ = \sum b_i$ and $l^- = l_w - l^+$. Consequently, $\alpha^+ = \frac{l_w}{l_+}$ and $\alpha^- = \frac{l_w}{l^-}$. Lastly, p_i is defined as the probability score assigned at time i.

$$L_{bl}(P, G) = \frac{1}{l_w} \sum_{i=1}^{l_w} (\alpha^+ \cdot b_i \cdot log(p_i) + \alpha^- \cdot (1 - b_i) \cdot log(1 - p_i)) \tag{6}$$

Lastly, PEM's loss is defined as the sum of a binary logistic loss and a regression loss (mean squared error). The binary logistic loss is taken between M_{CC} and G_C, while the regression loss is taken between M_{CR} and G_C.

4 Feature Enhancements

In the baseline model, we noticed severe overfitting on the two-stream dataset after approximately 8 epochs. This is likely because the two-stream network represents each video as a $C \times T = 400 \times 100$ length tensor, regardless of the length of the video. We hypothesize that this representation of each video using 40,000 numbers might contribute to loss of information and might be the reason for quick overfitting. However, for the scope of this work, we did not have the capacity to process raw videos from scratch. Consequently, our contribution in this paper mostly pertains to proposing methods to combat overfitting using the two-stream representations of ActivityNet videos.

4.1 Temporal Shifts

In this data augmentation method, we propose to shift some randomly selected features along the temporal dimension by several time steps as seen in figure **??**. In this way, we can exchange information with video frames in neighbouring timesteps. Temporal shifts have been shown (TSM [15], Lin et. al. 2019, and PRN [9], Wang et. al. 2021) to be an effective way to improve performance in video recognition tasks without additional compute. Effectively, we are applying a data augmentation method to make the trained model more generalizable. Due to the offline nature of the video localization task and the BMN model, bidirectional shifts were used. There are two hyperparameters to tune when shifting: the percentage of features we want to shift, and the maximum amount by which we want to shift a feature. We experimented with moving 10, 20, and 30% randomly selected features left or right by a randomly selected integer step size between 1 to 10%, and found shift probability of 20% and max shift of 5% to be the best hyperparameters over this search space.

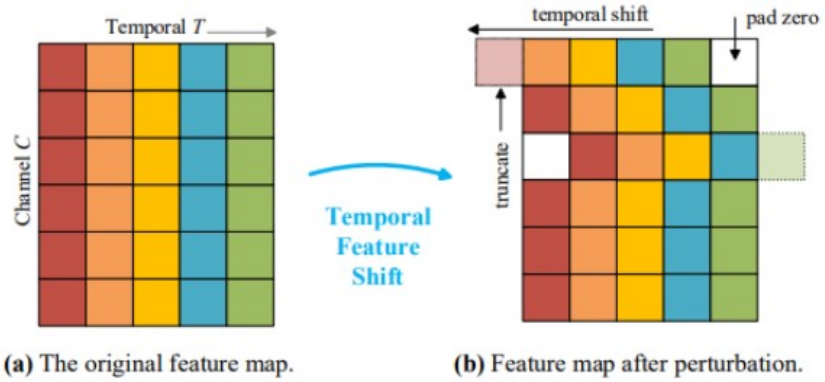

Figure 3: Temporal Shift Module, Wang et. al. [9] and Lin et. al. [15]

In Section 6.3, Ablation Studies, we use these hyperparameters to compare temporal shifts to other modules.

4.2 Global Information

For our second method, we propose a set of methods to augment our feature vectors obtained from the Two-Stream Network with global information. By concatenating information about the global context (the entire video clip), we hypothesize that the features at every time-step will have increased information, thus allowing the model to learn more and make better predictions.

4.2.1 Global Mean

Firstly, a naive approach was tested by simply averaging values of the features across all time-steps

$$f_{global} = \frac{1}{T} * \sum_{t=1}^{T} F_t \tag{7}$$

where the f_{global} is a $C \times 1$ vector representing the global mean feature for the video and F_t is a $C \times 1$ vector representing the feature at time t. Next, we augment the initial feature vectors by concatenating f_{global} with the original features at every time-step t. In the end, we have $F_{augmented}$ as a $C * 2 \times T$ matrix where the feature vector at each time-step contains both the original F_t and f_{global}. $F_{augmented}$ is then used as the training input.

