Supplementary Material:
TimeBalance: Temporally-Invariant and Temporally-Distinctive Video Representations for Semi-Supervised Action Recognition

Ishan Rajendrakumar Dave, Mamshad Nayeem Rizve, Chen Chen, Mubarak Shah
Center for Research in Computer Vision, University of Central Florida, Orlando, USA
{ishandave, nayeemrizve}@knights.ucf.edu, {chen.chen, shah}@crcv.ucf.edu

A. Overview

• Section B: Implementation details about network architectures and training setup.
• Section C: Ablation study for our framework.
• Section D: Supportive diagrams and explanation for our method.

B. Implementation Details

B.1. Network Architecture

B.1.1 Backbone

For teacher models \( f_I \) and \( f_D \), we utilize 3D-ResNet50 model from the implementation of Slow-R50 \cite{2} of official PyTorchVideo. For experiments with 3D-ResNet18, we utilize its official PyTorch implementation r3d18\(^2\).

B.1.2 Non-Linear Projection Head

We use non-linear projection head \( g(\cdot) \) during the self-supervised pretraining of temporally-invariant and temporally-distinctive teachers to reduce the dimensions of the representation. We utilize Multi-layer Perceptron (MLP) as a non-linear projection head to project 2048-dimensional model features to 128-dimensional vectors in normalized representation space. The design of MLP is as follows, where \( \text{nn} \) indicates \texttt{torch.nn} PyTorch package:

\[
\begin{align*}
\text{nn.Linear}(2048, 512, \text{bias} = \text{True}) \\
\text{nn.BatchNorm1d}(512) \\
\text{nn.ReLU}(\text{inplace}=\text{True}) \\
\text{nn.Linear}(512, 128, \text{bias} = \text{False}) \\
\text{nn.BatchNorm1d}(128)
\end{align*}
\]

B.2. Training Details

For all weight updates, we utilize Adam Optimizer \cite{3} with default parameters \( \beta_1 = 0.9 \) and \( \beta_2 = 0.999 \) with a base learning rate \((\alpha_I, \alpha_D, \alpha_S)\) of 1e-3. For all training, we utilize a linear warmup of 10 epochs. A patience-based learning rate scheduler is also used, which drops the learning rate to its half value on a loss plateau.

C. Additional Ablations

C.1. Loss function for teacher supervision

In order to distill teacher knowledge, we study three different loss functions as \( L_{\text{unsup}} \) and report the results in Table 1. For these experiments, we use 3D-Resnet50 as the student model on the UCF101 dataset \cite{4}. We observe that all three losses perform reasonably while \( L_2 \) performs the best, which we use as the default loss in our method.

<table>
<thead>
<tr>
<th>Unlabeled Supervision</th>
<th>UCF101 % Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5%</td>
</tr>
<tr>
<td>( L_2 )</td>
<td>53.48</td>
</tr>
<tr>
<td>KL-Divergence</td>
<td>52.62</td>
</tr>
<tr>
<td>JS-Divergence</td>
<td>50.91</td>
</tr>
</tbody>
</table>

Table 1. Ablation of different Teacher Losses. \( L_2 \) distillation loss performs the best, which we use in our default setting.

C.2. Student \( f_S \) from Scratch

We perform experiments with student from random initialization and compare them with the prior methods in Table 2.

D. Method

D.1. Loss for Labeled and Unlabeled set

In Fig. 1, we show the handling of labeled and unlabeled data in the semi-supervised training of student \( f_S \). For la-
<table>
<thead>
<tr>
<th>Backbone</th>
<th>UCF101 5%</th>
<th>UCF101 20%</th>
<th>HMDB 40%</th>
<th>HMDB 60%</th>
<th>Kinetics 10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours (scratch student) R3D-50</td>
<td><strong>53.1 (+26.1)</strong></td>
<td><strong>83.0 (+31.3)</strong></td>
<td><strong>52.1 (+19.2)</strong></td>
<td><strong>54.3 (+15.4)</strong></td>
<td><strong>60.8 (+2.4)</strong></td>
</tr>
<tr>
<td>Ours (scratch student) R3D-18</td>
<td>46.7 (+1.9)</td>
<td>78.2 (+2.1)</td>
<td>49.1 (+2.6)</td>
<td>52.9 (+3.2)</td>
<td>54.4 (+0.7)</td>
</tr>
</tbody>
</table>

Table 2. Experiments with Student trained from Random Initialization. (+n) shows absolute improvement over the prior best work.

![Loss computations](image)

Figure 1. Loss computations in labeled and unlabeled data. (a) In case of Labeled data, the student \( f_S \) gets supervision from supervised cross-entropy loss from label \( y^{(i)} \) and unsupervised \( L_2 \) loss from teachers. (b) For unlabeled set, the student is only trained with the unsupervised loss from teachers. Details are in Sec 3.2 of the main paper.

D.2. Temporally-Distinctive pretraining using unpooled features

Since \( L_{D1} \) deals with temporally-pooled(averaged) features, it promotes temporal-distinctiveness for the pooled features. Similar to that, [1] designs a contrastive objective that promotes temporally-distinctive representation on the unpooled features. We call it unpooled temporal-distinctive objective \( L_{D2} \), which is illustrated in Fig. 2.

References


[5] Junfei Xiao, Longlong Jing, Lin Zhang, Ju He, Qi She, Zong-wei Zhou, Alan Yuille, and Yingwei Li. Learning from temporal gradient for semi-supervised action recognition. In Pro-
Figure 2. Temporally-Distinctive Contrastive Objective for Temporally-unpooled features $\mathcal{L}_{D2}$: A time-duration of the video can be represented in two different ways: (1) Pooled features of the short(local) clip (2) Unpooled feature slice of the long(global) clip. In this contrastive objective, we maximize the agreement between temporally-aligned pooled and unpooled features.

---
