CLUSTERING-ASSISTED FOREGROUND AND BACKGROUND SEPARATION FOR WEAKLY-SUPERVISED TEMPORAL ACTION LOCALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Weakly-supervised temporal action localization aims to localize action instances in videos with only video-level action labels. Existing methods mainly embrace a localization-by-classification pipeline that optimizes snippet-level prediction with a video classification loss. However, this formulation suffers from the discrepancy between classification and detection, resulting in the noisy foreground and background (F&B) snippets separation. To alleviate this problem, we propose to explore the underlying structure among the snippets by unsupervised snippet clustering, rather than heavily relying on the video classification loss. Specifically, we propose a novel clustering-assisted F&B separation network dubbed CASE, which achieves F&B separation by two core components: a snippet clustering component that groups the snippets into multiple latent clusters and a cluster classification component that attempts to further classify the cluster as foreground or background. In the absence of ground-truth labels to train these two components, we propose to adopt an online self-training algorithm that allows online interaction of pseudo-label rectification and model training. More importantly, we propose a distribution-constrained labeling strategy that utilizes different priors to regularize the distribution of the pseudo-labels, so as to reinforce the quality of the pseudo-labels. With the aid of the online self-training algorithm and distribution-constrained labeling strategy, our method is able to exploit the latent clusters that are simultaneously typical to snippets and discriminative to F&B. Thereby, the cluster assignments of the snippets can be associated with their F&B labels to enable the F&B separation. The effectiveness of the proposed CASE is demonstrated by the experimental results on three publicly available benchmarks: THUMOS14, ActivityNet v1.2 and v1.3.

1 INTRODUCTION

Temporal action localization (TAL) (Shou et al., 2016) is a task to localize the temporal boundaries of action instances and recognize their categories in videos. In recent years, numerous works put effort into the fully supervised manner and gain great achievements. Albeit successful, these methods require extensive manual frame-level annotations, which is expensive and time-consuming. Without the requirement of frame-level annotations, weakly-supervised TAL (WTAL) has received increasing attention, as it allows us to detect the action instances with only video-level action labels.

There has been a large spectrum of WTAL methods developed in the literature. With only video-level labels, mainstream methods employ a localization-by-classification pipeline, which formulates WTAL as a video action classification problem to learn a temporal class activation sequence (T-CAS). Under this pipeline, the foreground (*i.e.*, action) and background separation remains an open question, since the video-level labels have no cue for background class. There are two types of approaches to solve it. The first type (Wang et al., 2017; Zhang et al., 2021) is based on the multiple instance learning (MIL), which uses the T-CAS to select the most confident snippets for each action class. The second type (Nguyen et al., 2019; Liu et al., 2019) introduces an attention mechanism to learn class-agnostic foreground weights that indicates the probabilities of the snippets belonging to foreground. Despite recent progress, the methods typically rely on the video classification loss to guide the learning of the T-CAS or the attention weights. There is an inherent drawback that the loss is easily minimized by the salient snippets (Min & Corso, 2020) and fails to explore the distribution

Figure 1: Conceptual illustration of CASE. In snippet clustering, we divide the snippets (or frames) into multiple clusters with explicit characteristics. In cluster classification, we classify the clusters as foreground or background.

of the whole snippets, resulting in erroneous T-CAS or attention weights. This issue is rooted in the supervision gap between the classification and detection tasks. There are some attempts (Pardo et al., 2021; Luo et al., 2020) proposed to generate snippet-level pseudo-labels to bridge the gap. The pseudo-labels, however, are derived from the unreliable T-CAS or attention weights.

In the literature, deep clustering (Chang et al., 2017), which automatically partitions the samples into different groups, is proven to be capable of revealing the intrinsic distribution of the samples in many label-scarce tasks (Asano et al., 2019; Caron et al., 2020; Fini et al., 2021; Li et al., 2022b). A natural issue arises: is it possible to adopt the clustering to capture the distribution of the snippets? Since the clustering can be conducted in a self-supervised manner, it is immune to the video classification loss. This indicates the great potential of clustering technique in WTAL, especially in the challenging F&B separation. With the aim of F&B separation, a naive solution is to group the snippets into two clusters (one for foreground and the other for background). Whereas, we empirically find that it doesn't work well (*cf.* Sec. 4.3). We argue that the reason that accounts for the failure is that the snippets, regardless of the foreground or background, differ dramatically in appearance (*cf.* Fig. 1 (a)). As a result, it may be difficult for a self-supervised model to group them together. Fortunately, in the real-world videos, there may be common characteristics (*e.g.*, 'interview', 'running') shared by a group of snippets (*cf.* Fig. 1 (b)). Compared with learning two clusters for F&B in the complex video content, it may be easier to explore the snippet clusters with clear and distinctive characteristics, which can be achieved by a clustering algorithm with multiple clusters. Furthermore, it can be observed that the characteristics of clusters are sometimes indicative cues to F&B separation. For example, we can confidently classify the 'running' cluster to foreground and the 'interview' cluster to background according to cluster-level characteristics. Consequently, it is promising to further take advantage of the cluster-level representations to assist F&B separation.

In light of the above discussion, we propose a **C**lustering-**A**ssisted F&B **SE**paration (CASE) network. Specifically, we first build a standard WTAL baseline that provides a primary estimation of the F&B snippets. Then a clustering-based F&B separation algorithm (*cf.* Fig. 1) is introduced to refine the F&B separation. At the heart of this algorithm lies a snippet clustering component for dividing the snippets into multiple clusters, alongside a cluster classification component for classifying the clusters as foreground or background. Considering there is no ground-truth label available to train the components, we develop an online self-training algorithm that optimizes the pseudo-labels and the model in an online fashion. More importantly, it is desired that the learned clusters are both *typical* to the snippets and *discriminative* to F&B classes. By 'typical', we mean that each snippet is assigned to one and only one cluster, and each cluster contains a considerable number of snippets. Such a property will force the model to comprehensively characterize the distribution of the snippets. To this end, we devise a distribution-constrained strategy to impose additional constraints on the distribution of pseudo-labels. Specifically, for the snippet clustering, we impose an equipartition constraint on the marginal distribution of the snippet-level labels so that the cluster assignments of the snippets are diverse. Besides, we leverage the baseline to form a dynamic prior distribution for the labels so as to make the assignments more confident. As for the cluster classification, we propose to enforce the marginal distribution of cluster-level F&B labels to be consistent with that of snippet-level F&B labels, thereby improving the discrimination of clusters to F&B. After training the two components, we can transform the cluster assignments of the snippets to their F&B assignments, which can be used to refine the F&B separation of the baseline.

In summary, the contributions of this paper are as follows: 1) We propose a clustering-assisted F&B separation network named CASE for WTAL, which casts the problem of F&B separation as

a combination of the snippet clustering and cluster classification. 2) We propose an online self-training algorithm accompanied by a distribution-constrained labeling strategy to guide the snippet clustering and cluster classification. 3) Extensive experiments manifest that the proposed CASE achieves significant performance improvements compared to state-of-the-art approaches.

2 Related Work

Deep Visual Clustering. Traditional clustering methods, such as K-Means (MacQueen et al., 1967), are limited to low-dimensional data. With the aid of deep learning (LeCun et al., 2015), deep clustering methods are proposed to embed the original data in a lower-dimensional embedding space. Current approaches (Yang et al., 2019; 2020; Dang et al., 2020) could be roughly divided into two categories: The first one iteratively computes the clustering assignment from the up-to-date model and supervise the network training processes by the estimated information (Xie et al., 2016; Yang et al., 2016; Chang et al., 2017; Caron et al., 2018; Chang et al., 2019; Wu et al., 2019). The second one simultaneously learns both the feature representation and clustering assignment (Haeusser et al., 2018; Ji et al., 2019; Huang et al., 2020a) . Asano *et al.* (Asano et al., 2019) transform the cluster assignment problem to an optimal transport problem which can be solved efficiently through the Sinkhorn-Knopp algorithm. Caron *et al.* (Caron et al., 2020) use the algorithm of (Asano et al., 2019) to introduce a swapped mechanism that uses two random transformations of the same images to guide each other. In this work, we extend (Asano et al., 2019) from image classification to WTAL. More importantly, we incorporate it with task-specific labeling strategies.

Weakly-Supervised Temporal Action Localization. UntrimNet (Wang et al., 2017) is the pioneering work for WTAL, which is widely used as the basis for later methods. Based on the specific designs, most existing approaches fall into four groups. The first group aims to improve feature discrimination. Deep metric learning is explored in (Min & Corso, 2020; Narayan et al., 2019) to encourage intra-class compactness and inter-class dispersion. Recent works (Zhang et al., 2021; Li et al., 2022a) use contrastive learning to refine the snippet representations. The second group seeks to discover complete action regions. (Min & Corso, 2020; Singh & Lee, 2017; Zhong et al., 2018) propose to remove the discriminative parts or randomly hide snippets to press the models in exploring more action regions. (Liu et al., 2019; Islam et al., 2021) design a multi-branch framework to discover complementary snippets. The third group focuses on the learning of attention weights. (Zhai et al., 2020; Nguyen et al., 2018) design the losses to regularize the values of the attention weights. Recently, efforts are made in (Pardo et al., 2021; Luo et al., 2020) to generate pseudo-labels for the attention weight training. Despite the success of these methods, their pseudo-labels are derived from the primary predictions that need to be optimized using the video classification loss. Yet, we generate the cluster pseudo-labels of snippets via self-labeling that is essentially independent of the video classification loss. The last group is the most closely related to ours. These methods (Liu et al., 2021; Luo et al., 2021; Huang et al., 2021a; Wang et al., 2021) introduce auxiliary classes in addition to the action classes and background class. (Liu et al., 2021) proposes to learns two feature sub-spaces respectively for the actions and the contexts. (Luo et al., 2021; Wang et al., 2021; Liu et al., 2019) propose to mine the "action units" or "sub-actions" that are shared by different action categories. Class-specific sub-actions are also explored in (Huang et al., 2021b;a). Our proposed method is superior to these methods in three noticeable aspects. 1) The learning of the auxiliary classes in these methods is driven by the video classification loss. Yet, we develop the clusters in a self-supervised manner, which is complementary to the video-level supervision. 2) These methods devise multiple loss terms to regularize the auxiliary classes. In contrast, we introduce the regularization in pseudo-labeling, which can be resolved in a principled way. 3) Our method significantly outperforms these methods by a large margin. Recently, Huang *et al.* (Huang et al., 2022) propose to maintain an asynchronous memory bank for storing the class-wise representative snippets derived from each video. On the contrary, we use an end-to-end clustering head to learn several clusters.

3 CASE

In this section, we elaborate on the proposed method, namely CASE, which is depicted in Fig. 2.

3.1 Notations and Baseline

In each training iteration, we first sample a batch of videos $\{\boldsymbol{V}_b\}_{b=1}^{B}$ with a batch size of B. For each video V_b, we can only access its video label $\bar{\boldsymbol{Y}}_b \in \mathbb{R}^{K^V}$, where K^V is the number of action classes.

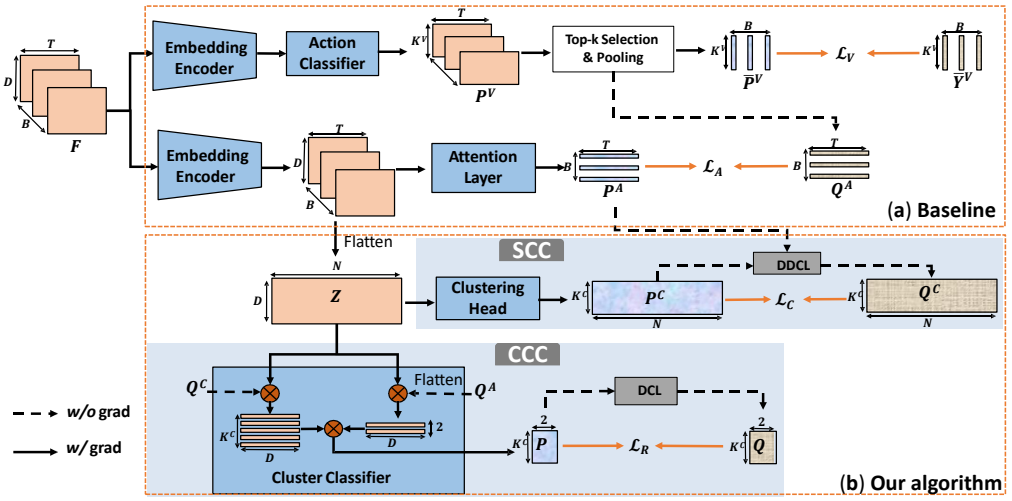

Figure 2: The framework of CASE. The upper box (a) shows the baseline model, which consists of a video classification branch and an attention branch. The bottom box (b) presents our proposed clustering-based F&B separation algorithm. It is comprised of a snippet clustering component (SCC) and a cluster classification component (CCC), where the dynamic distribution-constrained labeling (DDCL) and distribution-constrained labeling (DCL) are respectively employed.

