MobileVOS: Real-Time Video Object Segmentation Contrastive Learning meets Knowledge Distillation

Roy Miles* Mehmet Kerim Yucel Bruno Manganelli Albert Saà-Garriga
Samsung Research UK

Abstract

This paper tackles the problem of semi-supervised video object segmentation on resource-constrained devices, such as mobile phones. We formulate this problem as a distillation task, whereby we demonstrate that small space-time-memory networks with finite memory can achieve competitive results with state of the art, but at a fraction of the computational cost (32 milliseconds per frame on a Samsung Galaxy S22). Specifically, we provide a theoretically grounded framework that unifies knowledge distillation with supervised contrastive representation learning. These models are able to jointly benefit from both pixel-wise contrastive learning and distillation from a pre-trained teacher. We validate this loss by achieving competitive $\mathcal{J}\&\mathcal{F}$ to state of the art on both the standard DAVIS and YouTube benchmarks, despite running up to $\times 5$ faster, and with $\times 32$ fewer parameters.

1. Introduction

Video Object Segmentation (VOS) is a foundational task in computer vision, where the aim is to segment and track objects in a sequence of frames. VOS is the backbone of many applications such as video editing, autonomous driving, surveillance, and augmented reality [55]. In this work, we focus on semi-supervised VOS (SVOS), where the mask annotations for only the initial frame are provided. SVOS is a notoriously difficult task, where one needs to model the motion and appearance changes under severe occlusion and drifts, while performing well in a class-agnostic manner.

SVOS approaches can be divided into three main methods, namely, online fine-tuning [4, 20, 27, 36, 43, 53], object flow [12, 35, 40, 50, 61, 63], and offline matching [8, 21, 52, 64, 65]. Memory-based matching methods have shown impressive results in the last years. These approaches [9, 10, 26, 29, 32, 34, 41, 47, 54, 57] leverage memory banks to encode and memorize previous frames. They find feature correspondences between the memory and the current frame to predict the next object masks. Despite their good results, memory-networks tend to suffer from a trade-off between the memory usage and accuracy [26]. Storing a reduced number of frames [9, 26, 54], storing local segmentation features [37, 57] (i.e. features of only a part of the image), or segmenting only on keyframes [59], have been proposed as methods to tackle this tradeoff. Recent methods are now accurate and fast on high-end GPUs, but there is no method in the literature that is capable of retaining state-of-the-art results while performing in real-time on mobile phones.

Our work addresses real-time SVOS on resource-constrained devices. Specifically, we focus on memory-based networks and begin to bridge the gap between infinite memory and finite memory networks. In contrast with existing methods that do so by architectural [26] and memory-bank design changes [9, 54], we adopt a novel approach that addresses the problem through knowledge distillation. Specifically, we propose a pixel-wise representation distillation loss that aims to transfer the structural information of a large, infinite memory teacher to that of a smaller, finite memory student. We then show that using a simple boundary-aware sampling strategy can improve training convergence. Finally, we provide a natural generalisation to encompass a supervised pixel-wise contrastive representation objective. Using ground-truth labels as another venue for structural information, we interpolate between representation distillation and contrastive learning with a hyper-parameter, and use it as a unified loss. Our results on popular SVOS datasets show that we are competitive with state-of-the-art [26], while running $\times 5$ faster and having $\times 32$ fewer parameters. Our model, without any pruning or quantisation, runs comfortably in real-time on a mobile device. The summary of our main contributions is given as follows:

- We unify knowledge distillation and supervised contrastive learning to design a novel representation-based loss that bridges the gap between the large, infinite memory models and those with small, finite memory. Additionally, we show that a simple boundary-aware pixel sampling strategy can further improve the results and model convergence.

