University of Amsterdam

MSc Artificial Intelligence
Master Thesis

Self-supervised Video Representation Learning with Cross-Stream Prototypical Contrasting

---

by
Martine Toering
11302925

March 3, 2023

48 ECTS
Project Time: November 2020 - July 2021

Supervisor:
Vincent Tao Hu, MSc

Assessor:
Prof. Dr. Cees G. M. Snoek

Co-supervisors:
Ioannis Gatopoulos, MSc
Maarten C. Stol, MSc

BrainCreators

Abstract

Instance-level contrastive learning techniques, which rely on data augmentation and a contrastive loss function, have found great success in the domain of visual representation learning. They are not suitable for exploiting the rich dynamical structure of video however, as operations are done on many augmented instances. We propose "Video Cross-Stream Prototypical Contrasting", a novel method which predicts consistent prototype assignments from both RGB and optical flow views, operating on sets of samples. Specifically, we alternate the optimization process; while optimizing one of the streams, all views are mapped to one set of stream prototype vectors. Each of the assignments is predicted with all views except the one matching the prediction, pushing representations closer to their assigned prototypes. As a result, more efficient video embeddings with ingrained motion information are learned, without the explicit need for optical flow computation during inference. We obtain state-of-the-art results on nearest-neighbour video retrieval and action recognition, outperforming previous works on UCF101 and HMDB51 using the S3D and the R(2+1)D backbones.

Keywords

Video Understanding, Self-supervised Learning,
Representation Learning, Action Recognition,
Video Retrieval

Acknowledgments

The completion of any master thesis depends on the encouragement and guidelines of many others. I would like to take this opportunity to express my gratitude to the people and institutions without whom I could not have been able to complete my studies.

I would like to deeply thank my supervisor, Vincent Tao Hu, for his continued support and help. He has been instrumental for my academic journey. Throughout the project, he has provided me with many valuable inputs and was always present to help me. I am extremely grateful.

Similarly, words can not sum up the gratitude that I owe my mentor, Ioannis Gatopoulous. Thank you for taking me under your wing, giving me valuable perspectives on my work and being so pleasant to work with during the entire project.

I would like to thank the University of Amsterdam and the master's programme, for providing me with resources to perform experiments, and Prof. Cees Snoek for being part of my thesis committee.

I would like to thank BrainCreators and my fellow interns, for the environment and support I received while conducting the thesis. In particular, I would like to thank Maarten Stol for providing me with this opportunity and for the trust he had in me during the internship.

Most of all, I would like to thank Alex, for always being there for me and for giving me all the love, kindness and support.

Lastly, this would not have been possible without friends and family, especially my parents. Thank you all.

Contents

1 Introduction

The ability to understand motion is one of the most essential functions of our visual system. Despite significant advances in recent years, video understanding remains one of the key tasks in computer vision. Many systems in deployment for visual inspections or autonomous vehicles rely on processing static frames, while the ever-changing world around us necessitates interpreting movements. With the abundance of video in everyday life, retrieval of videos based solely on their content is becoming increasingly relevant for filtering and recommendation.

1.1 Context

Video recognition methods have been largely driven by the image domain, where deep learning has led to outstanding breakthroughs (Deng et al., 2009; He et al., 2016; Krizhevsky et al., 2012). Incorporating prior knowledge through pretraining is widely used to ease the dependency on scarcely available labeled data. However, replicating the successes of image methods for video is challenging because of greater practical disadvantages with gathering, storing, and manually annotating a substantial amount of data. As complex models overfit more easily, it is evident that larger datasets are needed. Moreover, annotating videos is a laborious and complex process because of the unclear definition of human actions. Such ambiguities could lead to a label space gap in datasets and consequently between models. In many practical applications, the familiar pretraining and finetuning mechanism would not transfer well to the video domain, with solutions remaining task-specific. It can even be argued that especially video is too high-dimensional for direct supervision, as labels can not capture inherent structure.

Representation learning methods that do not require labels encode generic knowledge from the data. Learning robust and generalizable representations is a fundamental goal of machine vision. Besides, it is more consistent with the workings of our visual system since humans do not learn from thousands of labeled examples. Self-supervised learning techniques can obtain supervisory signals from the data itself. Methods based on instance-level contrasting have significantly reduced the gap with supervised learning in image-based tasks (Chen et al., 2020b; He et al., 2020; van den Oord et al., 2019) and video (Han et al., 2020b; Qian et al., 2021). These contrastive learning frameworks require an augmentation module that obtains multiple views of one instance, and a loss function that contrasts between augmented views of instances. The objective can be viewed as instance discrimination: producing higher similarity scores between augmentations of the same instances, rather than with those that belong to different ones (*negative examples*). This learning paradigm can be thought of as more biologically plausible since humans can learn abstract relationships between concepts through their distinguishing features naturally.

1.2 Motivation

Given the issues raised, the objective of this thesis is to examine new approaches for video recognition tasks without needing labels. Self-supervision alleviates issues associated with

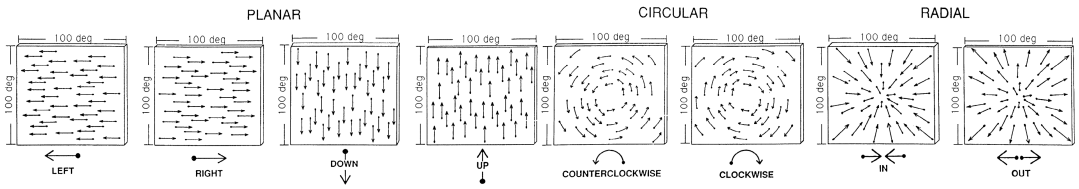

Figure 1.1: **Directionally selective neurons** in the dorsomedial region of the medial superior temporal area (MSTd) in the brain are sensitive one or several of the planar, circular or radial motion stimuli shown.

annotation by exploiting correlation present in the data. Video in particular contains a rich inherent structure because of spatiotemporal coherence and motion. Besides learning useful representations for video-based tasks, we aim to create a more plausible learning framework for video while contributing to the shift in research towards including generic prior knowledge in vision models. Recent contrastive instance learning methods rely on approximating the loss via noise-contrastive estimation (NCE), i.e. reducing the number of pairwise comparisons by sampling negative examples from the data. Despite the success of these methods, they suffer from a few fundamental issues. The methods rely heavily on data augmentation to create multiple views of instances in order to learn powerful representations. Furthermore, the number of pairwise comparisons to negative examples necessary for good representations is high. As a result, a vast amount of negative examples has to be obtained which often relies on either memory banks (He et al., 2020) or large batch sizes (Chen et al., 2020b), potentially limiting the wider applicability further in an already compute-intensive field.

To adopt these techniques into the video domain efficiently, we make the following observations. First, we notice that though video also provides natural augmentation with viewpoint changes, illumination (jittering) and deformation, still spatiotemporal coherence and motion are not explicitly used. We are inspired by the two-streams hypothesis for vision processing in the brain (Goodale and Milner, 1992; Schneider, 1969), suggesting two pathways: the ventral stream involved in object recognition and the dorsal stream locating objects and recognizing motion. Neurons in the dorsal stream of the cerebral cortex respond to rotating and expanding patterns of motion (Figure 1.1). Motion without appearance information can be a rich source of information for humans (Johansson, 1973), however more recent works propose that it is likely that the streams are not independent and involve considerable interaction instead (McIntosh and Schenk, 2009). Second, we argue that the focus in contrastive learning research should be on improving the quality of comparisons and the task itself, to improve towards a stronger self-supervision framework. We believe instance-level contrastive learning is inefficient and neglects the use of semantic similarity between instances. Low similarity scores are produced for a large pool of negative pairs regardless of their semantic similarity, resulting in undesirable distances between samples in the embedding. To resolve this, several works have explored alternatives to random sampling for negative examples (Cao et al., 2020; Chuang et al., 2020), such as hard negative mining (Kalantidis et al., 2020; Robinson et al., 2021). We are instead interested in leaving instance-level comparisons and include mappings to *prototypes* (defined as representatives of semantically similar groups of features), providing a possible benefit on video representations without any potential costs from distance searches in the data.

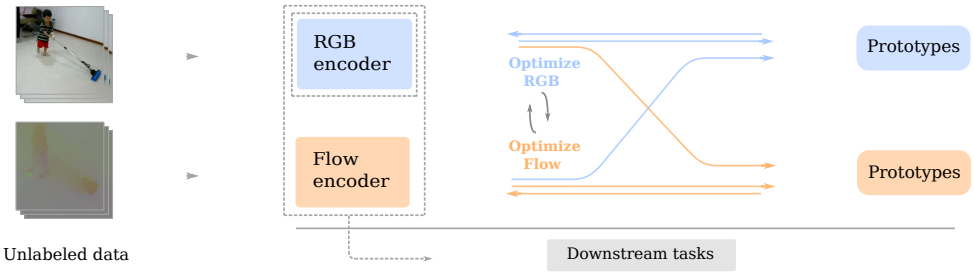

Figure 1.2: **RGB and optical flow** are used as two streams in the training of one stream by predicting consistent prototype assignments from features. By also alternating the training, we transfer knowledge cross-stream from motion (flow) to appearance (RGB) useful for downstream video tasks with optional optical flow.

1.3 Contributions

In this thesis, we present a novel method called **Vi**deo **C**ross-Stream Prototypical **C**ontrasting (ViCC) where we consider RGB and optical flow as distinct views for video contrastive learning, to influence appearance and motion learning respectively. The two input streams and spatiotemporal augmentations are united into one framework. In each iteration of the optimization of one stream, views are assigned to a set of prototypes and assignments are subsequently predicted from the features, see Figure 1.2.

Our contributions can be summarized as below.

- We introduce a novel visual-only self-supervised learning framework for video that contrasts using sets of views from two streams (RGB and flow). We demonstrate the benefits of operating on stream prototypes over contrastive instance learning, avoiding unnecessary comparisons and hence computations, while improving accuracy.
- We propose a new training mechanism for video, in which RGB and flow streams are interconnected in two ways: prototypes are predicted from both streams and the optimization process is alternated. As motion information is transferred to the RGB model, we can discard the optical flow network in deployment scenarios depending on speed and efficiency requirements.
- We perform extensive ablation studies to provide an in-depth analysis of our method. Our result reaches state-of-the-art on UCF101 (Soomro et al., 2012) and HMDB51 (Kuehne et al., 2011) on the two backbones S3D (Xie et al., 2018) and R(2+1)D (Tran et al., 2018).

1.4 Outline

This thesis is organized into 7 sections. In Section 2, we lay the foundations for several fundamental components of our method. Related work is reviewed in Section 3. Our proposed method, ViCC, is described in Section 4. In Section 5, we discuss experiments, including model ablations and evaluations on downstream video recognition tasks. In Section 6, we provide a discussion on our method and elaborate on possible future work.

2 Background

In this section, we introduce two important foundations of our proposed method: optical flow for video deep learning and contrastive instance learning. We start with a review of spatiotemporal learning in Section 2.1. We discuss representation learning without labels and self-supervision in Section 2.2. Finally, in Section 2.3 we introduce the contrastive learning paradigm and build preliminaries for contrastive self-supervision techniques while motivating their usage.

2.1 Spatiotemporal learning

Most neural network training in computer vision is achieved using knowledge from human-annotated labels. Convolutional neural networks (ConvNets), being spatial-agnostic and channel-specific, have been the core structure used for learning image representations. Large scale datasets such as ImageNet (Deng et al., 2009) were collected, computational power was increased and various influential image architectures were developed triggering the surge of deep learning in vision (Krizhevsky et al., 2012; Simonyan and Zisserman, 2015; Szegedy et al., 2015). Residual connections (He et al., 2016) allow for deeper networks using shortcut connections that reuse activations from previous layers by signal addition. ImageNet pretrained models have been proven useful for various tasks including segmentation and action classification.

2.1.1. Dense trajectories and optical flow

In video recognition, we are interested in both recognizing spatial structures as well as capturing motion from the temporal dimension. Among the most notable traditional methods (*i.e.* before the deep learning era) are handcrafted local spatiotemporal features based on Dense Trajectories. Improved Dense Trajectories (IDT) (Wang and Schmid, 2013) track densely sampled points in frames at several spatial scales using dense optical flow, resulting in pixel trajectories. Local descriptors are computed from regions around the trajectories based on motion and appearance heuristics including histograms of flow (HOF) (Laptev et al., 2008) and motion boundary histograms (MBH) (Dalal and Triggs, 2005). A fundamental component is optical flow, the form of apparent motion that results from camera motion or moving objects. Optical flow calculation is done by estimating a motion field that captures the displacement of pixels between consecutive frames, working with the assumption of brightness constancy. With its many solutions and challenges (*e.g.* the aperture problem), optical flow estimation has a rich literature (Farnebäck, 2003; Horn and Schunck, 1981) and has seen a revival of interest through neural networks (Dosovitskiy et al., 2015a; Ilg et al., 2017). Optical flow has been used as a component in numerous vision applications (Godard et al., 2015; Han et al., 2020b; Li et al., 2016; Simonyan and Zisserman, 2014), including deep learning-based approaches for video.

2.1.2. Video deep learning

The developments in image recognition have accelerated progress in video-based tasks, where extensions were proposed for the temporal component. Methods for combining video frames using ConvNets were first studied on the large dataset Sports-1M (Karpathy et al., 2014).

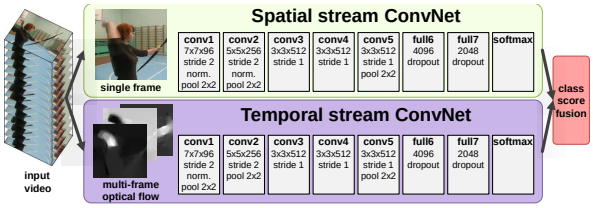

Figure 2.1: **The architecture of the first two-stream networks** (Simonyan and Zisserman, 2014). The spatial ConvNet works on single frames. The Temporal ConvNet takes an input that contains multiple flow obtained through stacking the flow channels $d_t^{x,y}$ of L consecutive frames. Softmax scores are fused by *e.g.* averaging.

