

IET Image Processing

Special issue Call for Papers

**Be Seen. Be Cited.
Submit your work to a new
IET special issue**

Connect with researchers and
experts in your field and share
knowledge.

Be part of the latest research
trends, faster.



Read more



The Institution of
Engineering and Technology

ORIGINAL RESEARCH

Class-wise boundary regression by uncertainty in temporal action detection

Yunze Chen^{1,2}  | Mengjuan Chen¹ | Qingyi Gu¹ 

¹Center of Precision Sensing and Control, Institute of Automation, Chinese Academy of Sciences, Beijing, China

²School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

Correspondence

Qingyi Gu, Center of Precision Sensing and Control, Institute of Automation, Chinese Academy of Sciences, Beijing, China.

Email: qingyi.gu@ia.ac.cn

Funding information

Scientific Instrument Developing Project of the Chinese Academy of Sciences, Grant/Award Number: YJKYYQ20200045

Abstract

Temporal action detection is a crucial aspect of video understanding. It aims to classify the action as well as locate the start and end boundaries of the action in the untrimmed videos. As deep learning is frequently utilized, the accuracy of annotation is crucial to boundary localization. However, it is observed that some annotation instances are ambiguous and the ambiguity varies between categories. To solve the problem above, a Gaussian model is built to estimate the boundary uncertainty for each instance. Based on instance uncertainty, category uncertainty is applied to describe the uncertainty of each category. By combining instance and category uncertainty, the boundaries of the selected proposals are refined and the ranking of candidate proposals is adjusted. Furthermore, overcorrection is avoided for categories with a high level of uncertainty. With the uncertainty approach, state-of-the-art performance is achieved: 57.5% on THUMOS14 (mAP@0.5) and 35.4% on ActivityNet (mAP@Avg).

1 | INTRODUCTION

With the progress of technology, an ever-increasing number of videos are saved and made available for a variety of daily activities. As a fundamental aspect of video understanding, temporal action detection can be applied to a variety of fields, including video content analysis and video recommendation. This task is not only to determine which actions are depicted in the video, but also to find their temporal boundaries, that is, their start and end times. Because most temporal action detection algorithms rely significantly on deep learning, the quality of the predicted proposals is dependent on the precision of the boundary annotations. To limit the impact of unclear annotations, the ambiguity of annotation boundaries are represented by instance uncertainty in [1].

Based on instance uncertainty, we observe that the difficulty of locating the boundaries of different categories varies. The labeling of some action types is ambiguous. These ambiguities include ambiguity in action definitions, ambiguity in action boundaries, and ambiguity in video content, as shown in Figure 1. This labeling uncertainty leads to prediction bias, which is difficult to correct.

To describe the uncertainty of various categories, we create a metric called category uncertainty. With these criteria, we aim to avoid over-refinement of proposals that fall into high-uncertainty categories. Specifically, we model the uncertainty of each instance in the training process. To describe the uncertainty of various categories, we create a metric called category uncertainty. Throughout the testing phase, we apply the reliability score module to select reliable proposals based on the instance uncertainty score. By combining the instance uncertainty and the category uncertainty, we refine the scores as well as the boundary of the predicted proposal simultaneously in uncertainty voting. Compared to [1], we introduce category uncertainty to describe the uncertainty of each category. Based on this metric, we propose the uncertainty voting module. In this module, we refine the boundaries and adjust the ranking of candidate proposals. In addition to the one-stage and two-stage methods, we deploy our methodology in the anchor-free method AFSD [2], and achieve state-of-the-art performance. Our contribution can be summarized as:

1. We introduce category uncertainty to describe the annotation accuracy of each category.

This is an open access article under the terms of the [Creative Commons Attribution-NonCommercial-NoDerivs](https://creativecommons.org/licenses/by-nc-nd/4.0/) License, which permits use and distribution in any medium, provided the original work is properly cited, the use is non-commercial and no modifications or adaptations are made.

© 2022 The Authors. *IET Image Processing* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology.



FIGURE 1 (a) Ambiguity of action definitions. The upper part of (a) is the complete Hammer Throw. The bottom half of (a) is a partial replay of the complete action. However, the partial replay is still labelled as a complete Hammer Throw. (b) Ambiguity in the action's boundaries. There is only a fraction of a second between the begin and end of Billiards. As a result, determining the end boundary of Billiards is difficult. For labeling convenience, the white ball touched by the club, the white ball touched by the target ball (yellow ball in (b)), and the target ball into the hole may all be labeled as the end boundary for labeling purposes. (c) Ambiguity in video content. In this instance, the start and end boundaries of volleyball spiking are accurately labeled. However, there are other actions of other people in this video, making it difficult for the model to learn volleyball spiking

2. According to category uncertainty, the uncertainty voting module is designed to refine proposals belonging to categories with low uncertainty.
3. We deploy our uncertainty method on a one-stage, two-stage and anchor-free framework to demonstrate the effectiveness of our method. On the anchor-free method AFSD [2], we achieve state-of-the-art performance. The performance is 57.5% on THUOMS14 (mAP@0.5) and 35.4% on ActivityNet (mAP@Avg).

The remainder of this work is arranged in the following manner. Section 2 provides background on temporal action localization and uncertainty learning in deep neural networks. Section 3 describes the proposed temporal action detection framework. In Section 4, the proposed method is validated by experimental results and compared to earlier research. And Section 5 concludes the paper and provides an overview of future directions.