Afterwards, a sequence of T snippets is sampled from each video, and RGB features $\boldsymbol{F}_b^{\text{RGB}} \in \mathbb{R}^{T\times D}$ and optical-flow features $\boldsymbol{F}_b^{\text{Flow}} \in \mathbb{R}^{T\times D}$ are extracted with pre-trained feature extractors. Here D is the channel dimension. Following (Zhai et al., 2020; Yang et al., 2021), we use an RGB stream and an optical-flow stream to process $\boldsymbol{F}_b^{\text{RGB}}$ and $\boldsymbol{F}_b^{\text{Flow}}$, respectively. Unless otherwise specified, we only illustrate one of them and omit the superscripts of 'RGB' and 'Flow' in the rest of this paper.

For the baseline, following (Wang et al., 2017), we deploy a two-branch framework that contains a video classification branch and an attention branch. In the former branch, we first feed the input features $\boldsymbol{F}_b$ to an embedding encoder, and then pass the resulting embeddings into an action classifier with K^V classes to get the T-CAS dubbed $\boldsymbol{P}_b^{\boldsymbol{V}} \in \mathbb{R}^{T\times K^V}$. In the latter branch, $\boldsymbol{F}_b$ is first passed through another embedding encoder to obtain the snippet embeddings, and the embeddings are then sent to a one-dimension attention layer to extract class-agnostic attention weights dubbed $\boldsymbol{P}_b^{\boldsymbol{A}} \in \mathbb{R}^T$ that represent the foreground probabilities of the snippets.

Here, we apply the widely-used MIL to train the video classification branch. Briefly (see Appendix E.1 for details), we first calibrate T-CAS with the attention weights to suppress background snippets. Then we select k snippets with highest activations of each class to build the video scores $\bar{\boldsymbol{P}}_b \in \mathbb{R}^{K^V}$. Finally, we optimize a video classification loss with the known video labels $\bar{\boldsymbol{Y}}_b$:

$$\mathcal{L}_V = -\frac{1}{B}\sum_{b=1}^{B}\sum_{k=1}^{K^V} \bar{\boldsymbol{Y}}_{b,k} \cdot \log \bar{\boldsymbol{P}}_{b,k}. \tag{1}$$

Besides, in order to train the attention branch, we opt for the pseudo-label-based scheme proposed by (Ma et al., 2021) due to its conciseness and effectiveness. Specifically, we define the foreground pseudo-labels $\boldsymbol{Q}_b^{\boldsymbol{A}} \in \mathbb{R}^T$ as: the union of the selected snippets corresponding to the classes presented in $\bar{\boldsymbol{Y}}_b$ are positive (*i.e.*, 1), and the other snippets are negative (*i.e.*, 0). To improve the robustness of the model against label noise, we train this branch with the generalized binary cross-entropy loss (Zhang & Sabuncu, 2018) that softens the penalty in regions of high disagreement:

$$\mathcal{L}_A = \frac{1}{N_{\text{pos}}}\sum_{b=1}^{B}\sum_{t=1}^{T} \mathbb{1}_{[\boldsymbol{Q}_{b,t}^{\boldsymbol{A}}=1]} \cdot \frac{1-(\boldsymbol{P}_{b,t}^{\boldsymbol{A}})^{\gamma}}{\gamma} + \frac{1}{N_{\text{neg}}}\sum_{b=1}^{B}\sum_{t=1}^{T} \mathbb{1}_{[\boldsymbol{Q}_{b,t}^{\boldsymbol{A}}=0]} \cdot \frac{1-(1-\boldsymbol{P}_{b,t}^{\boldsymbol{A}})^{\gamma}}{\gamma}, \tag{2}$$

where $\mathbb{1}_{[*]}$ is an indicator function. $\gamma \in (0,1)$ controls the noise tolerance. N_{pos} and N_{neg} represents that number of positive snippets and negative snippets in the batch.

3.2 Clustering-based Foreground and Background Separation

The preceding baseline has two main drawbacks: 1) It is biased by the video classification loss and thus is unable to reveal the true distribution of the snippets. 2) It uses a binary classifier to directly

separate F&B snippets, bearing no intra-class variation. To overcome the drawbacks, we propose the clustering-based F&B separation algorithm, which realizes F&B separation by a snippet clustering component and a cluster classification component.

3.2.1 SNIPPET CLUSTERING

To allow mutual promotion of the learning of the attention layer and snippet clustering, we append the snippet clustering component (SCC) over the embeddings in the attention branch. For notation simplicity, we use N to denote the total number of snippets within a batch and $N = BT$. Thereafter, we denote the snippet embeddings by $Z \in \mathbb{R}^{N \times D}$. To obtain the cluster assignments of the snippets dubbed $\boldsymbol{P^C} \in \mathbb{R}^{N \times K^C}$, we feed Z into a clustering head composed of a classifier with K^C classes. K^C is the predefined number of clusters. Without any label to train the cluster head, we adopt an online self-training algorithm that simultaneously optimizes the cluster assignments $\boldsymbol{P^C}$ and the pseudo-labels $\boldsymbol{Q^C} \in \mathbb{R}^{N \times K^C}$. Formally, it is achieved by minimizing following objective:

$$\mathcal{L}_C = -\sum_{n=1}^{N}\sum_{k=1}^{K^C} \boldsymbol{Q^C}_{n,k} \cdot \log \boldsymbol{P^C}_{n,k} = \langle \boldsymbol{Q^C}, -\log \boldsymbol{P^C} \rangle \tag{3}$$

where $\langle * \rangle$ stands for Frobenius dot-product. Minimizing $\mathcal{L}_C$ *w.r.t* $\boldsymbol{Q^C}$ is equivalent to pseudo-labeling. Minimizing $\mathcal{L}_C$ *w.r.t* $\boldsymbol{P^C}$ indicates minimizing a cross-entropy loss that optimizes the parameters of the model. These two processes are alternated within each iteration so that $\boldsymbol{Q^C}$ can co-evolve with $\boldsymbol{P^C}$. However, merely minimizing Eq. 3 may lead to a degenerate solution (Asano et al., 2019): most snippets are assigned to only a few clusters (termed imbalanced assignment issue). To tackle the issue, we propose a distribution-constrained labeling (DCL) strategy to impose constraints on the distribution of the pseudo-labels. Formally, we restrict $\boldsymbol{Q^C}$ to be an element of transportation polytopes (Cuturi, 2013):

$$\boldsymbol{Q^C} \in \mathcal{Q}^C, \ \mathcal{Q}^C := \{\boldsymbol{Q^C} \in \mathbb{R}_+^{N \times K^C} | \boldsymbol{Q^C} \mathbf{1}^{K^C} = \boldsymbol{\alpha^C}, \boldsymbol{Q^C}^\top \mathbf{1}^N = \boldsymbol{\beta^C}\}, \tag{4}$$

where $\mathbf{1}^N$ denotes the vectors of all ones of dimension N. $\boldsymbol{\alpha^C} \in \mathbb{R}^N$ and $\boldsymbol{\beta^C} \in \mathbb{R}^{K^C}$ are the marginal projections of $\boldsymbol{Q^C}$ onto its rows and columns, respectively. Since the pseudo-label of each sample belongs to a probability distribution, it is obvious that $\boldsymbol{\alpha^C} = \mathbf{1}^N$. $\boldsymbol{\beta}_k^{\boldsymbol{C}}$ represents the proportions of the snippets assigned to the k-th cluster. Here, we employ the equipartition constraint (Asano et al., 2019; Caron et al., 2020; Fini et al., 2021), *i.e.*,

$$\boldsymbol{\beta^C} = \frac{N}{K^C} \mathbf{1}^{K^C}. \tag{5}$$

It indicates that each cluster is assigned with the same number of snippets, thus encouraging the assigned labels to be diverse and preventing the imbalanced assignment issue. Without any other prior knowledge, equipartition is a good inductive bias from the perspective of Occam's razor, since it is one of the simplest possible behaviors. Then minimizing $\mathcal{L}_C$ *w.r.t* $\boldsymbol{Q^C}$ becomes an instance of optimal transport problem (Cuturi, 2013; Asano et al., 2019). Computationally, it is quite expensive to solve it. Following (Cuturi, 2013), we introduce an entropy term to the objective:

$$\min_{\boldsymbol{Q^C} \in \mathcal{Q}^C} \langle \boldsymbol{Q^C}, -\log \boldsymbol{P^C} \rangle - \frac{1}{\epsilon} \mathrm{H}(\boldsymbol{Q^C}), \tag{6}$$

where $\mathrm{H}(\boldsymbol{Q^C}) = -\sum_{n,k} \boldsymbol{Q}_{n,k}^{\boldsymbol{C}} \cdot \log \boldsymbol{Q}_{n,k}^{\boldsymbol{C}}$ is the entropy of $\boldsymbol{Q^C}$. ϵ is a hyper-parameter. The advantage of this regularization term is that the solver of Eq. 6 can be given as:

$$\boldsymbol{Q^C} = \mathrm{diag}(\boldsymbol{u})(\boldsymbol{P^C})^\epsilon \mathrm{diag}(\boldsymbol{v}), \tag{7}$$

where $\boldsymbol{u}$ and $\boldsymbol{v}$ are two renormalization vectors that can be computed by the Sinkhorn-Knopp algorithm (Cuturi, 2013). The algorithm is highly efficient on GPU as it only involves a couple of matrix multiplications, enabling online computation. Given the optimal $\boldsymbol{Q^C}$, we then minimize $\mathcal{L}_C$ *w.r.t* $\boldsymbol{P^C}$, which also optimizes the feature encoder and the attention layer.

3.2.2 CLUSTER CLASSIFICATION

It is desired that the clusters are discriminative to F&B separation. But so far, there is no explicit connection between the clustering and F&B separation. To mitigate the gap, the cluster classification component (CCC) is proposed to enforce each cluster to be classified into foreground class or

background class. It turns out to be an unsupervised classification problem. Motivated by prototype-based metric learning (Snell et al., 2017; Chen et al., 2021), we propose to express the clusters and the F&B classes by their prototypes (*i.e.*, the class-wise centers of the embeddings). Then a cluster classifier is constructed by comparing the similarities between the cluster prototypes and F&B prototypes. Specifically, we attain the prototype of each cluster by pseudo-labels as:

$$\bar{\boldsymbol{Z}}_k^{\boldsymbol{C}} = \frac{\sum_{n=1}^{N} \boldsymbol{Q}_{n,k}^{\boldsymbol{C}} \cdot \boldsymbol{Z}_n}{\sum_{n=1}^{N} \boldsymbol{Q}_{n,k}^{\boldsymbol{C}}}, k \in \{1, 2, ..., K^C\}. \tag{8}$$

