VideoLT: Large-scale Long-tailed Video Recognition

Xing Zhang[1*] Zuxuan Wu[2,3*] Zejia Weng[2] Huazhu Fu[4]
Jingjing Chen[2,3] Yu-Gang Jiang[2,3†] Larry Davis[5]
[1]Academy for Engineering and Technology, Fudan University,
[2]Shanghai Key Lab of Intel. Info. Processing, School of Computer Science, Fudan University,
[3] Shanghai Collaborative Innovation Center on Intelligent Visual Computing,
[4]Inception Institute of Artificial Intelligence, [5]University of Maryland

Abstract

Label distributions in real-world are oftentimes long-tailed and imbalanced, resulting in biased models towards dominant labels. While long-tailed recognition has been extensively studied for image classification tasks, limited effort has been made for the video domain. In this paper, we introduce ***VideoLT****, a large-scale long-tailed video recognition dataset, as a step toward real-world video recognition. VideoLT contains 256,218 untrimmed videos, annotated into 1,004 classes with a long-tailed distribution. Through extensive studies, we demonstrate that state-of-the-art methods used for long-tailed image recognition do not perform well in the video domain due to the additional temporal dimension in videos. This motivates us to propose FrameStack, a simple yet effective method for long-tailed video recognition. In particular, FrameStack performs sampling at the frame-level in order to balance class distributions, and the sampling ratio is dynamically determined using knowledge derived from the network during training. Experimental results demonstrate that FrameStack can improve classification performance without sacrificing the overall accuracy. Code and dataset are available at:* `https://github.com/17Skye17/VideoLT`.

1. Introduction

Deep neural networks have achieved astounding success in a wide range of computer vision tasks like image classification [18, 19, 39, 41], object detection [14, 28, 34, 35], *etc.* Training these networks requires carefully curated datasets like ImageNet and COCO, where object classes are uniformly distributed. However, real-world data often have a long tail of categories with very few training samples, posing significant challenges for network training. This results in biased models that perform exceptionally well on head classes (categories with a large number of training samples) but poorly on tail classes that contain a limited number of samples (see Figure 1).

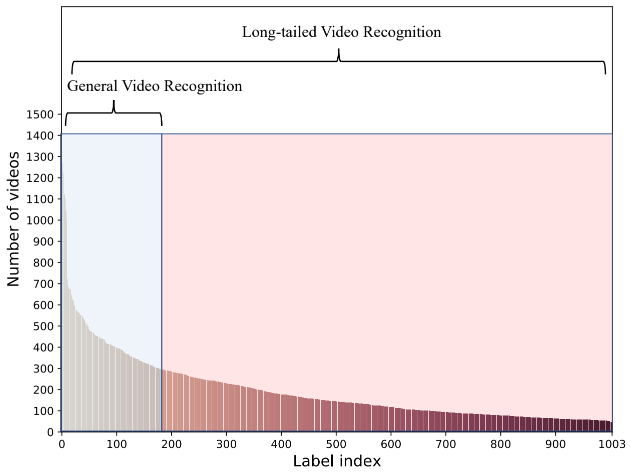

Figure 1. Long-tailed video recognition. General video recognition methods are overfitted on head classes, while long-tailed video recognition focuses on the performance of both head and tail classes, especially on tail classes. (Blue box is the head class region, red box is the region of medium and tail classes.)

Recently, there is a growing interest in learning from long-tailed data for image tasks [4, 10, 22, 29, 40, 42, 47, 49, 53]. Two popular directions to balance class distributions are re-sampling and re-weighting. Re-sampling [8, 11, 16, 22, 53] methods up-sample tail classes and down-sample head classes to acquire a balanced data distribution from the original data. On the other hand, re-weighting methods [4, 10, 27, 40, 47, 52] focus on designing weights to balance the loss functions of head and tail classes. While extensive studies have been done for long-tailed image classification tasks, limited effort has been made for video classification.

While it is appealing to directly generalize these methods from images to videos, it is also challenging since for classification tasks, videos are usually weakly labeled—only

[*] Equal contribution. [†] Corresponding author.

a single label is provided for a video sequence and only a small number of frames correspond to that label. This makes it difficult to apply off-the-shelf re-weighting and re-sampling techniques since not all snippets [1] contain informative clues—some snippets directly relate to the target class while others might consist of background frames. As a result, using a fixed weight/sampling strategy for all snippets to balance label distributions is problematic. For long-tailed video recognition, we argue that balancing the distribution between head and tail classes should be performed at the frame-level rather than at the video (sample)-level—more frames in videos from tail classes should be sampled for training and vice versa. More importantly, frame sampling should be dynamic based on the confidence of neural networks for different categories during the training course. This helps preventing overfitting for head classes and underfitting for tail classes.

To this end, we introduce FrameStack, a simple yet effective approach for long-tailed video classification. FrameStack operates on video features and can be plugged into state-of-the-art video recognition models with minimal surgery. More specifically, given a top-notch classification model which preserves the time dimension of input snippets, we first compute a sequence of features as inputs of FrameStack, *i.e*., for a input video with T frames, we obtain T feature representations. To mitigate the long tail problem, we define a temporal sampling ratio to select different number of frames from each video conditioned on the recognition performance of the model for target classes. If the network is able to offer decent performance for the category to be classified, we then use fewer frames for videos in this class. On the contrary, we select more frames from a video if the network is uncertain about its class of interest. We instantiate the ratio using running average precision (AP) of each category computed on training data. The intuition is that AP is a dataset-wise metric, providing valuable information about the performance of the model on each category and it is dynamic during training as a direct indicator of progress achieved so far. Consequently, we can adaptively under-sample classes with high AP to prevent over-fitting and up-sample those with low AP.