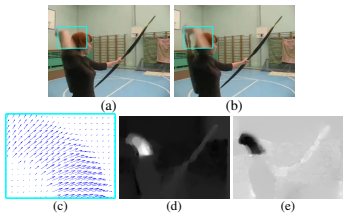

Figure 2.2: **Dense optical flow**: (a) and (b) show two consecutive frames, (c) shows the dense optical flow field, and (d) and (e) show the horizontal and vertical components. Figure adapted from Simonyan and Zisserman (2014).

As this turned out to capture only minimal temporal context, other adaptations were explored. Two-stream networks (Feichtenhofer et al., 2016; Simonyan and Zisserman, 2014) propose to leverage optical flow as a separate input modality by feeding it into one network while using the RGB frames from the spatial stream as input for another model. Figure 2.1 illustrates the architecture of the first two-stream network proposed by Simonyan and Zisserman (2014). As dense optical flow can be seen as displacement fields d_t between pairs of consecutive frames t and $t+1$, the horizontal component d_t^x and vertical component d_t^y can be used as image channels (See Figure 2.2). The flow channels were stacked to represent motion across L frames. A different strategy consists of a Long-Short Term Memory (LSTM) applied on feature maps (Donahue et al., 2016; Yue-Hei Ng et al., 2015) to address long-term dependency. This approach was found to lack in capturing fine-grained motion for action recognition, though recurrent-based approaches are effective for tasks in which sequence order plays a role such as future prediction (Oh et al., 2015; Villegas et al., 2017). 3D Convolutional Neural Networks (3DConvNets) learn spatiotemporal filters that move locally in space and time (Hara et al., 2018b; Ji et al., 2013; Tran et al., 2015; Varol et al., 2017). Inflated ConvNets enabled the integration of two-dimensional pretraining on ImageNet and trained on the large-scale dataset Kinetics, Two-stream 3DConvNets proved to have much potential (Carreira and Zisserman, 2017), giving rise to more research (Feichtenhofer et al., 2019; Qiu et al., 2017; Tran et al., 2018; Xie et al., 2018).

2.2 Representation learning: generative *vs.* discriminative

Due to the drawbacks of needing labeled data, increasing efforts have been devoted to researching visual representation learning without labels. Generative models such as the Variational Autoencoder (VAE) (Kingma and Welling, 2014) are latent variable models that encode the input data distribution. Although powerful methods, pixel-level generation is expensive and VAEs have not yet consistently shown to create useful representations for downstream tasks. Other unsupervised generative approaches include Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), deep belief networks (Hinton, 2010) and flow-based generative models (Dinh et al., 2015, 2017; Kingma and Dhariwal, 2018; Rezende and Mohamed, 2015).

2.2.1. Self-supervised learning

Recent progress in visual representation learning is largely due to the self-supervised learning paradigm (Doersch and Zisserman, 2017; Doersch et al., 2016). Self-supervised methods obtain supervisory signals from the underlying structure of the input data. This discriminative learning process is often performed through pretext tasks, which involves prediction problems where part of the data serves as labels. Aside from the regular use of self-supervision in natural language processing over the last decade, *e.g.* Mikolov et al. (2013) and Howard and Ruder (2018), tasks have been proposed for image recognition based on augmentations, including rotation (Gidaris et al., 2018), distortion (Dosovitskiy et al., 2015b) and patch shuffling (Doersch et al., 2016). Pretext tasks for video representation learning are often based on the temporal dimension. Examples include validating a frame ordering (Misra et al., 2016), sorting frames (Lee et al., 2017) and tracking (Wang and Gupta, 2015). Several attempts have been made at combining traditional clustering methods with representation learning, using cluster assignments as pseudo-labels (Asano et al., 2020b; Caron et al., 2018; Yan et al., 2020). Other self-supervised learning methods based on contrastive instance learning in the latent space have recently achieved state-of-the-art performance as pretraining for downstream tasks. Before discussing these techniques, we first introduce the broader learning paradigm of contrastive learning.

2.3 Contrastive learning

Contrastive learning is a learning method based on explicit contrasting between data samples. A class-level or instance-level representation is learned by forcing similarity scores of positive pairs higher than that of negative pairs. Several contrastive loss functions were proposed in the field of deep metric learning, which is aimed at learning a similarity metric from data. This can be applied in a one-shot learning setting where we have insufficient data per class (*e.g.* face prediction). Hadsell et al. (2006) create similar and dissimilar sets for every image and use these pairs in a contrastive loss with a Siamese network (Bromley et al., 1993) architecture. Schroff et al. (2015) introduced the triplet loss which involves a triple of samples: an anchor x, positive sample x_i from the same class and negative sample x_j from a different class. The triplet loss objective is for the embedding to satisfy that x stays close to the positive sample x_i, but far away from x_j in the latent space. Mining for hard negative samples is often needed to obtain performance (Chechik et al., 2009).

2.3.1. Contrastive instance learning

Contrastive instance learning (Chen et al., 2020b; He et al., 2020) can be defined as a self-supervised learning method which contrasts in the latent space by maximizing agreement between different augmented views of the same data instances. Three key components in this framework are *i)* a data augmentation module that transforms a given sample x into two views x_i and x_j by applying separate transformations t and t' sampled from the set of augmentations T, *ii)* the embedding function $f(\cdot)$ consisting of an encoder and a small MLP projection head that extracts feature vectors z_i and z_j from views, and *iii)* a contrastive loss function that contrasts between x_i and a set $\{x_k\}$ of augmented pairs that includes our positive pair.

Given a dataset $X = \{x_1, x_2, ..., x_n\}$, we aim to learn a function $f(\cdot)$ that maps X to $Z = \{z_1, z_2, ..., z_n\}$. The contrastive loss objective for a positive pair (i, j), referred to as the InfoNCE

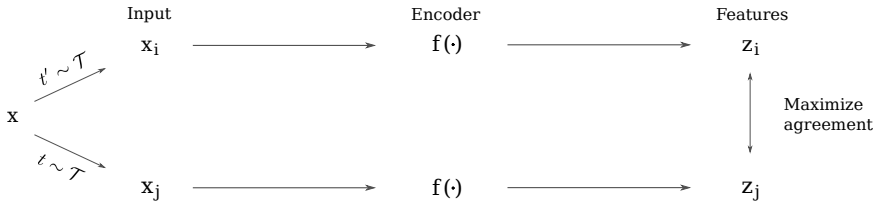

Figure 2.3: **Schematic overview of contrastive instance learning** (Chen et al., 2020b; He et al., 2020). Two different augmented versions of the same instance x are obtained, x_i and x_j. The encoder $f(\cdot)$, consisting of a embedding function (e.g. ResNet) and a small neural network projection head, is applied on these samples and outputs features. A comparison is then made between features from different augmentations of the same instance, maximizing their agreement, after which the backpropagation update is performed.

loss (Chen et al., 2020b; Sohn, 2016; van den Oord et al., 2019), is then given by

$$\mathcal{L}^{\text{InfoNCE}}(z_i, z_j) = -\log \frac{\exp(z_i \cdot z_j/\tau)}{\sum_{k \neq i} \exp(z_i \cdot z_k/\tau)}, \tag{2.1}$$

where τ is the temperature hyperparameter and $z_i \cdot z_j$ refers to the dot product between normalized vectors, *i.e.* cosine similarity. The final loss is computed for all available positive pairs. As shown by van den Oord et al. (2019), minimizing the contrastive loss is equivalent to maximizing a lower bound of the Mutual Information (MI) of z_i and z_j.

As opposed to pretext tasks that uses pixel-level cross-entropy loss, the contrastive loss is based in representation space which helps with invariance to certain irrelevant perceptual features such as background. In contrast to metric learning, the contrastive loss used in instance discrimination includes a way to leverage multiple negative examples per contrasting besides considering positive pairs of one instance, which naturally allows for improved representation learning given sufficient negative samples. As mining for hard negative examples without access to labels is difficult, the large improvements in performance compared to pretext tasks (Chen et al., 2020b; He et al., 2020) are the result of introducing many negative samples which requires large memory usage. Recent works, including ours, attempt to combat this issue (Caron et al., 2020; Grill et al., 2020).

3 Related Work

This section surveys studies related to our work, while positioning our method in relation to existing research. First, works related to representation learning and image-based tasks are discussed in Section 3.1. In Section 3.2 we discuss methods related to action recognition. Finally, we review video self-supervision in Section 3.3.

3.1 Representation learning

Representation learning methods without labels for visual data can roughly be categorized into works on either pretext tasks, loss functions, or clustering-based approaches. In earlier works, representations are often learned through pretext tasks by predicting pixel (regions) of images, such as context prediction of patches (Doersch et al., 2016), solving jigsaw puzzles (Noroozi and Favaro, 2016), predicting the angle of rotation (Gidaris et al., 2018), or performing colorization (Zhang et al., 2016). Exemplar ConvNets (Dosovitskiy et al., 2015b) apply image augmentations for the purpose of creating distorted versions to subsequently discriminate between a set of surrogate classes. Another line of work employs generative models in self-supervised pretext tasks, including the split-brain autoencoder (Zhang et al., 2017) and inpainting with a context encoder (Pathak et al., 2016).

3.1.1. Contrastive instance learning

Recent studies have made progress in developing loss functions for contrastive instance self-supervised learning, where contrasting is performed between augmentations of sampled instances. Instance discrimination considers each sample as its own class in the data. As such a classifier becomes computationally infeasible fast, Wu et al. (2018) use noise-contrastive estimation (NCE) (Gutmann and Hyvarinen, 2012) and a memory bank to store representations as their pool of negative samples. Other solutions include the work from Chen *et al*. (Chen et al., 2020c), which retrieves more negative samples by using large batch sizes. He *et al*. (He et al., 2020) propose a momentum encoder with a dynamic dictionary look-up with Momentum Contrast (MoCo). Another line of work contrasts between the global image and local patches (Hjelm et al., 2019; van den Oord et al., 2019). In Contrastive predictive Coding (CPC) van den Oord et al. (2019) defines this contrasting as past-present relationship, while Hjelm et al. (2019) uses a local-global feature relationship. Our method instead uses complementary modalities as main views and intuitively learns its own positive and negative examples from both feature spaces through the prototypes. Contrasting is done between instances and prototypes, going beyond instance-level learning while avoiding the need for substantial batch sizes (Chen et al., 2020b) or large memory banks (He et al., 2020).

Wang and Qi (2021) propose to use stronger data augmentations, facilitated through minimizing the Kullback–Leibler (KL) divergence between stronger and weaker augmentations. To tackle the efficiency of instance contrastive learning methods, alternative methods such as the work from Grill et al. (2020) were proposed, which attempts iteratively bootstrapping the representation instead of employing negative samples. The contrastive loss is reduced between two encoder networks: the online and the target network. Other possible methods can be found in self-labeling through clustering-based approaches, which we discuss next.

3.1.2. Clustering in latent space

An alternative to pretext tasks or contrastive losses is to directly learn pseudo-labels, such as using clustering-based methods for end-to-end visual representation learning. Combining clustering with representation learning to obtain pseudo-labels has been proposed in various self-supervised learning settings (Asano et al., 2020a; Caron et al., 2018, 2020; Li et al., 2021; Yan et al., 2020). Yan *et al*. (Yan et al., 2020) perform a clustering step and a representation learning step, using cluster assignments as pseudo-labels in the representation learning phase. Asano *et al*. (Asano et al., 2020b) propose a solution of degenerate solutions by casting clustering into an instance of the optimal transport problem. Caron *et al*. (Caron et al., 2020) use this clustering setup in a contrastive learning setting by enforcing consistency between different views in Swapping Assignments between Views (SwAV), comparing cluster assignments instead of individual features. Furthermore, an online clustering and simultaneous feature learning mechanism was proposed in (Zhan et al., 2020). Our objective is most similar to (Caron et al., 2020) and (Li et al., 2021), aligning cluster assignments for augmented instances in an online manner. However, we apply our method on video, use augmentation in the form of optical flow and alternate the training of models and prototypes to incorporate information in both streams.

3.2 Action recognition

Advances in action recognition architectures such as 3DConvNets have driven the field forwards, where numerous works are based around using both RGB and Flow in two-stream networks.

3.2.1. Two-stream networks

Two-stream architectures have been a structure used in many action recognition methods over the years. Using two separate pipelines for processing optical flow and RGB from frames, studies have shown that optical flow captures motion well and is an effective additional stream for action recognition (Carreira and Zisserman, 2017; Diba et al., 2019; Feichtenhofer et al., 2016; Simonyan and Zisserman, 2014). As optical flow computation can be computationally expensive, attempts have been made to incorporate flow information into one RGB-based network for action recognition, *e.g*. through knowledge distillation (Crasto et al., 2019; Stroud et al., 2020; Zhao and Snoek, 2019). Our proposed method instead keeps two streams and leverages an alternated optimization process to perform distillation through contrastive learning, avoiding the need for optical flow during inference but providing the option.

