Cost-Sensitive Learning for Long-Tailed Temporal Action Segmentation

BMVC 2024 Submission # 227

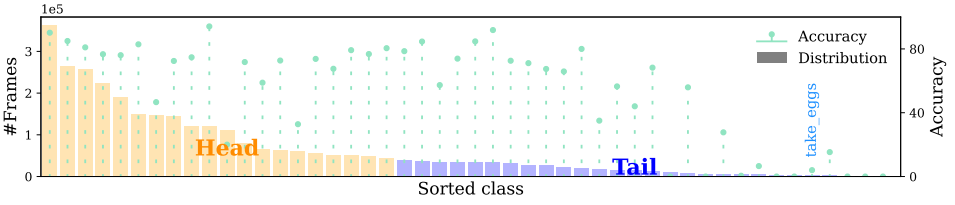
Abstract

Temporal action segmentation in untrimmed procedural videos aims to densely label frames into action classes. These videos inherently exhibit long-tailed distributions, where actions vary widely in frequency and duration. In temporal action segmentation approaches, we identified a bi-level learning bias. This bias encompasses (1) a class-level bias, stemming from class imbalance favoring head classes, and (2) a transition-level bias arising from variations in transitions, prioritizing commonly observed transitions. As a remedy, we introduce a constrained optimization problem to alleviate both biases. We define learning states for action classes and their associated transitions and integrate them into the optimization process. We propose a novel cost-sensitive loss function formulated as a weighted cross-entropy loss, with weights adaptively adjusted based on the learning state of actions and their transitions. Experiments on three challenging temporal segmentation benchmarks and various frameworks demonstrate the effectiveness of our approach, resulting in significant improvements in both per-class frame-wise and segment-wise performance.

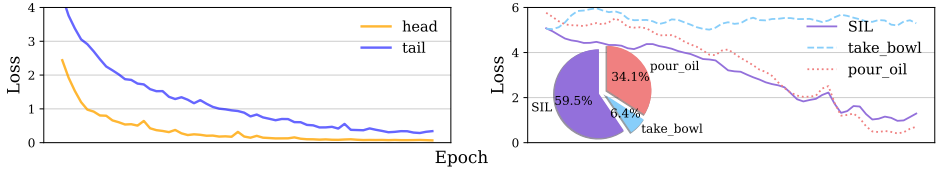
1 Introduction

Temporal action segmentation identifies actions in untrimmed procedural video sequences. These sequences often exhibit a long-tail distribution as shown in Fig. 1 (a) with tail actions that occur less frequently or have shorter durations. Despite this, state-of-the-art methods often overlook the long-tail, failing to recognize tail actions. For example, AsFormer [52] and DiffAct [31] exhibit zero accuracy on 5 and 4 out of 48 actions on Breakfast (see Fig. 1 (a) and Supplementary). The long-tail issue in action segmentation remains unexplored [10, 13, 14, 43, 52] due to the widespread use of global evaluation metrics across all samples which obscure the poor performance on tail actions.

Long tail learning on videos has predominantly been explored in action recognition [35, 54]. Action recognition [12, 29, 40] aims at classifying trimmed video clips as a whole, while temporal action segmentation focuses on frame-wise classification of untrimmed videos, necessitating the modeling of temporal dynamics and action transitions for precise segmentation. Conventional solutions to long-tail learning focus on reducing the class imbalance via loss re-weighting [9, 30], logit adjustment [32, 47], and post-hoc adjustment [21, 32]. These approaches operate under a class-independent assumption, overlooking temporal dependencies and dynamics in temporal action segmentation, thus leading to inaccurate segments and transitions. Consequently, striking a balance between improving segmentation accuracy and minimizing adverse impacts on learned temporal dynamics poses a significant challenge.



(a) Data & Class-wise accuracy distribution



(b) The bi-level learning bias

Figure 1: (a) Long-tail action distribution on Breakfast [24]. The long-tail distribution results in low accuracy on tail actions with AsFormer [52]. (b) Left: Head-tail loss curve shows slow convergence rate on tail actions, demonstrating the class-level learning bias. Right: Action *'take_eggs'* from tail shows skewed transition distribution(pie chart), *i.e.*, different transitions from { *'SIL'*, *'pour_oil'*, *'take_bowl'*} to *'take_eggs'*, and transition learning bias(loss curve, where common transition from *'pour_oil'* are better learned than *'take_bowl'*)

Our paper addresses the long tail issue in temporal action segmentation, bridging the research gap of long-tailed learning for untrimmed videos. Empirically, we observe a bi-level biased learning process attributed to the long-tail problem.

- Class imbalance leads to a **class-level learning bias**, which prioritizes learning head over tail actions, leading to different class convergence rates (Fig. 1 (b)). However, unlike the typical over-fitting to tail observed in long-tailed image classification and segmentation [2, 18, 38, 39, 48], we observe under-learnt tail actions in temporal segmentation. This is because the learning of tail is suppressed due to the temporal continuity of frame representation, see Fig. 2. Distinctly separating two consecutive actions, one being head and the other tail, is challenging as they share similar frame representations, especially at segment boundaries. This similarity in representation hinders independent learning of tail actions without adversely affecting head actions.

- Variations in action transitions introduce a **transition-level learning bias**. In Fig. 1 (b), for action *'take_eggs'*, the transition distribution from *'pour_oil'* or *'take_bowl'* to *'take_eggs'* is skewed. We observe a higher frequency of *'take_eggs'* preceded by *'pour_oil'*. Such frequent transitions, *e.g.* from *'pour_oil'*, tend to form stronger associations, resulting in learning gaps across transitions. For instance, *'take_eggs'* is more easily detected when preceded by *'pour_oil'* compared to *'take_bowl*.

To address these biases, we propose utilizing the class-wise accuracy to evaluate action learning state and transition-wise accuracy for transition learning state. These evaluations determine if an action or its transition is over- or under-learned by comparing them to their respective average accuracy. We design a constrained optimization problem targeting a balanced accuracy to reduce class-level bias. Constraints on temporal transition

learning are also imposed to address transition-level bias. Incorporating these constraints into a deep learning framework is nontrivial. To tackle this, we reframe the optimization as a Lagrangian min-max problem, which can be optimized by minimizing a surrogate cost-sensitive loss function. Our new loss function, a weighted cross-entropy formulation, adjusts weights adaptively based on the learning state of actions and their transitions.

Our contributions can be summarized as: (1) identifying the bi-level learning bias and the under-learned tail classes in temporal action segmentation, which differs from the common over-fitting trends observed in other tasks, (2) proposing a cost-sensitive loss that addresses these biases via a constraint optimization formulation, (3) conducting extensive evaluations on different backbones and datasets, showcasing notable performance improvements.