However, this results in samples with different time dimensions and such variable-length inputs are not parallel-friendly for current training pipelines. Motivated by recent data-augmentation techniques which blend two samples [43, 50] as virtual inputs, FrameStack performs temporal sampling on a pair of input videos and then concatenates re-sampled frame features to form a new feature representation, which has the same temporal dimension as its inputs. The resulting features can then be readily used for final recognition. We also adjust the the corresponding labels conditioned on the temporal sampling ratio.

[1] We use "snippet" to denote a stack of frames sampled from a video clip, which are typically used as inputs for video networks.

Moreover, we also collect a large-scale long tailed video recognition dataset, VideoLT, which consists of 256,218 videos with an average duration of 192 seconds. These videos are manually labeled into 1,004 classes to cover a wide range of daily activities. Our VideoLT have 47 head classes (#videos $>$ 500), 617 medium classes (100 $<$ #videos $<=$ 500) and 340 tail classes (#videos $<=$ 100), which naturally has a long tail of categories.

Our contributions are summarized as follows:

- We collect a new large-scale long-tailed video recognition dataset, VideoLT, which contains 256,218 videos that are manually annotated into 1,004 classes. *To the best of our knowledge, this is the first "untrimmed" video recognition dataset which contains more than 1,000 manually defined classes.*

- We propose FrameStack, a simple yet effective method for long-tailed video recognition. FrameStack uses a temporal sampling ratio derived from knowledge learned by networks to dynamically determine how many frames should be sampled.

- We conduct extensive experiments using popular long-tailed methods that are designed for image classification tasks, including re-weighting, re-sampling and data augmentation. We demonstrate that the existing long-tailed image methods are not suitable for long-tailed video recognition. By contrast, our FrameStack combines a pair of videos for classification, and achieves better performance compared to alternative methods. The dataset, code and results can be found at: `https://github.com/17Skye17/VideoLT`.

2. Related Work

2.1. Long-Tailed Image Recognition

Long-tailed image recognition has been extensively studied, and there are two popular strands of methods: re-weighting and re-sampling.

Re-Weighting A straightforward idea of re-weighting is to use the inverse class frequency to weight loss functions in order to re-balance the contributions of each class to the final loss. However, the inverse class frequency usually results in poor performance on real-world data [10]. To mitigate this issue, Cui *et al*. [10] use carefully selected samples for each class to re-weight the loss. Cao *et al*. [4] propose a theoretically-principled label-distribution-aware margin loss and a new training schedule DRW that defers re-weighting during training. In contrast to these methods, EQL [40] demonstrates that tail classes receive more discouraging gradients during training, and ignoring these

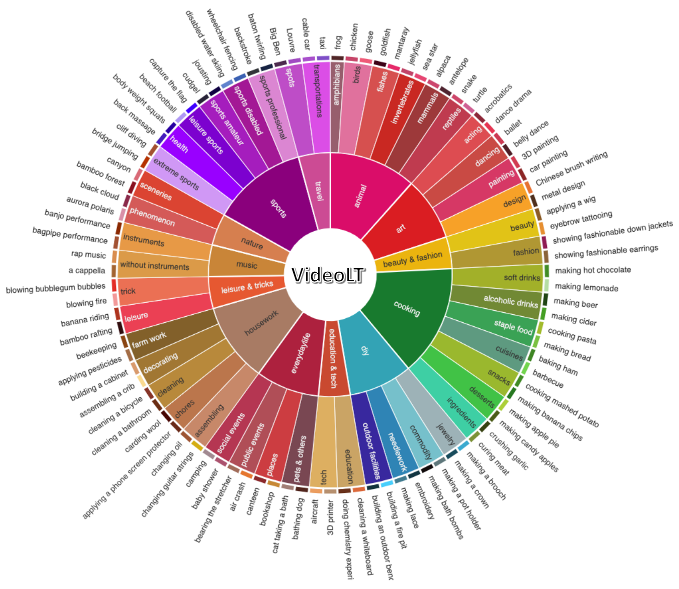

Figure 2. The taxonomy structure of VideoLT. There are 13 top-level entities and 48 sub-level entities, the children of sub-level entities are sampled. Full taxonomy structure can be found in Supplementary materials.

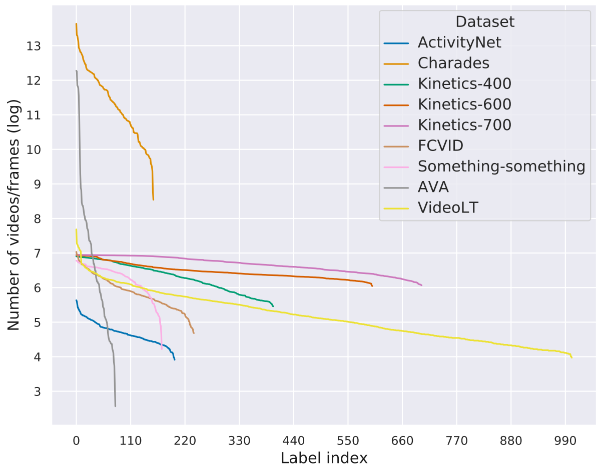

Figure 3. Class frequency distribution of existing video datasets and VideoLT. VideoLT has superior linearity in logarithmic coordinate system, which means the class frequency distribution of VideoLT is close to long-tailed distribution.

gradients will prevent the model from being influenced by those gradients. For videos, re-weighting the loss is sub-optimal as snippets that are used for training contain different amount of informative clues related to the class of interest—assigning a large weight to a background snippet from tail classes will likely bring noise for training.

