Reproducibility Companion Paper: Self-supervised Video Representation Learning Using Inter-intra Contrastive Framework

Li Tao
The University of Tokyo
Tokyo, Japan
taoli@hal.t.u-tokyo.ac.jp

Xueting Wang*
The University of Tokyo
Tokyo, Japan
wang_xueting@cyberagent.co.jp

Toshihiko Yamasaki
The University of Tokyo
Tokyo, Japan
yamasaki@hal.t.u-tokyo.ac.jp

Jingjing Chen
Fudan University
Shanghai, China
chenjingjing.tju@gmail.com

Steven Hicks
SimulaMet & Oslo Metropolitan University
Oslo, Norway
steven@simula.no

ABSTRACT

In this companion paper, we provide details of the artifacts to support the replication of “Self-supervised Video Representation Learning Using Inter-intra Contrastive Framework [5]”, which was presented at MM’20. The Inter-intra Contrastive (IIC) framework aims to extract more discriminative temporal information by extending intra-negative samples in contrastive self-supervised learning. In this paper, we first summarize our contribution. Then we explain the file structure of the source code and detailed settings. Since our proposal is a framework which contain a lot of different settings, we provide some custom settings to help other researchers to use our methods easily. The source code is available at <https://github.com/BestJuly/IIC>.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision representations**; **Activity recognition and understanding**; • **Information systems** → **Information extraction**.

KEYWORDS

Self-supervised learning, video representation, video recognition, video retrieval, spatio-temporal convolution

ACM Reference Format:

Li Tao, Xueting Wang*, Toshihiko Yamasaki, Jingjing Chen, and Steven Hicks. 2021. Reproducibility Companion Paper: Self-supervised Video Representation Learning Using Inter-intra Contrastive Framework. In *Proceedings of the 29th ACM International Conference on Multimedia (MM ’21)*, October 20–24, 2021, Virtual Event, China. ACM, New York, NY, USA, 3 pages. <https://doi.org/10.1145/3474085.3477939>

* Currently work at CyberAgent AI Lab, Inc.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MM ’21, October 20–24, 2021, Virtual Event, China

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8651-7/21/10...\$15.00

<https://doi.org/10.1145/3474085.3477939>

Figure 1: General idea of proposed method. Given video x_i , different views of this video are treated as positives, and those features are constrained to be close to each other. Data from other videos are treated as negatives. Temporal relations in the anchor view will be broken down to generate intra-negative samples, which are also treated as negatives to help the model learn temporal information.

1 CONTRIBUTION SUMMARY

We proposed an inter-intra contrastive (IIC) learning framework, which can not only learn discriminative features among video samples, but also distinguish temporal differences when original temporal information is changed. Traditional contrastive learning [6, 9] utilizes anchor, positive, and negative samples, and these samples are from different instances (i.e., videos). With IIC, we generate negative samples by breaking temporal relations in the anchor video. In this way, with contrastive self-supervised learning technologies, our framework can better extract temporal information from videos. The overview of our proposed framework is shown in Fig. 1. And we provide different option configurations in our framework such as data modalities, network backbones, intra-negative generation functions, etc.

2 ARTIFACTS DESCRIPTION

The artifacts mainly include the dataset, the source code, and the experimental environment settings. For reproducibility, we will introduce these part in details one by one.

2.1 Dataset

In our original paper [5], we used the UCF101 [4] and HMDB51 [2] datasets. The UCF101 dataset consists of 13,320 videos in 101 action categories. HMDB51 is comprised of 7,000 videos with a total of 51 action classes. We use three official splits for both UCF101¹ and HMDB51² datasets. To better eliminate artifacts from decoding videos, we use pre-computed images/frames from this repository³. For data splits, we also provide corresponding files in our codes, which has been transformed for convenient use.

2.2 Source code structure

Our source codes include 4 folders, paralleling with 4 python scripts under the first level directory.