*Work done while being an intern at Samsung Research UK

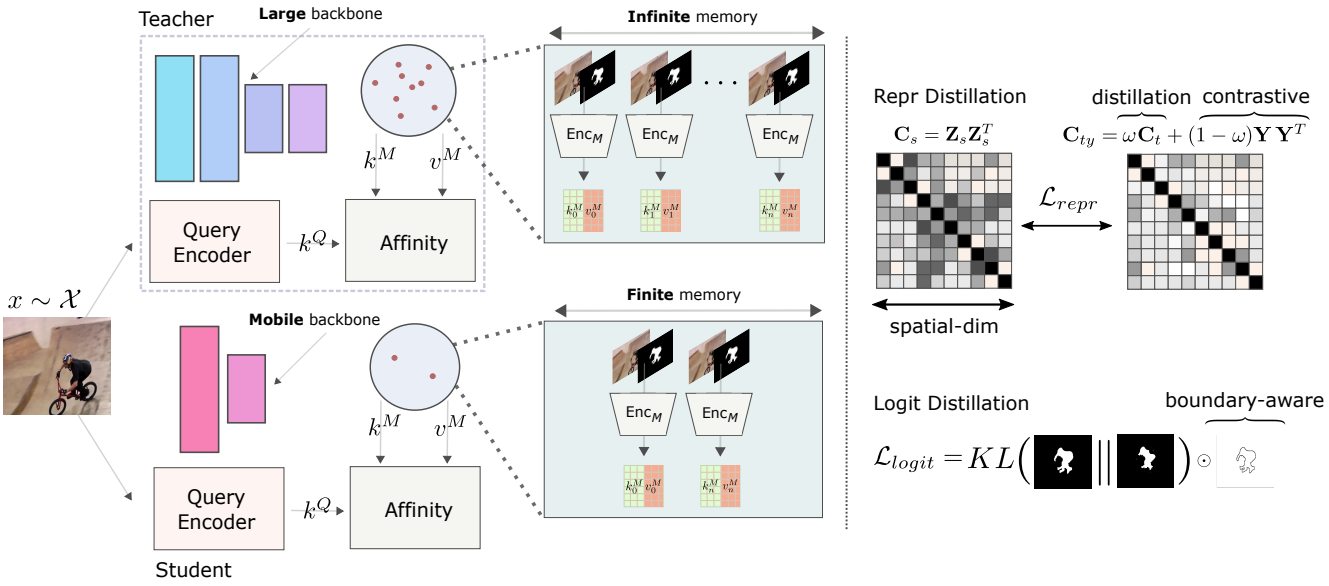

Figure 1. The proposed student-teacher framework using boundary-aware distillation. The teacher model utilises infinite memory, whereas the student model only retains memory from the previous and first frames/mask. Since the teacher has more memory, the task of distillation (equation 4 and 9) is then to learn features which are consistent across multiple frames. The auxiliary objective of contrastive representation learning encourages discriminative pixel-wise features (equation 7) around the boundary of objects.

- Using this unified loss, we show that a common network design can achieve results competitive to the state-of-the-art, while running up to $\times 5$ faster and having $\times 32$ fewer parameters.
- Without complex architectural or memory design changes, our proposed loss can unlock real-time performance (30FPS+) on mobile devices (i.e. Samsung Galaxy S22) while retaining competitive performance with state-of-the-art.

2. Related Work

Semi-Supervised Video Object Segmentation Some SVOS methods perform offline pretraining and then online per-mask finetuning to learn object-specific representations [4, 20, 27, 36, 43, 53]. Such methods are naturally unfit for deployment due to the finetuning, which is done for each object mask given in the first frame. Addressing the online adaptation requirement of such methods are flow-based approaches [12, 35, 40, 50, 61, 63], which model SVOS as a temporal mask propagation problem. These methods, however, lack long-term context and are known to suffer from errors accumulated due to occlusion and drift. Detection-based methods address [22, 27, 35, 51] error accumulation via detectors, however, their performance is bound by the detector's performance. Matching-based methods [8, 21, 52, 64, 65] match the features of the current and previous frame(s), and currently dominate the field due to the excellent results of memory-networks [41].