Re-Sampling. There are two popular types of re-sampling: over-sampling and under-sampling. Over-sampling [8, 16] typically repeats samples from tail classes and under-sampling [11] abandons samples from head classes. Recently, class frequency is used for class-balanced sampling [22, 30, 36, 53]. BBN [53] points out that training a model in an end-to-end manner on long-tailed data can improve the discriminate power of classifiers but damages representation learning of networks. Kang *et al*. [22] show that it is possible to achieve strong long-tailed recognition performance by only training the classifier. Motivated by these observations [22, 53], we decouple feature representation and classification for long-tailed video recognition. But unlike these standard re-sampling methods, we re-sample videos by concatenating frames from different video clips.

Mixup. Mixup [51] is a popular data augmentation method that linearly interpolates two samples at the pixel level and their targets as well. There are several recent methods improving mixup from different perspectives. For example Manifold Mixup [43] extends mixup from the input space to the feature space. CutMix [50] cuts out a salient image region and pastes it to another image. PuzzleMix [24] uses salient signals without removing the local properties of inputs. ReMix [9] designs disentangled mixing factors to handle imbalanced distributions and improve the performance on minority classes. Recently, several studies show that mixup is also powerful when dealing with the long tail problem [9, 53], because it brings higher robustness and smoother decision boundaries to models and it can reduce overfitting to head classes. Our approach is similar to mixup in that we also combine two videos and mix their labels. However, FrameStack operates on frame features along the temporal dimension, but more importantly the mixing ratio in FrameStack is dynamic based on knowledge from the network model.

2.2. General & Long-tailed Video Recognition

Extensive studies have been made on video recognition with deep neural networks [7, 12, 13, 20, 26, 33, 45] or training methods [46] for video recognition. These approaches focus on learning better features for temporal modeling, by developing plugin modules [26, 45] or carefully designing end-to-end network structures [12, 13]. State-of-the-art video recognition models mainly experiment with general video recognition datasets to demonstrate their capacity in modeling long-term temporal relationships [20, 45] or capturing short-term motion dynamics [7, 12, 26, 33]. However, limited effort has been made for long-tailed video recognition due to the lack of proper benchmarks. Zhu and Yang [54] propose Inflated Episodic Memory to address long-tailed visual recognition in both image and videos. However, the use of memory banks in [54] is resource-demanding. FrameStack is computationally more efficient using in-vitro knowledge from the network for resampling, and thus is more efficient without the need to attend to memory slots. Moreover, FrameStack is a plug-in data augmentation strategy that can be easily applied to model training.

3. VideoLT Dataset

We now introduce VideoLT in detail, which is a large scale benchmark designed for long-tailed video recognition. In contrast to existing video datasets that focus on action or activities [3, 7, 23, 25, 38], VideoLT is designed to be general and to cover a wide range of daily activities. We manually define a hierarchy with 13 top-level categories including: *Animal*, *Art*, *Beauty and Fashion*, *Cooking*, *DIY*, *Education and Tech*, *Everyday life*, *Housework*, *Leisure and Tricks*, *Music*, *Nature*, *Sports* and *Travel*. See Figure 2 for details. For each top-level class, we used ConceptNet to find sub-level categories. Finally, we selected $1,004$ classes for annotation. To obtain a more diverse video dataset, we not only use the defined categories in taxonomy system, but also expand tags with the same semantics. Then we use these tags to search and crawl videos from YouTube. For each category, duplicate videos and some very long videos are removed, and the number of videos for all categories is larger than 80. In order to ensure annotation quality, each video is labelled by three annotators and a majority voting is used to determine the final labels. See supplemental materials for more details.

VideoLT is split into a training set, a validation set and a test set using 70%, 10% and 20% of videos, respectively. To better evaluate approaches for long-tailed recognition, we define 47 head classes ($\#videos > 500$), 617 medium classes ($100 < \#videos <= 500$) and 340 tail classes ($\#videos <= 100$). See Supple. for details.

Comparisons with existing video datasets We visualize in Fig. 3 the class frequency distribution of the training and validation set from ActivityNet v1.3 [3], Charades [37], Kinetics-400 [7], Kinetics-600 [5], Kinetics-700 [6], FCVID [21], Something-something v1 [15], AVA [31] and VideoLT. VideoLT has superior linearity in logarithmic coordinate system, which means the class frequency distribution of VideoLT is close to a long-tailed distribution.

It is worth noting that YouTube-8M is a large scale dataset with 3,862 classes and 6.8 million videos [1]. With so many categories, the dataset naturally has a long tail distribution of classes as ours. However, the classes in YouTube-8M are inferred by algorithms automatically rather than manually defined. Each video class has at least 200 samples for training, which is two times higher than ours. In addition, it does not provide head, medium, tail classes for better evaluations of long-tailed approaches.