As for the F&B classes, we respectively calculate their prototypes as:

$$\bar{\boldsymbol{Z}}_k^{\boldsymbol{F}} = \frac{\sum_{n=1}^{N} \boldsymbol{Q}_{n,k}^{\boldsymbol{A}} \cdot \boldsymbol{Z}_n}{\sum_{n=1}^{N} \boldsymbol{Q}_{n,k}^{\boldsymbol{A}}}, \bar{\boldsymbol{Z}}_k^{\boldsymbol{B}} = \frac{\sum_{n=1}^{N} (1 - \boldsymbol{Q}_{n,k}^{\boldsymbol{A}}) \cdot \boldsymbol{Z}_n}{\sum_{n=1}^{N} (1 - \boldsymbol{Q}_{n,k}^{\boldsymbol{A}})}, \tag{9}$$

where $\boldsymbol{Q}_{n,k}^{\boldsymbol{A}}$ is the foreground pseudo-labels defined in Sec. 3.1. Then the relative similarity between the cluster prototype and the foreground prototype is calculated as:

$$\boldsymbol{P}_k^{\boldsymbol{F}} = \frac{\exp(\rho \cdot \cos(\bar{\boldsymbol{Z}}_k^{\boldsymbol{C}}, \bar{\boldsymbol{Z}}_k^{\boldsymbol{F}}))}{\exp(\rho \cdot \cos(\bar{\boldsymbol{Z}}_k^{\boldsymbol{C}}, \bar{\boldsymbol{Z}}_k^{\boldsymbol{F}})) + \exp(\rho \cdot \cos(\bar{\boldsymbol{Z}}_k^{\boldsymbol{C}}, \bar{\boldsymbol{Z}}_k^{\boldsymbol{B}}))}, \tag{10}$$

where $\cos(\cdot)$ denotes the cosine similarity function and ρ is the temperature. Obviously, the relative similarity between each cluster prototype and the background prototype is $\boldsymbol{P}_k^{\boldsymbol{B}} = 1 - \boldsymbol{P}_k^{\boldsymbol{F}}$. For the sake of simplicity, we denote the concatenation of $\boldsymbol{P}_k^{\boldsymbol{F}}$ and $\boldsymbol{P}_k^{\boldsymbol{B}}$ of all clusters by $\boldsymbol{P}^{\boldsymbol{R}} \in \mathbb{R}^{K^C \times 2}$, where $\boldsymbol{P}_k^{\boldsymbol{R}} = [\boldsymbol{P}_k^{\boldsymbol{F}}, \boldsymbol{P}_k^{\boldsymbol{B}}]$. Then $\boldsymbol{P}^{\boldsymbol{R}}$ is regarded as the prediction of the classifier. To optimize the classifier, we propose to employ the online self-training algorithm again and yet a revised distribution-constrained strategy. Specifically, the objective function is expressed as:

$$\mathcal{L}_R = \langle \boldsymbol{Q}^{\boldsymbol{R}}, -\log \boldsymbol{P}^{\boldsymbol{R}} \rangle, \;\; s.t.\; \boldsymbol{Q}^{\boldsymbol{R}} \in \mathcal{Q}^R, \mathcal{Q}^R := \{\boldsymbol{Q}^{\boldsymbol{R}} \in \mathbb{R}_+^{K^C \times 2} | \boldsymbol{Q}^{\boldsymbol{R}} \mathbf{1}_2 = \boldsymbol{\alpha}^{\boldsymbol{R}}, {\boldsymbol{Q}^{\boldsymbol{R}}}^\top \mathbf{1}_{K^C} = \boldsymbol{\beta}^{\boldsymbol{R}}\}, \tag{11}$$

where $\boldsymbol{Q}^{\boldsymbol{R}} \in \mathbb{R}^{K^C \times 2}$ is the pseudo-labels. $\boldsymbol{\alpha}^{\boldsymbol{R}} = \mathbf{1}^{K^C}$ is obvious. $\boldsymbol{\beta}^{\boldsymbol{R}} \in \mathbb{R}^2$ represents the proportions of clusters belonging to F&B. Instead of using equipartition constraint (*i.e.*, Eq. 5), we hereby use the distribution of $\boldsymbol{Q}_{n,k}^{\boldsymbol{A}}$ to estimate $\boldsymbol{\beta}^{\boldsymbol{R}}$. The intuition behind it is that each cluster contains approximately the same number of samples thanks to Eq. 5 used in Sec. 3.2.1, making the proportions of F&B clusters close to the proportions of F&B snippets. Hence, $\boldsymbol{\beta}^{\boldsymbol{R}}$ can be approximated as:

$$\boldsymbol{\beta}^{\boldsymbol{R}} = [\frac{1}{N} \sum_{n=1}^{N} \boldsymbol{Q}_n^{\boldsymbol{A}}, \; \frac{1}{N} \sum_{n=1}^{N} (1 - \boldsymbol{Q}_n^{\boldsymbol{A}})]. \tag{12}$$

Compared with Eq. 5, Eq. 12 is closer to the real distribution of the F&B snippets, leading to more discriminative clusters to F&B. Finally, we will get a loss term $\mathcal{L}_R$ in each iteration. With the loss, $\boldsymbol{Q}^{\boldsymbol{R}}$ will quickly converge to a stable status close to one-hot form (*c.f.* Sec. 4.3), indicating that a global and explicit relationship between the clusters and the F&B is established.

3.2.3 Dynamic Distribution-Constrained Labeling

In Eq. 6, an entropy term is introduced to make it tractable with affordable complexity. However, it may also cause a trivial solution in that the samples are assigned to different classes with the same probability. In practice, we find that the issue is serious in SCC, but not in CCC. This may be because the former involves much more instances and classes, rendering the algorithm harder to converge (*cf.* Fig. 3). The trivial solution indicates that the assignments between snippets and clusters are uncertain (termed uncertain assignment issue). To remedy the defect in SCC, it is necessary to replace the entropy term with a more informative one. Inspired by (Su & Hua, 2017), we propose a dynamic distribution-constrained labeling (DDCL) strategy to introduce a dynamic prior distribution for $\boldsymbol{Q}^{\boldsymbol{C}}$, namely, $\hat{\boldsymbol{Q}}^{\boldsymbol{C}} \in \mathbb{R}^{N \times K^C}$. In $\hat{\boldsymbol{Q}}^{\boldsymbol{C}}$, we force that the F&B snippets have relatively high probabilities of belonging to the F&B clusters, respectively. Specifically, we first rank the snippets according to their foreground probabilities (*i.e.*, $\boldsymbol{P}^{\boldsymbol{A}}$) from small to large. Let us denote the normalized ranked orders of the N snippets by $\boldsymbol{R} \in \{\frac{1}{N}, \frac{2}{N}, \ldots, 1\}^N$. Then in $\hat{\boldsymbol{Q}}^{\boldsymbol{C}}$, the snippets with high $\boldsymbol{R}$ are preferred to be assigned to the clusters with high foreground probabilities (*i.e*, $\boldsymbol{Q}_{:,0}^{\boldsymbol{R}}$) and vice versa. Mathematically, we define $\hat{\boldsymbol{Q}}^{\boldsymbol{C}}$ as a 2D Gaussian distribution:

$$\hat{\boldsymbol{Q}}_{nk}^{\boldsymbol{C}} = \frac{1}{\sigma\sqrt{2\pi}} \exp\big(-\frac{|\boldsymbol{R}_n - \boldsymbol{Q}_{k,0}^{\boldsymbol{R}}|^2}{2\sigma^2}\big), \tag{13}$$

where σ is the standard deviation. The reader may consider that $\boldsymbol{P^A}$ is an alternative to $\boldsymbol{R}$. But as shown in Appendix F.2, $\boldsymbol{R}$ is more comparable with $\boldsymbol{Q^R}$ than $\boldsymbol{P^A}$. Thereafter, we replace the Eq. 6 with following objective:

$$\min_{\boldsymbol{Q^C} \in \mathcal{Q}^C} \langle \boldsymbol{Q^C}, -\log \boldsymbol{P^C} \rangle + \frac{1}{\epsilon} \mathrm{KL}(\boldsymbol{Q^C} || \hat{\boldsymbol{Q}}^{\boldsymbol{C}}), \tag{14}$$

where $\mathrm{KL}(\boldsymbol{Q^C} || \hat{\boldsymbol{Q}}^{\boldsymbol{C}})$ is the Kullback-Leibler (KL) divergence between $\boldsymbol{Q^C}$ and $\hat{\boldsymbol{Q}}^{\boldsymbol{C}}$. Compared with the original entropy term, the advantage of the KL term is that it can use the confident $\boldsymbol{Q^R}$ to improve the confidence of $\boldsymbol{Q^C}$. Further taking the constraint of Eq. 4 into account, we can derive the solution for Eq. 14 as follows (please refer to Appendix H for the derivation):

$$\boldsymbol{Q^C} = \mathrm{diag}(\boldsymbol{u})(\hat{\boldsymbol{Q}}^{\boldsymbol{C}} \cdot (\boldsymbol{P^C})^{\epsilon}) \, \mathrm{diag}(\boldsymbol{v}). \tag{15}$$

Eq. 15 can also be efficiently computed by the Sinkhorn-Knopp algorithm. It is worth noting that our purpose is significantly different from (Su & Hua, 2017), a method for sequence matching. It is originally designed to enforce that only elements with similar temporal positions on two sequences are matched. On the contrary, we aim to inherit the basic information from the attention layer to enhance the typicality of the clusters.

3.3 Training and Testing

Two-stream Co-Labeling. In our framework, there are several procedures of pseudo-labeling that can be summarized with a unified formulation as $\boldsymbol{Q} = \Psi(\boldsymbol{P})$. Here $\boldsymbol{P}$ is the prediction of the model, Ψ is the function of generating pseudo-labels, $\boldsymbol{Q}$ is the pseudo-labels. To improve the quality of the pseudo-labels, inspired by (Zhai et al., 2020), we propose the two-stream co-labeling strategy. Specifically, we aggregate the predictions of RGB and optical-flow streams to generate the modality-sharing pseudo-labels, *i.e.*, $\boldsymbol{Q} = \Psi(0.5\boldsymbol{P}^{\mathrm{RGB}} + 0.5\boldsymbol{P}^{\mathrm{Flow}})$. We refer to Appendix E.2 for details.

Joint Training. We train all the components jointly in an end-to-end fashion. The overall training objective can be written as:

$$\mathcal{L} = (\mathcal{L}_V + \lambda_A \mathcal{L}_A) + \lambda_C \mathcal{L}_C + \lambda_R \mathcal{L}_R, \tag{16}$$

where λ_* represents the weight. As the attention layer and our proposed algorithm share the same embedding encoder, the joint training also improves the capacity of the attention layer (*cf.* Sec.4.3).

Clustering-Assisted Testing. Previous works (Zhang et al., 2021; Qu et al., 2021) have shown that the foreground probability from the attention layer (*i.e.*, $\boldsymbol{P^A} \in \mathbb{R}^N$) is helpful to localize action instances in the test phase. With the probabilities that the snippets belong to the clusters and the probabilities that the clusters belong to the foreground, we can obtain another foreground probabilities of the snippets dubbed $\hat{\boldsymbol{P}}^{\boldsymbol{A}} \in \mathbb{R}^{\boldsymbol{N}}$ (other than $\boldsymbol{P^A}$) based on the *law of total probability*:

$$\hat{\boldsymbol{P}}^{\boldsymbol{A}}_n = \sum_k^{K^C} \hat{\boldsymbol{P}}^{\boldsymbol{C}}_{n,k} \cdot \boldsymbol{Q}^{\boldsymbol{R}}_{:,0} \quad \textit{i.e.,} \quad \hat{\boldsymbol{P}}^{\boldsymbol{A}} = \boldsymbol{P^C} \boldsymbol{Q}^{\boldsymbol{R}}_{:,0}. \tag{17}$$

$\boldsymbol{Q^R}$ is computed within each batch in the training phase, which is infeasible in the test phase. Considering that $\boldsymbol{Q^R}$ is stable during training, we simply use $\boldsymbol{Q^R}$ of the last training iteration for testing. Moreover, as shown in Table 5, $\boldsymbol{P^A}$ and $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ are complementary, thus we fuse $\boldsymbol{P^A}$ and $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ by convex combination: $\ddot{\boldsymbol{P}}^{\boldsymbol{A}} = 0.5\boldsymbol{P^A} + 0.5\hat{\boldsymbol{P}}^{\boldsymbol{A}}$. $\ddot{\boldsymbol{P}}^{\boldsymbol{A}}$ can be therefore used for testing.

4 Experiments

4.1 Datasets and Evaluation Metric

THUMOS14 (Jiang et al., 2014) contains untrimmed videos with 20 classes. We use the 200 videos in validation set for training and the 213 videos in testing set for evaluation. **ActivityNet** (Caba Heilbron et al., 2015) has two release versions, *i.e.*, **ActivityNet v1.3** and **ActivityNet v1.2**. ActivityNet v1.3 covers 200 action categories, with a training set of 10, 024 videos and a validation set of 4, 926 videos. ActivityNet v1.2 is a subset of ActivityNet v1.3, and covers 100 action categories, with 4, 819 and 2, 383 videos in the training and validation set, respectively. We follow the standard evaluation protocol by reporting mean Average Precision (mAP) values under different temporal intersection over union (tIoU) thresholds.

Table 1: Comparisons of performance on THUMOS14. The AVG(0.1:0.5), AVG(0.3:0.7), AVG represent the average mAP under IoU thresholds of 0.1:0.5, 0.3:0.7 and 0.1:0.7, respectively.