2 | RELATED WORKS

2.1 | Temporal action localization

Recently, great progress has been made in deep learning, which facilitates the development of temporal action localization. The

methodology for this task can be divided into two categories: single-stage and two-stage. The one-stage approach combines proposal generation and classification into a single end-to-end framework for high efficiency. Inspired by SSD [3], [4] designs a one-dimensional temporal convolution to generate multiple temporal action anchors for proposal generation. Furthermore, SSTAD [5] employs a recurrent neural network (RNN) architecture to perform proposal generation and classification simultaneously. To provide valid proposal boundaries and classification results, Decouple-SSAD [6] employs two branches: one for regression and the other for classification. GTAN [7] introduces a Gaussian kernel to dynamically optimize the time scale of each action proposal.

On the other hand, the two-stage strategy first generates action proposals and then classifies them to achieve high performance. By incorporating 3D regional convolutional networks (C3D [8]), R-C3D [9] optimizes localization and classification loss in an end-to-end framework. Inspired by the end-to-end framework Faster R-CNN [10], TAL-Net [11] adds context information into proposals. And to make better use of contextual information, PGCN [12] considers the proposal-proposal interactions as well as leverages proposal relationships. Furthermore, AFSD [2] presents an efficient and effective anchor-free temporal localization strategy that yields state-of-the-art results. However, each of these methods suffers from boundary ambiguity, resulting in unreliable proposals or

inaccurate boundaries. To reduce the impacts of unreliability boundary, we introduce uncertainty metrics to describe the accuracy of annotations.

2.2 | Uncertainty learning in deep neural networks (DNNs)

To improve the robustness and interpretability of discriminative deep neural networks (DNNs), researchers are increasingly turning to the uncertainty approach. There are two types of uncertainty approaches: model uncertainty and data uncertainty. The model uncertainty captures the noise in the deep neural network's parameters, which can be minimized by increasing the training data amount. Data uncertainty, on the other hand, captures the noise in the given training data. As a result, data uncertainty does not alter as the amount of training data grows. This study is based on the uncertainty of the data. Previous methods primarily focused on the image task's uncertainty. For example, ambiguity in bounding box labeling in object detection [13], and ambiguity caused by blurry pictures in face recognition [14]. In the action localization task, the ambiguity of boundary labeling, as well as the ambiguity of action definition, are both present. Furthermore, the level of uncertainty differs for different categories of actions. Therefore, we build a Gaussian model to estimate the boundary uncertainty for each instance. Based on instance uncertainty, we apply category uncertainty to describe the uncertainty of each category. By combining instance and category uncertainty, we refine the boundaries of the selected proposals and adjust the ranking of candidate proposals.

3 | METHOD

3.1 | Baseline architecture

3.1.1 | Base feature extraction

Temporal action detection aims to discover precise temporal boundaries and classes of action instances in the untrimmed video. Due to the limitation of computational resources, it is impossible to feed the untrimmed video straight into the visual coder for feature extraction (untrimmed videos are often very long, up to several minutes). A typical strategy is to divide the video into different segments at equally sized intervals. Formally, the input video with frame l can be separated into l_s segments by dividing into time intervals σ .

$$S = \{s_n\}_{n=1}^{l_s}, \quad l_s = \frac{l}{\sigma}. \quad (1)$$

Then, each segment is input into a pre-trained visual coding system, such as two-stream [15] or I3D [16] to extract spatial and temporal feature vectors, respectively. Finally, these two features are linked together for further processing (Figure 2).

3.1.2 | Proposal generation

Equipped with the extracted features, we forecast the location of the instance and the action score using the proposal and classification branches, respectively. For the proposal branch, it's usual to build a set of anchors in different scales to determine the default proposal location (p_c, p_w) , where p_c and p_w are the default center and width, respectively. To locate the temporal boundaries of action instances accurately, each proposal outputs two predictions by 1D convolution: 1) Regression parameters of the proposal $(\Delta p_c, \Delta p_w)$, indicating the offset of the default temporal center and width. 2) The overlap score p_{ov} , which indicates the intersection-over-union (IoU) score between the proposal and its closest ground-truth segment. Finally, the regressed boundaries are:

$$\begin{aligned} x_s &= p_c + \beta_1 p_w \Delta p_c - \frac{1}{2} p_w e^{\beta_2 \Delta p_w} \\ x_e &= p_c + \beta_1 p_w \Delta p_c + \frac{1}{2} p_w e^{\beta_2 \Delta p_w} \end{aligned} \quad (2)$$

where x_s and x_e denote the start and end position of the action, respectively. β_1 and β_2 are hyper-parameters.

The classification branch, on the other hand, is responsible for determining which category the action instance belongs to. For each instance in proposal branch, the classification branch outputs the classification score $\mathbf{p} = [p_0, p_1, \dots, p_C]$, indicating the probability that the action instance belongs to C categories and one background.

3.1.3 | Post-processing

After predicting the location and action score, each proposal can be represented as:

$$\begin{aligned} \psi &= \{\mathcal{B}, S\} \\ \mathcal{B} &= \{x_s, x_e\} \\ S &= p_{ov} \times \arg\max \left(\{p_r\}_{r=1}^C \right). \end{aligned} \quad (3)$$

\mathcal{B} is the detection boundary of the proposal, where x_s and x_e represent the predicted start and end boundary of action, respectively. S represents the final score of the proposal, which is obtained by multiplying the overlap score p_{ov} and the maximum classification score p_r . For these proposals, we perform NMS [17] to remove any redundant predictions (in Algorithm 1).