4.2.2 Global Features from Learnable weights

Instead of simply taking the average values of features as the global information, a weight matrix (W_{global}) is used to produce a linear projection of the global information. W_{global} will be added to the model as a learnable parameter so that through training, W_{global} learns to extract more relevant global information from the features.

$$f_{global} = F_t \cdot W_{global} \tag{8}$$

4.2.3 Squeeze and Excitation

In the final iteration of utilizing global information, we took inspiration from research by Hu, Jie, Li Shen, and Gang Sun on Squeeze and Excitation networks [18]. The research proposes using a Squeeze and Excitation module to explicitly model channel-wise inter-dependencies across spatial dimensions for input images. This is then used to scale the channel values accordingly. Adapting to our application, we use a similar Squeeze and Excitation module to model inter-dependencies across the 100 time-steps.

$$f_{mean} = \frac{1}{T} * \sum_{t=1}^{T} F_t \tag{9}$$

$$s = \sigma(W_2 \cdot \delta(W_1 \cdot f_{mean})) \tag{10}$$

$$F_{weighted} = F * s \tag{11}$$

As seen from the equations above, first, we average each feature value over all time-steps which mimics the squeeze operation in the original research. This serves to describe how expressive a particular feature is throughout the entire video. To obtain the feature-wise inter-dependencies, we perform two linear transformations with rectified-linear[19] and sigmoid non-linearities respectively (excitation operation). This provides us with the scaling factors for every feature and this is used to scale the original video features accordingly as shown in the final equation.

4.3 Reverse Ensemble Method

Another proposed method to improve the model is to train two BMN models instead of one. The main difference between the two trained models is that the first model would be trained on the normal features just like the baseline model while the second model is trained on the reversed features and labels. This means that the reversed model would be fed the video features in reversed order while the labels of the actions in the reversed videos are also reversed. To achieve this, the $C \times T$ features of each video would be reversed on the first axis while the ground-truth label segments of each video are also reversed before creating the ground-truth maps (G_C, G_S, G_E) for the loss functions. This is done so that the actions in the labels would correspond to the actions in the videos.

After training the two models, we ensemble the results of the two models together during inference as seen in figure 4. To achieve this, the corresponding reversed model output (M_{CC}, M_{CR}, P_S, P_E) would be reversed back to forward time. After that, the two model outputs would be ensembled before inference is performed. By combining the model trained on the reversed features, the output should theoretically be more receptive to frames before and after the actions similar to the Bi-Directional LSTM model.

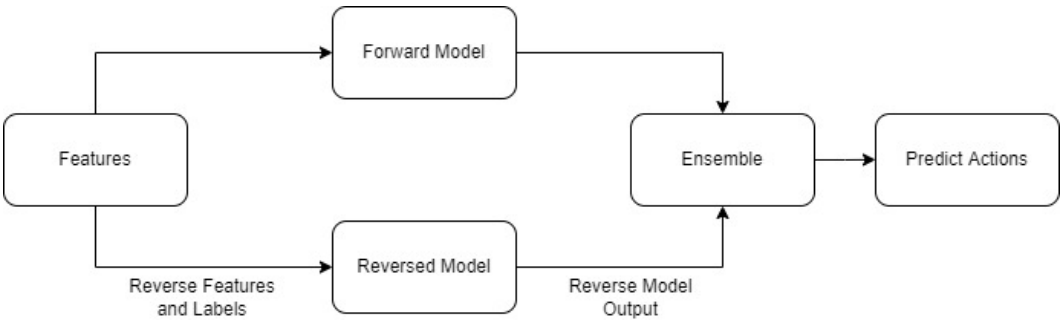

Figure 4: Reverse Ensemble Pipeline

5 Dataset

We trained our implementation of the BMN model on the ActivityNet-1.3 dataset [1] which contains videos encompassing a large range of human activity. It features 200 different types of simple human activities compiled to approximately 849 hours worth of videos collected from YouTube. The dataset's annotations are crowd-sourced via the Amazon Mechanical Turk and classify the types of actions performed in the video and the start and end timestamps of each action. Specifically, version 1.3 contains a total of 19,994 untrimmed videos and is divided into three separate subsets: training, validation, and testing data, by a ratio of 2:1:1. Each video includes an average of 1.41 activities annotated with temporal boundaries and the test videos' ground-truth annotations are not made public. Since ActivityNet is the most popular benchmark in the temporal activity detection and localization domains and used in many of the previous related works cited above, it serves as a good way to compare our performances with the existing state-of-the-art. We used pre-extracted features of each video based on the two-stream network [5] and did not apply any further preprocessing to the dataset other than data augmentation techniques during our experiments.