2 Related works

Temporal Action Segmentation employs various architectures such as temporal convolutional networks (TCN) [13, 25, 26, 27, 43], transformers [1, 52], and diffusion models [31]. These architectures expand the temporal receptive field [13, 43] and aggregate temporal dynamics [1, 52], facilitating information exchange across frames. To address the over-segmentation in such backbones, several approaches like boundary smoothing [20, 50] and refinement [1] has been proposed. Moreover, to incorporate temporal constraints in these backbones, differentiable temporal logic [51] and activity grammar [15] are utilized.

Long-Tail Learning involves various techniques. Re-sampling methods either undersample the head [2, 3, 42] or oversample the tail [11, 16]. Re-weighting assigns different weights to classes [9, 19, 49] or samples [30, 37]. Logit adjustment modifies margins based on class priors [4, 32] or compensation terms [46, 47, 55]. Post-hoc adjustment includes normalizing classification weights [21, 22, 53] or modifying thresholds [7, 23]. These methods have been extended to object detection/segmentation [28, 46] and video classification [35, 54].

Temporal action segmentation differs from these tasks due to temporal correlations between frames and segments. The long-tail issue in this domain remains unexplored. Our work addresses the long-tail in temporal action segmentation, aiming to tackle learning biases while accounting for temporal dynamics.

3 Method

In temporal action segmentation, a classifier f maps a video sequence $X \in \mathbb{R}^{D \times T}$ represented with pre-computed features [6] to a sequence of actions $Y \in [L]^T$. Here, D is the feature dimension, T indicates the number of frames, and L represents the number of classes. Classifier f is typically a neural network backbone such as MSTCN [13] or AsFormer [52], where segmentation is usually framed as frame-wise classification.

Our paper presents a cost-sensitive learning framework to tackle the long-tail issue in action segmentation. We evaluate the learning states of both actions (class-level) and action transitions (transition-level) using a transition-based confusion tensor in Section 3.1. We then formulate a learning-aware constrained optimization problem that is transformed it into a new cost-sensitive loss [17, 33, 36] in Section 3.2. We provide details on training with the cost-sensitive loss and a new post-processing technique for inference in Section 3.3.

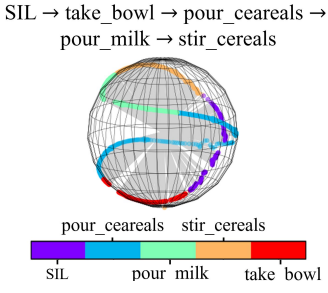


Figure 2: The t-SNE of the frame-wise representations for a video of making cereal exhibits a strong temporal continuity.

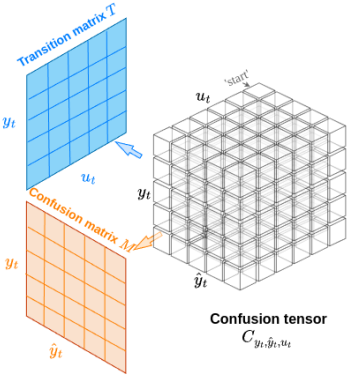


Figure 3: Transition-based confusion tensor.

3.1 Class- & Transition-level Learning States

A video sequence with N actions and T frames can be labelled either on a frame-wise basis, $Y = \{y_t\}_{t=1}^{T}$, with frame index t, or on a segment-wise basis, $Y = \{a_n\}_{n=1}^{N}$, with segment index n. The action segment $a_n = (s_n, e_n, l_n)$ represents a segment with start time s_n, end time e_n, and label l_n. For a timestamp $t \in [s_n, e_n]$, its frame label $y_t = l_n$.

To reduce the class-level learning bias, it is important to guarantee all classes are equally learned, namely achieving a uniform learning across classes. Similarly, all transitions should be learned equally to reduce transition-level learning bias. We assess the learning state for an action or a transition during training using its corresponding accuracy on training set. This can be calculated with a transition-based confusion tensor. Given a classifier f, the ijk^{th} entry of its confusion tensor C is defined as

$$C_{i,j,k}[f] = \mathbb{E}_{(X,Y)}[\mathbb{1}(y_t = i, \hat{y}_t = j, u_t = k)] \tag{1}$$

where for the t^{th} frame, y_t is the ground truth, $\hat{y}_t$ is the prediction from f, u_t represents the previous action of the current frame, namely, $\forall t \in [s_n, e_n], y_t = l_n, u_t = l_{n-1}$, with $l_{-1} =$ 'start'.

Based on the confusion tensor C, we can derive the corresponding confusion matrix M and transition matrix T shown in Fig. 3 as

$$M_{i,j}[f] = \sum_k C_{i,j,k}[f], \quad T_{i,k} = \sum_j C_{i,j,k}[f] \tag{2}$$

Note the transition matrix T remains constant regardless of the classifier f. It can be derived solely from the training dataset and remains consistent across variants of models or initialization. For simplicity, we omit the dependency of T on f.

Then, for an action class i with a transition from another class k, $k \neq i$, the learning states for class i and the transition associated i and k are formulated as the corresponding accuracy:

$$Acc_i[f] = \frac{M_{i,i}[f]}{\pi_i} = \frac{\sum_k C_{i,i,k}[f]}{\pi_i}, \quad Tacc_{k\to i}[f] = \frac{C_{i,i,k}[f]}{T_{i,k}}, \tag{3}$$

where π_i is the class prior $p(y = i) = \sum_k T_{i,k}$.

3.2 Cost-sensitive Learning with Constraint Optimization

We propose a new learning objective to mitigate class-level learning bias. The objective is to maximize per-class accuracy $\max_f \sum_i Acc_i[f]$, ensuring equal attention to all classes.

To further reduce the biased transition learning, we define transition constraints to reduce the learning variance. Let V_T denote the set of valid action transitions observed in the training set; specifically, $V_T = \{(k \to i), T_{i,k} > 0\}$. One way to regularize the transition learning is to penalize under-learned transitions:

$$\forall (k \to j) \in V_T, \quad Tacc_{k\to i}[f] \geq \varepsilon \overline{Tacc}, \quad \text{where} \quad \overline{Tacc} = \frac{1}{|V_T|} \sum_{(k\to i)\in V_T} Tacc_{k\to i}[f], \tag{4}$$

where ε is a tolerance hyperparameter which set to 0.9 in our implementation. $\overline{Tacc}$ is the average accuracy over all transitions. To simplify the problem, we detach $\overline{Tacc}$ from the classifier f, *i.e.*, $\overline{Tacc}$ is not considered as a function of f. We set it as a hyper-parameter and update it every epoch during training.