Following STM [41], many variants of memory-network based methods have emerged [9, 10, 26, 29, 32, 34, 41, 47, 54, 57]. STCN reformulated memory-networks and decoupled the masks and images in memory to increase efficiency [10]. Per-Clip VOS [42] proposed to process whole clips instead of individual frames. GSFM [32] proposed to leverage frequency-spectra to better model intra-frame spatial dependencies. XMem [9] proposed a three layered memory bank scheme to perform equally well on long videos. QDMN [33] proposed a quality assessment module to select informative frames to store in the memory. Several methods proposed to leverage spatial redundancy to perform localized segmentation/matching [37, 57]. RDE-VOS [26] maintains a single template in memory whereas SwiftNet [54] performs pixel-wise updates to ensure a constant memory cost. Despite the advances in efficiency, existing methods are far from feasible for mobile deployment, either due to subpar accuracy or excessive resource consumption.

Knowledge Distillation (KD) uses the predictions of a larger model (teacher) to provide additional supervision to a much smaller model (student) during training. KD is a common technique for model compression, making it useful for deployment scenarios. Its usage was first proposed in the context of image classification [19] and has since been extended to depth estimation [31, 67], segmentation [6], object detection [48] and image translation [69] tasks. In addition to simple logit distillation [19], some of the most successful frameworks look at the representations before the final fully connected layer since they can preserve some of

the structural information about the input. CRD [49] extended the supervised contrastive losses to incorporate the teachers' representations. This was later extended by both WCoRD [5] and ITRD [39] using the Wasserstein distance and an information theoretic framework, respectively. Although there is a rich literature on distillation for dense prediction tasks [6, 31, 48, 69], our work, to the best of our knowledge, is the first to leverage KD for SVOS.

Self-Supervision (SS) aims to learn a stable representation from unlabelled data by leveraging known invariances through the use of data augmentations. They can be grouped into contrastive [7] and de-correlation based [2, 68] methods. However, recent work has demonstrated the duality between these two paradigms under mild assumptions [16]. On a similar note, knowledge distillation shares very close similarities to self-supervision [58] and its usage in SVOS has motivated this work. Although several works have focused on self-supervision for video object segmentation [62, 70, 71], unsupervised distillation for segmentation [17], or general self-distillation [15, 38], there is no work exclusively focusing on self-supervised contrastive learning for SVOS. In addressing this gap we also provide a very general framework that is unified with knowledge distillation for improving the efficacy of training small mobile models. Theoretically, we also provide a different perspective of our loss and its direct equivalence to a generalised notion of mutual information.

Our work is the first to achieve real-time performance on mobile devices and the first to introduce distillation in the context of SVOS. RDE-VOS [26] introduces a guidance loss, which can be seen as a form of memory distillation, but they also impose significant architectural changes. On the other hand, MobileVOS performs competitively while running up to $\times 5$ faster with $\times 32$ fewer parameters.

3. Method

In this work we begin to bridge the performance gap between small finite memory space-time networks and their large infinite memory counterparts. This work is the first to open the possibility for real-time high-quality video object segmentation on resource-constrained devices, such as mobile phones. We propose to approach this problem through a knowledge distillation perspective and in doing so we develop a novel unification with contrastive learning. In the following sections, we introduce the distillation loss and its connection with information theory. Subsequent sections then show a series of natural extensions to this loss through boundary-aware sampling and a joint contrastive objective. We note that our framework is also very general: it can be applied to any other SVOS method and is likely applicable to other dense prediction tasks.

3.1. Representation Distillation

In light of recent works in the field of knowledge distillation [5, 39, 49], we propose to use the representation right before the final fully-connected layer. Using earlier layers may hurt the student's performance [49] through differing inductive biases, whereas the collapsed output space will have lost a lot of the structural information that can benefit the knowledge transfer.