4. FrameStack

We now introduce FrameStack, a simple yet effective approach for long-tailed video recognition. In image recognition tasks, input samples always correspond to their corresponding labels. However, for video recognition, snippets, which might not contain informative clues due to the weakly-labeled nature of video data, could also be sampled from video sequences for training. Popular techniques with a fixed re-sampling/re-weighting strategy for long-tailed image recognition are thus not applicable, since they will amplify noise in background snippets when calculating losses.

To mitigate imbalanced class distributions for video tasks, FrameStack re-samples training data at the frame level and adopts a dynamic sampling strategy based on knowledge learned by the network itself. The rationale behind FrameStack is to dynamically sample more frames from videos in tail classes and use fewer frames for those from head classes. Instead of directly sampling raw RGB frames to balance label distributions, we operate in the feature space by using state-of-the-art models that are able to preserve the temporal dimension in videos [13, 26] [2]. This allows FrameStack to be readily used as a plugin module for popular models to address the long-tail problem in video datasets without retraining the entire network.

More formally, we represent a video sequence with L frames as $\mathcal{V} = \{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_L\}$, and its labels as $\boldsymbol{y}$. We then use a top-notch model (more details will be described in the experiment section) to compute features for $\mathcal{V}$, and the resulting representations are denoted as $\boldsymbol{V} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_L\}$. To determine how many frames should be selected from $\boldsymbol{V}$ for training a classifier, we compute a running Average Pecision (rAP) during training to evaluate the performance of the network for each category on the entire dataset. For each mini-batch of training samples, we record their predictions and the groundtruth. After an epoch, we calculate the ap for each class on the training set. We refer to this metric as running AP since the parameters of the model are changing every mini-batch. While it is not accurate as standard average precision, it provides relative measurement about how the model performs for different classes. If the model is very confident for certain categories as suggested by rAP, we simply use fewer frames and vice versa.

However, this creates variable-length inputs for different samples in a batch, which is not parallel-friendly for current GPU architectures. In addition, it is difficult to directly translate rAP to the number of samples to be used. To address this issue, FrameStack operates on a pair of video samples, $(\boldsymbol{V}_i, \boldsymbol{y}_i), (\boldsymbol{V}_j, \boldsymbol{y}_j)$, which are randomly selected in a batch. Based on their ground-truth labels, we can obtain

[2] Most top-notch recognition models do not perform temporal downsampling till the end of networks.

the corresponding rAPs, rAP_i and rAP_j for classes $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$, respectively. We then define a temporal sampling ratio as:

$$\beta = \frac{rAP_i}{rAP_i + rAP_j}, \quad (1)$$

where β indicates the relative performance of classes $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ by the network so far. Then the number of frames that are sampled from $\boldsymbol{V}_i$ and $\boldsymbol{V}_j$ are L_i and L_j, respectively:

$$\begin{array}{l} L_i = \lfloor (1-\beta) \times L \rfloor \\ L_j = \lfloor \beta \times L \rfloor \end{array} . \quad (2)$$

We then produce two new snippets $\widehat{\boldsymbol{V}_i}$ and $\widehat{\boldsymbol{V}_j}$ with length L_i and L_j derived from $\boldsymbol{V}_i$ and $\boldsymbol{V}_j$ respectively through uniform sampling. By concatenating $\widehat{\boldsymbol{V}_i}$ and $\widehat{\boldsymbol{V}_j}$, we obtain a new sample $\widetilde{\boldsymbol{V}}$ whose length is L:

$$\widetilde{\boldsymbol{V}} = \texttt{Concat}([\widehat{\boldsymbol{V}_i}\ ;\ \widehat{\boldsymbol{V}_j}]). \quad (3)$$

Now $\widetilde{\boldsymbol{V}}$ becomes a multi-label snippet containing categories of $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$. We associate $\widetilde{\boldsymbol{V}}$ with a multi-label vector scaled by β:

$$\tilde{\boldsymbol{y}} = (1-\beta) \times \boldsymbol{y}_i + \beta \times \boldsymbol{y}_j, \quad (4)$$

Then, $\widetilde{\boldsymbol{V}}$ and $\tilde{\boldsymbol{y}}$ can then be used by temporal aggregation modules for classification. Note that at the beginning of the training process, recognition accuracies are pretty low for all categories, and thus β is not accurate. To remedy this, when $(rAP_i + rAP_j) < 1e-5$, we set β to 0.5 to sample a half of frames from the two videos. Algorithm 1 summarizes the overall training process.

It is worth pointing out that FrameStack shares similar spirit as mixup [51], which interpolates two samples linearly as data augmentations to regularize network training. Here, instead of mixing frames, we concatenate sampled video clips with different time steps to address the long-tailed video recognition problem. As will be shown in the experiments, FrameStack outperforms mixup by clear margins in the context of video classification. FrameStack can be regarded as a class-level re-balancing strategy which is based on average precision of each class, we also use focal loss [27] which adjusts binary cross-entropy based on sample predictions.