6 Experiments

6.1 Evaluation Metrics

To evaluate the proposals generated by the model, we calculated the Average Recall (AR) over different IoU thresholds ranging from 0.5 to 1.0 with 0.05 increments. This would mean that any labeled proposal with an IoU score that is more than the threshold would be labeled as correctly labeled and vice versa. Next, the AR is then plotted over different Average Number of proposals (AN) such that we could obtain the area under curve (AUC) as our primary evaluation metric. In our case, AN is a range between 1 and 100 proposals taken from the model's output. Therefore, we would compute the AR across the different AN ranges, and to call an AR at different AN values, we would use the notation AR@AN.

6.2 Experimental Setup

We used two distributed Nvidia Titan Xp GPUs with 12GB memory each for training our models. For ablation studies, each method mentioned in feature enhancements was incorporated in isolation from others, and all hyperparameters not specific to these modules were kept constant. These hyperparameter settings were arrived through finetuning during experiment runs prior to conducting ablations. General hyperparameter settings include the use of Adam optimizer with initial learning rate of 1e-3, and weight decay 1e-4 and a ReduceLROnPlateau scheduler with patience 2 and decay factor 0.5. Module-specific hyperparameters include: (1) Shift probability of 0.2 and maximum temporal shift of 5 in either direction for the Temporal Shift module; (2) Hidden dimension of size 100 for the Squeeze and Excite module; and (3) Same hyperparameter settings as foward counterparts for reverse ensembling. We'd also like to note that while the baseline didn't have dropout layers, we added dropout layers with dropout probability 0.4 to both the TEM and PEM modules of the model consistently in all our ablation studies.

6.3 Results

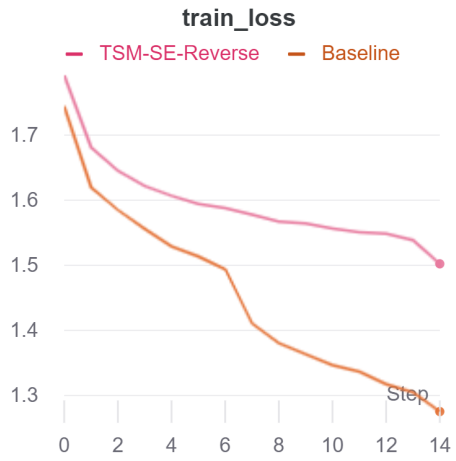

Figure 5: Total Train Loss

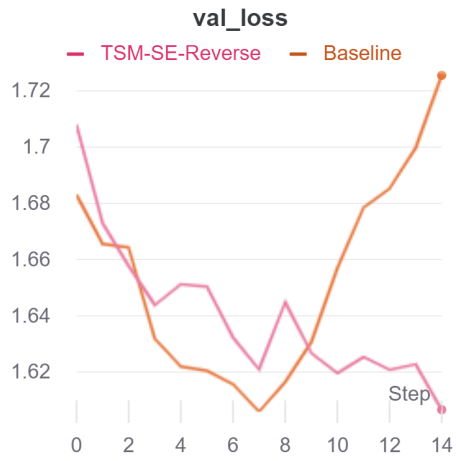

Figure 6: Total Val Loss

	BMN Results [3]	Our Baseline Reproduction	TSM-SE-Ensemble
AR@1	33.6%	33.5 %	33.4%
AR@5	49.9%	48.0 %	49.8%
AR@10	57.1%	55.1 %	57.3%
AR@100	75.5%	75.1 %	75.4%
AUC	67.5%	66.6 %	67.5%

Table 1: Average Recall and AUC Results

From our results shown in Table 1, we are able to achieve an overall AUC of 67.5% using a combination of temporal shifted data features, squeeze and excitation module and ensembling with a reversed version of the model. This result represents a 0.9% increase in AUC compared to our reproduction of the baseline model.

Looking at Figure 5, we observe that the baseline implementation experiences strong signs of overfitting starting from epoch 7 because the baseline validation loss start to inflect strongly upwards from that epoch on-wards. This is likely due to the lack of expressivity of the feature vectors extracted by the two-stream network. Furthermore, when we compare the validation loss of the baseline with our improved version, we can see that our model still manages to decrease both training and validation loss till the last epoch. As such, we note that a combination of using temporal shifted data features, squeeze and excitation module and ensembling with a reversed version of the model enables the model to better handle the issue of overfitting and outperform the baseline.