More concretely, we propose a pixel-wise representation distillation loss that can transfer the structural information between these two models. The same augmented frames are fed into both networks and a distillation loss is applied at the representation level (see Figure 1). To extract structural information, we propose to construct correlation matrices, $\mathbf{C}_s, \mathbf{C}_t \in \mathbb{R}^{HW \times HW}$, from the two sets of representations, where HW is the spatial size. These matrices will then capture the relationship between all pairs of pixels.

$$\mathbf{C}_s = \mathbf{Z}_S \mathbf{Z}_S^T, \quad \mathbf{C}_t = \mathbf{Z}_T \mathbf{Z}_T^T \tag{1}$$

where $\mathbf{Z}_S \in \mathbb{R}^{HW \times d_s}$ and $\mathbf{Z}_T \in \mathbb{R}^{HW \times d_t}$ denote the d-dimensional $L2$ normalised student and teacher representations provided before the final point-wise convolution and upsampling layers. Motivated by recent improvement in the choice of distance metrics for distillation tasks [39], we derive the final representation loss as follows:

$$\mathcal{L}_{repr} = \frac{1}{|\ \mathbf{C}_s\ |} \left(\log_2 \|\mathbf{C}_s\|^2 - \log_2 \|\mathbf{C}_s \odot \mathbf{C}_t\|^2 \right) \tag{2}$$

where $\odot$ is the Hadamard product and $\|\cdot\|$ is the Frobenius norm. The two components of this loss can be interpreted as a regularisation term, and a correlation alignment between the two models, while the $\log_2$ operator is used here to improve the robustness to spurious correlations. An interesting property of this loss formulation is that it is equivalent to maximising the pixel-wise mutual information between the student and teacher representations (see Supplementary for derivation).

$$\mathcal{L}_{Distill} = \mathbf{H}_2(\mathbf{Z}_S) - \mathbf{H}_2(\mathbf{Z}_S; \mathbf{Z}_T) \tag{3}$$
$$= -\mathbf{I}_2(\mathbf{Z}_S; \mathbf{Z}_T) \tag{4}$$

where $\mathbf{H}_2$ and $\mathbf{I}_2$ are matrix-based estimators [46] resembling Rényi's entropy and mutual information of order 2 respectively. From this perspective, the two terms can instead be interpreted as maximising the joint entropy subject to an entropy regularisation.

3.2. Unification with contrastive learning

The effectiveness of knowledge distillation alone can be dependant on the relative capacity gap between the student

and teacher models. More formally, this training regime scales poorly as the capacity gap diminishes. To address this constraint, we propose a vast generalisation of equation 2 to encompass pixel-wise contrastive learning. Other works [58] have already shown that using self-supervision as an auxiliary task can improve conventional distillation, but there has yet to be a single unification of the two.

Since we have the dense ground truth classes, we can construct an additional target correlation matrix $(\mathbf{C}_y)_{ij} \in \{0, 1\}$ as follows:

$$\mathbf{C}_y = \mathbf{Y}\mathbf{Y}^T \tag{5}$$

where $\mathbf{Y} \in \mathbb{R}^{HW \times 2}$ are the spatially-downsampled, one-hot-encoded labels for the 2 classes (object and background). We can then couple the two target correlation matrices to provide a way of interpolating between these two training regimes.

$$\mathbf{C}_{ty} = \omega \mathbf{C}_t + (1 - \omega)\mathbf{Y}\mathbf{Y}^T \tag{6}$$

where $\omega \in [0, 1]$ is a hyperparameter and $\mathbf{C}_{ty}$ can now be substituted into equation 2. By considering a representation $\mathbf{Z}$ and the case where $\omega = 0$, the loss will now reduce to a familiar supervised contrastive learning (SCL) setting [23].