3.2.2. Single-stream RGB networks

In recent years, single-stream RGB networks using 3DConvNets have shown to be very competitive (Carreira and Zisserman, 2017; Hara et al., 2018b). Because 3D convolutional operations are computationally more expensive than 2D convolutions, several variants of 3D convolutional operations were proposed that target improved efficiency (Feichtenhofer et al., 2019; Qiu et al., 2017; Tran et al., 2018; Xie et al., 2018). One example is R(2+1)D (Tran et al., 2018), which uses a Convolutional Recurrent Block that applies 2D spatial filtering followed by 1D convolution on the temporal dimension. Non-local networks (Wang et al., 2018) attempt local decomposition of 3D convolutions using non-local operations by computing the response as a weighted sum of the features at all positions. SlowFast (Feichtenhofer et al., 2019) studies decomposition at a global level and proposes a backbone network with two pathways: the slow

pathway processes with low frame rate for spatial semantics, and the fast pathway processes motion. Feichtenhofer (2020) employs a network expansion method in which dimensions are progressively extended to create efficient video architectures. Research aimed at processing longer videos (Tang et al., 2018; Wu et al., 2019; Yue-Hei Ng et al., 2015) often needs innovative solutions such as pooling or explicit long-term modeling because of limited computational resources and restrictions imposed by the convolution operation.

3.3 Video self-supervision

Following the image domain, video research has been conducted on self-supervised methods in combination with pretext tasks or contrastive losses. In the video domain, self-supervised approaches exploring pretext tasks are often based on the temporal dimension, such as the order of frames or clips (Fernando et al., 2017; Lee et al., 2017; Misra et al., 2016; Xu et al., 2019), learning the arrow of time (Pickup et al., 2014; Wei et al., 2018), or pace (Benaim et al., 2020; Cho et al., 2020; Wang et al., 2020; Yao et al., 2020b). Pretext tasks that were previously explored in the image domain have been proposed and extended for video, including rotation prediction (Jing et al., 2019) and space-time puzzles (Kim et al., 2019). Other approaches include leveraging the consistency in frames by temporal correspondence (Lai and Xie, 2019; Lai et al., 2020), tracking patches (Wang and Gupta, 2015; Wang et al., 2019b), future frame prediction (Goroshin et al., 2015; Vondrick et al., 2016) or future feature prediction (Han et al., 2019, 2020a).

3.3.1. Contrastive losses

Qian et al. (2021) applied the instance discrimination task on video clips and found that both temporal augmentations as well as spatial augmentation in a clip-wise consistent manner are important. Han et al. (2019, 2020a) predict future feature representations from patches from the past context following a CPC intution, using three types of negative examples: spatial negatives, temporal negatives and hard negatives originating from a different video. Tschannen et al. (2020) propose a framework built on a hierarchy of losses for video and Romijnders et al. (2021) extend this framework with an object-level loss. Yao et al. (2020a) proposes a similar hierarchical idea, namely to consider tree perspectives of spatial, spatiotemporal and sequential coherence. The combination of image augmentation, temporal coherence between frames and global-local correspondence between features was also explored (Hjelm and Bachman, 2020). Kong et al. (2020) leverages cycle correspondence in video for representation learning.

3.3.2. Optical flow and distillation

Multiple works explore optical flow for self-supervision (Han et al., 2020a,b; Mahendran et al., 2018). Mahendran *et al*. (Mahendran et al., 2018) use optical flow as supervision for RGB. Tian *et al*. (Tian et al., 2020) first explore the use of RGB and optical flow as views for contrastive learning. Clustering combined with Improved Dense Trajectories (IDT) was also studied (Tokmakov et al., 2020) Most similar to our work are (Tian et al., 2020) and (Han et al., 2020b), which both employ RGB and flow in a two-stream manner for contrastive learning. Han *et al*. (Han et al., 2020b) use an alternated training process and samples hard positive examples from the other stream in Co-training Contrastive Learning Representations (CoCLR). Different from this work, we do not employ instance-level contrastive learning. As we use prototype mappings of our features and subsequently predict feature assignments, our streams leverage a stronger interplay. We also incorporate informed negative examples from both streams through our prototypes and we do not use a momentum encoder (He et al., 2020). Several works avoid

flow computation during inference while utilizing it during label-free training (Gavrilyuk et al., 2021; Han et al., 2020b; Mahendran et al., 2018) which is related to our work.

3.3.3. Multi-modal approaches

Video allows for a multi-modal approach by using information such as audio (Alwassel et al., 2020; Asano et al., 2020a) and text (Miech et al., 2020; Sun et al., 2019) to learn from correspondence between modalities. Alwassel *et al*. (Alwassel et al., 2020) use a cross-modal audio-video iterative clustering and relabeling algorithm. Asano *et al*. (Asano et al., 2020a) employ both RGB and audio in a simultaneous clustering and representation learning setting, following (Asano et al., 2020b). Our method strictly speaking does not leverage multiple modalities as we use an optical flow representation extracted from the RGB representation, without introducing any external information. However, our work similarly leverages the interplay of complementary information and could therefore be used alternatively as a multi-modal approach, *e.g*. leveraging audio in addition to optical flow in order to improve representations further.

4 Method

In this thesis, we study whether we can improve self-supervised contrasting for video by leveraging prototypes and optical flow, thereby proposing a novel method: Video Cross-Stream Prototypical Contrasting (ViCC). We briefly revisit the contrastive instance loss, and explain how we can use RGB and optical flow separately to predict and learn prototypes in Section 4.1. Finally, we introduce our cross-stream interplay and the steps of our algorithm in Section 4.2.

4.1 Predicting stream prototype assignments

Contrastive instance learning (Chen et al., 2020b; He et al., 2020; van den Oord et al., 2019) is a representation learning method that contrasts in the latent space by maximizing the agreement between features from multiple transformed views of the same data instances. As the method is self-supervised, no labels are required for representation learning. The loss calculation for one positive pair follows equation 2.1, where the final loss is computed for all available positive pairs. Given a positive pair however, a sufficiently large number of negative examples through the set of negative augmented instances $\{x_k\}$ needs to be available for which storage of features besides the mini-batch is often needed. The contrastive learning mechanism also neglects to take into account the informativeness of samples, given that all negative examples are treated in the same manner.

In our proposed method we avoid instance-level contrasting by using for each stream a set of *prototypes* in our contrasting. Furthermore, we extend the augmentation module by considering RGB frames and optical flow as views. Mathematically, given a video clip x we first consider the two streams as views, obtaining $x = \{x^1, x^2\}$ which describe a RGB and a flow sample respectively. The objective is to learn the stream representations $z^1 = f_1(x^1)$ and $z^2 = f_2(x^2)$ through learning their encodings $f_1(\cdot)$ and $f_2(\cdot)$. Each of the encoders has a set of K trainable prototype vectors, $\{c_1^1, ..., c_K^1\} \in C_1$ and $\{c_1^2, ..., c_K^2\} \in C_2$, implemented as a linear layer in the networks.

Consider only the training of one encoder f_s on its own stream s where $s \in \{1, 2\}$. We denote the corresponding prototype set as matrix C_s with columns $c_1^s, ..., c_k^s$. Given input sample x^s, we obtain two augmented versions $\{x_i^s, x_j^s\}$. After applying the encoder $f_s(\cdot)$ we obtain features $\{z_i^s, z_j^s\}$. The features are mapped to the set of prototypes C_s to obtain cluster assignments $\{q_i^s, q_j^s\}$, as detailed in the following section. The features and assignments are subsequently used in the following prediction loss:

$$\mathcal{L}_s^{\text{Single-stream}}(z_i^s, z_j^s) = l_s\left(z_j^s, q_i^s\right) + l_s\left(z_i^s, q_j^s\right) . \tag{4.1}$$

Each of the terms represents the cross-entropy loss between the stream prototype assignment q and the probability obtained by a softmax on the similarity between z and C_s:

$$l_s\left(z_j^s, q_i^s\right) = -\sum_k q_i^{s,(k)} \log \frac{\exp(z_i^s \cdot c_k^s/\tau)}{\sum_{k'} \exp(z_i^s \cdot c_{k'}^s/\tau)} , \tag{4.2}$$

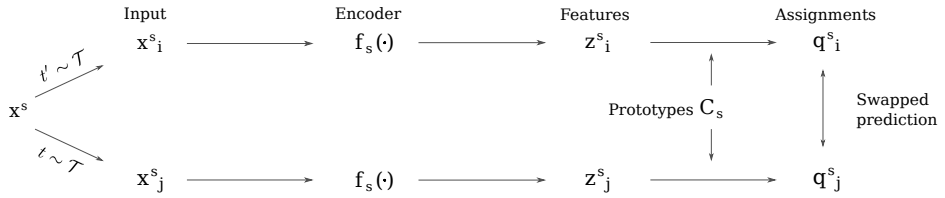

Figure 4.1: **Schematic overview of the single-stream loss** for stream s (RGB or flow). Features from different augmentations of the same sample are obtained in identical manner as in contrastive instance learning (see Section 2.3.1). Features are then matched to prototypes C_s, obtaining assignments q_s which are in turn predicted using the other view, following the SwAV method (Caron et al., 2020).

where τ is a temperature hyperparameter. The objective is to maximize the agreement of prototype assignments from multiple views of one sample (RGB or flow). Features are contrasted indirectly through comparing their prototype assignments. The total loss of training the encoder f_s on its own stream is taken over all videos and pairs of data augmentations, minimized with respect to both f_s and C_s. See Figure 4.1 for an overview of the single-stream loss.

4.1.1. Learning stream prototype assignments

The assignments $\{q_i^s, q_j^s\}$ are computed by matching features $\{z_i^s, z_j^s\}$ to prototypes C_s. In essence, we need to consider the cross-entropy for assigning each z to C_s and perform a mapping to assign labels automatically. Optimizing q directly leads to degeneracy. Following (Asano et al., 2020b; Caron et al., 2020) a uniform split of the features across prototypes is enforced, which avoids the collapse of assignments to one prototype. Given our feature vectors Z whose columns are $z_1, ..., z_B$, we map them to C_s and optimize using an Optimal Transport (Peyré and Cuturi, 2019) solver the mapping $Q = q_1, .., q_B$:

$$\max_{Q \in \boldsymbol{Q}} \operatorname{Tr}\left(Q^T C_s^T Z\right) + \epsilon H(Q), \tag{4.3}$$

where $H(Q)$ is the entropy of Q which acts as a regularizer. The ϵ parameter controls the uniformity of the assignment where a low value helps to avoid collapse. Following (Caron et al., 2020), we restrict the transportation polytope to mini-batches:

$$\boldsymbol{Q} = \{Q \in \mathbb{R}^{K \times B} | \; Q\mathbb{1}_B = \frac{1}{K}\mathbb{1}_K, Q^T\mathbb{1}_K = \frac{1}{B}\mathbb{1}_K\}, \tag{4.4}$$

where $\mathbb{1}_K$ denotes a vector of all ones with dimension K. We preserve soft assignments Q^* and the solution of the transportation polytope, solved efficiently using the Sinkhorn-Knopp algorithm (Cuturi, 2013) can be written as follows:

$$Q^* = \operatorname{Diag}(\alpha) \exp\left(\frac{1}{\epsilon} C_s^T Z\right) \operatorname{Diag}(\beta), \tag{4.5}$$

where α and β denote renormalization vectors such that Q results in a probability matrix (Cuturi, 2013). As the amount of batch features B is usually smaller than the number of prototypes K, we increase or available features B by adopting a queue mechanism that stores features from previous epochs.

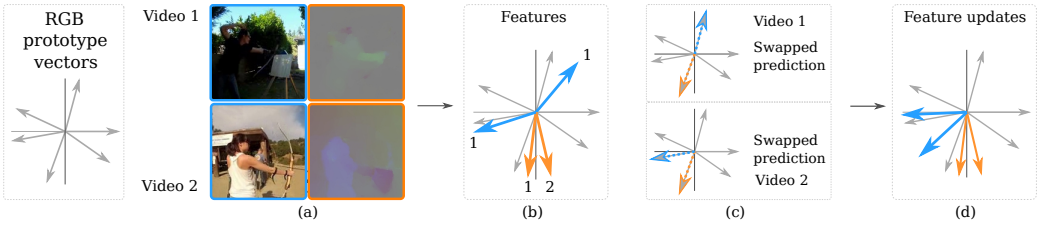

Figure 4.2: **Intuition of learning cross-stream**. Given two videos from the same semantic class with varying backgrounds, their optical flow patterns can look very similar (a). As a result, RGB feature vectors are further apart in space than the optical flow features (b), and after the features are matched to the prototypes a similar phenomenon could occur for the assignments (c). By enforcing consistent assignments of the views for each video, the RGB features from the two videos will eventually move closer in space to flow features after feature updates, and therefore closer to each other (d).

4.2 Learning cross-stream

We are now interested in using information from both streams in the training of the encoders f_1 and f_2. Leaving aside the use of data augmentation and the update of the prototypes, our motivation for using optical flow and RGB together is illustrated in Figure 4.2. The purpose of augmentation in the RGB space as applied in the single-stream stage is to obtain invariance to irrelevant features, including background and viewpoint. We hypothesize that improved representations can be obtained through introducing optical flow to RGB, as optical flow can provide a motion-only viewpoint which naturally allows for neglecting unimportant perceptual features. We will now formally describe our approach.