3.2 | Standard multi-class classification and boundary regression

As mentioned above, the baseline architecture generates three attributes for each proposal: (1) Start and end boundaries of action instances; (2) IoU between the proposal and the ground truth that is closest to it. (3) Scores of the proposal belong-

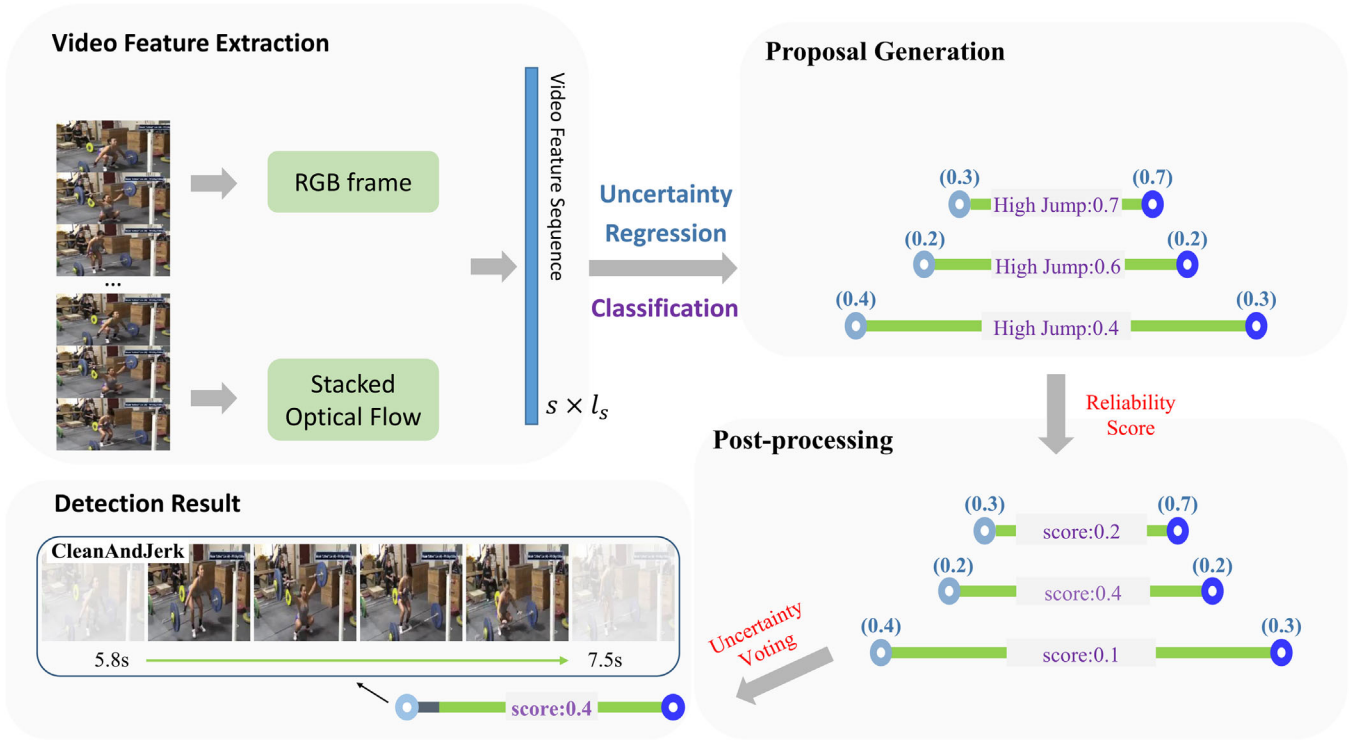


FIGURE 2 Given an untrimmed video, we encode the RGB frames and optical flow into the video feature sequence. With the feature sequence, we acquire a series of candidate proposals using uncertainty regression and classification branch, respectively. The candidate proposals include classification score and uncertainty score of the boundary. In post-processing, the reliability score combines the uncertainty score of the boundary with the classification score to choose reliable proposals. Furthermore, uncertainty voting sorts the candidate proposals and revises the boundaries of the selected proposals by incorporating category uncertainty

ALGORITHM 1 Uncertainty Post-processing

B is $N \times 2$ matrix of initial proposal boundaries. S contains corresponding final proposal scores. U denotes the boundary uncertainty variance.