$$\mathcal{L}_{SupCon} = -\frac{1}{\mid \mathbf{C}_s \mid} \log_2 \sum_i \frac{\sum_{j \in \mathcal{P}_i} sim(\mathbf{Z}_i, \mathbf{Z}_j)}{\sum_k sim(\mathbf{Z}_i, \mathbf{Z}_k)} \tag{7}$$

where *sim* is the cosine similarity between two individual pixels in a representation and $\mathcal{P}_i$ is the set of positive indices for i-th pixel (see supplementary). Intuitively, for a given pixel, the numerator attracts the positives, while the denominator repels the negatives. The connection between these two regimes is illustrated in the following diagram:

$$\mathcal{L}_{SupCon} \xleftarrow{\omega=0} \mathcal{L}_{repr} \xrightarrow{\omega=1} \mathcal{L}_{Distill} \tag{8}$$

where the choice of ω is motivated by the availability and relative performance of a pre-trained teacher model.

3.3. Boundary-aware sampling

Most prediction errors occur on the boundary of objects (see Figure 5). Additionally, constructing the correlation matrix for all pixels is far too computationally expensive. Motivated by both of these two points, we propose to only sample pixels around and near the boundary of the objects. This sampling strategy restricts the distillation gradients to only flow through pixels that lead to downstream prediction errors, while also enabling a much more computationally efficient formulation. Since each frame also has a different boundary, the normalisation term (Equation 2) will now average over the size of these object boundaries, thus allowing the loss to naturally uniformly weight both small and large objects evenly. We additionally observe that this modification can improve the overall model convergence, as shown in Figure 5.

Algorithm 1 PyTorch-style pseudocode for MobileVOS

```
# f_s, f_t: Student and teacher network
# y: Ground-truth one-hot encoded labels
# y_s, y_t: Student and teacher logits
# z_s, z_t: Student and teacher representations
for x, y in loader:
  # Forward pass
  z_s, y_s = f_s(x)
  z_t, y_t = f_t(x)
  z_s = embed_s(z_s)

  # Cross entropy loss
  loss = bootstrap_poly_cross_entropy(y_s, y)

  # Normalise representations
  z_s_norm = F.normalize(z_s, dim=1)
  z_t_norm = F.normalize(z_t, dim=1)

  # Compute correlation-matrices
  c_ss = matmul(z_s_norm, z_s_norm.T)
  c_tt = matmul(z_t_norm, z_t_norm.T)

  # Interpolate between KD and SCL
  y_d = downsample(y)
  yy = matmul(y_d, y_d.T)

  r = ω * c_tt + (1 - ω) * yy
  loss += log2(c_ss.pow(2).sum()) / len(z_s)
  loss -= log2((c_ss * r).pow(2).sum()) / len(z_s)

  # Logit distillation
  prob_s = softmax(y_s / τ, dim=1)
  prob_t = softmax(y_t / τ, dim=1)
  loss += kl(prob_t, prob_s)

  # Optimisation step
  loss.backward()
  optimizer.step()
```

3.4. Logit distillation

An additional KL divergence term is introduced at the logit space between the two models, which is common in the distillation literature [49].

$$\mathcal{L}_{logit} = \frac{1}{HW} \sum KL\left(\mathbf{p}_S(\tau) \parallel \mathbf{p}_T(\tau)\right) \tag{9}$$

where $\mathbf{p}_S, \mathbf{p}_T$ are the student and teacher probabilities parameterised by a temperature term τ for softening ($\tau > 1$) or sharpening ($\tau < 1$) the two predictions. The summation indices have been omitted for brevity. The only distinction from conventional logit distillation is that we only use the pixels around or near the boundary of the objects. The final loss is given as follows:

$$\mathcal{L} = \mathcal{L}_{cross-entropy} + \mathcal{L}_{logit} + \mathcal{L}_{repr} \tag{10}$$

The complete training pipeline can be seen in Figure 1 and the PyTorch-style pseudo-code is given in Algorithm