Algorithm 1: Pseudo code of FrameStack.

```
Result: Updated $rAP$ list. Updated model $f_\theta$
Input: Dataset $D = \{(\boldsymbol{V}_i, \boldsymbol{y}_i)\}_{i=1}^n$. Model $f_\theta$
Initialize $rAP = 0, \varepsilon = 1e-5$
/* $M$: videos in a mini-batch */
for $e \in MaxEpoch$ do
    for $(\boldsymbol{V}, \boldsymbol{y}) \in M$ do
        $((\boldsymbol{V}_i, \boldsymbol{y}_i)), (\boldsymbol{V}_j, \boldsymbol{y}_j)) \leftarrow$ Sampler($D$, $M$)
        if $(rAP_i + rAP_j) < \varepsilon$ then
            $\beta = 0.5$
        else
            $\beta = \frac{rAP_i}{rAP_i + rAP_j}$
        $L_i \leftarrow \lfloor (1-\beta) \times L \rfloor$
        $L_j \leftarrow \lfloor \beta \times L \rfloor$
        /* $\widehat{\boldsymbol{V}_i}$, $\widehat{\boldsymbol{V}_j}$: Uniformly sample
           $L_i$, $L_j$ frames from $\boldsymbol{V}_i$, $\boldsymbol{V}_j$
           */
        $\widehat{\boldsymbol{V}_i} \leftarrow Uniform(\boldsymbol{V}_i[L_i])$
        $\widehat{\boldsymbol{V}_j} \leftarrow Uniform(\boldsymbol{V}_j[L_j])$
        $\widetilde{\boldsymbol{V}} \leftarrow Concat[\widehat{\boldsymbol{V}_i}, \widehat{\boldsymbol{V}_j}]$
        $\tilde{\boldsymbol{y}} \leftarrow (1-\beta) \times \boldsymbol{y}_i + \beta \times \boldsymbol{y}_j$
    end
    $\mathcal{L}(f_\theta) \leftarrow \frac{1}{M} \sum_{(\widetilde{\boldsymbol{V}}, \tilde{\boldsymbol{y}}) \in M} \mathcal{L}((\widetilde{\boldsymbol{V}}, \tilde{\boldsymbol{y}}); f_\theta)$
    $f_\theta \leftarrow f_\theta - \delta \nabla_\theta \mathcal{L}(f_\theta)$
    /* $rAP$: A list of running
       average precision for each
       class */
    $rAP \leftarrow$ APCalculator
    return $rAP$
end
```

5. Experiments

5.1. Settings

Implementation Details. During training, we set the initial learning rate of the Adam optimizer to 0.0001 and decrease it every 30 epochs; we train for a maximum of 100 epochs by randomly sampling 60 frames as inputs, and the batch size is set to 128. At test time, 150 frames are uniformly sampled from raw features. For FrameStack, we use a mix ratio η to control how many samples are mixed in a mini-batch, and set η to 0.5. In addition, the clip length of FrameStack L is set to 60.

Backbone networks. To validate the generalization of our method for long-tailed video recognition, we follow the experimental settings as Decoupling [22]. We use two popular backbones to extract features including: ResNet-101 [18] pretrained on ImageNet and ResNet-50 [18] pretrained on ImageNet. We also experiment with TSM [26] using a ResNet-50 [26] as its backbone and the model is pretrained on Kinetics-400. We take the features from the penultimate layer of the network, resulting in features with a dimension of 2048. We decode all videos at 1 fps, and then resize frames to the size of 256 and crop the center of them as inputs; these frames are uniformly sampled to construct a sequence with a length of 150. Note that we do not finetune the networks on VideoLT due to computational limitations. Additionally, we hope FrameStack can serve as a plugin module to existing backbones with minimal surgery. Further, given features from videos, we mainly use a non-linear

LT-Methods	ResNet-50 Overall	ResNet-50 [500,+∞) Head	ResNet-50 [100,500) Medium	ResNet-50 [0,100) Tail	ResNet-50 Acc@1	ResNet-50 Acc@5	ResNet-101 Overall	ResNet-101 [500,+∞) Head	ResNet-101 [100,500) Medium	ResNet-101 [0,100) Tail	ResNet-101 Acc@1	ResNet-101 Acc@5
baseline	0.499	0.675	0.553	0.376	0.650	0.828	0.516	0.687	0.568	0.396	0.663	0.837
LDAM + DRW	0.502	0.680	0.557	0.378	0.656	0.811	0.518	0.687	0.572	0.397	0.664	0.820
EQL	0.502	0.679	0.557	0.378	0.653	0.829	0.518	0.690	0.571	0.398	0.664	0.838
CBS	0.491	0.649	0.545	0.371	0.640	0.820	0.507	0.660	0.559	0.390	0.652	0.828
CB Loss	0.495	0.653	0.546	0.381	0.643	0.823	0.511	0.665	0.561	0.398	0.656	0.832
mixup	0.484	0.649	0.535	0.368	0.633	0.818	0.495	0.660	0.546	0.381	0.641	0.824
Ours	**0.516**	**0.683**	**0.569**	**0.397**	**0.658**	**0.834**	**0.532**	**0.695**	**0.584**	**0.417**	**0.667**	**0.843**

Table 1. Results and comparisons of different approaches for long-tailed recognition using features extracted from ResNet-50 and ResNet-101 (aggregated with a non-linear model), our method FrameStack outperforms other long-tailed methods designed for image classification by clear margins.

classifier with two fully-connected layers to aggregate them temporally. To demonstrate the effectiveness of features, we also experiment with NetVLAD for feature encoding [2] with 64 clusters and the hidden size is set to 1024.