Consider again the encoder f_s and prototypes C_s from the stream s that is optimized in one alternation. We now add a second stream t where $t \in \{1, 2\}$ and $s \neq t$. We use the encoder f_t with *frozen* weights and obtain samples $\{x_i^t, x_j^t\}$ and features $\{z_i^t, z_j^t\}$. By matching these features to prototypes C_s the assignments $\{q_i^t, q_j^t\}$ are obtained. Given f_s, C_s, and f_t, all initialized with prior representations learned on their own stream, the loss function for the prediction problem consist of four main parts:

$$\begin{aligned} &\mathcal{L}_s^{\text{Cross-stream}}\left(z_i^s, z_j^s, z_i^t, z_j^t\right) = \\ &L_s\left(z_i^s, z_j^s, z_j^t, q_i^t\right) + L_s\left(z_i^s, z_j^s, z_i^t, q_j^t\right) + \\ &L_s\left(z_j^s, z_i^t, z_j^t, q_i^s\right) + L_s\left(z_i^s, z_i^t, z_j^t, q_j^s\right), \end{aligned} \tag{4.6}$$

where the function L_s measures the fit between three features z and an assignment q. For instance, the first of the L_s terms is given by:

$$L_s\left(z_i^s, z_j^s, z_j^t, q_i^t\right) = l_s\left(z_i^s, q_i^t\right) + l_s\left(z_j^s, q_i^t\right) + l_s\left(z_j^t, q_i^t\right). \tag{4.7}$$

The total loss function therefore consist of 12 terms. Each of the terms l_s again represents the cross-entropy between one feature z and one assignment q, *e.g.*:

$$l_s(z_i^s, q_i^t) = -\sum_k q_i^{t,(k)} \log \frac{\exp(z_i^s \cdot c_k^s/\tau)}{\sum_{k'} \exp(z_j^s \cdot c_{k'}^s/\tau)}, \tag{4.8}$$

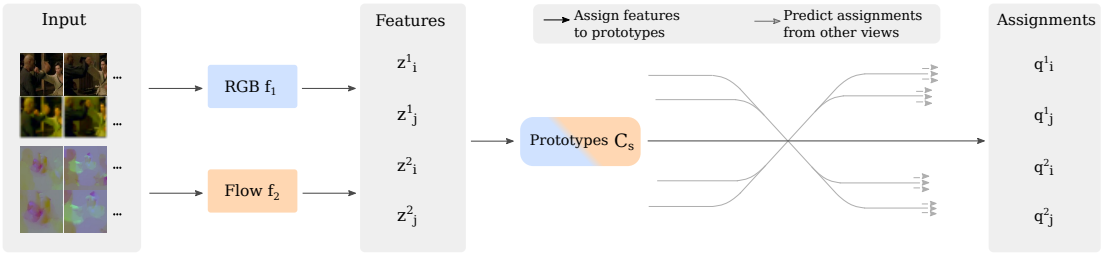

Figure 4.3: **Video Cross-Stream Prototypical Contrasting.** Two different augmented samples are obtained for both RGB and flow. The encoders f_1 and f_2 map samples from RGB and flow respectively to obtain features $z_i^1, z_j^1, z_i^2, z_j^2$, which are in turn assigned to either RGB or flow prototype vectors, depending on which stream $s \in \{1, 2\}$ is optimized. Next, the stream prototype assignments $q_i^1, q_j^1, q_i^2, q_j^2$ are predicted using features only from the three other views. The encoder and prototypes from the optimized stream are updated by backpropagation, while the other encoder remains fixed.

where we predict the assignment q_i^t from stream t (obtained by matching corresponding feature z_i^t to the prototypes C_s) using one of the augmented features z_i^s from stream s.

In summary, we predict assignments from each of the four views using features originating from three views, see Figure 4.3. In the prediction of each q, we avoid the use of the feature z where s is equal to t (same stream) *and* i is equal to j (same augmentation). This setup forces the features to capture the same information by predicting consistent assignments from them. The total loss for cross-stream training on stream s is taken over all videos and pairs of augmentations, optimized with respect to f_s and C_s.

4.2.1. Alternation

The optimization process from this section is then performed *vice versa* on the other stream. For example, we first optimize our RGB encoder f_1 and the corresponding prototypes C_1 as our f_s and C_s using views from both f_s (RGB) and f_t (flow). Next, we optimize our flow encoder f_2 and prototypes C_2 as our f_s and C_s, and use RGB as our f_t. Detailed pseudocode is provided in the next section.

4.2.2. ViCC Algorithm

Our complete algorithm is structured as follows. *Stage 1) Single-stream.* In the first stage, the two encoders f_1 and f_2 and their prototypes C_1 and C_2 are initialized from scratch and trained

Algorithm 4.1 ViCC Algorithm

```
Stage one: Single-Stream
    Train RGB
    Train Flow
Stage two: Cross-Stream
    for N alternations do
        Train RGB
        Train Flow
    end for
```

using their own input stream, following Equation 4.1. *Stage 2) Cross-stream.* In the second stage, cross-stream, the two models are trained in an alternating fashion using input from both streams. In one alternation, one of the streams s with encoder f_s and prototypes C_s is encouraged to predict mappings consistently following Equation 4.6, leveraging complementary information from the other stream through assigning views from f_t to C_s. Both the prototype mappings and the alternation process in our cross-stream mechanism serve as means for transferring knowledge from motion (flow) to RGB. See Algorithm 4.1 for an overview of the procedure.

At the inference stage, depending on speed *vs.* accuracy requirements, both the RGB model f_1 trained with ViCC self-supervision can be used for downstream tasks as well as both RGB f_1 and flow f_2 by averaging predictions from the models.

4.2.3. Example code for ViCC

In Algorithm 4.2 we provide pseudocode in PyTorch-like style for the implementation of the cross-stream stage of ViCC-RGB. For the definition of the function `sinkhorn` that describes the Sinkhorn-Knopp algorithm we refer to (Caron et al., 2020).

Algorithm 4.2 Pseudocode for ViCC-RGB-2 in PyTorch-like style

```
# rgb_model: encoder network for RGB
# flow_model: encoder network for flow, frozen
# temp: temperature
for rgb, flow in loader: # B samples
  # two augmented versions for two streams
  rgb_i, flow_i = aug(rgb_i, flow_i)
  rgb_j, flow_j = aug(rgb_j, flow_j)
  # get RGB and flow embeddings: 2B x D
  z_rgb = cat(rgb_model(rgb_i), rgb_model(rgb_j))
  z_flow = cat(flow_model(flow_i), flow_model(flow_j))
  # get similarity with prototypes C_rgb, C_rgb in D x K
  sim_rgb_i, sim_rgb_j = mm(z_rgb, C_rgb)
  sim_flow_i, sim_flow_j = mm(z_flow, C_rgb)
  # compute assignments
  with torch.no_grad():
    q_rgb_i, q_rgb_j, q_flow_i, q_flow_j = sinkhorn(sim_rgb_i), sinkhorn(sim_rgb_j),
                                           sinkhorn(sim_flow_i), sinkhorn(sim_flow_j)
  # convert similarity scores to probabilities
  p_rgb_i, p_rgb_j, p_flow_i, p_flow_j = softmax(sim_rgb_i / temp), softmax(sim_rgb_j / temp),
                                         softmax(sim_flow_i / temp), softmax(sim_flow_j / temp)

  # predict cluster assignments using three other views
  l_rgb_i = q_rgb_i * log(p_rgb_j) + q_rgb_i * log(p_flow_i) + q_rgb_i * log(p_flow_j)
  l_rgb_j = q_rgb_j * log(p_rgb_i) + q_rgb_j * log(p_flow_i) + q_rgb_j * log(p_flow_j)
  l_flow_i = q_flow_i * log(p_rgb_i) + q_flow_i * log(p_rgb_j) + q_flow_i * log(p_flow_j)
  l_flow_j = q_flow_j * log(p_rgb_i) + q_flow_j * log(p_rgb_j) + q_flow_j * log(p_flow_i)
  # combine for total loss for rgb model
  loss = - 1/4 * (1/3 * l_rgb_i + 1/3 * l_rgb_j + 1/3 * l_flow_i + 1/3 * l_flow_j)
  # optimizer update and normalize prototypes
  loss.backward()
  update(rgb_model.params), update(C_rgb)
  with torch.no_grad():
    C_rgb = normalize(C_rgb, dim=0, p=2)
```

5 Experiments

To show the applicability of our representations, we perform evaluations on several downstream tasks for video recognition. Our goal is to study whether our method, ViCC, can learn effective video representations. The experimental settings are detailed in Section 5.1. We compare our method with our baseline and analyze the impact of components through ablation studies in Section 5.2. More comparisons with other self-supervised works for video are provided in Section 5.3. We show qualitative evaluations of the representations in Section 5.4 and conclude with an analysis of our prototypes in Section 5.5.

5.1 Experimental setup

We use two datasets for our experiments: HMDB51 (Kuehne et al., 2011) and UCF101 (Soomro et al., 2012). UCF101 consists of 13K videos over 101 human action classes. HMDB51 is another widely used action recognition dataset and contains around 7K videos over 51 action classes. UCF101 and HMDB51 are both divided into three train/test splits. For self-supervised training we use UCF101 training split 1 without class labels. For downstream evaluation we use UCF101 and HMDB51 and evaluate on split 1 for both datasets, following prior work (Han et al., 2020b).

5.1.1. Data preprocessing

From the source videos at 25fps, input video clips are extracted at random time stamps. Our input video clips have a spatial resolution of 128×128 pixels. We use clips of 32 frames as input, without temporal downsampling for S3D. For R(2+1)D, we use input clips of 16 frames with temporal downscaling at rate 2. For optical flow, we use the widely used TV-L1 algorithm (Zach et al., 2007), and we follow practice in (Carreira and Zisserman, 2017; Han et al., 2020b) for flow preprocessing. This means that we truncate large vectors with more than a value of 20 in both channels, transform the values to range [0, 255] and append a third channel of 0s to consider them as frames. We apply spatial data augmentation on RGB and flow clips in a consistent manner across frames. Random cropping, horizontal flipping, Gaussian blur and color jittering are used. In terms of temporal augmentation we take clips at different time stamps with 50% probability.

5.1.2. Implementation

As our base encoder architecture we use S3D (Xie et al., 2018). We also test our method with the R(2+1)D-18 (Tran et al., 2018) architecture, following recent works (Asano et al., 2020a; Crasto et al., 2019). A 2-layer MLP projection head is used during self-supervised training that projects the backbone output to 128 dimensional space following Chen et al. (2020b). In line with SwAV (Caron et al., 2020), we employ a linear layer updated by backpropagation as the prototype implementation. The projection head and the prototype layer are removed for downstream evaluation. We use K=300 as the number of prototypes. The temperature τ is set to 0.1, the Sinkhorn regularization parameter ϵ is set to 0.05 and we perform 3 iterations of the Sinkhorn algorithm. Batch shuffle is used according to MoCo (He et al., 2020) to avoid the model exploiting local intra-batch information leakage for trivial solutions.

5.1.3. Training details

The single-stream stage consists of 300 epochs. Next, the cross-steam stage is initialized with models from the single-stream stage and is trained for two cycles. In one cross-stream cycle, we first train RGB for 100 epochs and then flow for 100 epochs, each time taking the newest models, following CoCLR (Han et al., 2020b). For single-stream, the prototypes are frozen during the first 100 epochs of training. For cross-stream, the prototypes are directly updated from the start of the training. We run all our experiments with 4 Titan RTX GPUs with a batch size of 48. During self-supervised training, we use a queue that consists of stored features for a more accurate assignment with the Sinkhorn-Knopp algorithm. With a total batch size of $48 \times 4 = 192$, we adopt a queue of size 1920 to store features from the last 10 batches. The queue is introduced when the evolution of features is slowing down, *i.e.* when the decrease of the loss function is moderate. For single-stream RGB (RGB-1) we introduce the queue at 150 epochs and for Flow-1 we introduce the queue at 200 epochs. For the cross-stream stage, we introduce the queue at 25 epochs in each alternation. SGD with LARS (You et al., 2017) is used as the optimizer. A learning rate of 0.6, a weight decay of 10^{-6} and a cosine learning rate schedule with a final learning rate of 6×10^{-4} are chosen.

5.1.4. Evaluation methods

We evaluate the quality of our learned video representation using two downstream video understanding tasks: nearest neighbour video retrieval and action recognition. In the former, nearest neighbour video retrieval is performed without any supervised finetuning. We follow common protocol (Büchler et al., 2018; Misra and van der Maaten, 2020; Xu et al., 2019) by using videos form the test set as queries for k nearest-neighbour (kNN) retrieval in the training set. We report Recall at k (R@K) meaning that we mark the retrieval as correct if a video of the same class appears among the top kNN. In the latter downstream task of action recognition, we initialize with our representation and evaluate two settings: linear probe and finetuning. For linear probe, we freeze the entire network and add a linear classifier to train for the downstream task. For finetuning, the entire network with linear layer is trained end-to-end. We report Top-1 accuracy for both settings. Data augmentation similar to the self-supervision stage is used except for Gaussian blur. At inference, we follow the ten-crop procedure, meaning that we spatially obtain ten crops: the center crop, four corners and the horizontal flipped version of aforementioned crops. The moving-window approach is used for taking clips followed by averaging the predictions.

5.2 Model ablations

5.2.1. Impact of training stages

In Table 5.1 results are shown for several stages of our method in order to evaluate the improvement that cross-stream (Stage 1) has over single-stream (Stage 2). We report action recognition and nearest-neighbour video retrieval on UCF101 split 1 and include CoCLR (Han et al., 2020b) as our baseline model, as it uses the contrastive instance loss on RGB and flow with additional positive examples. Training settings are kept identical across self-supervised models. All methods are trained on 500 epochs in total. Evaluated on nearest-neighbour retrieval, we observe that our RGB-1 network gains a significant performance benefit when learning and predicting from optical flow in stage 2, shown as RGB-2 (62.1% *vs*. 40.0%). Furthermore, when combining predictions from the RGB-2 model and the Flow-2 model, both trained with cross-

Method	Stage	Input	Classification: Linear Acc	Classification: Finetune Acc	Retrieval: No labels R@1
ViCC-RGB-1	1	RGB	49.2	81.8	40.0
ViCC-Flow-1	1	Flow	71.9	87.9	55.5
ViCC-RGB-2	2	RGB	**72.2**	**84.3**	**62.1**
ViCC-Flow-2	2	Flow	75.5	88.7	59.7
CoCLR	2	R+F	72.1	87.3	55.6
ViCC-R+F-2	2	R+F	**78.0**	**90.5**	**65.1**

Table 5.1: **Improvement of ViCC** from single-stream (stage 1) to cross-stream (stage 2) evaluated on action recognition and nearest-neighbour retrieval on UCF101. CoCLR (Han et al., 2020b) is included as a baseline comparison. R+F denotes the result obtained by averaging predictions of RGB and flow models.