Supervision	Method	mAP @ IoU (%)							AVG (0.1:0.5)	AVG (0.3:0.7)	AVG
		0.1	0.2	0.3	0.4	0.5	0.6	0.7			
Full	S-CNN (Shou et al., 2016) (CVPR'16)	47.7	43.5	36.4	28.7	19.0	10.3	5.3	35.0	19.9	27.3
	TAL-Net (Chao et al., 2018) (CVPR'18)	59.8	57.1	53.2	48.5	42.8	33.8	20.8	52.3	39.8	45.1
	GTAN (Long et al., 2019) (CVPR'19)	69.1	63.7	57.8	47.2	38.8	-	-	55.3	-	-
	RCL (Wang et al., 2022) (CVPR'22)	-	-	70.1	62.3	52.9	42.7	30.7	-	57.1	-
Weak	UntrimNet (Wang et al., 2017) (CVPR'17)	44.4	37.7	28.2	21.1	13.7	-	-	29.0	-	-
	W-TALC (Paul et al., 2018) (ECCV'18)	55.2	49.6	40.1	31.1	22.8	-	7.6	39.8	-	-
	BaS-Net (Lee et al., 2020) (AAAI'20)	58.2	52.3	44.6	36.0	27.0	18.6	10.4	43.6	27.3	35.3
	DGAM (Shi et al., 2020) (CVPR'20)	60.0	54.2	46.8	38.2	28.8	19.8	11.4	37.0	-	-
	TSCN (Zhai et al., 2020) (ECCV'20)	63.4	57.6	47.8	37.7	28.7	19.4	10.2	47.0	28.8	37.8
	WUM (Lee et al., 2021) (AAAI'21)	67.5	61.2	52.3	43.4	33.7	22.9	12.1	51.6	32.9	41.9
	MSA (Huang et al., 2021a) (TIP'21)	65.5	58.9	49.1	40.0	31.4	18.8	10.6	49.0	30.0	39.2
	ACM-Net (Qu et al., 2021) (TIP'21)	68.9	62.7	55.0	44.6	34.6	21.8	10.8	53.2	33.4	42.6
	CoLA (Zhang et al., 2021) (CVPR'21)	66.2	59.5	51.5	41.9	32.2	22.0	13.1	50.3	32.1	40.9
	UGCT (Yang et al., 2021) (CVPR'21)	69.2	62.9	55.5	46.5	35.9	23.8	11.4	54.0	34.6	43.6
	ASL (Ma et al., 2021) (CVPR'21)	67.0	-	51.8	-	31.1	-	11.4	-	-	-
	D2-Net (Narayan et al., 2021) (ICCV'21)	65.7	60.2	52.3	43.4	36.0	-	-	51.5	-	-
	FAC-Net (Huang et al., 2021c) (ICCV'21)	67.6	62.1	52.6	44.3	33.4	22.5	12.7	52.0	33.1	42.2
	ACGNet (Yang et al., 2022) (AAAI'22)	68.1	62.6	53.1	44.6	34.7	22.6	12.0	52.6	33.4	42.5
	DCC (Li et al., 2022a) (CVPR'22)	69.0	63.8	55.9	45.9	35.7	24.3	**13.7**	54.1	35.1	44.0
	RSKP (Huang et al., 2022) (CVPR'22)	71.3	65.3	55.8	47.5	**38.2**	**25.4**	12.5	55.6	35.9	45.1
	ASM-Loc (He et al., 2022) (CVPR'22)	71.2	65.5	57.1	46.8	36.6	25.2	13.4	55.4	35.8	45.1
	CASE	**72.3**	**66.9**	**59.2**	**49.3**	37.7	24.2	**13.7**	**57.1**	**36.8**	**46.2**

4.2 Comparison with the State-of-the-art (SOTA) Methods

In Table 1, we compare our method with SOTA WTAL methods and several fully supervised method on THUMOS14. We can observe that our method evidently outperforms the previous WTAL methods. Remarkably, we surpass the SOTA approach ASM-Loc (He et al., 2022) by 1.7% on the average mAP (0.1:0.5). Furthermore, our method obtains competitive results even compared with some fully supervised methods, although we utilize much less supervision. In Appendix F.1, we thoroughly compare our method with SOTA approaches on ActivityNet v1.2 and v1.3, and our method also gains substantial promotion over previous SOTA methods on these two datasets.

4.3 Ablation Study

Contribution of each component. In Table 2, we evaluate the contribution of each component. To illustrate the quality of F&B separation, we additionally report the binary classification accuracy of the F&B snippets, namely, 'ACC'. The baseline obtains 38.3% on mAP and 73.6% on ACC (*cf.* 1-th row), which is further analyzed in Appendix F.2. Compared with the baseline, we can see that each proposed component contributes to the performance. For example, SCC and CCC yield a gain of 2.0% (*cf.* 1,2-th rows) and 1.1% (*cf.* 3,4-th rows) on mAP , respectively. The reason may be that via deliberated snippet clustering, the embedding space is shaped as well-structured, eventually benefiting F&B separation of the attention layer. After the SCC and CCC are equipped, we further employ the CAT, which brings about a remarkable promotion of 2.3% on mAP (*cf.* 7,8-th rows). In addition, DDCL boosts the performance by 1.1% (*cf.* 3,4-th rows). We will conduct exhaustive ablation studies to these components in the following sections. TSCL improves the method by over 2% on mAP (*cf.* 6,8-th rows), which may be attributed to the improved quality of the pseudo-labels (Zhai et al., 2020). The TSCL involves multiple procedures, which will be elaborately investigated in Appendix F.2. After incorporating all the components together, our method increases the performance from 38.3% to 46.2% on mAP and 73.6% to 77.5% on ACC.

Are the multiple clusters necessary? We propose to cluster the snippets into multiple classes (*i.e.*, $K^C > 2$), although only F&B separation is required. To verify the effectiveness of our design, we compare the performances under different settings of K^C in Table 3. As can be seen, a small K^C usually result in an inferior performance, while the results become stable when increasing K^C beyond 16. Notably, $K^C = 2$ causes a performance decline of 0.7% relative to the baseline. The reason may be that the clustering results deviate too much from the true distribution of the F&B snippets. Hence, clustering into multiple clusters is necessary.

Effect of cluster classification component (CCC). In Table 5, we investigate the effects of CCC. $\lambda_R = 0$ indicates that the loss $\mathcal{L}_R$ doesn't take effect. $\boldsymbol{\beta^R} = [0.5, 0.5]$ indicates that the equipartition constraint is used for setting $\boldsymbol{\beta^R}$. Notably, for our CASE, $\lambda_R \neq 0$ and $\boldsymbol{\beta^R}$ is set based on the distribution of pseudo F&B snippets (*i.e.*, Eq. 12). We can see that that using $\mathcal{L}_R$ leads to the simultaneous promotion of $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ and $\boldsymbol{P}^{\boldsymbol{A}}$. Besides, our proposed $\boldsymbol{\beta^R}$ is superior to the equipartition constraint (*i.e.*, Eq. 5). These observations clearly corroborate the effectiveness of our design.

Effect of distribution-constrained labeling (DCL) In Table 4, we provide detailed analysis of DCL. Without loss of generality, we will utilize DCL used in SCC as an illustration. 'PL' indicates

Table 2: Contribution of each component. DDCL, TSCL, and CAT indicate dynamic distribution-constrained labeling, two-stream co-labeling, and clustering-assisted testing, respectively. When CAT is equipped, $\ddot{\boldsymbol{P}}^{\boldsymbol{A}}$ is used for inference; otherwise, $\boldsymbol{P}^{\boldsymbol{A}}$ is used.

Row	SCC	CCC	DDCL	TSCL	CAT	mAP	ACC
1						38.3	73.6
2	✓					40.3	74.5
3	✓			✓		42.0	75.4
4	✓	✓		✓		43.1	75.9
5	✓	✓		✓	✓	45.4	77.1
6	✓	✓	✓		✓	44.1	76.7
7	✓	✓	✓	✓		43.9	76.5
8	✓	✓	✓	✓	✓	46.2	77.5

Table 3: The effect of K^C. $K^C = 0$ indicates the baseline.

K^C	0	2	4	8	16	32	48	64
mAP	38.3	37.6	38.6	39.6	40.3	40.1	39.9	39.9

Table 4: Ablation study on DCL under different settings, where $\bar{\boldsymbol{P}}^{\boldsymbol{C}} = \sum_{n=1}^{N} \boldsymbol{P}_n^{\boldsymbol{C}}/N \in \mathbb{R}^{K^C}$.

	PL(Hard)	PL (Soft)	DCL(Hard)	DCL (Soft)
mAP	40.8	41.2	44.7	46.2
$\mathrm{H}(\boldsymbol{P}^{\boldsymbol{C}})$	0.01	2.77	1.25	1.37
$\mathrm{H}(\bar{\boldsymbol{P}}^{\boldsymbol{C}})$	0.01	2.77	2.75	2.76

Table 5: The effect of CCC. λ_R is the weight of the loss $\mathcal{L}_R$. $\boldsymbol{\beta}^{\boldsymbol{R}}$ is the marginal distribution of $\boldsymbol{Q}^{\boldsymbol{R}}$. Here the performances evaluated with $\boldsymbol{P}^{\boldsymbol{A}}$, $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ and $\ddot{\boldsymbol{P}}^{\boldsymbol{A}}$ are reported.

	$\boldsymbol{P}^{\boldsymbol{A}}$	$\hat{\boldsymbol{P}}^{\boldsymbol{A}}$	$\ddot{\boldsymbol{P}}^{\boldsymbol{A}}$
CASE	43.9	44.1	46.2
$\lambda_R = 0$	43.4	43.1	44.6
$\boldsymbol{\beta}^{\boldsymbol{R}} = [0.5, 0.5]$	42.7	42.2	43.5

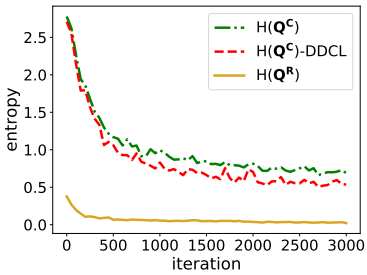

Figure 3: The evolution of the entropy of $\boldsymbol{Q}^{\boldsymbol{R}}$ and $\boldsymbol{Q}^{\boldsymbol{C}}$ during training. $\mathrm{H}(\boldsymbol{Q}^{\boldsymbol{C}})$ and $\mathrm{H}(\boldsymbol{Q}^{\boldsymbol{C}})$-DDCL represent that the terms without and with DDCL, respectively.

that the distribution constraint of Eq. 4 is disabled. 'Hard' and 'Soft' refer to the use of one-hot and soft labels, respectively. The one-hot label is obtained by applying the argmax operator to the soft label. In addition to mAP, we also report the average entropy of the cluster assignments $\mathrm{H}(\boldsymbol{P}^{\boldsymbol{C}})$ and the average entropy of the proportions of clusters $\mathrm{H}(\bar{\boldsymbol{P}}^{\boldsymbol{C}})$. The smaller $\mathrm{H}(\boldsymbol{P}^{\boldsymbol{C}})$, the less serious the uncertain assignment issue. The larger $\mathrm{H}(\bar{\boldsymbol{P}}^{\boldsymbol{C}})$, the less serious the imbalanced assignment issue. It is observed that: 1) PL (Hard) and PL (Soft) suffer from extremely serious imbalanced assignment and uncertain assignment respectively, resulting in evident performance degradation. Compared with them, DCL gets much better performance and meanwhile these two issues are greatly alleviated; 2) DCL (Hard) lags behind DCL (Soft). An explanation for this is that obtaining the hard labels is more aggressive than gradient updates, leading to a worse solution (Caron et al., 2020).

Effect of dynamic distribution-constrained labeling (DDCL). In Table 2, it is shown that the DDCL is helpful. To further understand the contribution of DDCL, we show the evolution of the entropy of $\boldsymbol{Q}^{\boldsymbol{C}}$ (*i.e.*, $\mathrm{H}(\boldsymbol{Q}^{\boldsymbol{C}})$) of the models with and without DDCL in Fig. 3. Besides, we also illustrate $\mathrm{H}(\boldsymbol{Q}^{\boldsymbol{R}})$ for comparison. Since $\boldsymbol{Q}^{\boldsymbol{C}}$ and $\boldsymbol{Q}^{\boldsymbol{R}}$ involve different numbers of classes, a direct comparison is not entirely fair. Still, we provide it as an indication. We can observe that $\mathrm{H}(\boldsymbol{Q}^{\boldsymbol{R}})$ converges quickly to a small value, indicating that the optimization can easily result in a solution close to one-hot form. Yet, $\mathrm{H}(\boldsymbol{Q}^{\boldsymbol{C}})$ converges slowly and keeps a large value at the end of training. On the other hand, once the DDCL is introduced, this issue is alleviated with lower $\mathrm{H}(\boldsymbol{Q}^{\boldsymbol{C}})$, which is helpful to prevent the uncertain assignment issue and obtain better performance.