Evaluation Metrics. To better understand the performance of different methods for long-tailed distribution, we calculate mean average precision for head, medium and tail classes in addition to dataset-wise mAP, Acc@1 and Acc@5. Long-tailed video recognition requires algorithms to obtain good performance on tail classes without sacrificing overall performance, which is a new challenge for existing video recognition models.

5.2. Results

Comparisons with SOTA methods. We compare FrameStack with three kinds of long-tailed methods that are widely used for image recognition tasks (See details and other extensions including square-root sampling and two-stage methods in the Supplementary material.):

- **Re-sampling**: We implement class-balanced sampling (CBS) [22, 36] which uses equalized sampling strategies for data from different classes. In a mini-batch, it takes a random class and then randomly samples a video, and thus videos from head and tail classes share the same probability to be chosen.

- **Re-weighting**: It takes sampling frequency of each class into consideration to calculate weights for cross-entropy or binary cross-entropy. We conduct experiments with, Class-balanced Loss [10], LDAM Loss [4] and EQL [40].

- **Data augmentation**: We use the popular method Mixup [51] for comparisons. For fair comparisons, mixup is performed in the feature space as FrameStack. In particular, mixup mixes features from two videos in a mini-batch by summing their features frame-by-frame.

Table 1 summarizes the results and comparisons of different approaches on VideoLT. We observe that the performance of tail classes are significantly worse compared to that of head classes using both ResNet-50 and ResNet-101 for all methods. This highlights the challenge of long-tailed video recognition algorithms. In addition, we can see popular long tail recognition algorithms for image classification tasks are not suitable for video recognition. Class-balanced sampling and class-balanced losses result in slightly performance drop, compared to the baseline model without using any re-weighting and re-sampling strategies; LDAM+DRW and EQL achieve comparable performance for overall classes and tail classes. For mixup, its performance is even worse compared to the baseline model, possibly due to the mixing between features makes training difficult. Instead, our approach achieves better results compared with these methods. In particular, when using ResNet-101 features, FrameStack achieves an overall mAP of 53.2%, which is 1.6% and 1.4% better compared to the baseline and the best performing image-based method (*i.e.*, LDAM+DRW and EQL). Furthermore, we can observe that although CB Loss brings slightly better performance on tail classes, this comes at the cost of performance drop for overall classes. Compared to the CB Loss, FrameStack significantly improves the performance by 2.1% for tail classes without sacrificing the overall results.

Extensions with more powerful backbones. We also experiment with a TSM model using a ResNet-50 as its backbone to demonstrate the compatibility of our approach with more powerful networks designed for video recognition. In addition, we use two feature aggregation methods to derive a unified representations for classification. The results are summarized in Table 2. We observe similar trends as Ta-

	LT Methods	Overall	[500,+∞) Head	[100,500) Medium	[0,100) Tail
Nonlinear Model	baseline	0.565	0.757	0.620	0.436
	LDAM + DRW	0.565	0.750	0.620	0.439
	EQL	0.567	0.757	0.623	0.439
	CBS	0.558	0.733	0.612	0.435
	CB Loss	0.563	0.744	0.616	0.440
	Mixup	0.548	0.736	0.602	0.425
	Ours	**0.580**	**0.759**	**0.632**	**0.459**
NetVLAD Model	baseline	0.660	0.803	0.708	0.554
	LDAM + DRW	0.627	0.779	0.675	0.519
	EQL	0.665	**0.808**	**0.713**	0.557
	CBS	0.662	0.806	0.708	0.558
	CB Loss	0.666	0.801	0.712	**0.566**
	Mixup	0.659	0.800	0.706	0.556
	Ours	**0.667**	0.806	**0.713**	**0.566**

Table 2. Results and comparisons using TSM (ResNet-50). Top: features aggregated using a non-linear model; Bottom: features aggregated using NetVLAD.

ble 1 using a nonlinear model—FrameStack outperforms image-based long-tailed algorithms by 1.5% and 2.3% for overall and tail classes, respectively. In addition, we can see that features from the TSM model pretrained on Kinetics are better than image-pretrained features (58.0% *vs*. 53.2%). Furthermore, we can see that our approach is also compatible with more advanced feature aggregation strategies like NetVLAD. More specifically, with NetVLAD, FrameStack outperforms the baseline approach by 0.7% and 1.2% for overall classes and tail classes, respectively.

5.3. Discussion

We now conduct a set of studies to justify the contributions of different components in our framework and provide corresponding discussion.

Effectiveness of AP. Throughout the experiments, we mainly use AP as a metric to adjust the number of frames used in FrameStack. To test the effectiveness of AP, we also experiment with a constant β and class frequency. The results are summarized in Table 3. For a constant β, we test a variant of β with $\beta = 0.5$ throughout the training which takes the same number of frames for all classes. Results show the overall mAP on Nonlinear and NetVLAD model decrease at 10.2% and 2.0% respectively compared to the baseline, which suggests that sampling more frames for tail classes and fewer frames for head classes is a more practical strategy in long-tailed video recognition scenario. Class frequency is another popular metric widely used for image-based long-tailed recognition [22, 30, 36, 53]. In