Then, the objective on per-class accuracy and above constraints combine to the problem:

$$\max_f \sum_{i,k} \frac{C_{i,i,k}[f]}{\pi_i} \qquad \text{s.t.} \ \forall (k \to i) \in V_T, \ \frac{C_{i,i,k}[f]}{T_{i,k}} \geq \varepsilon \overline{Tacc}. \tag{5}$$

Optimizing Eq. (5) with constraints is challenging. A common strategy is to relax the constraints and reformulate the objective as a Lagrangian. Eq. (5) can be reformulated as an equivalent Lagrangian min-max problem $\mathcal{L}(f, \lambda)$ by introducing Lagrange multipliers λ:

$$\max_f \min_{\lambda \in \mathbb{R}_+} \sum_{i,k} \frac{C_{i,i,k}[f]}{\pi_i} + \sum_{i,k} \mathbb{1}(T_{i,k} > 0) \lambda_{i,k} \left(\frac{C_{i,i,k}[f]}{T_{i,k}} - \varepsilon \overline{Tacc} \right) \frac{T_{i,k}}{\pi_i}, \tag{6}$$

where a constant term $\frac{T_{i,k}}{\pi_i}$ for a given transition pair $k \to i$ is multiplied by each transition constraint to balance the magnitudes between the objective and constraints.

The Lagrangian is solved iteratively by maximizing f while fixing the multipliers λ and minimizing λ while keeping f fixed. In practice, instead of the full optimization at each max/min iteration, we only take a few update steps for gradient descent to update the classifier f and for projected gradient descent to update the Lagrange multiplier. The detailed training algorithm can be found in Supplementary.

Step 1. Maximizing the Lagrangian $\mathcal{L}(f)$ with fixed λ leads to the following objective:

$$\max_f \sum_{i,k} G_{i,i,k} C_{i,i,k}[f] + \text{constant}, \tag{7}$$

where G is a *gain tensor* representing the gain of the correct classification and transition, while 'constant' absorbs terms not depending on f. Given a transition from action k, the slice $G_{:,:,k}$ from the gain tensor G is a diagonal matrix, and $G_{i,i,k} = (1 + \mathbb{1}(T_{i,k} > 0)\lambda_{i,k})/\pi_i$. Optimizing Eq. (7) is equivalent to minimizing a re-weighted loss in Eq. (8), which is proven to be calibrated for this diagonal gain matrix [33, 34]. See Supplementary for the proof.

$$l_{CE}(y_t, u_t, X) = -G_{y_t, y_t, u_t} \log(p(y_t \mid X)) \tag{8}$$

The formulation in Eq. (7) represents a more generalized form of cost-sensitive learning [33]. Standard cost-sensitive learning is typically formulated as naive reweighting based

on the class frequency, treating each class independently. In contrast, our proposed reweighting factor $G_{i,i,k}$ considers class inter-dependencies, incorporating an extra term that models the transitions as shown in Eq. (9). This extra term allows adaptive adjustment of the reweighting factor for a given action based on its current transition learning state.

$$G_{i,i,k} = \underbrace{\frac{1}{\pi_i}}_{\text{action prior}} + \underbrace{\frac{\lambda_{i,k}\mathbb{1}(T_{i,k}>0)}{\pi_i}}_{\text{transition learning state}} \tag{9}$$

Step 2. Minimizing the Lagrangian $\mathcal{L}(\lambda)$ is done by projected gradient descent. The gradients of the Lagrangian objective $\mathcal{L}$ in Eq. (6) with respect to λ is estimated as $\nabla_\lambda \mathcal{L}$. The multipliers λ are updated with gradient descent and projected back to $\mathbb{R}_+$ as

$$\lambda^{(l+1)} = \max\{0, \lambda^{(l)} - \gamma \nabla_\lambda \mathcal{L}\}, \tag{10}$$

where γ is the step size for updating the multipliers, l is the iteration index.

3.3 Training and Inference

Empirically, over-emphasizing the per-class performance will hurt the global performance. To achieve a better trade-off, we introduce a hyper-parameter, τ, to modify the gain tenser as $\tilde{G}_{i,j,k} = G^{\tau}_{i,j,k}$ for the frame-wise loss in Eq. (8). A small τ will smooth the weights of the loss function, favoring global performance. Conversely, a large τ emphasizes reducing biased learning and enhancing per-class performance. We estimate the confusion tensor C every epoch on training set to update Lagrangian multipliers λ and the gain tensor G. Importantly, this modification only affects training, leaving the inference stage unchanged.

Similar to [21], we identify that the final classifier is biased. Inspired by Nearest Class Mean(NCM) [44], we propose a new post-processing technique to further mitigate the long-tail impact. Specifically, instead of relying on the classifier, we make predictions using frame representations based on NCM, which involves computing mean representations for each class and performing nearest neighbor search using Euclidean distance. Applying frame-level NCM, however, disregard the temporal continuity and lead to over-segmentation. We propose Segment Nearest Class Mean (S-NCM) to address this. We first leverage the classifier's predictions $\hat{y}$ to detect segment boundaries b and then utilize frame-wise NCM predictions $\hat{v}$ for labelling each segment through major voting, namely the frames in each segment share the same prediction $\tilde{y}$.

$$\tilde{y}_{b_i:b_{i+1}} = \text{mod}(\hat{v}_{b_i}, \hat{v}_{b_i+1}, \cdots, \hat{v}_{b_{i+1}}) \quad \text{where} \quad b = \{t, \text{ if } \hat{y}_t != \hat{y}_{t+1}, \forall t \in [1, T-1]\} \tag{11}$$

4 Experiments

4.1 Dataset, Implementation, and Evaluation

Dataset. We evaluate our framework on three benchmarks: Breakfast Actions [24], 50Salads [45] and the recently released Assembly101 [41]. We split the actions in these datasets into Head and Tail groups based on the class frequency (see details in Supplementary).

Implementation details. We consider three backbones: a temporal convolution model MSTCN [13], a transformer model ASFormer [52], and a state of the art diffusion-based model DiffAct [31]. All models are retrained using the released source codes based on I3D

Table 1: Per-class & global result summary across datasets and backbones over 3 runs. The column 'G_F1' represents the global F1 score with IOU threshold 0.25 over all samples.