Effect of clustering-assisted testing (CAT). In Table 5, we compare the performances of $\boldsymbol{P}^{\boldsymbol{A}}$, $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ and the fused one $\ddot{\boldsymbol{P}}^{\boldsymbol{A}}$ under different settings. It can be seen that, in all these cases, $\ddot{\boldsymbol{P}}^{\boldsymbol{A}}$ consistently outperforms both $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ and $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$, proving that $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ and $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ are complementary to each other. To gain further insights, we provide some visualized examples in Appendix G.

5 Conclusion

In this paper, we propose a new WTAL framework, namely CASE, which focuses on leveraging the snippet clustering to help the F&B separation. CASE first utilizes a snippet clustering component to divide the snippets into multiple clusters, and then employs a cluster classification component to classify the clusters as foreground or background. Moreover, we integrate an online self-training algorithm and a distribution-constrained labeling strategy to optimize these two components. Thereafter, the cluster assignments of the snippets can be used to refine their F&B scores. Extensive analysis manifests that CASE is powerful on all the benchmarks.

6 Reproducibility Statement

We have uploaded the main code of our CASE to https://anonymous.4open.science/r/CASE-1275/README.md for the reviewers.

References

YM Asano, C Rupprecht, and A Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *ICLR*, 2019. 2, 3, 5, 21

Alberto Borobia and Rafael Cantó. Matrix scaling: A geometric proof of sinkhorn's theorem. *Linear algebra and its applications*, 268:1–8, 1998. 21

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015. 7

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018. 3

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020. 2, 3, 5, 9

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 15

Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *ICCV*, 2017. 2, 3

Jianlong Chang, Yiwen Guo, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep discriminative clustering analysis. *arXiv preprint arXiv:1905.01681*, 2019. 3

Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *CVPR*, 2018. 8

Yinbo Chen, Zhuang Liu, Huijuan Xu, Trevor Darrell, and Xiaolong Wang. Meta-baseline: exploring simple meta-learning for few-shot learning. In *ICCV*, 2021. 6

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013. 5, 21

Zhiyuan Dang, Cheng Deng, Xu Yang, and Heng Huang. Multi-scale fusion subspace clustering using similarity constraint. In *CVPR*, 2020. 3

Enrico Fini, Enver Sangineto, Stéphane Lathuilière, Zhun Zhong, Moin Nabi, and Elisa Ricci. A unified objective for novel class discovery. In *ICCV*, 2021. 2, 5

Junyu Gao, Mengyuan Chen, and Changsheng Xu. Fine-grained temporal contrastive learning for weakly-supervised temporal action localization. In *CVPR*, 2022. 16

Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018. 15

Philip Haeusser, Johannes Plapp, Vladimir Golkov, Elie Aljalbout, and Daniel Cremers. Associative deep clustering: Training a classification network with no labels. In *GCPR*, 2018. 3

Bo He, Xitong Yang, Le Kang, Zhiyu Cheng, Xin Zhou, and Abhinav Shrivastava. Asm-loc: Action-aware segment modeling for weakly-supervised temporal action localization. In *CVPR*, 2022. 8, 16

Jiabo Huang, Shaogang Gong, and Xiatian Zhu. Deep semantic clustering by partition confidence maximisation. In *CVPR*, 2020a. 3

Linjiang Huang, Yan Huang, Wanli Ouyang, and Liang Wang. Relational prototypical network for weakly supervised temporal action localization. In *AAAI*, 2020b. 16

Linjiang Huang, Yan Huang, Wanli Ouyang, and Liang Wang. Modeling sub-actions for weakly supervised temporal action localization. *Transactions on Image Processing*, 2021a. 3, 8, 16

Linjiang Huang, Yan Huang, Wanli Ouyang, and Liang Wang. Two-branch relational prototypical network for weakly supervised temporal action localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021b. 3

Linjiang Huang, Liang Wang, and Hongsheng Li. Foreground-action consistency network for weakly supervised temporal action localization. In *ICCV*, 2021c. 8, 16

Linjiang Huang, Liang Wang, and Hongsheng Li. Weakly supervised temporal action localization via representative snippet knowledge propagation. In *CVPR*, 2022. 3, 8, 16

Ashraful Islam, Chengjiang Long, and Richard Radke. A hybrid attention mechanism for weakly-supervised temporal action localization. In *AAAI*, 2021. 3

Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *ICCV*, 2019. 3

Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes, 2014. 7

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 3

Pilhyeon Lee, Youngjung Uh, and Hyeran Byun. Background suppression network for weakly-supervised temporal action localization. In *AAAI*, 2020. 8, 16, 17

Pilhyeon Lee, Jinglu Wang, Yan Lu, and Hyeran Byun. Weakly-supervised temporal action localization by uncertainty modeling. In *AAAI*, 2021. 8, 16, 17

Jingjing Li, Tianyu Yang, Wei Ji, Jue Wang, and Li Cheng. Exploring denoised cross-video contrast for weakly-supervised temporal action localization. In *CVPR*, 2022a. 3, 8, 16

Ruihuang Li, Shuai Li, Chenhang He, Yabin Zhang, Xu Jia, and Lei Zhang. Class-balanced pixel-level self-labeling for domain adaptive semantic segmentation. In *CVPR*, 2022b. 2

Daochang Liu, Tingting Jiang, and Yizhou Wang. Completeness modeling and context separation for weakly supervised temporal action localization. In *CVPR*, 2019. 1, 3

Ziyi Liu, Le Wang, Wei Tang, Junsong Yuan, Nanning Zheng, and Gang Hua. Weakly supervised temporal action localization through learning explicit subspaces for action and context. In *AAAI*, 2021. 3

Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *CVPR*, 2019. 8

Wang Luo, Tianzhu Zhang, Wenfei Yang, Jingen Liu, Tao Mei, Feng Wu, and Yongdong Zhang. Action unit memory network for weakly supervised temporal action localization. In *CVPR*, 2021. 3, 16

Zhekun Luo, Devin Guillory, Baifeng Shi, Wei Ke, Fang Wan, Trevor Darrell, and Huijuan Xu. Weakly-supervised action localization with expectation-maximization multi-instance learning. In *ECCV*, 2020. 2, 3, 16

Junwei Ma, Satya Krishna Gorti, Maksims Volkovs, and Guangwei Yu. Weakly supervised action selection learning in video. In *CVPR*, 2021. 4, 8, 14, 16, 17

James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *BSMSP*, 1967. 3

Kyle Min and Jason J Corso. Adversarial background-aware loss for weakly-supervised temporal activity localization. In *ECCV*. Springer, 2020. 1, 3

Sanath Narayan, Hisham Cholakkal, Fahad Shabaz Khan, and Ling Shao. 3c-net: Category count and center loss for weakly-supervised action localization. In *ICCV*, 2019. 3

Sanath Narayan, Hisham Cholakkal, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. D2-net: Weakly-supervised action localization via discriminative embeddings and denoised activations. In *ICCV*, 2021. 8, 16

Phuc Nguyen, Ting Liu, Gautam Prasad, and Bohyung Han. Weakly supervised action localization by sparse temporal pooling network. In *CVPR*, 2018. 3

Phuc Xuan Nguyen, Deva Ramanan, and Charless C Fowlkes. Weakly-supervised action localization with background modeling. In *ICCV*, 2019. 1

Alejandro Pardo, Humam Alwassel, Fabian Caba, Ali Thabet, and Bernard Ghanem. Refineloc: Iterative refinement for weakly-supervised action localization. In *WACV*, 2021. 2, 3

Sujoy Paul, Sourya Roy, and Amit K Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification. In *ECCV*, 2018. 8, 16

Sanqing Qu, Guang Chen, Zhijun Li, Lijun Zhang, Fan Lu, and Alois Knoll. Acm-net: Action context modeling network for weakly-supervised temporal action localization. *Transactions on Image Processing*, 2021. 7, 8, 14, 16

Baifeng Shi, Qi Dai, Yadong Mu, and Jingdong Wang. Weakly-supervised action localization by generative attention modeling. In *CVPR*, 2020. 8

Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, 2016. 1, 8

Zheng Shou, Hang Gao, Lei Zhang, Kazuyuki Miyazawa, and Shih-Fu Chang. Autoloc: Weakly-supervised temporal action localization in untrimmed videos. In *ECCV*, 2018. 16

Krishna Kumar Singh and Yong Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *ICCV*, 2017. 3

Richard Sinkhorn. Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, 74(4):402–405, 1967. 21

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 6

Bing Su and Gang Hua. Order-preserving wasserstein distance for sequence matching. In *CVPR*, 2017. 6, 7, 21

Binglu Wang, Xun Zhang, and Yongqiang Zhao. Exploring sub-action granularity for weakly supervised temporal action localization. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021. 3

Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *CVPR*, 2017. 1, 3, 4, 8, 16

Qiang Wang, Yanhao Zhang, Yun Zheng, and Pan Pan. Rcl: Recurrent continuous localization for temporal action detection. In *CVPR*, 2022. 8

Jianlong Wu, Keyu Long, Fei Wang, Chen Qian, Cheng Li, Zhouchen Lin, and Hongbin Zha. Deep comprehensive correlation mining for image clustering. In *ICCV*, 2019. 3

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *ICML*, 2016. 3

Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *CVPR*, 2016. 3

Wenfei Yang, Tianzhu Zhang, Xiaoyuan Yu, Tian Qi, Yongdong Zhang, and Feng Wu. Uncertainty guided collaborative training for weakly supervised temporal action detection. In *CVPR*, 2021. 4, 8, 16

Xu Yang, Cheng Deng, Feng Zheng, Junchi Yan, and Wei Liu. Deep spectral clustering using dual autoencoder network. In *CVPR*, 2019. 3

Xu Yang, Cheng Deng, Kun Wei, Junchi Yan, and Wei Liu. Adversarial learning for robust deep clustering. In *NeurIPS*, 2020. 3

Zichen Yang, Jie Qin, and Di Huang. Acgnet: Action complement graph network for weakly-supervised temporal action localization. In *AAAI*, 2022. 8, 16

C Zach, T Pock, and H Bischof. A duality based approach for realtime tv-l 1 optical flow. *Pattern Recognition*, 2007. 15

Yuanhao Zhai, Le Wang, Wei Tang, Qilin Zhang, Junsong Yuan, and Gang Hua. Two-stream consensus network for weakly-supervised temporal action localization. In *ECCV*, 2020. 3, 4, 7, 8, 16

Can Zhang, Meng Cao, Dongming Yang, Jie Chen, and Yuexian Zou. Cola: Weakly-supervised temporal action localization with snippet contrastive learning. In *CVPR*, 2021. 1, 3, 7, 8, 16

Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*, 2018. 4

Jia-Xing Zhong, Nannan Li, Weijie Kong, Tao Zhang, Thomas H. Li, and Ge Li. Step-by-step erasion, one-by-one collection: A weakly supervised temporal action detector. In *ACMMM*, 2018. 3

A OVERVIEW

In the supplementary material, we offer more information about: 1) Source code (*cf.* B); 2) List of acronyms (*cf.* C); 3) Summarization of the DCL strategy (*cf.* D); 4) Implementation details (*cf.* E); 5) Additional experimental results (*cf.* F); 6) Visualization results (*cf.* G); 7) Some theoretical results (*cf.* H); 8) Discussion on limitations and future works (*cf.* I).

B SOURCE CODE

The main code of our CASE can be found in the anonymous website: https://anonymous.4open.science/r/CASE-1275/README.md.

C LIST OF ACRONYMS

Table 6: Key acronyms in this paper.

Short	**Long**
WTAL	Weakly-supervised Temporal Action Localization
F&B	Foreground and Background
MIL	Multiple Instance Learning
CASE	Clustering-Assisted F&B Separation Network
SCC	Snippet Clustering Component
CCC	Cluster Classification Component
DCL	Distribution-Constrained Labeling
DDCL	Dynamic Distribution-Constrained Labeling
TSCL	Two-Stream Co-Labeling
CAT	Clustering-Assisted Testing

D SUMMARIZATION OF DISTRIBUTION-CONSTRAINED LABELING

The distribution-constrained labeling (DCL) strategy is designed to constrain the distribution of pseudo-labels, which is an important technique in our method. What's more, we devise different variants of DCL in different parts of the main paper. Here we summarize them to make the relations between the variants easier to follow. Specifically, we first propose the primary version of DCL for snippet clustering, where we impose an equipartition constraint on the marginal distribution of the pseudo-labels to circumvent the imbalanced assignment issue (*cf.* Sec. 3.2.1). Thereafter, we present a dynamic version of DCL (named DDCL) to improve the snippet clustering. DDCL leverages the prediction of the baseline to form a prior distribution for the pseudo-labels so as to mitigate the uncertain assignment issue (*cf.* Sec. 3.2.3). The equipartition constraint and prior distribution together encourage the cluster assignments of the snippets to be individually certain and globally diverse, thereby resulting in more typical snippet clustering. To achieve feasible cluster classification, we make an improvement to the primary version of DCL by enforcing the marginal distribution of cluster-level F&B pseudo-labels to be consistent with that of snippet-level F&B pseudo-labels, thereby promoting the discrimination of clusters to F&B (*cf.* Sec. 3.2.2).