- **data**: containing training split information of UCF101 and HMDB51 datasets, transformed from official splits.
- **datasets**: containing codes for data preparation, which load video clips from frame folders.
- **lib**: containing computation of contrastive learning, including intra-negative version of Noise Contrastive Estimation (NCE) loss and the use of memory bank [9].
- **models**: containing different network backbones, including R3D [1], R(2+1)D [8], and C3D [7]. Detailed network architectures might be different from original papers and we follow previous works [3, 10] to construct these backbones.
- **train_ssl.py**: working as the main file for our IIC, which includes different settings in self-supervised video representation learning.
- **gen_neg.py**: working as the supporting file to generate intra-negative samples, jointly used with train_ssl.py.
- **retrieve_clips.py**: working as the evaluation file for learned features by using video retrieval as the evaluation task.
- **ft_classify.py**: working as the training and evaluation file for the downstream task, video recognition.
- **README_for_reproducibility**: working as the description for the source code and related information for this companion.
- **README.md**: working as the description for our github repo, containing different content but can be used to refer.

2.3 Experimental environment

Our source code is tested in the following environment.

- **Operating system**: Ubuntu 18.04.5 LTS with CPU Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz. The detailed Linux release is Linux version 4.15.0-122-generic (buildd@lgw01-amd64-036).

Table 1: Important parameters in self-supervised training that can be customized.

Parameter	Description	Default Value
batch_size	Training batch size	16
epochs	Total training epochs	240
learning rate	Initial learning rate for optimization	0.01
model	Choice for network backbone	r3d
dataset	Choice for training dataset	ucf101
modality	Choice for data modality	res
neg	Choice for intra-negative generation function	repeat
nce_k	Number of negative samples in NCE loss	1024
seed	Settings for random seeds	632

- **CUDA and GPU**: CUDA 10.0 with GPU Tesla V100-SXM2-32GB. Note that cuDNN should be disabled to ensure the 100% reproducible results for each run at any time.
- **Python**: Python 3.7.4 compiled with GCC 8.4.0. Python packages: torch==1.2.0, numpy==1.16.2, torchvision==0.4.0, tqdm==4.42.1, scikit-learn==0.21.3, accimage==0.1.1, pandas==0.25.3, Pillow==6.2.0, tensorboardX==1.9, tensorboard-logger==0.1.0.

3 EXPERIMENTS

For the training and evaluation of self-supervised video representation learning, the main training part is the self-supervised feature learning and video-level retrieval and video recognition should be treated as downstream tasks.

3.1 Self-supervised feature learning

We first provide important parameters in our framework in Table 1. There are some other hyper-parameter settings which are not listed in the table because they are not that important in training artifacts and we have freeze them in the scripts.

Experimental results can be reproduced by the script:

```
python train_ssl.py
```

which is equal to

```
python train_ssl.py --dataset=ucf101 --model=r3d
--modality=res --neg=repeat
```

Please not that our proposal is a framework, these explicit parameters can be replaced to fit other settings such as different datasets, different network backbones, different data modalities, and different intra-negative generation functions.

The training process will last for around 30 hours in our experimental settings using one GPU, and the trained models will be automatically saved in the ./ckpt directory. The last checkpoint ckpt_epoch_240.pth will be used in evaluation with downstream tasks.

Because our IIC is a framework containing different training settings, the corresponding settings should be specified in the further use of the model, especially for the network backbone and the data modality.

¹<https://www.crcv.ucf.edu/data/UCF101.php>

²<https://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database/>

³<https://github.com/feichtenhofer/twostreamfusion>

Table 2: Important parameters in video recognition.