Model	Breakfast					50salads					Assembly101				
	Per class				G_F1	Per class				G_F1	Per class				G_F1
	F1@{10,25,50}			Acc.		F1@{10,25,50}			Acc.		F1@{10,25,50}			Acc.	
MSTCN [13]	48.1	44.8	36.9	49.1	57.9	78.8	76.4	67.6	75.6	75.9	7.5	6.6	4.8	8.3	27.2
+ CB [9]	+0.9	+0.7	+0.3	+0.6	0.0	-0.6	-0.2	-0.8	-0.3	-0.4	+1.8	+1.7	+1.2	+1.5	-0.5
+ LA [32]	+1.0	+1.1	+0.1	+1.4	0.0	-0.2	-0.7	0.0	-0.3	-0.7	+2.1	+1.4	+1.2	+1.2	-1.1
+ Focal [30]	+0.2	-0.3	-1.2	-0.5	-0.4	+0.6	+0.5	+1.0	+0.4	+0.2	+1.9	+1.6	+0.5	+1.4	-0.2
+ τ-norm [21]	-1.1	-1.0	-1.0	-0.8	-0.9	-0.6	-0.5	-0.2	+0.2	-0.6	+0.1	+0.2	+0.1	-0.2	+0.2
+ ours(S-NCM)	**+8.1**	**+8.1**	**+5.7**	**+3.7**	**+6.1**	**+2.8**	**+3.1**	**+3.2**	**+1.7**	**+3.1**	**+4.1**	**+3.3**	**+2.0**	**+2.6**	**+2.3**
ASFormer [53]	57.9	54.7	45.4	52.3	69.9	85.1	82.6	75.3	81.5	82.3	9.2	7.6	5.2	9.2	30.4
+ CB [9]	+0.8	+1.0	+0.9	+0.8	-0.2	+0.2	+1.0	+0.9	+0.2	+0.9	0.0	-0.1	0.0	+0.2	-2.2
+ LA [32]	+1.4	+1.0	+1.3	+0.5	-0.2	+0.2	+0.9	+1.5	+0.4	+0.9	-0.1	+0.4	+0.1	+0.3	-1.9
+ Focal [30]	+1.0	+1.2	+0.4	-0.3	**+0.5**	+0.7	+0.8	+1.1	-0.3	+1.2	+1.5	+2.1	+1.1	+1.7	-0.1
+ τ-norm [21]	0.0	+0.2	+0.4	+0.8	-0.8	-0.1	0.0	+0.1	+0.1	-0.1	-1.9	-2.1	-1.3	-1.0	-7.7
+ ours(S-NCM)	**+3.1**	**+3.2**	**+3.6**	**+2.8**	**+0.5**	**+1.5**	**+2.2**	**+3.1**	**+1.6**	**+1.7**	**+4.3**	**+4.5**	**+3.5**	**3.5**	**+1.3**

features [6] pre-trained on Kinetics. Results on Breakfast and 50salads are reported based on standard 4- and 5-fold splits respectively, while for Assembly101, we employ the provided train-val-test split and report test results. All long-tailed methods are trained with the same settings as the original baseline.

Evaluation metrics. Three commonly used metrics [13, 43, 50, 52] are: frame-wise accuracy (Acc.), segment-wise edit score (Edit), and F1 score with IoU thresholds of 0.10, 0.25 and 0.50 (F1@10/25/50). Conventionally, these metrics are tabulated globally over all the frames, obscuring the performance of tail actions. To emphasize the performance of tail actions, we use balanced metrics commonly used in long-tailed works[21, 46, 47]. Specifically, we calculate the average of recall scores per class for frame-wise accuracy and use the per-class F1 score for the segment-wise evaluation.

4.2 Benchmark Results

Compared to existing long-tail methods (Table 1), our approach demonstrates superior per-class performance across all datasets and backbones. Existing methods such as CB [9] struggle with locating transitions; our method leverages constraints to detect transition boundaries and has substantial improvements in F1 scores. For example, we surpass the second best model LA [32] on F1 score by 8.3%, 3.5%, and 3.0% for Breakfast, 50Salads, and Assembly101 respectively for MSTCN backbone. Additionally, our approach has strong frame-wise accuracy because it dynamically adjusts the learning focus based on action and transition learning states. Competing methods such as CB [9] employ class-wise reweighting without considering the learning state. Focal loss [30] overemphasises frames at transition boundaries, even though these are ambiguous [10, 31]. Due to space constraints, we present the global performance of F1@25 score. Other global results can be found in Supplementary. The results demonstrate our method's ability to balance per-class and global performance.

Table 2 compares head versus tail group performance. Our methods' emphasis on transitions allows us to improve segment-wise performance for both head and tail classes. From a frame-wise perspective, our approach boosts tail classes without compromising the head classes. Notably, Focal loss [30] predominantly focuses on hard boundary frames from head classes due to their high frequency, thereby primarily improving head rather than tail classes.

We further evaluate our method with the SOTA DiffAct [51] backbone on Breakfast. Results in Table 3 demonstrate the effectiveness of our method on improving per class performance, particularly on tail actions, without sacrificing the global performance.

Table 2: Group-wise result summary across datasets and backbones.

Model	Breakfast				50salads				Assembly101			
	Accuracy		F1@25		Accuracy		F1@25		Accuracy		F1@25	
	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail
MSTCN	65.1	37.7	53.3	38.7	87.7	70.0	85.7	72.1	33.9	4.7	26.3	3.9
+ CB [9]	64.1	39.3	54.1	39.4	**88.4**	69.3	85.3	72.0	34.8	6.8	28.1	6.0
+ LA [32]	64.4	40.6	56.0	38.7	87.5	69.6	86.0	71.0	34.3	6.4	27.1	5.8
+ Focal [30]	**66.1**	36.1	53.6	38.0	88.3	70.3	84.8	73.3	**35.3**	6.6	26.3	6.4
+ τ-norm [21]	65.3	36.2	52.7	37.4	87.6	70.3	85.1	71.6	34.0	4.3	25.9	4.2
+ ours(S-NCM)	65.3	**44.0**	**64.5**	**44.6**	87.8	**72.5**	**87.7**	**75.7**	34.1	**8.7**	**31.7**	**7.6**
ASFormer	69.7	39.8	69.9	43.9	90.6	77.4	87.5	80.3	35.2	5.7	29.0	4.8
+ CB [9]	**70.2**	40.8	71.2	44.7	**90.9**	78.0	88.4	81.4	35.4	5.9	26.5	5.2
+ LA [32]	**70.2**	40.4	71.2	44.7	90.3	78.1	89.0	81.1	36.1	5.9	27.5	5.7
+ Focal [30]	69.9	39.1	71.3	44.9	89.7	77.3	88.1	81.8	**36.2**	7.9	29.4	7.9
+ τ-norm [21]	69.6	41.3	69.9	44.3	90.4	77.5	87.7	80.2	32.2	4.9	21.8	3.2
+ ours(S-NCM)	69.7	**44.7**	**72.5**	**47.5**	**90.9**	**79.5**	**90.1**	**82.6**	35.7	**10.9**	**33.9**	**10.5**

Table 3: Diffusion backbone trained on Breakfast. Left - per class & global performance; Right - group-wise performance.