$B = \{b_1, \dots, b_N\}$, $C = \{U_1, \dots, U_N\}$

$b = (x_s, x_e)$, $U = (\sigma_s^2, \sigma_e^2)$

$D \leftarrow \{\}$

$\mathcal{T} \leftarrow B$

Estimate S by Equation (13)

while $\mathcal{T} \neq \text{empty}$ **do**

$m \leftarrow \arg\max S$

$\mathcal{T} \leftarrow \mathcal{T} - b_m$

Refine S by Equation (14)

Refine b_m by Equation (15)

$D \leftarrow D \cup b_m$

end while

return D, S

ing to C categories. Therefore, we supervise each of these three attributes through location regression loss, overlap loss, and classification loss. Specifically, we use the Smooth L1 loss (S_{L1}) [18] in location regression to shift the proposal (x_s, x_e)

closer the ground truth (g_s, g_e) that is closest to it, which is determined as:

$$L_{loc} = S_{L1}(x_s - g_s) + S_{L1}(x_e - g_e). \quad (4)$$

And the overlap loss is calculated by Mean Square Error (MSE) loss:

$$L_{ov} = (p_{ov} - g_{ion})^2, \quad (5)$$

where g_{ion} is the IoU value between the proposal and the ground truth closest to it. For classification, we use the standard softmax loss:

$$L_{cls} = - \sum_{i=0}^C I_{i=t} \log(P_i) \text{ and } P_i = \frac{\exp(p_i)}{\sum_{j=0}^C \exp(p_j)}, \quad (6)$$

where $I_{i=t}$ is an indicator function that equals 1 if i is the ground truth class label t , otherwise 0.

Finally, we accumulate the losses of the three attributes. The overall training objective is defined as follows:

$$\mathcal{L} = \mathcal{L}_{cls} + \beta \mathcal{L}_{loc} + \gamma \mathcal{L}_{ov}, \quad (7)$$

where β and γ are the hyperparameters that need to be adjusted.

3.3 | Uncertainty modeling

However, inaccurate regression boundaries and unreliability of final scores may be caused by the ambiguity of action instance labeling. To mitigate the impact of this ambiguity, we estimate the uncertainty of the boundary in addition to the standard regression loss (uncertainty-aware boundary regression). This uncertainty indicator is further used to adjust the final score of the proposal (Reliability Score) and the regression boundary (Uncertainty Voting).

3.3.1 | Uncertainty-aware boundary regression

As mentioned above, we build a Gaussian model to reduce the impact of labeling ambiguity. This model locates the action instances' locations while simultaneously estimating the regression boundary's uncertainty:

$$P(g^j | x^j) = \frac{1}{\sqrt{2\pi}\sigma^i} e^{-\frac{(g^j - x^j)^2}{2(\sigma^i)^2}}, \quad (8)$$

where σ denotes the boundary uncertainty variance and g^j is the corresponding ground truth of the location prediction x^j . Equipped with this model, The learnable parameters $\hat{\Theta}$ that maximize the probability $P(y^j | x^j, \Theta)$ over N samples are then estimated.

$$\hat{\Theta} = \arg \max_{\Theta} \prod_{i=1}^N \frac{1}{\sqrt{2\pi}\sigma^i} e^{-\frac{(g^i - x^i)^2}{2(\sigma^i)^2}}. \quad (9)$$

The logarithm of $\hat{\Theta}$ is:

$$\arg \max_{\Theta} \left\{ -\frac{N}{2} \ln 2\pi - \frac{N}{2} \ln \left((\sigma^i)^2 \right) - \frac{1}{2(\sigma^i)^2} \sum_{i=1}^N (x^i - y^i)^2 \right\}. \quad (10)$$

The Gaussian likelihood estimation can be treated as a regression loss by omitting the correlation term (i.e. $\frac{N}{2} \ln 2\pi$) that does not depend on the estimated parameter Θ :

$$L'_{loc} \propto \frac{(g - x)^2}{2\sigma^2} + \frac{1}{2} \ln (\sigma^2). \quad (11)$$

For some tiny values of σ^2 , we notice that L'_{loc} can be negative. Because the duration of actions fluctuates greatly, this condition occurs from time to time, resulting in training errors. As a result, to avoid a negative loss, we adjust the regression loss as follows:

$$L'_{loc} = \frac{(g - x)^2}{2\sigma^2} + \frac{1}{2} \ln (\sigma^2 + 1). \quad (12)$$

3.4 | Uncertainty post-processing

In post-processing, the traditional localization process multiplies the classification and overlap scores as a detection criterion score for non-maximum suppress (NMS). However, unreliable proposals with high classification or overlap scores may be selected. To limit the impact of unreliable proposals, we employ the uncertainty score in conjunction with classification and overlap scores to pick more reliable proposals (i.e. Reliability score) before the NMS. And in the NMS process, the proposal score, as well as the related boundaries are regressed in uncertainty voting. In this regression approach, It is worth noting that we incorporate category uncertainty to avoid over-refinement of categories with high uncertainty. And the details are described below.

3.4.1 | Reliability score

We employ (σ_s, σ_e) in conjunction with classification and overlap scores to pick more reliable proposals. The modified proposal score is:

$$S = p_{ov} \times \max \left(\{p_r\}_{r=1}^C \right) \times S_{\text{Reliability}}, \quad (13)$$

$$S_{\text{Reliability}} = (1 - f(\sigma_s^2)) \times (1 - f(\sigma_e^2)), f(\sigma^2) = \frac{e^{\sigma^2}}{\sum e^{\sigma^2}},$$

where $f(\cdot)$ is a softmax function to normalize the reliability score. $\sum e^{\sigma^2}$ denotes the index summation of all input proposals' uncertainty variance.