Parameter	Description	Default Value
batch_size	Training batch size	16
epochs	Total training epochs	150
learning rate	Initial learning rate for optimization	1e-3
dataset	Choice for training dataset	ucf101
seed	Settings for random seeds	632

3.2 Evaluation 1: directly on learned features using video retrieval

Video retrieval is conducted with video-level features. Several video clip features are averaged when they are from the same video to represent the video feature. If the retrieved video has the same action category as the query one, it will be recorded as a successful case to calculate the accuracy. The evaluation script is

```
python retrieve_clips.py
    --ckpt=/path/to/ckpt_epoch_240.pth
```

which is equal to

```
python retrieve_clips.py
    --ckpt=/path/to/ckpt_epoch_240.pth --model=r3d
    --modality=res --dataset=ucf101 --merge=True
```

Here, the “--dataset” specifies the evaluation dataset, which can be unrelated to the training part to test the transferability of the learned features. Our IIC used only one model to handle different data modalities. Therefore, for the retrieval evaluation, we jointly use features from different modalities by default or use one data modality alone or by setting “--merge=False”.

The procedure for this part includes feature extraction and video-level retrieval. And the retrieval results will be shown in the screen, with top1, top5, top10, top20, and top50 retrieval accuracies.

3.3 Evaluation 2: fine-tuning in video recognition

The script for this part is

```
python ft_classify.py
    --ckpt=/path/to/ckpt_epoch_240.pth --model=r3d
    --modality=res --dataset=ucf101
```

In the downstream task video recognition, the previous trained model (i.e. ckpt_epoch_240.pth) is used to initialize model parameters. Hyper-settings such as network backbone and data modality should be the same as the previous trained model. For some other settings, we list important parts in Table 2 and validation will be conducted every epoch. The model with best validation performance will be saved and tested for video recognition. After training, testing will be automatically performed and the recognition accuracy will be shown on the command window.

4 REPRODUCIBILITY EFFORTS

In the code version for the original paper, some artifacts for random seeds are not fixed, and results will be changed for each run. We tried our best and find that by using the following codes, we can

ensure all the artifacts with randomness can be eliminate and the exactly the same results can be achieved for each run at any time in the same experimental environment.

```
seed = args.seed
torch.backends.cudnn.deterministic = True
torch.backends.cudnn.benchmark = False
random.seed(seed)
np.random.seed(seed)
os.environ['PYTHONHASHSEED'] = str(seed)
torch.manual_seed(seed)
torch.cuda.manual_seed(seed)
torch.cuda.manual_seed_all(seed)
```

5 CONCLUSION

In this paper, we provided the details of the artifacts of the paper “Self-supervised Video Representation Learning Using Inter-intra Contrastive Framework” for replication. The artifacts contain the dataset and the source code for experiments in the paper. Taking advantage of the source code, the experiments can be operated and customized.

ACKNOWLEDGMENTS

This work was partially financially supported by the Grants-in-Aid for Scientific Research Numbers JP19K20289 and JP18H03339 from Japan Society for the Promotion of Science (JSPS).

REFERENCES

- [1] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. 2018. Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA*. 18–22.
- [2] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. 2011. HMDB: a large video database for human motion recognition. In *2011 International Conference on Computer Vision*. IEEE, 2556–2563.
- [3] Dezhao Luo, Chang Liu, Yu Zhou, Dongbao Yang, Can Ma, Qixiang Ye, and Weiping Wang. 2020. Video Cloze Procedure for Self-Supervised Spatio-Temporal Learning. *arXiv preprint arXiv:2001.00294* (2020).
- [4] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012).
- [5] Li Tao, Xueting Wang, and Toshihiko Yamasaki. 2020. Self-supervised video representation learning using inter-intra contrastive framework. In *Proceedings of the 28th ACM International Conference on Multimedia*. 2193–2201.
- [6] Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2019. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849* (2019).
- [7] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*. 4489–4497.
- [8] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. 2018. A Closer Look at Spatiotemporal Convolutions for Action Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6450–6459.
- [9] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3733–3742.
- [10] Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. 2019. Self-supervised spatiotemporal learning via video clip order prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 10334–10343.