Model	Per class				G_F1
	F1@{10,25,50}			Acc.	
DiffAct [31]	63.3	61.5	53.6	56.2	75.5
+ CB [9]	-0.4	-0.5	-0.4	0.0	+0.3
+ LA [32]	-0.2	-0.4	-0.1	+0.9	+0.1
+ ours(S-NCM)	**+0.9**	**+0.5**	**+0.4**	**+1.8**	**+0.4**

Accuracy		F1@25	
Head	Tail	Head	Tail
74.9	42.8	77.6	49.9
74.1	43.3	77.5	49.1
75.4	44.5	**78.1**	49.0
74.5	**46.4**	77.6	**50.8**

4.3 Ablation Studies and Analysis

Components. We evaluate the contributions of the objective function (for class-level bias) and transitional constraints (for transition-level bias) in our proposed constraint optimization Eq. (5), alongside segment-wise post-processing S-NCM in Table 4. Incrementally incorporating the objective function and transitional constraints progressively enhances performance. When applied to the naive frame-wise NCM, despite significantly improving per-class accuracy, we observe a noticeable decline in the global F1 score due to over-segmentation. Our proposed S-NCM effectively tackles the over-segmentation issue, thus outperforming the naive NCM. Notably, even without the S-NCM component, our proposed cost-sensitive learning framework substantially enhances per-class performance while maintaining global performance. The inclusion of S-NCM further boosts the final results.

Table 4: Impact of components on AsFormer. First row is the baseline with no components.

Objective	Constraint	NCM	S-NCM	Breakfast					Assembly101				
				Per class				G_F1	Per class				G_F1
				F1@{10,25,50}			Acc.		F1@{10,25,50}			Acc.	
✗	✗	✗	✗	57.9	54.7	45.4	52.3	69.9	9.2	7.6	5.2	9.2	30.2
✓	✗	✗	✗	58.4	55.6	46.4	52.9	69.8	11.2	8.9	5.8	10.7	29.8
✓	✓	✗	✗	60.6	57.3	48.5	54.4	**70.8**	12.4	11.3	7.7	**12.1**	29.9
✓	✓	✓	✗	59.7	56.4	46.7	55.0	67.9	11.0	9.3	6.3	11.3	23.1
✓	✓	✗	✓	**61.0**	**57.9**	**49.0**	**55.1**	70.3	**13.5**	**12.1**	**8.7**	**12.1**	**31.7**

Impact of τ. Table 5 shows the impact of the hyperparameter τ, applied to gain tensor G. This hyperparameter determines the learning balance between global and per-class performance. A smaller τ results in smoother reweighting factors, prioritising global performance

but potentially obscuring the tail actions. Conversely, a large τ emphasises the learning of tail class, favouring per-class performance. A τ set too large may overemphasise tail actions, leading to performance drops. Compared to Breakfast, Assembly101 exhibits a larger scale and greater imbalance. The hyperparameter τ for Assembly101 is then set to a smaller value. More details regarding the selected τ can be found in Supplementary.

Table 5: Impact of threshold τ with Asformer.

τ	Breakfast					Assembly101				
	Per class		Global			Per class		Global		
	F1@25	Acc.	F1@25	Acc.	Edit	F1@25	Acc.	F1@25	Acc.	Edit
0.1	56.6	53.4	69.9	71.8	**74.0**	**12.1**	**12.7**	**31.7**	**40.8**	**32.9**
0.3	**57.9**	**55.1**	**70.3**	**72.1**	73.4	10.6	11.5	30.9	40.3	31.5
0.5	57.4	54.9	69.9	71.8	72.4	10.8	11.5	30.1	39.0	31.1
0.7	56.8	54.6	67.3	71.1	69.7	10.6	12.2	28.3	37.1	29.9

Lagrangian multiplier λ. We illustrate various types of evolution of accuracy and Lagrangian multipliers, λ, using transitions related to the action *'butter_pan'* in Fig. 4. Our constraints penalise transitions with learning speed slower than the average. An increasing multiplier indicates a violation of its corresponding constraint. In this plot, we observe that transitions to *'butter_pan'* from other actions exhibit varying learning states. For example, the transition from *'stir_dough'*, exhibits faster learning speed, with its accuracy surpassing the average accuracy, leading to its multiplier decreasing to zero. Conversely, for less frequent transitions, such as transition from *'spoon_flour'*, the corresponding λ keeps increasing, indicating that its learning state remains below average, prompting more attention towards this transition.

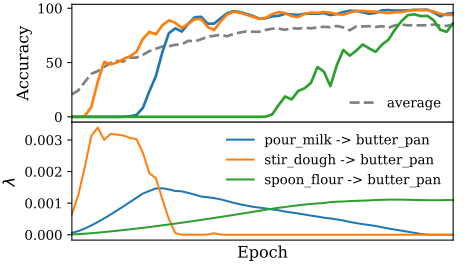


Figure 4: Transition accuracy and Lagrangian multiplier, λ, curves during training using AsFormer on Breakfast.

Computational cost. Our method requires additional computation costs for calculating the confusion tensor. We estimate the confusion tensor at every epoch for the full training set, which leads to a 30% longer training time. To mitigate this overhead, we could consider sampling a subset of the training set or employ an exponential moving average approach. The testing time complexity remains unaffected compared to the baseline.

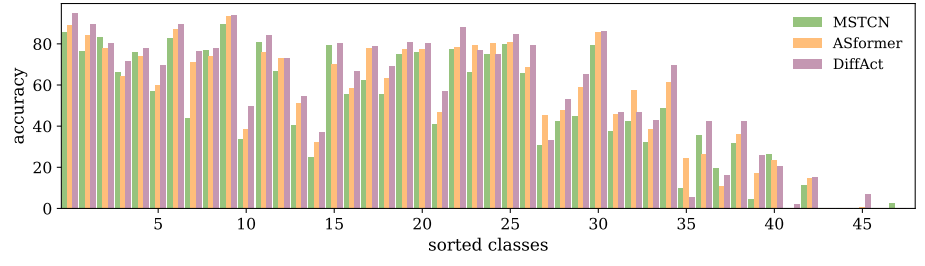
5 Conclusion

We propose a constrained optimization approach to address the bi-level learning bias in temporal action segmentation. The optimization includes an objective to reduce class-level bias arising from class imbalance, and extra transition constraints to reduce transition-level bias stemming from variations in transitions. The problem is transformed into a new cost-sensitive loss function with adaptively adjusted loss weights. Experiments on challenging benchmarks demonstrate the effectiveness of our approach.