3.4.2 | Uncertainty voting

In uncertainty voting, we refer to Soft-NMS [19] that update the candidate proposal's confidence score based on the overlap between the selected proposal and the candidate proposals. Specifically, we aim to award low ratings to candidate proposals that correspond to large boundary uncertainties. The formula for softnms is:

$$S_i = \begin{cases} S_i, & IoU(m, b_i) < N_t \\ S_i \cdot \frac{\sigma_b}{\sigma_e^i} (1 - f(\sigma_b)) (1 - IoU(m, b_i)), & IoU(m, b_i) \geq N_t \end{cases}, \quad (14)$$

where s_i is the candidate proposal i 's original score, $f(\cdot)$ is the sigmoid function, $\sigma_b = (\sigma_s + \sigma_e)/2$ is the average value of uncertainty boundary (σ_s, σ_e) , σ_e^i represents the category uncertainty of the proposal, which is derived by averaging the mean value of all proposals in the same category j . And $IoU(m, b_i)$ denotes the overlapping proportion of the selected proposal m and the candidate proposal b_i . On the other hand, uncertainty voting aims to regress the selected proposal's boundaries

through uncertainty:

$$b'_m = \frac{f(\sigma_c^j) p_m b_m / \sigma_{b,m}^2 + (1 - f(\sigma_c^j)) \sum_{i \neq m} p_i b_i / \sigma_{b,i}^2}{f(\sigma_c^j) p_m / \sigma_{b,m}^2 + (1 - f(\sigma_c^j)) \sum_{i \neq m} p_i / \sigma_{b,i}^2}, \quad (15)$$

$$p_i = e^{-(1 - IoU(m, b_i))^2 / \sigma_i}, IoU(m, b_i) > 0$$

where b'_m is the temporal boundary of the selected proposal m after regression, and σ_i is the hyperparameter. In this way, we aim to avoid over-regression on classes with high uncertainty. Specifically, for the category σ_c^j with large uncertainty, we reduce the weight of the boundary adjustment by $1 - f(\sigma_c^j)$ and increase the weight of the original boundary of the chosen proposal by $f(\sigma_c^j)$.

4 | EXPERIMENTS

4.1 | Experimental settings

4.1.1 | Datasets

We conduct experiments on two commonly used datasets THUMOS14 [20] and ActivityNet v1.3 [21]. For THUMOS14, it is standard procedure to train on a validation set containing 200 temporal annotations in 20 categories and to evaluate on 213 test videos. In this dataset, each video has an average of 15 action clips, and some videos have numerous action clips. Furthermore, the video length varies widely, ranging from a few seconds to over an hour. For the reasons stated above, it's challenging to perform temporal action detection on THUMOS14. ActivityNet v1.3 is a widely used benchmark for temporal action detection. It contains 10024 training, 4926 validation, 5044 test videos, and 200 action categories. Most of the videos in this dataset have instances of activity in a single category. This dataset is much larger than THUMOS14 in terms of the number of activity categories and the number of videos. We use a training subset for training and a validation subset for testing, as is standard practice [2, 12].

4.1.2 | Evaluation metrics

Implementation details

We use the uncertainty method on one-stage Tb-SSAD [1], two-stage P-GCN [12], and anchor-free approach AFSD [2], respectively. For a fair comparison, we stick to their initial setup in our studies. To extract the video's features, we fine-tune the I3D [16] model that was pre-trained on Kinetics. In the training process, Our model is trained by Adam [22] with a learning rate of 10^{-5} and weight decay of 10^{-3} . We apply random crop as well as horizontal flipping for data augmentation. The results of RGB and optical flow frames are averaged to determine final locations and class scores.

TABLE 1 Effectiveness of each component in uncertainty method based on [2] in THUMOS14. And the mAP of tIoU is calculated by using 0.5 threshold (mAP@0.5). For mAP@0.5, the basic framework performs at 55.5%. The performance is enhanced to 57.5% after incorporating three proposed approaches

Method	Performance				
Uncertainty modeling	×	✓	✓	✓	✓
Reliability score	×	×	✓	×	✓
Uncertainty voting	×	×	×	✓	✓
mAP@0.5(%)	55.5	56.1	56.8	57.0	57.5

4.2 | Ablation study

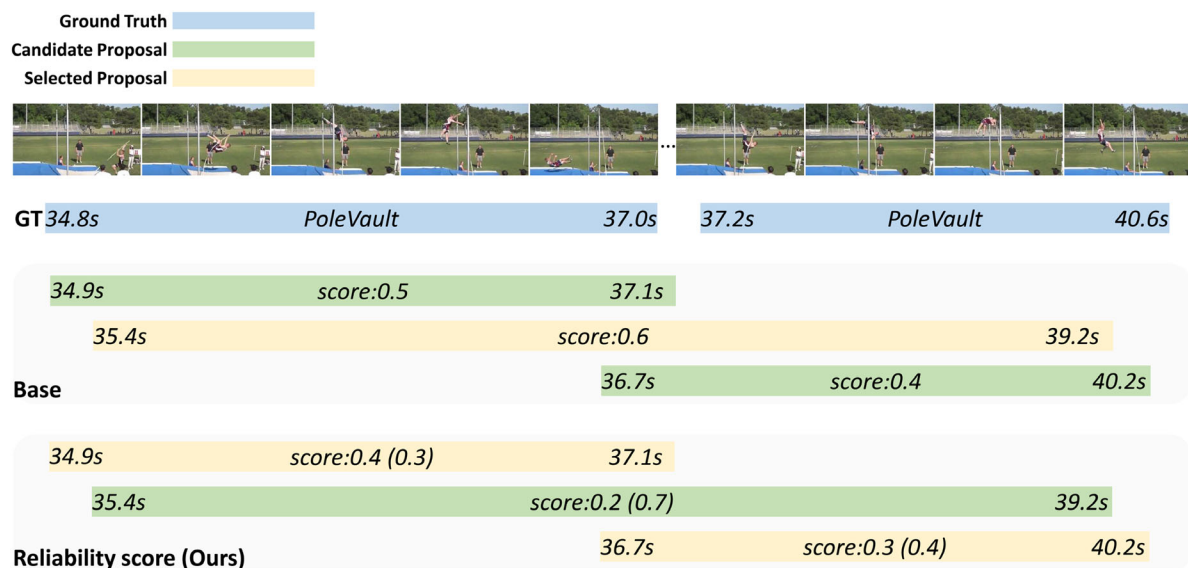
To verify the validity of the individual modules of uncertainty, we employ our algorithm on the state-of-the-art method [2]. As shown in Table 1, the model's performance improves from 55.5% to 56.1% by integrating uncertainty modeling in the training phase. With uncertainty modeling, the model's performance improves from 56.1% to 56.8% and 57.0%, respectively, when the reliability score and uncertainty voting are combined throughout the testing process. Finally, by combining the three modules above, our method achieves a 2.0% improvement on mAP@0.5 when compared to [2].