Model	CB Strategy	Overall	[500,+∞) Head	[100,500) Medium	[0,100) Tail
Nonlinear	-	0.516	0.687	0.568	0.396
	$\beta = 0.5$	0.414	0.589	0.460	0.308
	w/ CF	0.520	0.680	0.571	0.405
	w/ rAP	**0.532**	**0.695**	**0.584**	**0.417**
NetVLAD	-	0.668	0.775	**0.707**	0.584
	$\beta = 0.5$	0.648	0.758	0.684	0.567
	w/ CF	0.663	0.767	0.699	0.584
	w/ rAP	**0.670**	**0.780**	**0.707**	**0.590**

Table 3. Results and comparisons of using running AP and other variants of β to determine how many frames should be used from video clips.

particular, we take the inverse frequency of each class to compute the number of frames sampled for head and tail classes, and then concatenate two clips as FrameStack. We observe that resampling videos with class frequency results in 0.5% performance drop on the NetVLAD model. In contrast, using running average precision is a better way for resampling frames since it is dynamically derived based on knowledge learned by networks so far. As a result, it changes sampling rates based on the performance of particular classes, which prevents overfitting for top-performing classes and avoids under-fitting for under-performing categories at the same time. As aforementioned, treating weakly labeled videos as images and then resampling/reweighting them using class frequency might be problematic because some snippets might consist of background frames.

Effectiveness of loss functions. As mentioned above, our approach resamples data from different classes and it is trained with focal loss. We now investigate the performance of our approach with different loss functions and the results are summarized in Table 4. We observe that using FrameStack is compatible with both loss functions, outperforming the baseline model without any resampling/reweighting strategies. For the nonlinear model, FrameStack achieves better performance, while for NetVLAD, FrameStack with binary cross-entropy loss(BCE) is slightly better.

Effectiveness of mixing ratio. We also investigate how many samples that are mixed in a mini-batch, determined by a mixing ratio η, influence the performance. From Table 5 we find that as η increases, the performance of overall and tail classes increases at the beginning and then decreases—$\eta = 0.5$ reaches highest performance which is adopted in our FrameStack. It suggests that mixing all data in an epoch makes training more difficult.

FrameStack *vs*. Mixup. We compare the performance of FrameStack and Mixup for the overall and tail classes.

		TSM (ResNet-50)						ResNet-101					
	LT-Methods	Overall	[500,+∞) Head	[100,500) Medium	[0,100) Tail	Acc@1	Acc@5	Overall	[500,+∞) Head	[100,500) Medium	[0,100) Tail	Acc@1	Acc@5
NonLinear	baseline	0.565	0.757	0.620	0.436	0.680	0.851	0.516	0.687	0.568	0.396	0.663	0.837
	FrameStack-BCE	0.568	0.751	0.622	0.445	0.679	0.855	0.521	0.684	0.571	0.406	0.660	0.839
	FrameStack-FL	**0.580**	**0.759**	**0.632**	**0.459**	**0.686**	**0.859**	**0.532**	**0.695**	**0.584**	**0.417**	**0.667**	**0.843**
NetVLAD	baseline	0.660	0.803	0.708	0.554	0.695	0.870	0.668	0.775	0.707	0.584	0.700	**0.864**
	FrameStack-BCE	**0.669**	**0.807**	**0.715**	**0.568**	**0.711**	**0.872**	**0.671**	**0.781**	0.707	0.589	0.709	0.858
	FrameStack-FL	0.667	0.806	0.713	0.566	0.708	0.866	0.670	0.780	0.707	**0.590**	**0.710**	0.858

Table 4. Results of our approach using different loss functions and comparisons with baselines. FrameStack is complemented with focal loss on Nonlinear model and TSM (ResNet-50), ResNet-101 features.

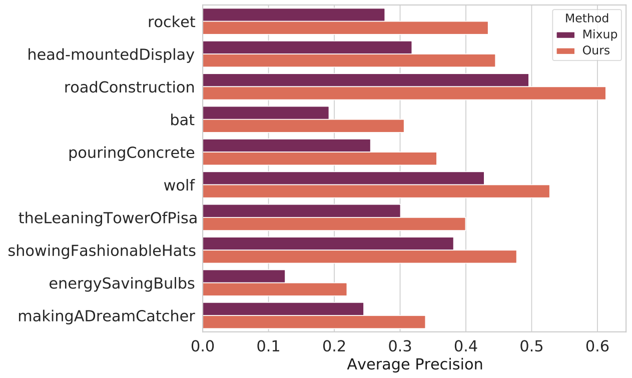

Figure 4. Top 10 among 1004 classes that FrameStack surpasses Mixup. 40% classes are action classes.

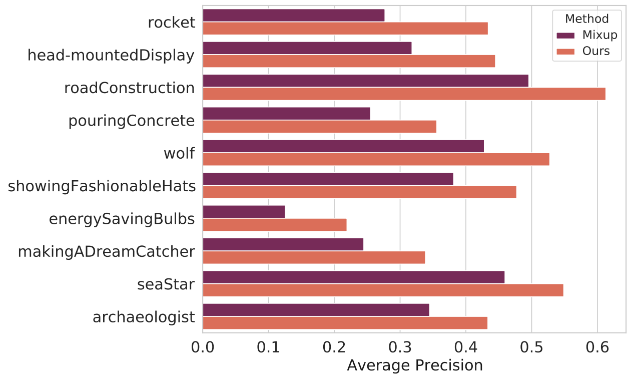

Figure 5. Top 10 among 340 tail classes that FrameStack surpasses Mixup. Comparing across Figure 4, we see FrameStack achieves better performance mainly on tail clases.