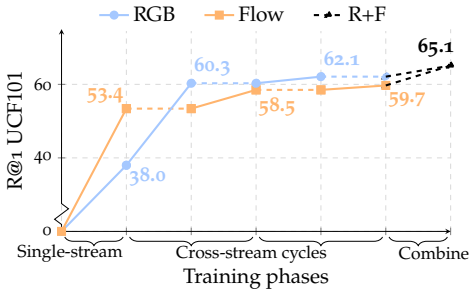

Figure 5.1: **Retrieval performance progress** on our training phases. RGB and flow are subsequently optimized in one cross-stream cycle, where a dotted line indicates no optimization. We report Top-1 Recall (R@1) on UCF101.

stream, we obtain a further performance boost shown as ViCC-R+F-2 (65.1% *vs*. 62.1%). We outperform CoCLR Han et al. (2020b) on retrieval by +9.5%, demonstrating the benefit of cross-stream prototype contrasting in ViCC. In linear probe downstream classification, our RGB-2 model again outperforms the RGB-1 one by a significant margin (72.2% *vs*. 49.2%). When end-to-end finetuned our self-supervised RGB-2 outperforms RGB-1 (84.3% *vs*. 81.8%). Further improvement is found by combining the predictions of the two streams, obtaining the result for R+F (90.5% *vs*. 84.3%). Here, our performance for R+F is on par with the RGB model from Han et al. (2020b).

As our cross-stream phase consists of cycles in which we alternate the training of streams, we further analyse the performance progress on video retrieval across training phases in Figure 5.1. We show the evolution from single-stream to cross-stream for both models, where cross-stream consists of two cycles in which RGB and Flow are trained alternately. It can be seen that representations for both models continue to improve after one cycle, indicating that the alternating scheme is beneficial for ViCC representations.

5.2.2. Ablations on stream views

We perform an ablation study on our model by investigating the importance of the streams used as views for both prediction and assignment. We first consider the number of features for prediction, where the normal setting is to use all other available views from streams s and t for prediction of each assignment q. We now study the setting where we use two features for prediction originating only from the other stream t. Table 5.2 shows results for both settings, reporting Top-1 accuracy on UCF101 action recognition using the finetuning protocol. We find that using two features results in a slightly worse performance overall, suggesting that more views are beneficial for prediction of the assignments despite originating from the same stream as the assignment. The second setting that we evaluate is only using the other stream t for assignment, where we map only the two features from stream t to prototypes C_s. Note, the prediction is performed as normal using all other available views. Both models are again slightly underperforming compared to using all views. The results for stream views in both settings suggest that the information used from the other stream in cross-stream training is of more significance than its own stream. Indeed, we find that ViCC is roobust against changes in views

	Streams for prediction		Streams for assignment	
Method	$s+t$	t	$s+t$	t
ViCC-RGB-2	84.3	83.8	84.3	84.1
ViCC-R+F-2	90.5	90.2	90.5	90.0

Table 5.2: **Ablations on streams** used as views for assignment and prediction. We report Top-1 accuracy on action recognition finetuning on UCF101.

	Number of prototypes		
Method	100	300	1000
ViCC-RGB-2	83.5	84.3	83.9
ViCC-R+F-2	89.2	90.5	90.0

Table 5.3: **Impact of number of prototypes**. We report Top-1 accuracy on action recognition finetuning on UCF101.

	Queue size		
Method	3840	1920	0
ViCC-RGB-2	84.5	84.3	84.7
ViCC-R+F-2	90.4	90.5	90.2

Table 5.4: **Impact of queue size.** We report Top-1 accuracy on action recognition finetuning on UCF101.

from its own stream as it almost performs in line with results using all views for both prediction and assignment.

5.2.3. Impact of number of prototypes

Here we evaluate the impact of the number of stream prototypes K. Explored previously by (Caron et al., 2020) on ImageNet (Deng et al., 2009), they found no significant impact on performance when varying the prototypes by several orders of magnitude using a sufficient large amount of prototypes. In Table 5.3, we show results on varying the number of prototypes to K={100, 1000}. We observe a slightly worse result for both settings for the RGB model and the R+F model. As we find no significant impact on the performance, our results are in line with previous work which suggests that the soft prototype mappings used for contrasting in ViCC are not necessarily a self-labeling approach similar to other pseudo-labeling approaches (Asano et al., 2020a,b; Caron et al., 2018; Gavrilyuk et al., 2021; Yan et al., 2020), despite the usefulness in contrasting for representation learning.

5.2.4. Ablation studies on queue size

We investigate the effect of the queue size on performance. The queue is used in the assignment of features to K prototypes. In theory, using more features on top of the current batch for the Sinkhorn-Knopp algorithm should result in a more accurate assignment. Results for queue sizes {3840, 1920, 0} are shown in Table 5.4. We report Top-1 accuracy on action recognition on UCF101 finetuning. For queue size 3840, we observe that the larger queue size is not necessary or beneficial for UCF101 self-supervised pretraining, as the differences in performance are minimal. Indeed, we find that using no queue almost performs on par with our default queue size of 1920. We conjecture that our mini-batches may already provide enough features for ViCC self-supervision on UCF101.

	Pretrain stage						Finetune	
Method	Dataset	Backbone	Param	Res	Frames	Modality	UCF101	HMDB51
OPN (Lee et al., 2017)	UCF101	VGG	8.6M	80	16	V	59.8	23.8
VCOP (Xu et al., 2019)	UCF101	R(2+1)D	14.4M	112	16	V	72.4	30.9
Var. PSP (Cho et al., 2020)	UCF101	R(2+1)D	14.4M	112	16	V	74.8	36.8
Pace Pred (Wang et al., 2020)	UCF101	R(2+1)D	14.4M	112	16	V	75.9	35.9
VCP (Luo et al., 2020b)	UCF101	R(2+1)D	14.4M	112	16	V	66.3	32.2
PRP (Yao et al., 2020b)	UCF101	R(2+1)D	14.4M	112	16	V	72.1	35.0
RTT (Jenni et al., 2020)	UCF101	R(2+1)D	14.4M	112	16	V	81.6	46.4
Pace Pred (Wang et al., 2020)	K-400	R(2+1)D	14.4M	112	16	V	77.1	36.6
XDC (Alwassel et al., 2020)	K-400	R(2+1)D	14.4M	224	32	V+A	86.8	52.6
SeLaVi (Asano et al., 2020a)	VGG-sound	R(2+1)D	14.4M	112	30	V+A	87.7	53.1
GDT (Patrick et al., 2020)	Audioset	R(2+1)D	14.4M	224	32	V+A	92.5	66.1
ViCC-RGB (*ours*)	UCF101	R(2+1)D	14.4M	128	32	V	**82.8**	**52.4**
ViCC-R+F (*ours*)	UCF101	R(2+1)D	14.4M	128	32	V	**88.8**	**61.5**
DPC (Han et al., 2019)	UCF101	R2D3D	14.2M	128	40	V	60.6	-
MemDPC (Han et al., 2020a)	UCF101	R2D3D	14.2M	224	40	V	69.2	-
MemDPC † (Han et al., 2020a)	UCF101	R2D3D	14.2M	224	40	V	84.3	-
Pace Pred (Wang et al., 2020)	UCF101	S3D-G	9.6M	224	64	V	87.1	52.6
CoCLR (Han et al., 2020b)	UCF101	S3D	8.8M	128	32	V	81.4	52.1
CoCLR † (Han et al., 2020b)	UCF101	S3D	8.8M	128	32	V	87.3	58.7
CoCLR † (Han et al., 2020b)	K-400	S3D	8.8M	128	32	V	90.6	62.9
SpeedNet (Benaim et al., 2020)	K-400	S3D-G	8.8M	128	32	V	81.1	48.8
MIL-NCE (Miech et al., 2020)	HTM	S3D	8.8M	224	32	V+T	91.3	61.0
CBT (Sun et al., 2019)	K-600	S3D	8.8M	112	16	V+T	79.5	44.6
ViCC-RGB (*ours*)	UCF101	S3D	8.8M	128	32	V	**84.3**	**47.9**
ViCC-R+F (*ours*)	UCF101	S3D	8.8M	128	32	V	**90.5**	**62.2**

Table 5.5: **Comparison with prior self-supervised works on end-to-end finetuning for video action recognition** on UCF101 and HMDB51. We report Top-1 accuracy and compare with self-supervision pretraining on UCF101. In grey color we show larger pretraining datasets such as K-400 (Carreira and Zisserman, 2017) and multi-modal approaches pretrained on VGG-sound (Chen et al., 2020a), Audioset (Gemmeke et al., 2017), HTM (Miech et al., 2019) and K-600 (Carreira et al., 2018) (where T is text, A is audio).

5.3 Comparison with state-of-the-art

In this section, we compare our method with self-supervised methods on action classification and nearest-neighbour video retrieval, reporting models from the cross-stream stage.

5.3.1. Action recognition

We compare with several self-supervised methods on action recognition in Table 5.5, reporting our results for two backbone architectures. We organized the methods by backbone and include settings such as resolution (Res), number of frames and number of parameters (Param) for a fairer comparison. We include several methods pretrained on larger training datasets for both visual-only and multi-modal methods. In the following, we compare with visual-only modality on the same training set, with visual-only on larger datasets, and with multi-modal approaches on end-to-end finetuning.

First, we significantly outperform previous approaches pretrained on UCF101 when considering the visual modality (V). On the S3D backbone, our R+F model (obtained by averaging

	Pretrain stage						Linear	
Method	Dataset	Backbone	Param	Res	Frames	Modality	UCF101	HMDB51
ViCC-RGB (*ours*)	UCF101	R(2+1)D	14.4M	128	32	V	**74.4**	**30.8**
ViCC-R+F (*ours*)	UCF101	R(2+1)D	14.4M	128	32	V	**78.3**	**45.2**
CoCLR (Han et al., 2020b)	UCF101	S3D	8.8M	128	32	V	70.2	39.1
CoCLR † (Han et al., 2020b)	UCF101	S3D	8.8M	128	32	V	72.1	40.2
CoCLR † (Han et al., 2020b)	K-400	S3D	8.8M	128	32	V	77.8	52.4
MIL-NCE (Miech et al., 2020)	HTM	S3D	8.8M	224	32	V+T	82.7	53.1
CBT (Sun et al., 2019)	K-600	S3D	8.8M	112	16	V+T	54.0	29.5
ViCC-RGB (*ours*)	UCF101	S3D	8.8M	128	32	V	**72.2**	**38.5**
ViCC-R+F (*ours*)	UCF101	S3D	8.8M	128	32	V	**78.0**	**47.9**

Table 5.6: **Comparison with self-supervised works on action recognition with linear probe** on UCF101 and HMDB51. We report Top-1 accuracy and compare with UCF101 self-supervision pretraining. In grey color we show larger pretraining datasets such as K-400 (Carreira and Zisserman, 2017) and multi-modal approaches pretrained on HTM (Miech et al., 2019) and K-600 (Carreira et al., 2018) (where T is text).

RGB and flow predictions) achieves a Top-1 accuracy of 90.5% on UCF101 and a Top-1 accuracy of 62.2% on HMDB51. Our approach outperforms the best model of Han *et al*. (Han et al., 2020b) by 3.2% on UCF101 and by 3.5% on HMDB51. We also achieve better performance than Pace Pred (Wang et al., 2020), which uses the S3D-G (Xie et al., 2018) backbone, on both UCF101 and HMDB51. Using the R(2+1)D backbone, we obtain a Top-1 accuracy of 82.8% on UCF101 and a Top-1 accuracy of 52.4% on HMDB51 for RGB. When combining RGB and Flow predictions (R+F), we obtain 88.8% and 61.5% on the datasets respectively. We outperform VCOP (Luo et al., 2020a), VCOP (Xu et al., 2019), PRP (Yao et al., 2020b) by a wide margin for both our models. With the R+F model we obtain a 7.2% increase on UCF101 and a 15.1% increase over RTT (Jenni et al., 2020), underlined in the table as the second best result. ViCC models therefore consistently outperform previous works on both backbones and evaluation datasets, where optical flow provides only an optional performance boost. Our RGB model also outperforms the RGB model of MemDPC (Han et al., 2020a), and performs on par with their combined RGB and flow model. Comparing against visual-only information using larger training sets, we outperform methods on that use Kinetics (K-400) pretraining on HMDB51, using UCF101 pretraining, such as Pace Pred (Wang et al., 2019a) for R(2+1)D and SpeedNet for S3D-G (Benaim et al., 2020). We also perform better on HMDB51 than some multi-modal approaches that use text (Miech et al., 2020) and audio (Asano et al., 2020b) for similar resolution, number of frames and backbone.

Finally, in Table 5.6 we compare against methods on linear probe. We outperform CoCLR (Han et al., 2020b) on the same training dataset by a significant margin (+5.9% on UCF101 and +7.7% on HMDB51 for R+F) on the same backbone.