In post-processing, we visualize the reliability score and uncertainty voting in Figures 3a and 3b, respectively. The reliability score is calculated by multiplying the uncertainty score to the final proposal score, resulting in the selection of the proposal with a low level of uncertainty. After the selected proposal is determined, uncertainty voting relies on the candidate proposal to correct the boundaries of the selected proposal to bring it closer to the ground truth. On the other hand, the selected proposal revises the ratings of other candidate proposals in uncertainty voting. Specifically, in addition to reducing candidate proposal scores based on the degree of overlap between candidate and selected proposals [19], candidate proposals with higher uncertainty get their scores reduced appropriately.

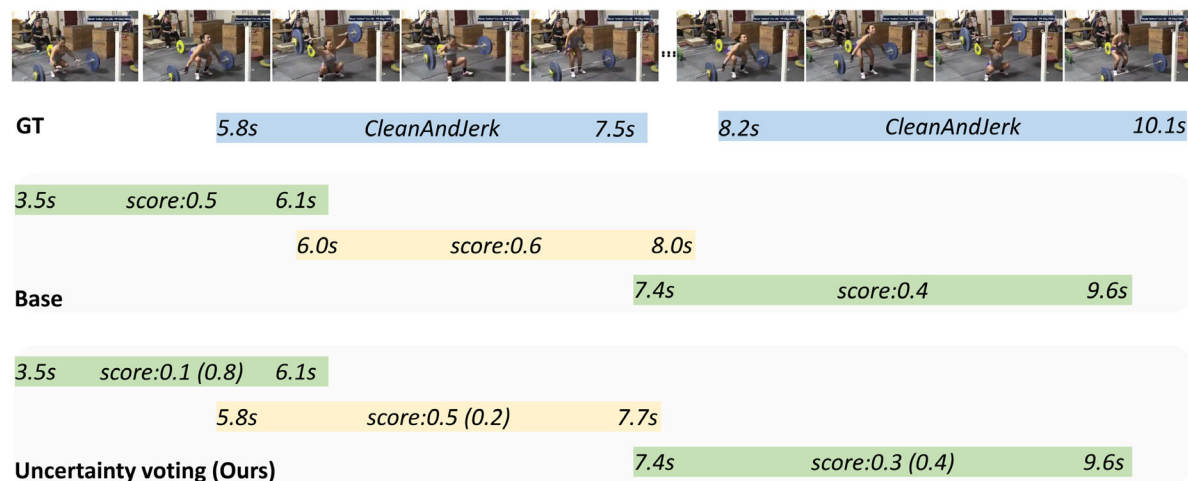
Due to the ambiguity of action definitions, the boundary of some categories is difficult to label accurately and carries a large uncertainty. This uncertainty results in a prediction deviation that is difficult to correct. Therefore, we focus on correcting the bounds corresponding to the categories with less uncertainty while relaxing the constraints for the categories with greater uncertainty by introducing a category uncertainty indicator σ_c . As shown in Figure 4, our method improves the performance of the classes with less uncertainty. On the other hand, categories with greater uncertainty have the same or reduced performance as before due to the relaxation of constraints.

4.3 | Comparison with state-of-the-art methods

To demonstrate the usefulness of the suggested uncertainty method, we deploy our method on three types of approaches: one-stage, two-stage, and anchor-free method. As shown in



(a) Visualization of reliability score.



(b) Visualization of uncertainty voting.