E IMPLEMENTATION DETAILS

E.1 BASELINE

Here we present more details about the multiple instance learning (MIL) scheme used in the baseline. Specifically, we first calibrate T-CAS $\boldsymbol{P}^{\boldsymbol{V}} \in \mathbb{R}^{B\times T\times K^V}$ with the attention weights $\boldsymbol{P}^{\boldsymbol{A}} \in \mathbb{R}^{B\times T}$ to highlight foreground snippets and suppress background snippets, resulting in the calibrated T-CAS (dubbed $\hat{\boldsymbol{P}}^{\boldsymbol{V}} \in \mathbb{R}^{B\times T\times K^V}$). It can be implemented in multiple ways. Here following (Ma et al., 2021; Qu et al., 2021), we fuse the scores by weighted summation, $\hat{\boldsymbol{P}}^{\boldsymbol{V}} = \omega \boldsymbol{P}^{\boldsymbol{V}} + (1-\omega)\boldsymbol{P}^{\boldsymbol{A}}$. ω is

a predefined weight. Thereafter, we select K snippets from each video for each class based on $\hat{\boldsymbol{P}}^{\boldsymbol{V}}$:

$$\Gamma_{b,c} = \arg \max_{\substack{\Gamma \subset \{1,..,T\} \\ |\Gamma|=K}} \sum_{\tau \in \Gamma} \hat{\boldsymbol{P}}^{\boldsymbol{V}}_{b,\tau,c} \quad \forall\, b \in \{1,..,B\}, c \in \{1,..,K^V\}, \tag{18}$$

where K is a hyper-parameter. Temporal pooling is applied over the selected snippets in $\Gamma_{b,c}$ to build video-level class prediction $\bar{\boldsymbol{P}} \in \mathbb{R}^{B \times K^V}$:

$$\bar{\boldsymbol{P}}_{b,c} = \text{Softmax}(\frac{1}{K} \sum_{\tau \in \Gamma_{b,c}} \boldsymbol{P}^{\boldsymbol{V}}_{b,\tau,c}). \tag{19}$$

Finally, $\bar{\boldsymbol{P}}_{b,c}$ is used to compute a video classification loss, as shown in Eq. 1 in the main paper. The $\Gamma_{b,c}$ is not only used to build the video-level scores, but also used to form the foreground pseudo-labels $\boldsymbol{Q}^{\boldsymbol{A}}$, as shown in the main paper.

E.2 Two-Stream Co-Labeling

We propose the two-stream co-labeling (TSCL) to improve the quality of the pseudo-labels by fusing the information of the RGB stream and optical-flow stream. Here we provide more details about it. TSCL will be applied in all procedures that would generate pseudo-labels (such as $\boldsymbol{Q}^{\boldsymbol{C}}$, $\boldsymbol{Q}^{\boldsymbol{R}}$). To be specific, for $\boldsymbol{Q}^{\boldsymbol{C}}$, we fuse the cluster assignments of RGB stream (dubbed $\boldsymbol{P}^{\boldsymbol{C},\text{RGB}}$) and that of Flow stream (dubbed $\boldsymbol{P}^{\boldsymbol{C},\text{Flow}}$) by:

$$\boldsymbol{P}^{\boldsymbol{C}} = 0.5\boldsymbol{P}^{\boldsymbol{C},\text{RGB}} + 0.5\boldsymbol{P}^{\boldsymbol{C},\text{Flow}}. \tag{20}$$

Then the pseudo-labels $\boldsymbol{Q}^{\boldsymbol{C}}$ is generated by:

$$\min_{\boldsymbol{Q}^{\boldsymbol{C}} \in \mathcal{Q}^C} \langle \boldsymbol{Q}^{\boldsymbol{C}}, -\log \boldsymbol{P}^{\boldsymbol{C}} \rangle. \tag{21}$$

As for $\boldsymbol{Q}^{\boldsymbol{R}}$, the prediction of cluster classifier of RGB stream (dubbed $\boldsymbol{P}^{\boldsymbol{R},\text{RGB}}$) and that of Flow stream (dubbed $\boldsymbol{P}^{\boldsymbol{R},\text{Flow}}$) are combined as follows

$$\boldsymbol{P}^{\boldsymbol{R}} = 0.5\boldsymbol{P}^{\boldsymbol{R},\text{RGB}} + 0.5\boldsymbol{P}^{\boldsymbol{R},\text{Flow}}. \tag{22}$$

Then the pseudo-labels $\boldsymbol{Q}^{\boldsymbol{R}}$ is generated by:

$$\min_{\boldsymbol{Q}^{\boldsymbol{R}} \in \mathcal{Q}^R} \langle \boldsymbol{Q}^{\boldsymbol{R}}, -\log \boldsymbol{P}^{\boldsymbol{R}} \rangle. \tag{23}$$

Moreover, the top-K selection used in Eq. 18 can be regarded as a procedure of defining the foreground and background snippets. Here, we also utilize the TSCL to improve the quality of the top-K selection. Specifically, we fuse the calibrated T-CAS of RGB stream (dubbed $\hat{\boldsymbol{P}}^{\boldsymbol{V},\text{RGB}}$) and that of optical-flow stream (dubbed $\hat{\boldsymbol{P}}^{\boldsymbol{V},\text{Flow}}$) as follows:

$$\hat{\boldsymbol{P}}^{\boldsymbol{V}} = 0.5\hat{\boldsymbol{P}}^{\boldsymbol{V},\text{RGB}} + 0.5\hat{\boldsymbol{P}}^{\boldsymbol{V},\text{Flow}}. \tag{24}$$

Then $\hat{\boldsymbol{P}}^{\boldsymbol{V}}$ is used for top-K selection. Notably, the results of the top-K selection also influences the definition of $\boldsymbol{Q}^{\boldsymbol{A}}$.

E.3 Training and Inference Details

TVL1 (Zach et al., 2007) is applied to extract optical-flow stream from RGB stream in advance. Each stream is divided into 16-frame snippets. We employ the I3D (Carreira & Zisserman, 2017) network pretrained on Kinetics-400 (Carreira & Zisserman, 2017) to extract snippet-level features from each stream, where the channel dimension D is 1024. The number of sampled snippets T is set to 750 for THUMOS14 and 50 for ActivityNet v1.2 and v1.3. Both streams share the same structure but have separate parameters. The embedding encoders are comprised of a temporal convolution layer with 512 channels and a ReLU layer. The action classifier consists of a FC layer and a Softmax layer. The cluster head is composed of a cosine classifier Gidaris & Komodakis (2018) with temperature of 10 and a Softmax layer. The attention layer consists of a FC layer and a Sigmoid layer. We set the classes K^C of the cluster head to 16 for THUMOS14 and 64 for ActivityNet v1.2 and v1.3. The batch size B is set to 16 for all datasets. The K for MIL is set to $T//8$ in THUMOS14 and

Table 7: Results on ActivityNet v1.2. AVG indicates the average mAP at IoU thresholds 0.5:0.05:0.95.

Method	mAP @ IoU			
	0.5	0.75	0.95	AVG
UntrimNet (Wang et al., 2017)	7.4	3.2	0.7	3.6
AutoLoc (Shou et al., 2018)	27.3	15.1	3.3	16.0
W-TALC (Paul et al., 2018)	37.0	12.7	1.5	18.0
BaS-Net (Lee et al., 2020)	38.5	24.2	5.6	24.3
RPN (Huang et al., 2020b)	37.6	23.9	5.4	23.3
EM-MIL (Luo et al., 2020)	37.4	-	-	20.3
TSCN (Zhai et al., 2020)	37.6	23.7	5.7	23.6
WUM (Lee et al., 2021)	41.2	25.6	6.0	25.9
CoLA (Zhang et al., 2021)	42.7	25.7	5.8	26.1
ASL (Ma et al., 2021)	40.2	-	-	25.8
D2-Net (Narayan et al., 2021)	42.3	25.5	5.8	26.0
ACGNet (Yang et al., 2022)	41.8	26.0	5.9	26.1
CASE	**43.6**	**26.9**	**6.3**	**27.6**

Table 8: Results on ActivityNet v1.3. AVG indicates the average mAP at IoU thresholds 0.5:0.05:0.95.

Method	mAP @ IoU			
	0.5	0.75	0.95	AVG
BaS-Net (Lee et al., 2020)	34.5	22.5	4.9	22.2
TSCN (Zhai et al., 2020)	35.3	21.4	5.3	21.7
WUM (Lee et al., 2021)	37.0	23.9	5.7	23.7
MSA (Huang et al., 2021a)	36.5	22.8	6.0	22.9
ACM-Net (Qu et al., 2021)	40.1	24.2	6.2	24.6
UGCT (Yang et al., 2021)	39.1	22.4	5.8	23.8
AUMN (Luo et al., 2021)	38.3	23.5	5.2	23.5
FAC-Net (Huang et al., 2021c)	37.6	24.2	6.0	24.0
DCC (Li et al., 2022a)	38.8	24.2	5.7	24.3
FTCL (Gao et al., 2022)	40.0	24.3	6.4	24.8
RSKP (Huang et al., 2022)	40.6	24.6	5.9	25.0
ASM-Loc (He et al., 2022)	41.0	24.9	6.2	25.1
CASE	**42.3**	**25.6**	**6.6**	**26.0**

$T//2$ in ActivityNet v1.2 and v1.3. The γ is set to 0.7. The ϵ is set to 20. The temperature ρ is set to 10. The standard deviation σ is set to 10. The ω is set to 0.25. The loss weights are set as $\lambda_A = 1, \lambda_C = 1, \lambda_R = 0.3$ for all datasets. We utilize Adam optimizer with a learning rate of 10^{-4} for all datasets. We run each experiment three times and report their mean accuracy for reliability.

In the inference stage, we first fuse the video-level scores $\bar{\boldsymbol{P}}$ (and snippet-level scores $\boldsymbol{P}$) of RGB stream and that of optical-flow stream by average. Then, we threshold on the video-level scores to determine the action categories. For the selected action class, following (Qu et al., 2021; Gao et al., 2022), we apply a threshold strategy on the calibrated T-CAS $\hat{\boldsymbol{P}}^{\boldsymbol{V}}$ to obtain action proposals. After obtaining the action proposals, we calculate the class-specific score for each proposal using the outer-inner-contrastive technique (Shou et al., 2018). To enrich the proposal pool, multiple thresholds are applied. The Non-Maximum Suppression (NMS) is used to remove duplicated proposals.

F Additional Experimental Results

F.1 Performances on ActivityNet v1.2 and v1.3

In Table 7 and Table 8, we comprehensively compare our proposed method with several state-of-art WTAL models on ActivityNet v1.2 and v1.3, respectively. It can be seen that our method achieve the best performances on both datasets. In ActivityNet v1.2, CASE significantly outperforms ACGNet (Yang et al., 2022) by 1.5% on AVG. In ActivityNet v1.3, CASE improves previous SOTA method ASM-Loc (He et al., 2022) by 0.9% on AVG. There results further prove the superiority of our method.

F.2 Additional Ablation Study

Detailed analysis of baseline model. We carry out several ablation experiments to study the components of the baseline. The results are illustrated in Table 9. It can be seen that ATM largely increases performance (*cf.* 1,2-th rows), demonstrating the significance of class-agnostic F&B separation. Besides, the generalized binary cross-entropy loss performs better than the traditional one (*cf.* 2,3-th rows), proving that enhancing the label noise tolerance is advantageous.

Detailed analysis of two-stream co-labeling (TSCL). In Table 2 of the main paper, we show the overall performance promotion when the TSCL is applied. But as described in Appendix E.2, TSCL actually involves multiple procedures, *e.g.*, Eq. 20, Eq. 22, Eq. 24. Hence, we offer the detailed analysis of each procedure in Table 10. From the table, it can be seen that the TSCL used in each procedure contributes to the final performance.