5.3.2. Nearest-neighbour video retrieval

Next, we compare with self-supervised approaches on nearest-neighbour clip retrieval in Table 5.7. All methods are pretrained on UCF101 and we report Recall@K. Our approach outperforms all previous self-supervised learning approaches by a significant margin on UCF101 and HMDB51 for both our backbone networks. We achieve a Top-1 Recall of 65.1% on UCF101 using the S3D backbone, outperforming the previous best by 9.2%. On HMDB51, we achieve a Top-1 Recall of 29.7%, which is a 8.8% increase on previous best. With the R(2+1)D backbone,

Method	Backbone	Modality	UCF101				HMDB51			
			R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
OPN (Lee et al., 2017)	VGG	V	19.9	28.7	34.0	40.6	-	-	-	-
ST Order (Büchler et al., 2018)	CaffeNet	V	25.7	36.2	42.2	49.2	-	-	-	-
ST-Puzzle (Kim et al., 2019)	R3D	V	19.7	28.5	33.5	40.0	-	-	-	-
VCOP (Xu et al., 2019)	R3D	V	14.1	30.3	40.4	51.1	7.6	22.9	34.4	48.8
Pace Pred (Wang et al., 2020)	R3D	V	23.8	38.1	46.4	56.6	-	-	-	-
Var. PSP (Cho et al., 2020)	R3D	V	24.6	41.9	51.3	62.7	-	-	-	-
RTT (Jenni et al., 2020)	R3D	V	26.1	48.5	59.1	69.6	-	-	-	-
MemDPC (Han et al., 2020a)	R2D3D	V	20.2	40.4	52.4	64.7	7.7	25.7	40.6	57.7
VCP (Luo et al., 2020b)	R(2+1)D	V	19.9	33.7	42.0	50.5	6.7	21.3	32.7	49.2
CoCLR (Han et al., 2020b)	S3D	V	55.9	70.8	76.9	82.5	26.1	45.8	57.9	69.7
ViCC-RGB (*ours*)	R(2+1)D	V	58.6	76.2	83.1	89.0	25.3	50.4	64.0	77.5
ViCC-R+F (*ours*)	R(2+1)D	V	59.9	77.6	84.6	90.6	28.3	52.7	65.3	77.0
ViCC-RGB (*ours*)	S3D	V	62.1	77.1	83.7	87.9	25.5	49.6	61.9	72.5
ViCC-R+F (*ours*)	S3D	V	**65.1**	**80.2**	**85.4**	**89.8**	**29.7**	**54.6**	**66.0**	**76.2**

Table 5.7: **Comparison with self-supervised methods on nearest-neighbour video retrieval**. All self-supervised methods are pretrained on UCF101 split 1. We show results on Top-k Recall (R@k) for k={1, 5, 10, 20} on UCF101 split 1 and HMDB51 split 1.

we obtain a Top-1 Recall of 58.6% on UCF101 and 25.3% on HMDB51 for RGB, and 59.9% on UCF101 and 28.3% on HMDB51 for R+F. Compared to other self-supervised works apart from the second best, the margins are significantly wider. We conclude that our cross-stream self-supervision model RGB learns useful motion features without needing optical flow during test time.

5.4 Qualitative results

5.4.1. Nearest-neighbour video retrieval

In Figure 5.2, we visualize query video clips from the UCF101 test set with its Top-3 nearest-neighbours from the UCF101 training set, retrieved using the ViCC representation without labels. The ground truth action labels are included above the video clips. We visualize results for both single-stream (RGB-1) and cross-stream (RGB-2). Our qualitative results further support the benefit of cross-stream training (RGB-2), showing that it helps to retrieve videos from the same semantic categories compared to the model only trained on single-stream, despite significant changes in appearance and background (*e.g.* *Swing* and *WalkingWithDog*). More difficult is the retrieval for the third query video from class *BlowDryHair*, but we again observe that cross-stream training helps to retrieve the first result from the correct class.

5.4.2. T-SNE Visualization

In this section, we visualize representations of the UCF101 test set using the t-SNE clustering algorithm (van der Maaten and Hinton, 2008) to project features to 2D. For clarity, only 10 random action classes are visualized with a limited amount of random features for each class. Figure 5.3 shows the t-SNE visualization of features extracted from single-stream (RGB-1) and cross-stream (RGB-2) trained using the same number of epochs (500). It can be observed that the inter-class distance between certain classes such as *CricketBowling* and *GolfSwing* is increased

Figure 5.2: **Nearest-neighbour retrieval results with our representations**. The query video from the UCF101 test set is shown on the left, the top-3 nearest neighbours from the UCF101 training set on the right. Each video is visualized with 2 frames and we show results for single-stream (RGB-1) and cross-stream (RGB-2). The action label is shown above the video (not used during training), where green denotes the correct label and red denotes an incorrect result. Best viewed in color.

from RGB-1 to RGB-2. Moreover, the intra-class distance is reduced for classes *FrisbeeCatch*, *BasketballDunk* and *ApplyEyeMakeup*, which can be attributed to the benefit of motion learning from the flow encoder in cross-stream.

5.5 Analysis of Prototypes

This section focuses on further analysis of the prototypes. The main purpose of the prototype sets in ViCC is to guide the contrasting of groups of views from streams in each iteration. In combination with the relatively stable performance observed when varying the number of prototypes, we conjecture that the prototypes are not a pseudo-labeling approach similar to other methods (Asano et al., 2020a,b; Caron et al., 2018; Gavrilyuk et al., 2021; Yan et al., 2020). Despite this intuition and our use of soft assignments, we investigate the prototypes by visualizing video samples assigned to the same prototypes when rounding the assignments. We also evaluate the rounded prototype assignments from several of our self-supervised stages on standard cluster evaluation metrics.

5.5.1. Visualization of Prototypes

In Figure 5.4 we show the hard assignment of video samples to random prototypes. Video samples with the highest similarity scores to the prototype clusters are visualized. Prototype scores are indicated on the samples and the ground truth class labels of the samples are indicated below the groups. We can observe that video samples assigned to the same prototypes share semantic similarity and even belong to the same action class, despite the fact that class labels

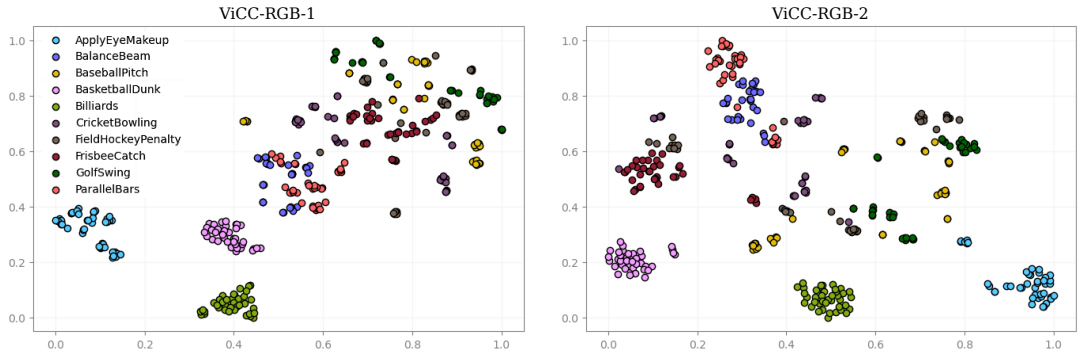

Figure 5.3: **t-SNE visualization** of the feature representations of UCF101 test set after 500 epochs of ViCC training. On the left RGB-1 single-stream is shown and on the right RGB-2 cross-stream.

Figure 5.4: **Visualization of rounded assignments to random ViCC prototypes** using videos from UCF101. Samples with high similarity scores (visualized on the samples) to the prototypes are shown. The ground truth labels of all the video samples are included below (not used during training).

are not used during ViCC training. The prototypes seem effective at grouping together views from the same semantic class label, as the samples visualized are all from the same class. These semantically similar sets in ViCC thereby provide an advantage for video representation learning over methods that use contrastive instance learning.

5.5.2. Cluster evaluation

In this section, we evaluate the hard assignment of our prototype sets with standard cluster evaluation measures as done in (Asano et al., 2020b; Caron et al., 2018). Although the ground truth number of clusters is not known in advance for self-supervised training, we set the number of prototypes to K=101 for evaluation purposes only to match the number of class labels for UCF101. The Hungarian algorithm (Kuhn, 1955) is then used to match self-supervised labels to the ground truth labels to obtain accuracy (Acc). We also report the Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), mean entropy per cluster (where the optimal number is 0) and mean maximal purity per cluster. The evaluation measures are detailed in Appendix A.1. For example, the NMI ranges from 0 (no mutual information) to 100% (implying

Method	Acc	NMI	ARI	Entropy	Max Purity
ViCC-RGB-1	32.3	62.5	16.4	1.6	36.8
ViCC-Flow-1	34.4	63.1	17.6	1.5	39.1
ViCC-RGB-2	40.8	67.8	24.5	1.4	45.1
ViCC-Flow-2	40.3	67.0	23.5	1.4	45.3

Table 5.8: **Cluster evaluation of ViCC prototypes** when rounding the assignments evaluated on the UCF101 test set.

perfect correlation between self-supervised labels and the ground truth labels). Table 5.8 shows that our prototypes from the cross-stream stage (RGB-2 and Flow-2) obtain better performance on all measures compared to prototypes learned only on their own stream (RGB-1 and Flow-1), achieving *e.g.* a higher NMI, lower mean entropy per cluster and higher mean maximal purity.

6 Discussion and Future work

Instance-level contrastive learning has reached performance levels challenging supervised pretraining on image recognition tasks, while providing the vital benefit of not needing annotation. Following the intuition that instance learning for video can be improved in terms of both task efficiency and the representations produced, we explored alternatives that touch upon two main improvements over the contrastive instance loss: using prototypes instead of instances, and incorporating optical flow as a second stream. We developed Video Cross-Stream Prototypical Contrasting (ViCC), inspired by recent works on both improvements. Here, we will provide a discussion on our contributions and expand on possibilities for future work. We touch upon the computational efficiency of components of our method in Section 6.1. We discuss the generality of our framework in Section 6.2.

6.1 Efficiency and model configuration

One motivation in the development of our method is to improve the efficiency of the contrastive learning task. Contrastive learning can be seen as 'learning only by comparison', without any supervision in the form of labels. The contributing factor of the main breakthrough of contrastive instance learning is the access to more computational resources and therefore GPU memory, allowing the contrasting with many features in each iteration. This recently enabled outstanding results compared to early works in metric learning. By computing the similarity between features and prototypes instead of between individual features as in contrastive instance learning, we hypothesized that this would produce a more cost-effective method. Instead of comparing to countless negative examples one at a time, we compare to multiple indirectly. An important benefit is that negative examples are not needed. Although we definitely improve the task of instance contrastive learning as demonstrated in the comparison to several baselines, our task is not entirely free from random sampling. This is because having access to more features allows the Sinkhorn-Knopp assignment to be more accurate, requiring a small storage space. As we have seen that additional features are not essential in our case, we leave for future work to investigate to what extent the storage space is needed when using larger datasets.

In our ablation studies, we investigated how using certain elements in training such as the number of stream views and the amount of prototypes influence results on several downstream video recognition tasks, to gain a better understanding of the behavior of our models. It was found that the performance benefit of our model in these studies was marginal. Therefore, we find it important to highlight that different configurations could potentially provide other benefits. For example, using fewer views for assignment and a smaller amount of prototypes could be desirable for the computational time in training because of fewer computations of the Sinkhorn-Knopp algorithm. Another component in our model that might need further attention because of potentially unnecessary configuration is the use of two sets of prototypes, one for each model. The cross-stream stage employs two sets because the prototypes are initialized with the prototypes obtained from the single-stream stage. As the cross-stream models gradually improve by assigning two streams to their prototypes sets during the whole cross-stream stage, we speculate that one prototype set shared between the two models might not make a large

difference in practice. However, one problem is how to obtain only one prototype set. A possible solution is to investigate whether it is possible to avoid initializing the cross-stream prototypes with the single-stream prototypes, and use one prototype set with random initialization instead. Future research could continue to explore this topic.

Lastly, one of our contributions is the motion knowledge transfer from the optical flow network to the RGB model, thereby avoiding the need for optical flow in deployment. A possible direction for future work could be studying the training scheme. In our brief exploration, we found no performance benefits and even divergence when using short alternation periods for training such as one or ten epochs. Future research is needed to confirm our initial findings and perhaps investigate whether improvements on the training time can be found by faster alternation while reaching similar performance levels.

6.2 Generality

We argue that learning from examples as done in our contrastive learning method is likely closer to biological processes in human vision than various other works, including supervised learning. It has been found that infants can rapidly learn prototypical representations from comparing objects (Mareschal and Quinn, 2001; Ribar et al., 2004; Younger and Cohen, 1985). Although neural networks are biologically inspired, there are probably numerous differences with processes in the human brain, in particular regarding training and backpropagation (Crick, 1989; Zipser and Rumelhart, 1993). It must be mentioned therefore that the connection of our method to the two-stream hypothesis is a rough comparison, albeit we do believe that processes in the human brain can provide guidance for vision algorithm development. One possible future direction to bring our method closer to human learning capabilities could be to introduce a hierarchy structure to the prototypes, since humans can abstract on every level. This can for instance be achieved by employing several clustering steps with varying numbers of clusters, or via the use of hyperbolic embeddings (Khrulkov et al., 2020; Nickel and Kiela, 2017) that are naturally suited to encode hierarchy, with benefits shown on the natural language processing domain (Gulcehre et al., 2019; Tifrea et al., 2018).

We hypothesized that our prototypes could be a useful element in learning video representations. From our experiments and visualizations, it seems that the purpose of our prototypes is to provide an intermediate step that possibly offers fewer comparisons, where the semantically similar groups formed by the assignments function as temporary categories instead of learnable consistent clusters. The impact of these prototypes being learned is therefore likely not an important component, although future work here is needed. To what extent the prototypes could be leveraged in other ways, for instance self-labeling, could also be examined.

Finally, our novel method provides a framework for using multiple modalities together with prototypical contrasting. Video in particular is well suited for using multiple types of input. Given the strength and flexibility of this framework we conjecture that more gains could be found in using external complementary information through multi-modal approaches, for instance by leveraging audio.

7 Conclusion

In this thesis, we present the Video Cross-Stream Prototypical Contrasting (ViCC) framework for self-supervised representation learning. We demonstrate the advantages of using similar semantic groupings of RGB and flow views over methods that use instance-level contrastive learning, avoiding redundant comparisons and improving performance. By learning through predicting consistent prototype assignments from views originating from both streams, ViCC effectively transfers knowledge from the motion representation to appearance and vice versa. We demonstrate state-of-the-art performance on downstream video recognition tasks using visual-only self-supervision.