FIGURE 3 The reliability score is used to select proposals that have a low level of ambiguity. On the other side, uncertainty voting redraws the boundaries of the selected proposal. Besides, it adjusts the candidate proposal's score based on the selected proposal

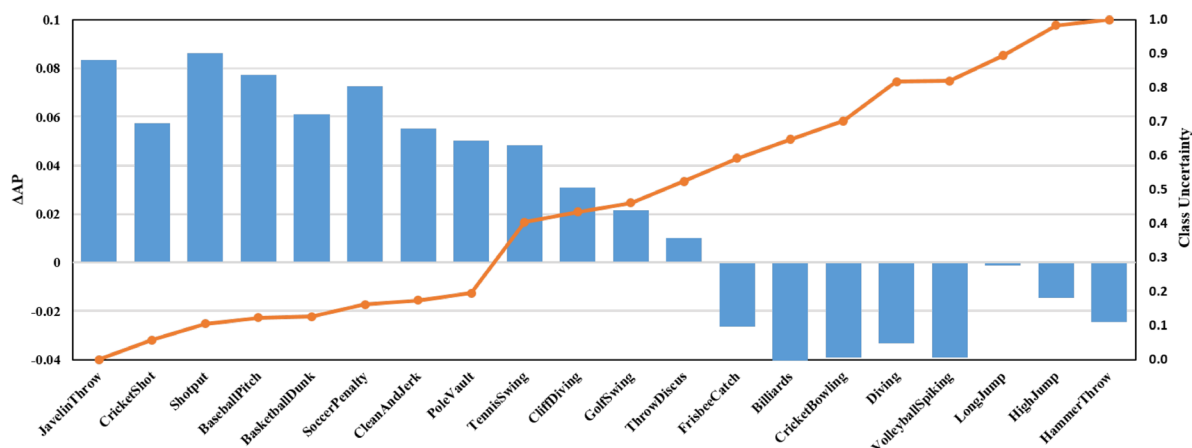


FIGURE 4 We demonstrate the performance improvement of our method in comparison to [2] (bars) and the uncertainty (lines) for each category (AP@0.5) in THUMOS14. Overall, lower class uncertainty correlates with higher performance gains

TABLE 2 Performance comparison with state-of-the-art methods on THUMOS14 and ActivityNet1.3. The performance are measured by mAP at different IoU thresholds. The average mAP is calculated in [0:3 : 0:1 : 0:7] on THUMOS14 and [0:5 : 0:05 : 0:95] on ActivityNet1.3. (Ours) represents the performance by adding our proposed uncertainty method.

Type	Model	THUMOS14					ActivityNet 1.3			
		0.3	0.4	0.5	0.6	0.7	0.5	0.75	0.95	Avg.
One-stage	SSAD [4]	43.0	35.0	24.6	15.4	7.7	-	-	-	-
	SS-TAD [5]	45.7	-	29.2	-	9.6	-	-	-	-
	SSN [23]	51.0	41.0	29.8	-	-	43.2	28.7	5.6	28.3
	Decouple-SSAD [6]	49.9	44.4	35.8	24.3	13.6	-	-	-	-
	GTAN [7]	57.8	47.2	38.8	-	-	52.6	34.1	8.9	34.3
	Tb-SSAD [1]	49.9	45.5	38.0	28.0	16.5	-	-	-	-
	Tb-SSAD (Ours)	53.0	48.5	42.1	31.9	19.7	-	-	-	-
Two-stage	R-C3D [9]	44.8	35.6	28.9	19.1	9.3	26.8	-	-	-
	TAL-Net [11]	53.2	48.5	42.8	33.8	20.8	38.2	18.3	1.3	20.2
	TAL-MR [24]	53.9	50.7	45.4	38.0	28.5	43.5	33.9	9.2	30.2
	TSA-Net [25]	61.2	55.9	46.9	36.1	25.2	48.7	32.0	9.0	31.9
	G-TAD [26]	54.5	47.6	40.3	30.8	23.4	50.4	34.6	9.0	34.1
	BC-GNN [27]	57.1	49.1	40.4	31.2	23.1	50.6	34.8	9.4	34.3
	PGCN [12]	63.6	57.8	49.1	-	-	48.3	33.2	3.3	31.1
	PGCN (Ours)	66.8	60.3	50.8	38.0	23.8	48.7	34.2	4.7	32.4
Anchor-free	AFSD [2]	67.3	62.4	55.5	43.7	31.1	52.4	35.3	6.5	34.4
	AFSD (Ours)	70.0	65.1	57.5	46.5	32.7	53.1	36.4	7.2	35.4

Table 2, our uncertainty method outperforms all three types of approaches significantly. On the one-stage method Tb-SSAD, our strategy improves the performance of the model from 38.0% to 42.1% for mAP@0.5 on the THUMOS14 dataset. On the two-stage method PGCN, the performance of the model improves from 49.1% to 50.8% on THUMOS14 (mAP@0.5) and from 31.1% to 32.4% in average mAP on ActivityNet v1.3. Finally, the model achieves the state-of-the-art (SOTA) performance by combining our method with the Anchor-free method AFSD. After incorporating our method, the performance of AFSD reaches 57.5% on THUMOS14 and 35.4% on ActivityNet v1.3.

5 | CONCLUSION

In this paper, we develop a Gaussian model to represent instance uncertainty. Based on the instance uncertainty, we propose category uncertainty, which allows the model to focus on refining the less uncertain category's boundary. By integrating instance and category uncertainty, we propose the uncertainty voting module to modify the boundaries of the selected proposals and the ranking of candidate proposals. We use a combination of one-stage, two-stage, and anchor-free networks in our method. Equipped with the anchor-free framework, we attain state-of-the-art performance, with 57.5% on THUMOS14 (mAP@0.5) and 35.4% on ActivityNet (mAP@Avg). Future work includes: (1) Apply uncertainty to the classifica-

tion and regression branch instead of focusing primarily on the regression branch. (2) Combine uncertainty with temporal information.