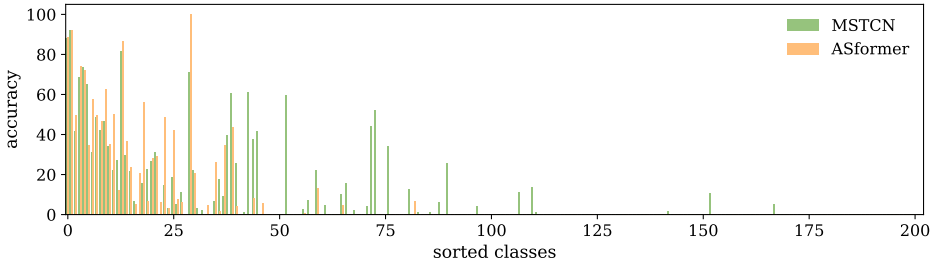
Supplementary

A. Long-tail Problem in Temporal Action Segmentation

Temporal action segmentation methods [13, 31, 52] often ignore the long-tail problem, leading to poor performance on tail classes. For instance, state-of-the-art models like MSTCN [13], ASFormer [52], and DiffAct [31] fail to predict tail classes accurately. On Breakfast dataset, MSTCN and ASFormer each have zero accuracy for 5 out of 48 classes, while DiffAct misses 4 classes entirely. On Assembly101 dataet, there are 30 classes do not appear in test set. Except those non-appeared classes, MSTCN and ASFormer achieve zero accuracy for 106 and 128 classes of 141 tail classes respectively. Details can be seen in Fig. 5.



(a) Breakfast



(b) Assembly101

Figure 5: Class-wise accuracy distribution.

B. Convert Optimization to Weighted Cross-entropy

Proposition 1. *Given a timestamp t and its previous action u_t, the optimal classifier of*

$$\max_{f} \sum_{i,j,k} G_{i,j,k} C_{i,j,k}[f]$$

for a gain matrix $G \in \mathbb{R}^{L\times L\times L+1}$ and the t^{th} frame takes the form:

$$f^*(X,u_t) \in \arg\max_{j\in[L]} \sum_i p_i(X) G_{i,j,u_t}$$

where $p_i(X)$ is the estimated conditional probability for class i at the current frame t by classifier f.

Proof.

$$\begin{aligned}\sum_{i,j,k} G_{i,j,k}C_{i,j,k}[f] &= \mathbb{E}_{(X,y_t,u_t)}[\sum_{i,j,k} G_{i,j,k}\mathbb{1}(y_t=i,\hat{y}_t=j,u_t=k)] \\ &= \mathbb{E}_{(X,y_t,u_t)}[\sum_{j} G_{y_t,j,u_t}\mathbb{1}(\hat{y}_t=j)] \\ &= \mathbb{E}_{(X,u_t)}\mathbb{E}_{(y_t|X,u_t)}[\sum_{j} G_{y_t,j,u_t}\mathbb{1}(\hat{y}_t=j)] \\ &= \mathbb{E}_{(X,u_t)}[\sum_{i,j} p_i(X)G_{i,j,u_t}\mathbb{1}(\hat{y}_t=j)]\end{aligned}$$

We use the fact in frame-wise classification where the prediction for y_t does not depend on u_t. It suffices to maximize the above objective point-wise to compute the Bayes-optimal classifier. To predict for a frame labelled as y_t of given input X and the label for the previous action u_t, the prediction should maximize the term in the expectation.

$$f^*(X,u_t) \in \arg\max_{j\in[L]} \sum_i p_i(X)G_{i,j,u_t}$$

where $G_{:,:,u_t}$ is a matrix, denoting a slice of G. □

In our case, the gain matrix $G_{:,:,u_t}$ is diagonal. The optimal classifier takes the form

$$f^*(X,u_t) \in \arg\max_{i\in[L]} p_i(X)G_{i,i,u_t} \propto \arg\min_{i\in[L]} -G_{i,i,u_t}\log p_i(X)$$

which is the reweighted cross entropy loss and is calibrated for the diagonal gain matrix.

C. Algorithm for Constrained Learning

The Lagrangian min-max problem can be solved by iteratively maximizing it over f with fixed multipliers, and minimizing it over multipliers λ while fixing f. Algorithm 1 shows the details. There are theoretical guarantees on convergence for the learned classifier [6, 8, 33].

Algorithm 1 Optimizing Per class Accuracy with Transitional Constraints

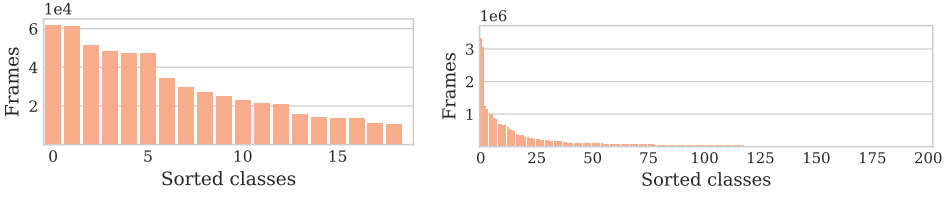
Input: Training set $\mathcal{D}$, Class prior $\pi \in \mathbb{R}_+^L$ and transition prior $T \in \mathbb{R}_+^{L\times(L+1)}$ derived from $\mathcal{D}$, Learning rate for multiplier $\gamma \in \mathbb{R}_+$, Cost-sensitive loss function l, Lagrangian objective $\mathcal{L}$

Initialize: Classifier f, Multiplier $\lambda \in \mathbb{R}_+^{L\times(L+1)}$

for epoch $l \leftarrow 0,\ldots,N$ **do**
  // Update G
  Calculate the gain tensor G based on π, T, and λ
  // Update f
  $f^{l+1} \in \arg\min_f \frac{1}{|\mathcal{D}|}\sum_{(X,Y)\in\mathcal{D}} l(y_t,\hat{y}_t,G)$ *// $Y=\{y_t\}$*
  // Update λ
  $C_{i,j,k}[f^{l+1}] = \frac{1}{|\mathcal{D}|}\sum_{(X,Y)\in\mathcal{D}} \mathbb{1}(y_t=i,\hat{y}_t=j,u_t=k)$ *// calculate confusion matrix*
  Calculate $\overline{Tacc}$ based on T and $C[f^{l+1}]$
  $\lambda_{i,k}^{l+1} = \max\{\lambda_{i,k}^l - \gamma\nabla_{\lambda_{i,k}}\mathcal{L}, 0\}$ *// gradients are calculated based on $\overline{Tacc}$*

D. Experimental Setting

Dataset. We evaluate our method on three benchmarks. (1) Breakfast comprises 1712 videos for breakfast preparation, featuring 48 action classes with an average duration of 2.3 minutes. (2)Assembly101 has a collection of 4321 videos focused on assembling and disassembling toys, with an average length of 7.1 minutes and 202 coarse action classes. (3) 50Salads contains 50 videos of making mixed salads, involving 19 actions. Data distribution of these datasets is illustrated in Fig. 6.