References

Humam Alwassel, Dhruv Mahajan, Bruno Korbar, Lorenzo Torresani, Bernard Ghanem, and Du Tran. Self-Supervised Learning by Cross-Modal Audio-Video Clustering. In *NeurIPS*, 2020.

Yuki M. Asano, Mandela Patrick, Christian Rupprecht, and Andrea Vedaldi. Labelling unlabelled videos from scratch with multi-modal self-supervision. In *NeurIPS*, 2020a.

Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *ICLR*, 2020b.

Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, William T. Freeman, Michael Rubinstein, Michal Irani, and Tali Dekel. SpeedNet: Learning the Speediness in Videos. In *CVPR*, 2020.

Jane Bromley, James W. Bentz, Léon Bottou, Isabelle Guyon, Yann Lecun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. *Pattern Recognition and Artificial Intelligence*, 07, 1993.

Uta Büchler, Biagio Brattoli, and Björn Ommer. Improving Spatiotemporal Self-Supervision by Deep Reinforcement Learning. In *ECCV*, 2018.

Yue Cao, Zhenda Xie, Bin Liu, Yutong Lin, Zheng Zhang, and Han Hu. Parametric Instance Classification for Unsupervised Visual Feature Learning. In *NeurIPS*, 2020.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep Clustering for Unsupervised Learning of Visual Features. In *ECCV*, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *NeurIPS*, 2020.

Joao Carreira and Andrew Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *CVPR*, 2017.

Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A Short Note about Kinetics-600. *ArXiv preprint arXiv:1808.01340*, 2018.

Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large Scale Online Learning of Image Similarity through Ranking. *Pattern Recognition and Image Analysis*, 5524, 2009.

Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. Vggsound: A Large-Scale Audio-Visual Dataset. In *ICASSP*, 2020a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*, 2020b.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved Baselines with Momentum Contrastive Learning. *ArXiv preprint arXiv:2003.04297*, 2020c.

H. Cho, Tae-Hoon Kim, H. J. Chang, and Wonjun Hwang. Self-Supervised Spatio-Temporal Representation Learning Using Variable Playback Speed Prediction. *ArXiv preprint arXiv:2003.02692*, 2020.

Ching-Yao Chuang, Joshua Robinson, Lin Yen-Chen, Antonio Torralba, and Stefanie Jegelka. Debiased Contrastive Learning. In *NeurIPS*, 2020.

Nieves Crasto, Philippe Weinzaepfel, Karteek Alahari, and Cordelia Schmid. MARS: Motion-Augmented RGB Stream for Action Recognition. In *CVPR*, 2019.

Francis Crick. The recent excitement about neural networks. *Nature*, 337, 1989.

Marco Cuturi. Sinkhorn Distances: Lightspeed Computation of Optimal Transportation Distances. In *NIPS*, 2013.

N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, 2005.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

Ali Diba, Vivek Sharma, Luc Van Gool, and Rainer Stiefelhagen. DynamoNet: Dynamic Action and Motion Network. In *ICCV*, 2019.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear Independent Components Estimation. In *ICLR Workshop*, 2015.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. In *ICLR*, 2017.

Carl Doersch and Andrew Zisserman. Multi-Task Self-Supervised Visual Learning. In *ICCV*, 2017.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised Visual Representation Learning by Context Prediction. In *ICCV*, 2016.

Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term Recurrent Convolutional Networks for Visual Recognition and Description. In *CVPR*, 2016.

Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow with Convolutional Networks. In *ICCV*, 2015a.

Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative Unsupervised Feature Learning with Exemplar Convolutional Neural Networks. *ArXiv preprint arXiv:1406.6909*, 2015b.

Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*, 2003.

Christoph Feichtenhofer. X3D: Expanding Architectures for Efficient Video Recognition. *ArXiv preprint arXiv:2004.04730*, 2020.

Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional Two-Stream Network Fusion for Video Action Recognition. In *CVPR*, 2016.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast Networks for Video Recognition. In *ICCV*, 2019.

Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-Supervised Video Representation Learning With Odd-One-Out Networks. In *CVPR*, 2017.

Kirill Gavrilyuk, Mihir Jain, Ilia Karmanov, and Cees G. M. Snoek. Motion-Augmented Self-Training for Video Recognition at Smaller Scale. *ArXiv preprint arXiv:2105.01646*, 2021.

Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio Set: An ontology and human-labeled dataset for audio events. In *ICASSP*, 2017.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised Representation Learning by Predicting Image Rotations. *ArXiv preprint arXiv:1803.07728*, 2018.

Clement Godard, Peter Hedman, Wenbin Li, and Gabriel J Brostow. Multi-view reconstruction of highly specular surfaces in uncontrolled environments. In *3DV*, 2015.

Melvyn A Goodale and A.David Milner. Separate visual pathways for perception and action. *Trends in Neurosciences*, 15, 1992.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *ArXiv preprint arXiv:1406.2661*, 2014.

Ross Goroshin, Joan Bruna, Jonathan Tompson, David Eigen, and Yann LeCun. Unsupervised Learning of Spatiotemporally Coherent Metrics. In *ICCV*, 2015.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised Learning. In *NeurIPS*, 2020.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. Hyperbolic attention networks. In *ICLR*, 2019.

Michael U Gutmann and Aapo Hyvarinen. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *JMLR*, 13, 2012.

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality Reduction by Learning an Invariant Mapping. In *CVPR*, 2006.

Tengda Han, Weidi Xie, and Andrew Zisserman. Video Representation Learning by Dense Predictive Coding. In *ICCV Workshop*, 2019.

Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented Dense Predictive Coding for Video Representation Learning. In *ECCV*, 2020a.

Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised Co-training for Video Representation Learning. In *NeurIPS*, 2020b.

Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. 3D ResNets for Action Recognition. https://github.com/kenshohara/3D-ResNets-PyTorch, 2018a.

Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? In *CVPR*, 2018b.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*, 2020.

Geoffrey Hinton. Deep belief nets. *Encyclopedia of Machine Learning*, 2010.

R. Devon Hjelm and Philip Bachman. Representation Learning with Video Deep InfoMax. *ArXiv preprint arXiv:2007.13278*, 2020.

R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *NeurIPS*, 2019.

Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17, 1981.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *ACL*, 2018.

Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *CVPR*, 2017.

S. Jenni, Givi Meishvili, and P. Favaro. Video Representation Learning by Recognizing Temporal Transformations. In *ECCV*, 2020.

Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3D Convolutional Neural Networks for Human Action Recognition. *PAMI*, 35, 2013.

Longlong Jing, Xiaodong Yang, Jingen Liu, and Yingli Tian. Self-Supervised Spatiotemporal Feature Learning via Video Rotation Prediction. *ArXiv preprint arXiv:1811.11387*, 2019.

Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, 14, 1973.

Yannis Kalantidis, Mert Bulent Sariyildiz, and Noe Pion. Hard Negative Mixing for Contrastive Learning. In *NeurIPS*, 2020.

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.

Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In *CVPR*, 2020.

Dahun Kim, Donghyeon Cho, and In So Kweon. Self-Supervised Video Representation Learning with Space-Time Cubic Puzzles. In *AAAI*, 2019.

Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. *ArXiv Preprint arXiv:1807.03039*, 2018.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *ArXiv Preprint arXiv:1312.6114*, 2014.

Quan Kong, W. Wei, Z. Deng, Tomoaki Yoshinaga, and T. Murakami. Cycle-Contrast for Self-Supervised Video Representation Learning. In *ArXiv preprint arXiv:2010.14810*, 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

Hilde Kuehne, Hueihan Jhuang, Estibaliz Garrote, Tomaso Poggio, and Thomas Serre. HMDB: A large video database for human motion recognition. In *ICCV*, 2011.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2, 1955.

Zihang Lai and Weidi Xie. Self-supervised Learning for Video Correspondence Flow. In *BMVC*, 2019.

Zihang Lai, Erika Lu, and Weidi Xie. MAST: A Memory-Augmented Self-supervised Tracker. In *CVPR*, 2020.

I. Laptev, M. Marszalek, C. Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008.

Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised Representation Learning by Sorting Sequences. In *ICCV*, 2017.

Junnan Li, Pan Zhou, Caiming Xiong, Richard Socher, and Steven C. H. Hoi. Prototypical Contrastive Learning of Unsupervised Representations. In *ICLR*, 2021.

Wenbin Li, Darren Cosker, and Matthew Brown. Drift robust non-rigid optical flow enhancement for long sequences. *Journal of Intelligent & Fuzzy Systems*, 31, 2016.

Dezhao Luo, Bo Fang, Yin-qing Zhou, Yucan Zhou, D. Wu, and Weiping Wang. Exploring Relations in Untrimmed Videos for Self-Supervised Learning. *ArXiv preprint arXiv:2008.02711*, 2020a.

Dezhao Luo, Chang Liu, Yu Zhou, Dongbao Yang, Can Ma, Qixiang Ye, and Weiping Wang. Video Cloze Procedure for Self-Supervised Spatio-Temporal Learning. In *AAAI*, 2020b.

Aravindh Mahendran, James Thewlis, and Andrea Vedaldi. Cross Pixel Optical Flow Similarity for Self-Supervised Learning. In *ACCV*, 2018.

Denis Mareschal and Paul C. Quinn. Categorization in infancy. *Trends in Cognitive Sciences*, 5, 2001.

Robert D McIntosh and Thomas Schenk. Two visual streams for perception and action: Current trends. *Neuropsychologia*, 47, 2009.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*, 2019.

Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-End Learning of Visual Representations from Uncurated Instructional Videos. In *CVPR*, 2020.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *ArXiv preprint arXiv:1301.3781*, 2013.

Ishan Misra and Laurens van der Maaten. Self-Supervised Learning of Pretext-Invariant Representations. In *CVPR*, 2020.

Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and Learn: Unsupervised Learning using Temporal Order Verification. In *ECCV*, 2016.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *NeurIPS*, 2017.

Mehdi Noroozi and Paolo Favaro. Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. In *ECCV*, 2016.

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard Lewis, and Satinder Singh. Action-Conditional Video Prediction using Deep Networks in Atari Games. In *NIPS*, 2015.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context Encoders: Feature Learning by Inpainting. *ArXiv preprint arXiv:1604.07379*, 2016.

Mandela Patrick, Yuki M. Asano, Polina Kuznetsova, Ruth Fong, João F. Henriques, Geoffrey Zweig, and Andrea Vedaldi. Multi-modal Self-Supervision from Generalized Data Transformations. *ArXiv preprint arXiv:2003.04298*, 2020.

Gabriel Peyré and Marco Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends in Machine Learning*, 11, 2019.

Lyndsey C. Pickup, Zheng Pan, Donglai Wei, YiChang Shih, Changshui Zhang, Andrew Zisserman, Bernhard Scholkopf, and William T. Freeman. Seeing the Arrow of Time. In *CVPR*, 2014.

A. J. Piergiovanni, Anelia Angelova, and Michael S. Ryoo. Evolving Losses for Unsupervised Video Representation Learning. In *CVPR*, 2020.

Rui Qian, Tianjian Meng, Boqing Gong, Ming-Hsuan Yang, Huisheng Wang, Serge Belongie, and Yin Cui. Spatiotemporal Contrastive Video Representation Learning. In *CVPR*, 2021.

Zhaofan Qiu, Ting Yao, and Tao Mei. Learning Spatio-Temporal Representation with Pseudo-3D Residual Networks. In *ICCV*, 2017.

Danilo Rezende and Shakir Mohamed. Variational Inference with Normalizing Flows. In *ICML*, 2015.

Rebecca J Ribar, Lisa M Oakes, and Thomas L Spalding. Infants can rapidly form new categorical representations. *Psychonomic bulletin & review*, 11, 2004.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive Learning with Hard Negative Samples. In *ICLR*, 2021.

Rob Romijnders, Aravindh Mahendran, Michael Tschannen, Josip Djolonga, M. Ritter, N. Houlsby, and M. Lucic. Representation learning from videos in-the-wild: An object-centric approach. In *WACV*, 2021.

Gerald Schneider. Two visual systems: Brain mechanisms for localization and discrimination are dissociated by tectal and cortical lesions. *Science*, 163, 1969.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. *CVPR*, 2015.

Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27, 1948.

Karen Simonyan and Andrew Zisserman. Two-Stream Convolutional Networks for Action Recognition in Videos. In *NIPS*, 2014.

Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*, 2015.

Kihyuk Sohn. Improved Deep Metric Learning with Multi-class N-pair Loss Objective. In *NIPS*, 2016.

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.

Jonathan C. Stroud, David A. Ross, Chen Sun, Jia Deng, and Rahul Sukthankar. D3D: Distilled 3D Networks for Video Action Recognition. In *WACV*, 2020.

Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Learning Video Representations using Contrastive Bidirectional Transformer. *ArXiv preprint arXiv:1906.05743*, 2019.

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

Yongyi Tang, Xing Zhang, Jingwen Wang, Shaoxiang Chen, Lin Ma, and Yu-Gang Jiang. Non-local NetVLAD Encoding for Video Classification. In *ECCV*, 2018.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive Multiview Coding. In *ECCV*, 2020.

Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. Poincar\'e glove: Hyperbolic word embeddings. *ArXiv preprint arXiv:1810.06546*, 2018.

P. Tokmakov, M. Hebert, and C. Schmid. Unsupervised Learning of Video Representations via Dense Trajectory Clustering. *ArXiv preprint arXiv:2006.15731*, 2020.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. In *ICCV*, 2015.

Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A Closer Look at Spatiotemporal Convolutions for Action Recognition. In *CVPR*, 2018.