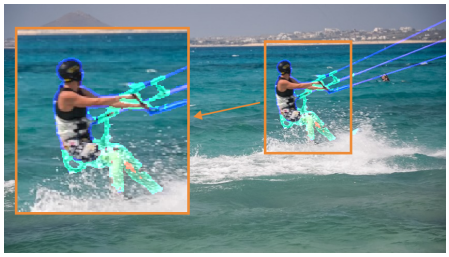

Figure 2. Prediction errors, shown in cyan, can typically occur on the boundaries of the segmented object, thus motivating the emphasis on distilling and contrasting boundary pixels.

1. For brevity, we omit the boundary-aware sampling in pseudo-code, but this can be straightforwardly implemented using a Sobel edge detector on the ground truth masks.

4. Experiments

We evaluate MobileVOS on two standard benchmarks, namely DAVIS [44, 45] and YouTube [60]. For all experiments, we train the student with an additional linear embedding layer and with a pre-trained STCN [10] as the teacher. Since most SVOS methods in the literature report the FPS metrics on different hardware, we further provide an additional fair comparison of our distilled models on the same sets of hardware. In these experiments, we consider both a server-grade GPU and a desktop-grade GPU, whereby our models are consistently at least twice as fast as competing methods for both long and short videos while being up to $\times 32$ smaller. The smallest distilled model is then deployed on a mobile phone for real-time segmentation, which opens up the possibility for many new on-device applications. Finally, we propose the use of model soups [56], for which are able to match the best performing RDE-VOS model that is specifically trained for YouTube.

4.1. Datasets and Metrics

DAVIS The DAVIS datasets are high-quality and high-resolution, densely-annotated videos for video object segmentation. DAVIS 2016 [44] provides single-object segmentation, with a validation set of 20 videos, while DAVIS 2017 [45] considers multi-object segmentation with both a validation and test set of 30 videos.

YouTube-VOS YouTube-VOS 2019 [60] is a large-scale benchmark dataset for multi-object segmentation. It consists of 3,471 videos for training and 507 videos for validation, across 65 categories. There are also 26 additional unseen categories in the validation set.

4.2. Implementation Details

To demonstrate the simplicity of our proposed representation loss, we provide its pseudo-code in algorithm 1. All of our experiments were trained using 4 NVIDIA A10 GPUs with a batch size of 8. For evaluating the FPS metrics on the DAVIS datasets, we used a single NVIDIA A10 GPU with a batch-size of 1. Adam [24] was used as the optimizer, with a learning rate of $1e-5$ and a weight decay of $1e-7$. The training details and architectures are primarily unchanged from that in the original STCN [10], but are discussed in detail in the following sections.

Architecture modifications Motivated by the goal to deploy these models on mobile devices, we adopt a few architectural modifications to the original STCN architecture. The value encoder is replaced with a MobileNetV2 [14] backbone, while the query encoder is either replaced with ResNet18 [18] or another MobileNetV2. The frame is also removed as an input to the value encoder and, unless otherwise specified, an ASPP module - commonly employed in the SVOS literature [26, 54] - is used before the decoder to introduce more global context. Finally, we select a fixed queue length of 2, since it enabled real-time performance on a mobile device, while maintaining the top-end accuracy (see figure 4).

Training details We follow the same training methodology proposed in STCN [10] except with two distinct modifications. Firstly, we extend the sequence length of videos to 5 to enable the student to learn features that are consistent across multiple frame predictions and secondly we use the poly cross-entropy loss [25] with $\epsilon = 1$. The poly loss can be seen as a natural generalisation of the cross-entropy and focal loss [30], which encourages the model to focus on the hard misclassified pixels.

Training stages All models are first trained on static images with synthetic deformations. The main training stage uses both the YouTube-VOS and DAVIS 2017 datasets for 600k iterations and with a batch size of 8. BatchNorm layers are also frozen throughout all training stages.