ACKNOWLEDGEMENTS

This work is supported by the Scientific Instrument Developing Project of the Chinese Academy of Sciences (No. YJKYYQ20200045).

CONFLICT OF INTEREST

The authors have declared no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in THUMOS14 at <http://crcv.ucf.edu/THUMOS14/> and in ActivityNet in <http://activity-net.org/>

ORCID

Yunze Chen  <https://orcid.org/0000-0002-6043-5028>

Qingyi Gu  <https://orcid.org/0000-0001-8332-5350>

REFERENCES

- Chen, Y., Chen, M., Wu, R., Zhu, J., Zhu, Z., Gu, Q.: Refinement of boundary regression using uncertainty in temporal action localization. In: British Machine Vision Conference. Springer, London (2020)
- Lin, C., Xu, C., Luo, D., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Fu, Y.: Learning salient boundary feature for anchor-free temporal action localization. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3319–3328. IEEE, Piscataway (2021)

3. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: *European Conference on Computer Vision*. Springer, Berlin (2016)
4. Lin, T., Zhao, X., Shou, Z.: Single shot temporal action detection. *Proceedings of the 25th ACM International Conference on Multimedia*. ACM, New York (2017)
5. Buch, S., Escorcia, V., Ghanem, B., Fei-Fei, L., Niebles, J.C.: End-to-end, single-stream temporal action detection in untrimmed videos. In: *British Machine Vision Conference*. Springer, London (2017)
6. Huang, Y., Dai, Q., Lu, Y.: Decoupling localization and classification in single shot temporal action detection. In: *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1288–1293. IEEE, Piscataway (2019)
7. Long, F., Yao, T., Qiu, Z., Tian, X., Luo, J., Mei, T.: Gaussian temporal awareness networks for action localization. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 344–353. IEEE, Piscataway (2019)
8. Tran, D., Bourdev, L.D., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4489–4497. IEEE, Piscataway (2015)
9. Xu, H., Das, A., Saenko, K.: R-c3d: Region convolutional 3d network for temporal activity detection. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5794–5803. IEEE, Piscataway (2017)
10. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 1137–1149 (2015)
11. Chao, Y.-W., Vijayanarasimhan, S., Seybold, B., Ross, D.A., Deng, J., Suktharankar, R.: Rethinking the faster r-cnn architecture for temporal action localization. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1130–1139. IEEE, Piscataway (2018)
12. Zeng, R., Huang, W., Tan, M., Rong, Y., Zhao, P., Huang, J., Gan, C.: Graph convolutional networks for temporal action localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7093–7102. IEEE, Piscataway (2019)
13. He, Y., Zhu, C., Wang, J., Savvides, M., Zhang, X.: Bounding box regression with uncertainty for accurate object detection. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2883–2892. IEEE, Piscataway (2019)
14. Shi, Y., Jain, A.K., Kalka, N.D.: Probabilistic face embeddings. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6901–6910. IEEE, Piscataway (2019)
15. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: *International Conference on Neural Information Processing Systems*. IEEE, Piscataway (2014)
16. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4724–4733. IEEE, Piscataway (2017)
17. Neubeck, A., Gool, L.V.: Efficient non-maximum suppression. In: *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3, pp. 850–855. IEEE, Piscataway (2006)
18. Girshick, R.B.: Fast r-cnn. In: *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448. IEEE, Piscataway (2015)
19. Bodla, N., Singh, B., Chellappa, R., Davis, L.S.: Soft-nms – improving object detection with one line of code. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5562–5570. IEEE, Piscataway (2017)
20. Jiang, Y.-G., Liu, J., Zamir, A.R., Toderici, G., Laptev, I., Shah, M., Suktharankar, R.: Thumos challenge: Action recognition with a large number of classes (2014)
21. Heilbron, F.C., Escorcia, V., Ghanem, B., Niebles, J.C.: Activitynet: A large-scale video benchmark for human activity understanding. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 961–970. IEEE, Piscataway (2015)
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. *CoRR abs/1412.6980* (2015)
23. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Tang, X., Lin, D.: Temporal action detection with structured segment networks. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2933–2942. IEEE, Piscataway (2017)
24. Zhao, P., Xie, L., Ju, C., Zhang, Y., Wang, Y., Tian, Q.: Bottom-up temporal action localization with mutual regularization. In: *European Conference on Computer Vision*. Springer, Berlin (2020)
25. Gong, G., Zheng, L., Bai, K., Mu, Y.: Scale matters: Temporal scale aggregation network for precise action localization in untrimmed videos. In: *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6. IEEE, Piscataway (2020)
26. Xu, M., Zhao, C., Rojas, D.S., Thabet, A.K., Ghanem, B.: G-tad: Sub-graph localization for temporal action detection. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10153–10162. IEEE, Piscataway (2020)
27. Bai, Y., Wang, Y., Tong, Y., Yang, Y., Liu, Q., Liu, Y.: Boundary content graph neural network for temporal action proposal generation. *arXiv abs/2008.01432* (2020)

How to cite this article: Chen, Y., Chen, M., Gu, Q.: Class-wise boundary regression by uncertainty in temporal action detection. *IET Image Process.* 16, 3854–3862 (2022). <https://doi.org/10.1049/ipr2.12599>