(a) 50salads (b) Assembly101

Figure 6: Data distribution of 50salads and Assembly101.

Head/Tail group split: We divide the action classes into head and tail groups based on the class frequency as in Table 6 and evaluate the performance of different methods on each group.

Table 6: Head/Tail class split criterion.

Dataset	Head		Tail	
	#classes	#frames	#classes	#frames
Breakfast	20	$\geq 5\times10^4$	28	$\leq 5\times10^4$
50salads	6	$\geq 4\times10^4$	13	$\leq 4\times10^4$
Assembly101	31	$\geq 1.8\times10^5$	171	$\leq 1.8\times10^5$

Hyperparameters: The used hyperparameters for each dataset, method, and backbone are shown in Table 7. We omit τ-norm [21] as the results always favour $\tau = 1.0$ for τ-norm. In our method, we fix the hyperparameter ε in Eq. (5) as 0.9, and the learning rate for multiplier γ in Algorithm 1 as 0.01.

Table 7: Hyperparameters summary

Data	Model	Focal [30] γ	CB [9] β	LA [32] τ	CSL(ours) τ
Breakfast	MSTCN	0.5	0.9	0.5	0.5
	AsFormer	1.5	0.9	0.1	0.3
	DiffAct	-	0.99	0.3	0.7
50salads	MSTCN	1.5	0.9	0.5	0.7
	AsFormer	0.5	0.99	0.3	0.9
Assembly	MSTCN	0.5	0.9	0.1	0.3
	AsFormer	0.5	0.9	0.3	0.1

E. Additional Results

Global performance. Evaluation in the main paper primarily focuses on per-class performance, as it better reflects the extent to which the long-tail problem is addressed. Since existing works in temporal action segmentation commonly report global performance, we also present detailed global results across different datasets, backbones, and methods in Table 8 for completeness. Notably, our method, which includes constraints for detecting transitions, demonstrates large improvements on global segment-wise metrics, *i.e.*, F1 and edit scores. Although our method may not always lead in frame-wise performance, it still delivers competitive results. Balancing global and balanced results is challenging due to the trade-off: improving tail often boosts per-class results at the expense of head performance, resulting in the drop in global results. Our method achieves a good trade-off by significantly enhancing per-class performance while still showing competitive results on global metrics.

Table 8: Result summary on global metrics.

Model	Breakfast					50salads					Assembly101				
	F1@{10,25,50}			Edit	Acc.	F1@{10,25,50}			Edit	Acc.	F1@{10,25,50}			Edit	Acc.
MSTCN	63.2	57.9	46.0	66.6	67.7	78.5	75.9	67.0	71.4	81.1	30.8	27.2	20.5	30.1	**39.8**
+ CB [9]	63.6	57.9	45.7	66.8	67.4	77.7	75.5	65.8	71.1	81.0	30.0	26.7	20.2	28.4	39.7
+ LA [53]	63.1	57.9	45.6	67.2	67.6	78.2	75.2	66.9	70.4	80.8	29.4	26.1	20.0	29.2	39.2
+ Focal [41]	63.1	57.5	45.5	67.3	**68.5**	78.8	76.1	67.6	70.8	81.7	30.6	27.0	20.0	30.7	39.2
+ τ-norm [24]	62.4	57.0	45.1	66.3	67.9	77.7	75.3	66.5	70.8	81.1	31.1	27.4	20.7	30.5	39.6
+ ours(S-NCM)	**69.3**	**64.0**	**50.9**	**67.7**	67.5	**81.3**	**79.0**	**70.2**	**74.0**	**81.8**	**32.9**	**29.5**	**22.8**	**30.8**	39.1
ASFormer	75.5	69.9	56.1	74.5	**72.4**	84.8	82.3	75.1	79.0	85.2	34.4	30.4	21.5	31.8	41.1
+ CB [9]	75.6	69.7	55.8	74.9	71.9	84.9	83.2	75.7	78.7	85.8	32.6	28.2	20.1	30.6	41.0
+ LA [53]	75.6	69.7	56.3	74.9	**72.4**	84.9	83.2	76.3	78.3	85.3	32.3	28.5	20.9	30.2	**41.3**
+ Focal [41]	**75.7**	**70.4**	56.2	**75.2**	72.3	85.7	83.5	75.7	79.6	84.6	34.1	30.3	22.4	32.1	41.2
+ τ-norm [24]	74.9	69.1	55.7	73.6	72.2	84.7	82.2	75.2	78.9	85.2	26.4	22.7	15.9	24.3	38.5
+ ours(S-NCM)	75.3	**70.4**	**57.5**	74.3	72.1	**86.0**	**84.0**	**77.8**	**80.3**	**86.0**	**34.8**	**31.7**	**23.8**	**32.9**	40.8

Transition detection. The transition constraints help focus on learning hard transitions. Fig. 7 presents the distribution of transition accuracy, as defined in Eq. (3) on Breakfast test set. Transitions are sorted according to the baseline performance. The results indicate that the model trained under the defined the constraints can detect more transitions, particularly in the tail section where the baseline model struggles to recognise them, demonstrating the efficacy of our transition constraints. Specifically, the baseline Asformer can detect 132 out of 167 transitions, while our cost-sensitive learning(CSL) method successfully detects 11 more transitions. Besides, our method achieves higher average transition accuracy 56.1% than the baseline 54.3%.

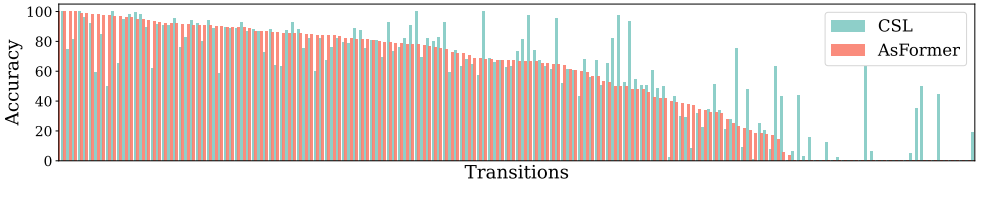


Figure 7: Transition accuracy for AsFormer on Breakfast testset.

References

[1] Nadine Behrmann, S Alireza Golestaneh, Zico Kolter, Juergen Gall, and Mehdi Noroozi. Unified fully and timestamp supervised temporal action segmentation via sequence to sequence translation. In *European Conference on Computer Vision*, pages 52–68. Springer, 2022.

[2] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 106:249–259, 2018.