Table 9: Ablation study on the baseline. VTB, ATB and GBCE indicate video classification branch, attention branch, and generalized binary cross-entropy loss, respectively. Notably, if GBCE is not used, we use the traditional binary cross-entropy loss to train the ATB.

Row	VTB	ATB	GBCE	mAP
1	✓			31.0
2	✓	✓		38.0
3	✓	✓	✓	38.3

Table 10: Ablation study on the TSCL under different settings.

Row	Eq. 24	Eq. 22	Eq. 20	mAP
1				44.1
2	✓			44.7
3	✓	✓		45.3
4	✓		✓	45.7
5		✓	✓	45.6
6	✓	✓	✓	46.2

Table 11: Performance on different baselines.

Method	mAP
BaS-Net (Lee et al., 2020)	35.3
BaS-Net + CASE	39.3
WUM (Lee et al., 2021)	40.1
WUM + CASE	43.1
ASL (Ma et al., 2021)	42.4
ASL + CASE	45.4

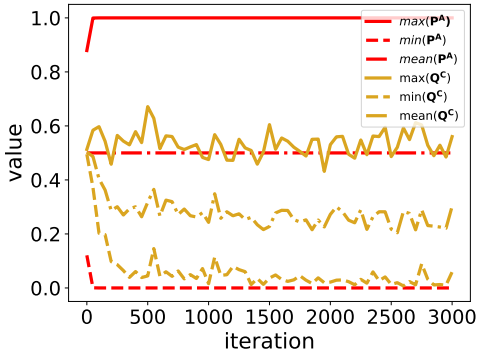

Figure 4: The maximum, average, and minimum values of $\boldsymbol{P^A}$ and $\boldsymbol{Q^C}$ of each iteration during training.

Table 12: Average mAP (mean ± std) of the baseline and CASE.

	AVG (0.1:0.5)	AVG (0.3:0.7)	AVG
Baseline	47.8±0.22	28.8±0.19	38.3±0.20
CASE	57.1±0.24	36.8±0.17	46.2±0.18

$\boldsymbol{P^A}$ vs. $\boldsymbol{R^A}$. As mentioned in Sec. 3.2.3 of the main paper, we use the distance between the normalized ranking indices of the snippets $\boldsymbol{R^A}$ and the pseudo-labels of the clusters $\boldsymbol{Q^C}$ to compute a 2D gaussian distribution (*cf.* Eq. 13). In theory, $\boldsymbol{R^A}$ can be replaced by $\boldsymbol{P^A}$. However, we experimentally find that the performance of using $\boldsymbol{P^A}$ is inferior to that of using $\boldsymbol{R^A}$ (*i.e.*, 45.7 for $\boldsymbol{P^A}$ *vs.* 46.2 for $\boldsymbol{R^A}$ on average mAP). To explain it, we show the statistics (*i.e.*, maximum, average, and minimum) of $\boldsymbol{P^A}$ and $\boldsymbol{Q^C}$ in Fig. 4. The statistics are computed over each batch (*i.e.*, iteration). Notably, the maximum, average, and minimum of $\boldsymbol{R^A}$ are always $\frac{1}{N} \simeq 0.$, $0.5 + 0.5\frac{1}{N} \simeq 0.5$ and $1.$, respectively. As we can see, compared with $\boldsymbol{P^A}$, $\boldsymbol{R^A}$ is more comparable to $\boldsymbol{Q^C}$ in range. For example, both the averages of $\boldsymbol{Q^C}$ and $\boldsymbol{R^A}$ are around 0.5 and evidently larger than that of $\boldsymbol{P^A}$. This well proves the correctness of our design.

Generalization capability to different baselines. In Table 11, we integrate our method into different baselines. For fairness, we use the default settings of these methods. It can be seen that our method is able to consistently and significantly improve the performances, which well demonstrates the great generalization ability of CASE on different baselines.

Standard deviation for multiple runs. In Table 12, we report the standard deviation of the mAP performance of our method and the baseline, which is computed from 3 runs with different random seeds. It can be seen that the performance of CASE is stable under different random seeds, suggesting that the architecture of CASE itself is the main reason for performance promotion rather than randomness.

G VISUALIZATION

In this section, we further examine the effectiveness of our method with visualizations.

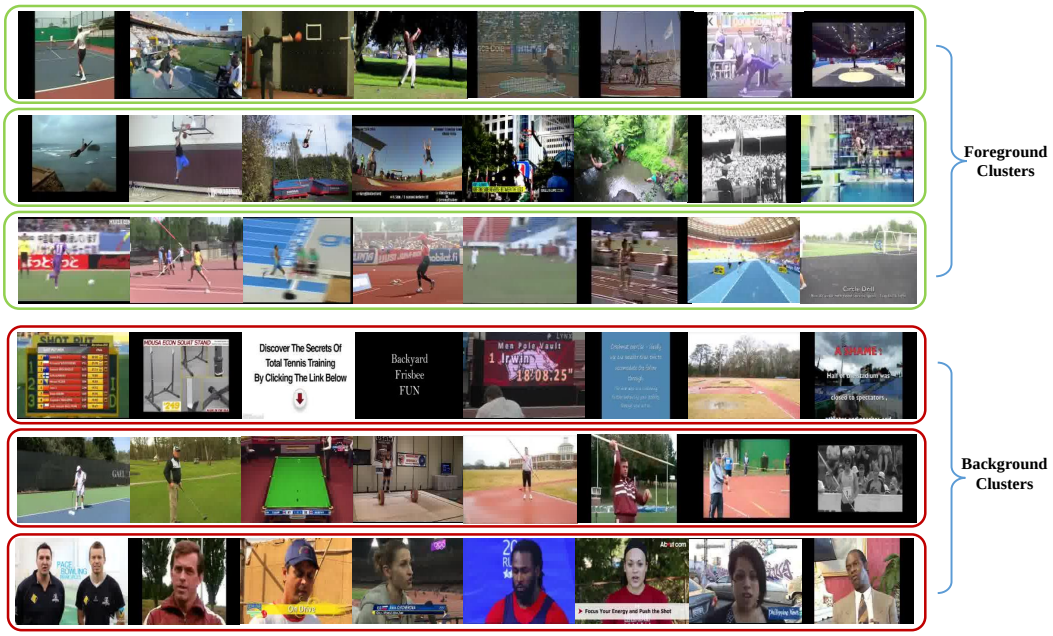

Figure 5: Qualitative results of the snippet clustering and cluster classification. We show three clusters belonging to foreground class at the top and there clusters belonging to background class at the bottom.

Predictions of SCC and CCC. We firstly show some examples of the snippet clustering results and cluster classification results in Fig. 5, where different images are sampled from different snippets. It can be seen that 1) The snippets within a cluster share some common characteristics. 2) The snippets within the foreground/background clusters commonly belong to the foreground/background class. These results well prove the effectiveness of the SCC and CCC.

Visualization of snippet embeddings and cluster prototypes. We visualize the snippet embeddings and cluster prototypes by using tSNE plot in Fig. 6. In detail, we show the embeddings of 10,000 snippets sampled from THUMOS14 and the prototypes of all 16 clusters. The snippet embeddings are derived from the attention branch, and the cluster prototypes are computed from the embeddings based on the cluster assignments of the snippets. From Fig. 6, we can observe that: 1) Each cluster is surrounded by a group of snippets, demonstrating that the clusters are typical to the snippets. 2) The clusters are separable to foreground class and background class. 3) The boundary between foreground snippets and background snippets of our method is more clear than that of baseline in the embedding space.

Comparison between $\boldsymbol{P^A}$ and $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$. We illustrate some examples of F&B separation results in Fig. 7 for comparing $\boldsymbol{P^A}$ and $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$. We have following observations. First, $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ activates more complete action regions compared to $\boldsymbol{P^A}$ (see the regions of '2', '4') and has more clear and more accurate action boundaries (see the regions of '3', '5'). The reason is that compared with $\boldsymbol{P^A}$, $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ is more independent to the video classification loss, and thus can capture more comprehensive distribution of the snippets. Second, $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ is not always better than $\boldsymbol{P^A}$ (see the region of '1'). When training $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$, CASE treats the snippets as independent samples, and thus is unable to make use of the video-level cues and temporal continuity, leading to abnormal detection. These observations verify that $\boldsymbol{P^A}$ and $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$ are complementary to each other.

Comparison between CASE and baseline. In Fig. 8, four visualized examples are provided to illustrate the differences between the F&B separation of CASE and that of baseline. It can be observed that: 1) CASE is advantageous to capture fine-grained patterns of snippets that are helpful to distinguish different snippets (see the solid boxes). For example, the region of '4' represents

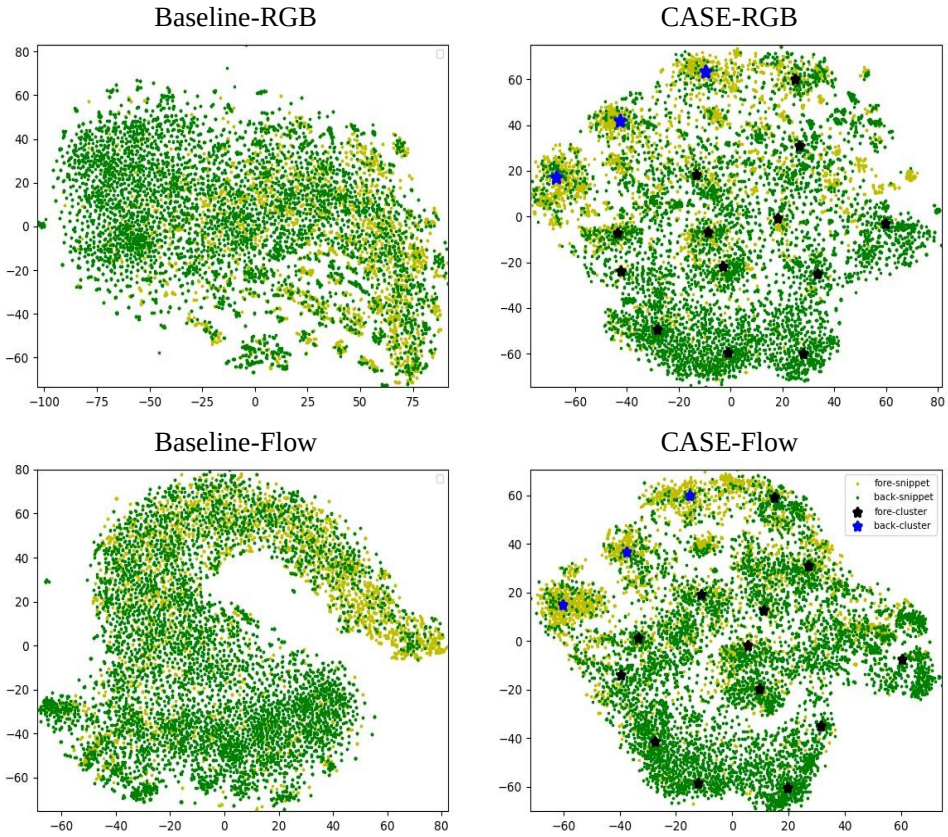

Figure 6: Visualization of the snippet embeddings and cluster prototypes. The left one is visualization for baseline and the right for our model. '-RGB' and '-Flow' indicate that the embeddings are produced in the RGB stream and optical-flow stream, respectively. 'fore-snippet', 'back-snippet', 'back-cluster', and 'back-cluster' indicate the foreground snippets, background snippets, foreground cluster prototypes, and background cluster prototypes, respectively.