6.4 Ablation Studies

Modules				Metrics	
TSM	Squeeze n Excite	Reverse	Ensemble	AUC	AR@100 Recall
Our Reproduction of Baseline				*66.6%*	*75.1%*
x				67.2%	75.2%
	x			67.4%	75.3%
		x		66.6%	74.8%
			x	66.9%	75.0%
x			x	67.2%	75.1%
x	x			**67.5%**	75.3%
x	x		x	**67.5%**	**75.4%**

Table 2: Ablation Studies

To understand the contribution of each component in the model, we conducted a set of ablation studies, with results on the ActivityNet validation dataset displayed in Table 2. The combination of TSM, Squeeze and Excite, and our reverse ensemble model contributes the most significant increase of 0.3 % in AR @ 100 Recall and 0.9% in AUC. The second best performance was achieved by TSM and Squeeze and Excite with AR @ 100 Recall of 0.2 % over the Baseline and 0.6 % in AUC. It's interesting to note that the inclusion of Squeeze and Excite contributes the most for both metrics. While the ensemble of the baseline model and our reversely mapped model did not outperform the baseline on its own, when combined with TSM and Squeeze and Excite, it was able to accomplish the best predictive performance.

6.5 Error Analysis

To elucidate the different causes of misclassification, we conducted error analyses across video samples of varying IoU scores below 0.5 (examples with IoU equal to or greater than 0.5 were categorized to be correct) and categorized several potential sources of errors based on the sample's reported accuracy. The predominant cause of error for video examples with a lower IoU performance of [0, 0.3] was the ground truth's annotations, namely its impreciseness in action labelling. One common case was where the ground truth label failed to distinguish the foreground from the background context; the said action did not directly occur in the video but the background did include objects or images that were related to a certain action. For instance, in the "True Shingle Warranty Comparison, CHATEAU ROOFING" video, the action is labeled as "roof shingle removal" when the speaker merely verbally advertises his own roof shingle removal services in front of a static stock image of a roof in the background throughout the entire 90 seconds. In this example, the ground truth failed to tell the agent's action apart from the visual cues provided in the background, which naturally confused the model to also incorrectly identify a nonexistent "action."

Other detected cases of ground truth mislabelling include situations when the range do not contain the labeled action, but rather the state before the said action occurs. In "Attack of the Curlers", the removal process of curlers is not directly shown, but rather the before and after of removing are

compared and the labeled range includes the "before" state. We also found multiple examples where the action is simply an inaccurate and abstracted summary of what actually happens in the video. In "Waxing Furniture", the ground truth states that the identified action is "painting furniture," when in reality, the labeled time range shows a person demonstrating how one can wax-paint a painted cabinet. We hypothesize that such inconsistency in the training label precision and sequence detection, as they were left uncontrolled for, inevitably impacted the model's learning process.

For video samples that resulted in a higher IoU performance of [0.3, 0.5], model errors primarily originated from a single action spanning the entire length of the video or appearing in multiple moments throughout the video. In video "Jan Zelezny - 95.34m and 95.66m – Sheffield 29/08/1993", the video is merely 7 seconds long and the action of "javelin throw" spans the entire length, for which the proposal correctly suggests the entire length. In videos "My 3 year old's first make up tutorial" and "Lifting Challenge" , the ground truth labels action of " putting on makeup" and "snatch" at five and three separate moments respectively. This again resulted in teaching the model to propose the entire chunks of the videos in majority of its top 4950 proposals and miss the exact ranges by more than 10 seconds in its shorter proposals. We hypothesize that the first case hindered the model's ability to accurately pinpoint the exact timestamp of the action's occurrence, while the second, despite adding variance and richness to training data, did not facilitate the model's learning in appropriately narrowing down its predicted ranges to correctly capture each and every fragment.

7 Discussion and Future Work

A major limitation that we faced is the the lack of expressivity of the feature vectors generated from the two-stream network. This severely restricted our ability to experiment with the methods that increase model complexity due to the problem of overfitting. Future work can experiment with alternative feature extractors as MMAction2 by OpenMMLab [21]. Furthermore, future work could also leverage the multi-modalities of the video dataset and perform multi-modal learning to augment the visual features with audio information[12]. This would allow for more model complexity and room to experiment.

While our work successfully achieved a superior performance over the baseline, future experiments can build upon our exploration of data augmentation methods to further improve generalization. The foreground-background confusion was the main source of errors for video samples with lower IoU's where the model failed to discriminate between videos that feature a true action and those do not. This phenomenon can potentially be mitigated through following three methods: 1) adding "pseudo video samples" with just static backgrounds and labelling them as "no action" 2) calculating the amount of motion involved in each sample over time to add as a feature and let the model learn to disregard static images (by incorporating sequence-based models such as attention mechanisms[20]). or 3) detecting static backgrounds and replacing them with randomized backgrounds to prevent the model from learning a "pattern" in the background as a clue for the action label.