[3] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International conference on machine learning*, pages 872–881. PMLR, 2019.

[4] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

[5] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[6] Robert S Chen, Brendan Lucier, Yaron Singer, and Vasilis Syrgkanis. Robust optimization for non-convex objectives. *Advances in Neural Information Processing Systems*, 30, 2017.

[7] Guillem Collell, Drazen Prelec, and Kaustubh Patil. Reviving threshold-moving: a simple plug-in bagging ensemble for binary and multiclass imbalanced data. *arXiv preprint arXiv:1606.08698*, 2016.

[8] Andrew Cotter, Heinrich Jiang, Maya R Gupta, Serena Wang, Taman Narayan, Seungil You, and Karthik Sridharan. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *J. Mach. Learn. Res.*, 20(172): 1–59, 2019.

[9] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277, 2019.

[10] Guodong Ding, Fadime Sener, and Angela Yao. Temporal action segmentation: An analysis of modern technique. *arXiv preprint arXiv:2210.10352*, 2022.

[11] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, pages 1–8, 2003.

[12] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6824–6835, 2021.

[13] Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3575–3584, 2019.

[14] Shang-Hua Gao, Qi Han, Zhong-Yu Li, Pai Peng, Liang Wang, and Ming-Ming Cheng. Global2local: Efficient structure search for video action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16805–16814, 2021.

[15] Dayoung Gong, Joonseok Lee, Deunsol Jung, Suha Kwak, and Minsu Cho. Activity grammars for temporal action segmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

[16] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[17] Yin-Yin He, Peizhen Zhang, Xiu-Shen Wei, Xiangyu Zhang, and Jian Sun. Relieving long-tailed instance segmentation via pairwise class balance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7000–7009, 2022.

[18] Ting-I Hsieh, Esther Robb, Hwann-Tzong Chen, and Jia-Bin Huang. Droploss for long-tail instance segmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 1549–1557, 2021.

[19] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5375–5384, 2016.

[20] Yuchi Ishikawa, Seito Kasai, Yoshimitsu Aoki, and Hirokatsu Kataoka. Alleviating over-segmentation errors by detecting action boundaries. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2322–2331, 2021.

[21] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[22] Byungju Kim and Junmo Kim. Adjusting decision boundary for class imbalanced learning. *IEEE Access*, 8:81674–81685, 2020.

[23] Gary King and Langche Zeng. Logistic regression in rare events data. *Political analysis*, 9(2):137–163, 2001.

[24] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 780–787, 2014.

[25] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.

[26] Peng Lei and Sinisa Todorovic. Temporal deformable residual networks for action segmentation in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6742–6751, 2018.

[27] Shi-Jie Li, Yazan AbuFarha, Yun Liu, Ming-Ming Cheng, and Juergen Gall. Ms-tcn++: Multi-stage temporal convolutional network for action segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020. doi: 10.1109/TPAMI.2020.3021756.

[28] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10991–11000, 2020.

[29] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7083–7093, 2019.

[30] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[31] Daochang Liu, Qiyue Li, Anh-Dung Dinh, Tingting Jiang, Mubarak Shah, and Chang Xu. Diffusion action segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10139–10149, 2023.

[32] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*, 2020.

[33] Harikrishna Narasimhan and Aditya K Menon. Training over-parameterized models with non-decomposable objectives. *Advances in Neural Information Processing Systems*, 34:18165–18181, 2021.

[34] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952, 2017.

[35] Toby Perrett, Saptarshi Sinha, Tilo Burghardt, Majid Mirmehdi, and Dima Damen. Use your head: Improving long-tail video recognition. *arXiv preprint arXiv:2304.01143*, 2023.

[36] Harsh Rangwani, Sho Takemori, Kato Takashi, Yuhei Umeda, et al. Cost-sensitive self-training for optimizing non-decomposable metrics. *Advances in Neural Information Processing Systems*, 35:26994–27007, 2022.

[37] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR, 2018.

[38] Dvir Samuel and Gal Chechik. Distributional robustness loss for long-tail learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9495–9504, 2021.

[39] Dvir Samuel, Yuval Atzmon, and Gal Chechik. From generalized zero-shot learning to long-tail with class descriptors. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 286–295, 2021.

[40] Fadime Sener, Dipika Singhania, and Angela Yao. Temporal aggregate representations for long-range video understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 154–171. Springer, 2020.

[41] Fadime Sener, Dibyadip Chatterjee, Daniel Shelepov, Kun He, Dipika Singhania, Robert Wang, and Angela Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21096–21106, 2022.

[42] Li Shen, Zhouchen Lin, and Qingming Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*, pages 467–482. Springer, 2016.

[43] Dipika Singhania, Rahul Rahaman, and Angela Yao. Coarse to fine multi-resolution temporal convolutional network. *arXiv preprint arXiv:2105.10859*, 2021.

[44] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[45] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738, 2013.

[46] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11662–11671, 2020.

[47] Jiaqi Wang, Wenwei Zhang, Yuhang Zang, Yuhang Cao, Jiangmiao Pang, Tao Gong, Kai Chen, Ziwei Liu, Chen Change Loy, and Dahua Lin. Seesaw loss for long-tailed instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9695–9704, 2021.

[48] Tong Wang, Yousong Zhu, Chaoyang Zhao, Wei Zeng, Jinqiao Wang, and Ming Tang. Adaptive class suppression loss for long-tail object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3103–3112, 2021.

[49] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. *Advances in neural information processing systems*, 30, 2017.

[50] Zhenzhi Wang, Ziteng Gao, Limin Wang, Zhifeng Li, and Gangshan Wu. Boundary-aware cascade networks for temporal action segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 34–51. Springer, 2020.

[51] Ziwei Xu, Yogesh Rawat, Yongkang Wong, Mohan S Kankanhalli, and Mubarak Shah. Don't pour cereal into coffee: Differentiable temporal logic for temporal action segmentation. *Advances in Neural Information Processing Systems*, 35:14890–14903, 2022.

[52] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. *arXiv preprint arXiv:2110.08568*, 2021.

[53] Junjie Zhang, Lingqiao Liu, Peng Wang, and Chunhua Shen. To balance or not to balance: A simple-yet-effective approach for learning with long-tailed distributions. *arXiv preprint arXiv:1912.04486*, 2019.

[54] Xing Zhang, Zuxuan Wu, Zejia Weng, Huazhu Fu, Jingjing Chen, Yu-Gang Jiang, and Larry S Davis. Videolt: large-scale long-tailed video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7960–7969, 2021.

[55] Yan Zhao, Weicong Chen, Xu Tan, Kai Huang, and Jihong Zhu. Adaptive logit adjustment loss for long-tailed visual recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3472–3480, 2022.