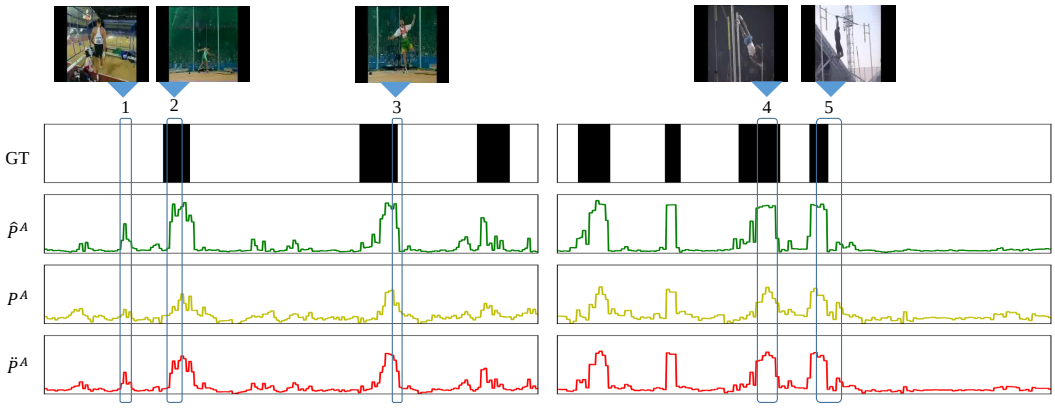

Figure 7: Qualitative results of two videos on THUMOS14. We show $\hat{\boldsymbol{P}}^{\boldsymbol{A}}$, $\boldsymbol{P}^{\boldsymbol{A}}$ and $\ddot{\boldsymbol{P}}^{\boldsymbol{A}}$, and ground-truth. The top numbers indicate some noteworthy regions.

the area near the boundary of a 'diving' action instance, where the background regions are visually similar to action regions. CASE can accurately classify the snippets to F&B classes while baselines cannot, showing that CASE is able to capture the underlying fine-grained structure of the snippets. 2) CASE performs worse than the baseline in some 'suspicious' regions (see the dashed boxes). To

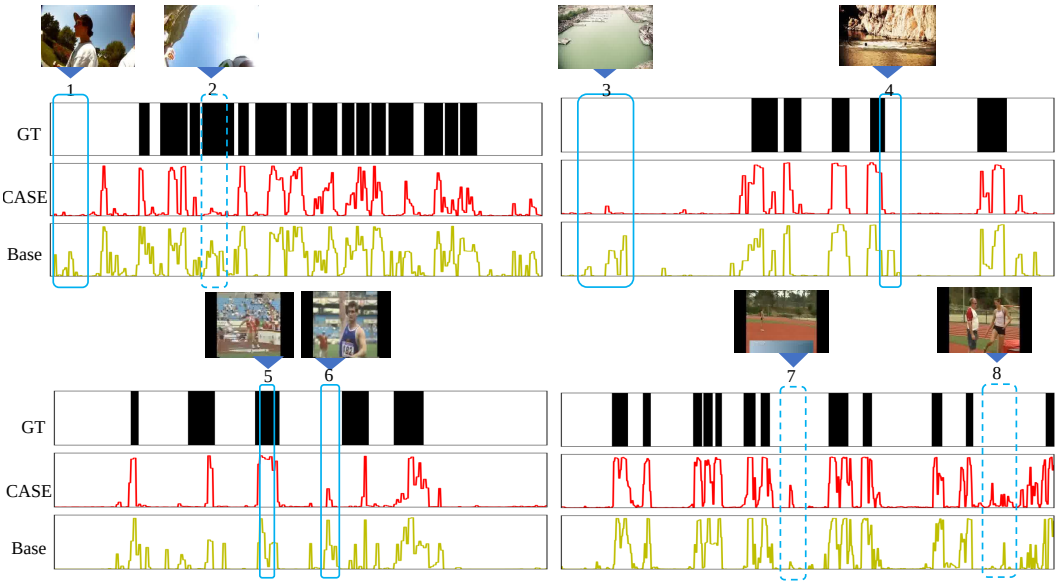

Figure 8: Comparison between our CASE and the baseline. The solid/dashed boxes represent the regions where CASE performs better/worse than the baseline.

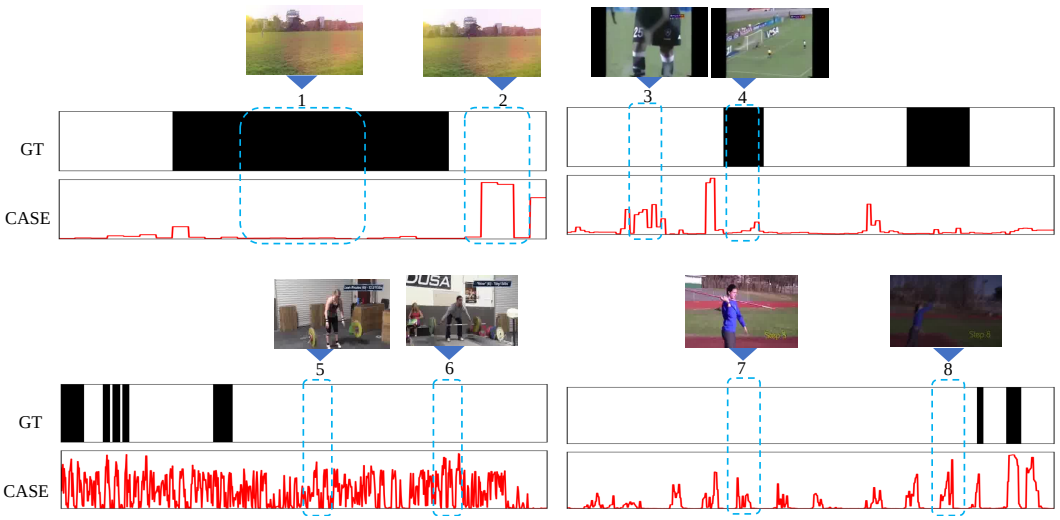

Figure 9: Samples of failure cases. We highlight the regions with wrong predictions by dashed boxes.

name a few, in the region of '8', there is an athlete who raises her leg, causing CASE to mistake the region for an action instance. It may be avoided by the baseline model, because the video-level labels used to train the baseline can offer instructive information for the potential action types within the videos.

Failure cases. We also show several failure cases in Fig. 9. The failure cases are caused by 1) low quality of images, *e.g.*, '1' and '8'; 2) indistinguishable body motions, *e.g.*, '3' and '7'; 3) small objects, *e.g.*, '2' and '4'; 4) incorrect annotation, *e.g.*, '5' and '6'.

H THEORETICAL PROOFS

Here we provide the derivation of the solution of following problem.

$$\begin{aligned} &\min \langle \boldsymbol{Q}^{\boldsymbol{C}}, -\log \boldsymbol{P}^{\boldsymbol{C}} \rangle + \frac{1}{\epsilon} \mathrm{KL}(\boldsymbol{Q}^{\boldsymbol{C}} || \hat{\boldsymbol{Q}}^{\boldsymbol{C}}) \quad s.t., \boldsymbol{Q}^{\boldsymbol{C}} \in \mathcal{Q}^{C} \\ &\mathcal{Q}^{C} = \{\boldsymbol{Q}^{\boldsymbol{C}} \in \mathbb{R}_{+}^{N \times K^{C}} | \boldsymbol{Q}^{\boldsymbol{C}} \mathbf{1}^{K^{C}} = \boldsymbol{\alpha}^{\boldsymbol{C}}, \boldsymbol{Q}^{\boldsymbol{C}\top} \mathbf{1}^{N} = \boldsymbol{\beta}^{\boldsymbol{C}}\}. \end{aligned} \tag{25}$$

For notation simplicity, we remove the superscript C. Then the problem is rewritten as

$$\begin{aligned} &\min \langle \boldsymbol{Q}, -\log \boldsymbol{P} \rangle + \frac{1}{\epsilon} \mathrm{KL}(\boldsymbol{Q} || \hat{\boldsymbol{Q}}) \quad s.t., \boldsymbol{Q} \in \mathcal{Q} \\ &\mathcal{Q} = \{\boldsymbol{Q} \in \mathbb{R}_{+}^{N \times K} | \boldsymbol{Q} \mathbf{1}^{K} = \boldsymbol{\alpha}, \boldsymbol{Q}^{\top} \mathbf{1}^{N} = \boldsymbol{\beta}\}. \end{aligned} \tag{26}$$

To address the problem, we first write the Lagrangian function of Eq. 26 as follows:

$$\begin{aligned} \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\mu}, \boldsymbol{\nu}) &= \langle \boldsymbol{Q}, -\log \boldsymbol{P} \rangle + \frac{1}{\epsilon} \mathrm{KL}(\boldsymbol{Q} || \hat{\boldsymbol{Q}}) + \boldsymbol{\mu}^{\top} (\boldsymbol{Q} \mathbf{1}^{K} - \boldsymbol{\alpha}) + \boldsymbol{\nu}^{\top} (\boldsymbol{Q}^{\top} \mathbf{1}^{N} - \boldsymbol{\beta}) \\ &= \sum_{n=1}^{N} \sum_{k=1}^{K} (-\boldsymbol{Q}_{n,k} \log \boldsymbol{P}_{n,k} + \frac{1}{\epsilon} \boldsymbol{Q}_{n,k} \log \frac{\boldsymbol{Q}_{n,k}}{\hat{\boldsymbol{Q}}_{n,k}} + \boldsymbol{\mu}_{n} \boldsymbol{Q}_{n,k} + \boldsymbol{\nu}_{k} \boldsymbol{Q}_{n,k}) - \boldsymbol{\mu}^{\top} \boldsymbol{\alpha} - \boldsymbol{\nu}^{\top} \boldsymbol{\beta} \end{aligned} \tag{27}$$

where $\boldsymbol{\mu} \in \mathbb{R}^{N}$ and $\boldsymbol{\nu} \in \mathbb{R}^{K}$ are the dual variables so that $\boldsymbol{Q} \mathbf{1}^{K} = \boldsymbol{\alpha}$ and $\boldsymbol{Q}^{\top} \mathbf{1}^{N} = \boldsymbol{\beta}$. The derivative of $\mathcal{L}(\boldsymbol{Q}, \boldsymbol{\mu}, \boldsymbol{\nu})$ *w.r.t.* $\boldsymbol{Q}_{n,k}$ is:

$$\frac{\partial \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\mu}, \boldsymbol{\nu})}{\partial \boldsymbol{Q}_{n,k}} = -\log \boldsymbol{P}_{n,k} + \frac{1}{\epsilon} \log \frac{\boldsymbol{Q}_{n,k}}{\hat{\boldsymbol{Q}}_{n,k}} + \frac{1}{\epsilon} + \boldsymbol{\mu}_{n} + \boldsymbol{\nu}_{k}. \tag{28}$$

Note that, the optimal $\boldsymbol{Q}$ exists and is unique, as both the objective and the constraint in Eq. 26 are convex. Hence, to obtain the optimal $\boldsymbol{Q}$, we set $\frac{\partial \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\mu}, \boldsymbol{\nu})}{\partial \boldsymbol{Q}_{n,k}} = 0$, and then get:

$$\boldsymbol{Q}_{n,k} = e^{-\frac{1}{2} - \epsilon \boldsymbol{\mu}_{n} - \frac{1}{2}} (\hat{\boldsymbol{Q}}_{n,k} \boldsymbol{P}_{n,k}^{\epsilon}) e^{-\frac{1}{2} - \epsilon \boldsymbol{\nu}_{k}}, \tag{29}$$

Let us denote $\boldsymbol{S} = \hat{\boldsymbol{Q}} \cdot \boldsymbol{P}^{\epsilon}$. Obviously, all elements of $\boldsymbol{S}$ are strictly positive. According to (Sinkhorn, 1967; Borobia & Cantó, 1998; Su & Hua, 2017), there exist diagonal matrices $\mathrm{diag}(\boldsymbol{u})$ and $\mathrm{diag}(\boldsymbol{v})$ with strictly positive diagonal elements so that $\mathrm{diag}(\boldsymbol{u}) S \,\mathrm{diag}(\boldsymbol{v})$ belongs to $\mathcal{Q}$.

In summary, the optimal $\boldsymbol{Q}$ has the form as:

$$\boldsymbol{Q} = \mathrm{diag}(\boldsymbol{u}) \boldsymbol{S} \,\mathrm{diag}(\boldsymbol{v}) = \mathrm{diag}(\boldsymbol{u}) (\hat{\boldsymbol{Q}} \cdot \boldsymbol{P}^{\epsilon}) \,\mathrm{diag}(\boldsymbol{v}). \tag{30}$$

where $\boldsymbol{u} \in \mathbb{R}^{N}$ and $\boldsymbol{v} \in \mathbb{R}^{K}$ are two renormalization vectors that makes the resulting matrix $\boldsymbol{Q}$ to be a probability matrix, which can be efficiently computed by the iterative Sinkhorn-Knopp algorithm (Cuturi, 2013). We refer to (Cuturi, 2013; Asano et al., 2019; Su & Hua, 2017) for more details.

I LIMITATIONS AND FUTURE WORKS

1) In this work, we mainly focus on the class-agnostic F&B separation due to its significance and difficulty to WTAL. But for the WTAL task, classifying different action classes is also required, which is beyond the scope of our current framework.

2) The CASE needs a WTAL baseline to provide semantic-level information of F&B classes so as to classify the clusters as foreground or background. A more self-contained clustering-based framework is our future work.