Specifically, we compute the difference of average precision for each class between FrameStack and Mixup. In Figure 4, we visualize the top 10 classes from 1,004 classes that FrameStack outperforms Mixup and find that 40% of them are action classes. When comparing Figure 4 and Figure 5 together, we observe that the 80% of the top 10 classes are tail classes, which shows FrameStack is more effective than Mixup especially in recognizing tail classes.

η	Overall	[500,+∞) Head	[100,500) Medium	[0,100) Tail	Acc@1	Acc@5
0	0.668	0.775	0.707	0.584	0.700	**0.864**
0.3	0.667	0.780	0.707	0.586	0.710	0.860
0.5	**0.670**	**0.780**	0.707	**0.590**	0.710	0.858
0.7	0.669	0.780	0.707	0.585	0.709	0.860
0.9	0.668	0.774	0.704	0.588	0.706	0.859

Table 5. Effectiveness of the mixing ratio η, test results are based on ResNet-101 feature and NetVLAD Model.

6. Conclusion

This paper introduced a large-scale long-tailed video dataset—VideoLT with an aim to advance research in long-tailed video recognition. Long-tailed video recognition is a challenging task because videos are usually weakly labeled. Experimental results show existing long-tailed methods that achieve impressive performance in image tasks are not suitable for videos. In our work, we presented FrameStack, which performs sampling at the frame level by using running AP as a dynamic measurement. FrameStack adaptively selects different number of frames from different classes. Extensive experiments on different backbones and aggregation models show FrameStack outperforms all competitors and brings clear performance gains on both overall and tail classes. Future directions include leveraging weakly-supervised learning [32, 44], self-supervised learning [17, 48] methods to solve long-tailed video recognition.

Acknowledgement

This work was supported in part by National Natural Science Foundation of China (#62032006).

References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016. 4

[2] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *CVPR*, 2016. 6

[3] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015. 4

[4] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *NIPS*, 2019. 1, 2, 6

[5] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018. 4

[6] Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset. *arXiv preprint arXiv:1907.06987*, 2019. 4

[7] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 3, 4

[8] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *JAIR*, 2002. 1, 3

[9] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: Rebalanced mixup. In *ECCV*, 2020. 3

[10] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, 2019. 1, 2, 6

[11] Chris Drumnond. Class imbalance and cost sensitivity: Why undersampling beats oversampling. In *ICML-KDD Workshop*, 2003. 1, 3

[12] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *CVPR*, 2020. 3

[13] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 3, 4

[14] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 1

[15] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, 2017. 4

[16] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *ICIC*, 2005. 1, 3

[17] Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. In *NIPS*, 2020. 8

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 5

[19] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 1

[20] Noureldien Hussein, Efstratios Gavves, and Arnold WM Smeulders. Timeception for complex action recognition. In *CVPR*, 2019. 3

[21] Yu-Gang Jiang, Zuxuan Wu, Jun Wang, Xiangyang Xue, and Shih-Fu Chang. Exploiting feature and class relationships in video categorization with regularized deep neural networks. *IEEE TPAMI*, 2017. 4

[22] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 2020. 1, 3, 5, 6, 7

[23] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 4

[24] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *ICML*, 2020. 3

[25] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, 2011. 4

[26] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, 2019. 3, 4, 5

[27] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1, 5

[28] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1

[29] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019. 1

[30] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018. 3, 7

[31] Naila Murray, Luca Marchesotti, and Florent Perronnin. Ava: A large-scale database for aesthetic visual analysis. In *CVPR*, 2012. 4

[32] Phuc Nguyen, Ting Liu, Gautam Prasad, and Bohyung Han. Weakly supervised action localization by sparse temporal pooling network. In *CVPR*, 2018. 8

[33] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *ICCV*, 2017. 3

[34] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 1

[35] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1

[36] Li Shen, Zhouchen Lin, and Qingming Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *ECCV*, 2016. 3, 6, 7

[37] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016. 4

[38] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 4

[39] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 1

[40] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, 2020. 1, 2, 6

[41] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 1

[42] Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *NIPS*, 2020. 1

[43] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019. 2, 3

[44] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *CVPR*, 2017. 8

[45] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3

[46] Chao-Yuan Wu, Ross Girshick, Kaiming He, Christoph Feichtenhofer, and Philipp Krahenbuhl. A multigrid method for efficiently training video models. In *CVPR*, 2020. 3

[47] Tong Wu, Qingqiu Huang, Ziwei Liu, Yu Wang, and Dahua Lin. Distribution-balanced loss for multi-label classification in long-tailed datasets. In *ECCV*, 2020. 1

[48] Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In *CVPR*, 2019. 8

[49] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. In *NIPS*, 2020. 1

[50] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 2, 3

[51] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 3, 5, 6

[52] Xiao Zhang, Zhiyuan Fang, Yandong Wen, Zhifeng Li, and Yu Qiao. Range loss for deep face recognition with long-tailed training data. In *ICCV*, 2017. 1

[53] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *CVPR*, 2020. 1, 3, 7

[54] Linchao Zhu and Yi Yang. Inflated episodic memory with region self-attention for long-tailed visual recognition. In *CVPR*, 2020. 3