Michael Tschannen, Josip Djolonga, Marvin Ritter, Aravindh Mahendran, Xiaohua Zhai, Neil Houlsby, Sylvain Gelly, and Mario Lucic. Self-Supervised Learning of Video-Induced Visual Invariances. *ArXiv Preprint arXiv:1912.02783*, 2020.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. *ArXiv preprint arXiv:1807.03748*, 2019.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *JMLR*, 9, 2008.

Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term Temporal Convolutions for Action Recognition. *ArXiv preprint arXiv:1604.04494*, 2017.

Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing Motion and Content for Natural Video Sequence Prediction. In *ICLR*, 2017.

Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating Videos with Scene Dynamics. In *NIPS*, 2016.

Heng Wang and Cordelia Schmid. Action Recognition with Improved Trajectories. In *ICCV*, 2013.

Jiangliu Wang, Jianbo Jiao, Linchao Bao, Shengfeng He, Yunhui Liu, and Wei Liu. Self-supervised Spatio-temporal Representation Learning for Videos by Predicting Motion and Appearance Statistics. In *CVPR*, 2019a.

Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. Self-supervised Video Representation Learning by Pace Prediction. In *ECCV*, 2020.

Xiao Wang and Guo-Jun Qi. Contrastive Learning with Stronger Augmentations. *ArXiv preprint arXiv:2104.07713*, 2021.

Xiaolong Wang and Abhinav Gupta. Unsupervised Learning of Visual Representations using Videos. In *ICCV*, 2015.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local Neural Networks. *ArXiv preprint arXiv:1711.07971*, 2018.

Xiaolong Wang, Allan Jabri, and Alexei A. Efros. Learning Correspondence from the Cycle-Consistency of Time. In *CVPR*, 2019b.

Donglai Wei, Joseph J Lim, and Andrew Zisserman. Learning and Using the Arrow of Time. In *CVPR*, 2018.

Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-Term Feature Banks for Detailed Video Understanding. In *CVPR*, 2019.

Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised Feature Learning via Non-Parametric Instance-level Discrimination. In *CVPR*, 2018.

Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking Spatiotemporal Feature Learning: Speed-Accuracy Trade-offs in Video Classification. In *ECCV*, 2018.

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.

Xueting Yan, Ishan Misra, Abhinav Gupta, Deepti Ghadiyaram, and Dhruv Mahajan. ClusterFit: Improving Generalization of Visual Representations. In *CVPR*, 2020.

Ting Yao, Yiheng Zhang, Zhaofan Qiu, Yingwei Pan, and Tao Mei. SeCo: Exploring Sequence Supervision for Unsupervised Representation Learning. In *AAAI*, 2020a.

Yuan Yao, Chang Liu, Dezhao Luo, Yu Zhou, and Qixiang Ye. Video Playback Rate Perception for Self-supervisedSpatio-Temporal Representation Learning. In *CVPR*, 2020b.

Yang You, Igor Gitman, and Boris Ginsburg. Large Batch Training of Convolutional Networks. *ArXiv preprint arXiv:1708.03888*, 2017.

Barbara A. Younger and Leslie B. Cohen. How infants form categories. *19*, 1985.

Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond Short Snippets: Deep Networks for Video Classification. In *CVPR*, 2015.

C. Zach, T. Pock, and H. Bischof. A Duality Based Approach for Realtime TV-L1 Optical Flow. In *DAGM-Symposium*, 2007.

Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew Soon Ong, and Chen Change Loy. Online Deep Clustering for Unsupervised Representation Learning. In *CVPR*, 2020.

Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful Image Colorization. *ArXiv Preprint arXiv:1603.08511*, 2016.

Richard Zhang, Phillip Isola, and Alexei A. Efros. Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction. *ArXiv preprint arXiv:1611.09842*, 2017.

Jiaojiao Zhao and Cees G. M. Snoek. Dance with Flow: Two-in-One Stream Action Detection. In *CVPR*, 2019.

Chengxu Zhuang, Tianwei She, Alex Andonian, Max Sobol Mark, and Daniel Yamins. Unsupervised Learning from Video with Deep Neural Embeddings. *ArXiv Preprint arXiv:1905.11954*, 2020.

David Zipser and David E Rumelhart. The neurobiological significance of the new learning models. *Computational neuroscience*, 1993.

[heading=classicthesis]

List of Tables

List of Figures

List of Abbreviations

3DConvNets 3D Convolutional Neural Networks. 10, 14

ARI Adjusted Rand Index. 30, 47

CoCLR Co-training Contrastive Learning Representations. 15, 23, 24, 27

CPC Contrastive predictive Coding. 13, 15

IDT Improved Dense Trajectories. 9, 15

KL Kullback–Leibler. 13

MI Mutual Information. 12, 47

MoCo Momentum Contrast. 13, 22

NCE noise-contrastive estimation. 7, 13

NMI Normalized Mutual Information. 30, 31, 47

SwAV Swapping Assignments between Views. 14, 18, 22

ViCC Video Cross-Stream Prototypical Contrasting. 8, 17, 21, 22, 24, 25, 27–30, 32, 34, 44, 45

Appendix

A.1 Cluster evaluation metrics details

Let (X, Y) be a pair of random variables and $P_{(X,Y)}$ their joint distribution. The marginal distributions are P_X and P_Y. The Shannon entropy H (Shannon, 1948), quantifying the amount of uncertainty about the outcome of the random variable, is given by:

$$H(X) = -\sum_{x \in X} p_X(x) \log p_X(x). \tag{A.1}$$

The Mutual Information (MI) (Shannon, 1948) expresses how much knowing one of the random variables reduces uncertainty about the other. The MI is given by

$$MI(X;Y) = \sum_{x \in X, y \in Y} p_{(X,Y)}(x,y) \log \frac{p_{(X,Y)}(x,y)}{p_X(x) p_Y(y)} = H(Y) - H(Y|X). \tag{A.2}$$

The Normalized Mutual Information (NMI) is given by

$$NMI(X;Y) = \frac{MI(X;Y)}{\sqrt{H(X)H(Y)}}. \tag{A.3}$$

The Adjusted Rand Index (ARI), defined as the Rand Index (RI) corrected for change grouping of features, is defined as

$$ARI = \frac{RI - \mathbb{E}[RI]}{\max(RI) - \mathbb{E}[RI]}. \tag{A.4}$$

The ARI measure ranges from -1 to 1, with a value of 1 indicating identical agreement, and 0 meaning random agreement. The RI can be seen as a measure of correct decisions made by the clustering compared to the ground truth labels:

$$RI = \frac{TP + TN}{TP + FP + TN + FN}, \tag{A.5}$$

where TP is the number of true positives, TN indicates true negatives, FP indicates false positives and FN indicates false negatives. The *mean entropy per cluster*, as defined by (Asano et al., 2020a), is given by

$$<H> = \frac{1}{K} \sum_{k \in K} H(p(y|\hat{y}_k = k)), \tag{A.6}$$

where $\hat{y}$ are clusters and $p(y|\hat{y}_k = k)$ is the distribution of ground-truth labels for cluster k. Finally, to express the semantic purity of each cluster, the *mean maximal purity per cluster* (Asano et al., 2020a), ranging from $1/K$ to 100, can be defined as

$$<p_{\max}> = \frac{1}{K} \sum_{k \in K} \max(p(y|\hat{y}_k = k)). \tag{A.7}$$

A.2 More comparison with self-supervised works on action recognition

In Table A.1 we list more results from self-supervised methods evaluated on finetuning for action recognition. Our method still outperforms all methods pretrained on UCF101. We also outperform several methods pretrained on the larger dataset K-400, and achieve competitive performance compared to (Qian et al., 2021).

	Pretrain stage						Finetune	
Method	Dataset	Backbone	Param	Res	Frames	Modality	UCF101	HMDB51
OPN (Lee et al., 2017)	UCF101	VGG	8.6M	80	16	V	59.8	23.8
VCOP (Xu et al., 2019)	UCF101	R(2+1)D	14.4M	112	16	V	72.4	30.9
Var. PSP (Cho et al., 2020)	UCF101	R(2+1)D	14.4M	112	16	V	74.8	36.8
Pace Pred (Wang et al., 2020)	UCF101	R(2+1)D	14.4M	112	16	V	75.9	35.9
VCP (Luo et al., 2020b)	UCF101	R(2+1)D	14.4M	112	16	V	66.3	32.2
PRP (Yao et al., 2020b)	UCF101	R(2+1)D	14.4M	112	16	V	72.1	35.0
RTT (Jenni et al., 2020)	UCF101	R(2+1)D	14.4M	112	16	V	81.6	46.4
Pace Pred (Wang et al., 2020)	K-400	R(2+1)D	14.4M	112	16	V	77.1	36.6
MotionFit (Gavrilyuk et al., 2021)	K-400	R(2+1)D	14.4M	112	32	V	88.9	61.4
XDC (Alwassel et al., 2020)	K-400	R(2+1)D	14.4M	224	32	V+A	86.8	52.6
SeLaVi (Asano et al., 2020a)	VGG-sound	R(2+1)D	14.4M	112	30	V+A	87.7	53.1
GDT (Patrick et al., 2020)	Audioset	R(2+1)D	14.4M	224	32	V+A	92.5	66.1
ViCC-RGB (*ours*)	UCF101	R(2+1)D	14.4M	128	32	V	**82.8**	**52.4**
ViCC-R+F (*ours*)	UCF101	R(2+1)D	14.4M	128	32	V	**88.8**	**61.5**
DPC (Han et al., 2019)	UCF101	R2D3D	14.2M	128	40	V	60.6	-
MemDPC (Han et al., 2020a)	UCF101	R2D3D	14.2M	224	40	V	69.2	-
MemDPC † (Han et al., 2020a)	UCF101	R2D3D	14.2M	224	40	V	84.3	-
Pace Pred (Wang et al., 2020)	UCF101	S3D-G	9.6M	224	64	V	87.1	52.6
CoCLR (Han et al., 2020b)	UCF101	S3D	8.8M	128	32	V	81.4	52.1
CoCLR † (Han et al., 2020b)	UCF101	S3D	8.8M	128	32	V	87.3	58.7
CoCLR (Han et al., 2020b)	K-400	S3D	8.8M	128	32	V	87.9	54.6
CoCLR † (Han et al., 2020b)	K-400	S3D	8.8M	128	32	V	90.6	62.9
MotionFit (Gavrilyuk et al., 2021)	K-400	S3D	8.8M	224	64	V	90.1	50.6
SpeedNet (Benaim et al., 2020)	K-400	S3D-G	8.8M	128	32	V	81.1	48.8
MIL-NCE (Miech et al., 2020)	HTM	S3D	8.8M	224	32	V+T	91.3	61.0
CBT (Sun et al., 2019)	K-600	S3D	8.8M	112	16	V+T	79.5	44.6
ELo (Piergiovanni et al., 2020)	K-400	S3D	8.8M	224	32	V+T	93.8	67.4
ViCC-RGB (*ours*)	UCF101	S3D	8.8M	128	32	V	**84.3**	**47.9**
ViCC-R+F (*ours*)	UCF101	S3D	8.8M	128	32	V	**90.5**	**62.2**
VCOP (Xu et al., 2019)	UCF101	R3D	14.2M	112	16	V	64.9	29.5
Var. PSP (Cho et al., 2020)	UCF101	R3D	14.2M	112	16	V	69.0	33.7
VCP (Luo et al., 2020b)	UCF101	R3D	14.2M	112	16	V	66.0	31.5
PRP (Yao et al., 2020b)	UCF101	R3D	14.2M	112	16	V	66.5	29.7
RTT (Jenni et al., 2020)	UCF101	R3D	14.2M	112	16	V	77.3	47.5
RotNet3D (Jing et al., 2019)	K-400	R3D	33.6M	224	16	V	62.9	33.7
ST-Puzzle (Kim et al., 2019)	K-400	R3D	33.6M	224	16	V	65.8	33.7
DPC (Han et al., 2019)	K-400	R3D	14.2M	128	40	V	68.2	34.5
VIE (Zhuang et al., 2020)	K-400	R3D	14.2M	112	40	V	72.3	44.8
CVRL (Qian et al., 2021)	K-400	R3D-50	36.1M	224	16	V	92.1	65.4
Spatio-Temp (Wang et al., 2019a)	UCF101	C3D	58.3M	112	16	V	58.8	32.6
VCOP (Xu et al., 2019)	UCF101	C3D	58.3M	112	16	V	65.6	28.4
Var. PSP (Cho et al., 2020)	UCF101	C3D	58.3M	112	16	V	70.4	34.3
Pace Pred (Wang et al., 2020)	UCF101	C3D	58.3M	112	16	V	68.0	-
VCP (Luo et al., 2020b)	UCF101	C3D	58.3M	112	16	V	68.5	32.5
PRP (Yao et al., 2020b)	UCF101	C3D	58.3M	112	16	V	69.1	34.5
RTT (Jenni et al., 2020)	UCF101	C3D	58.3M	112	16	V	68.3	38.4
ST-Puzzle (Kim et al., 2019)	K-400	C3D	58.3M	112	16	V	60.6	28.3
Spatio-Temp (Wang et al., 2019a)	K-400	C3D	58.3M	112	16	V	61.2	33.4

Table A.1: **More comparison with prior self-supervised works on end-to-end finetuning for video action recognition** on UCF101 and HMDB51. We report Top-1 accuracy and compare with self-supervision pretraining on UCF101 including methods that use the backbones R3D (Hara et al., 2018a) and C3D (Tran et al., 2015). In grey color we show larger pretraining datasets such as K-400 (Carreira and Zisserman, 2017) and multi-modal approaches pretrained on VGG-sound (Chen et al., 2020a), Audioset (Gemmeke et al., 2017), HTM (Miech et al., 2019) and K-600 (Carreira et al., 2018) (where T is text, A is audio).