8 Conclusion

In this paper, we identified the severe overfitting challenge present in the state-of-the-art BMN model caused by the feature vectors extracted by the two-stream network. Furthermore, we sought to improve its generalizability through three different techniques including Temporal Shift augmentation method, Squeeze and Excite module, and ensembling with another model trained on time-reversed video features and labels. Through our extensive experiments, we demonstrate that a combination of these three techniques largely curb the issue of overfitting. Although severely limited by the expressivity of the feature vectors, we managed to achieve an AUC score of 67.5%, resulting in a 0.9 point increase compared to our reproduction of the baseline.

9 Acknowledgements

This work was advised by our mentor, Kai Hu, and our assigned Teaching Assistant, Zilin Si. We truly appreciate their guidance on devising new methods to train, improve, and debug our model and their overall continued support throughout the semester.

We'd also like to extend our gratitude to JJBoy for their licensed open-source implementation (available here) of the baseline BMN paper which we referenced to reproduce the baseline model, prior to adding our experiments and modules for improvement.

References

[1] Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J. (2015). Activitynet: A large-scale video benchmark for human activity understanding. In Proceedings of the ieee conference on computer vision and pattern recognition (pp. 961-970).

[2] Wang, L., Qiao, Y., Tang, X. (2014). Action recognition and detection by combining motion and appearance features. THUMOS14 Action Recognition Challenge

[3] Lin, T., Liu, X., Li, X., Ding, E., Wen, S. (2019). Bmn: Boundary-matching network for temporal action proposal generation. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 3889-3898).

[4] Lin, T., Zhao, X., Su, H., Wang, C., Yang, M. (2018). Bsn: Boundary sensitive network for temporal action proposal generation. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 3-19).

[5] Simonyan, K., Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. arXiv preprint arXiv:1406.2199.

[6] Buch, S., Escorcia, V., Shen, C., Ghanem, B., Carlos Niebles, J. (2017). Sst: Single-stream temporal action proposals. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (pp. 2911-2920).

[7] Escorcia, V., Heilbron, F. C., Niebles, J. C., Ghanem, B. (2016, October). Daps: Deep action proposals for action understanding. In European Conference on Computer Vision (pp. 768-784). Springer, Cham.

[8] Zeng, R., Huang, W., Tan, M., Rong, Y., Zhao, P., Huang, J., Gan, C. (2019). Graph convolutional networks for temporal action localization. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 7094-7103).

[9] Wang, X., Qing, Z., Huang, Z., Feng, Y., Zhang, S., Jiang, J., ... Sang, N. (2021). Proposal Relation Network for Temporal Action Detection. arXiv preprint arXiv:2106.11812.

[10] Zhao, C., Thabet, A. K., Ghanem, B. (2021). Video Self-Stitching Graph Network for Temporal Action Localization. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 13658-13667).

[11] Wang, X., Zhang, S., Qing, Z., Shao, Y., Gao, C., Sang, N. (2021). Self-supervised learning for semi-supervised temporal action proposal. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1905-1914).

[12] Bagchi, A., Mahmood, J., Fernandes, D., Sarvadevabhatla, R. K. (2021). Hear Me Out: Fusional Approaches for Audio Augmented Temporal Action Localization. arXiv preprint arXiv:2106.14118.

[13] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

[14] Tran, D., Ray, J., Shou, Z., Chang, S. F., Paluri, M. (2017). Convnet architecture search for spatiotemporal feature learning. arXiv preprint arXiv:1708.05038.

[15] Lin, J., Gan, C. and Han, S., 2019. Tsm: Temporal shift module for efficient video understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 7083-7093).

[16] J. Gao, Z. Yang, and R. Nevatia. Cascaded boundary regression for temporal action detection. In Proceedings of the British Machine Vision Conference, 2017. 3

[17] T. Lin, X. Zhao, and Z. Shou. Single shot temporal action detection. In Proceedings of the 2017 ACM on Multimedia Conference, pages 988–996. ACM, 2017. 2, 3

[18] Hu, J., Shen, L., Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7132-7141).

[19] Agarap, A. F. (2018). Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375.

[20] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

[21] Contributors, MMAction. "Openmmlab's next generation video understanding toolbox and benchmark." (2020).