4.3. Comparisons to State-of-the-art

Similar to RDE-VOS [26], we use **Constant Cost (CC)** to denote methods with finite memory during inference. We consider two different key encoder backbones, namely a ResNet18 and a MobileNetV2. For the MobileNetV2 model, we also show results with and without ASPP. Since the ResNet architectures achieve close to the original STCN performance, we choose to use a purely contrastive loss i.e. $\omega = 0.0$. This helps avoid overfitting to the teacher's predictions on known classes and improve its generalisation to the unknown classes on YouTube. In contrast, for the much smaller MobileNet architectures, we use $\omega = 0.95$. These much smaller models are unlikely to overfit, and thus can benefit more from the distillation objective. It is worth noting that, in all cases, the models are also jointly trained with logit distillation, where $\tau = 0.1$.

DAVIS We compare MobileVOS against previous state-of-the-art methods on the DAVIS 2016 and DAVIS 2017 validation splits. The results for DAVIS 2016 are shown in Table 1, in which our best performing model is just 0.3 $\mathcal{J}\&\mathcal{F}$ shy of STCN and 0.2 $\mathcal{J}\&\mathcal{F}$ shy of RDE-VOS, while running $4\times$ and $3\times$ faster than these models, respectively.

Table 2 shows our results on DAVIS 2017. We are highly competitive, where we are only slightly worse (0.3 $\mathcal{J}\&\mathcal{F}$) than our STCN teacher model. Note that we are nearly $5\times$ and $4\times$ faster than STCN and RDE-VOS, which shows our runtime performance is still competitive despite tracking multiple objects.

Method	CC	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	FPS
RMNet† [57]	✗	88.8	88.9	88.7	11.9
STM† [41]	✗	89.3	88.7	89.9	6.3
MiVOS†* [11]	✗	91.0	89.7	92.4	16.9
STCN†* [10]	✗	91.7	90.4	93.0	26.9
BATMAN [66]	✗	**92.5**	**90.7**	**94.2**	-
GSFM [32]	✗	91.4	90.1	92.7	≈ 8.9
XMem† [9]	✗	91.5	90.4	92.7	**29.6**
XMem†* [9]	✗	92.0	**90.7**	93.2	**29.6**
GCNet [28]	✓	86.6	87.6	85.7	25.0
CFBI† [64]	✓	89.9	88.7	91.1	5.9
SwiftNet† [54]	✓	90.4	**90.5**	90.3	25.0
RDE-VOS† [26]	✓	91.1	89.7	92.5	35.0
RDE-VOS†* [26]	✓	**91.6**	90.0	**93.2**	35.0
MobileVOS					
ResNet18†	✓	90.6	89.7	91.6	**100.1**
ResNet18†*	✓	91.4	90.3	92.6	**100.1**
↳ model soup†	✓	91.3	90.2	92.5	**100.1**
MobileNetV2†	✓	90.5	89.5	91.5	81.8
↳ wo/ ASPP†	✓	90.1	89.0	91.1	86.0

Table 1. Results on the DAVIS 2016 validation set. CC denotes constant cost during the inference. † indicates YouTube-VOS is added during the training stage. ∗ denotes BL30K is added during the training stage. For both CC and non-CC methods, the best results are highlighted in **bold**, while the second best results are underlined. FPS was averaged over 3 runs.

YouTube-VOS Table 3 shows a comparison of our method against other state-of-the-art methods on the large-scale YouTube-VOS 2019 validation set. Our method outperforms RDE-VOS in the case where no BL30K pre-training is used and is competitive in the case where it is used. Unlike RDE-VOS, which trains with different loss weights for the YouTube evaluation to avoid overfitting on unseen classes, we report predictions using the same set of weights as in the DAVIS evaluation. We also observe only a 1.9 $\mathcal{J}\&\mathcal{F}$ drop with respect to the STCN teacher model.

Method	CC	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	FPS
STM†z	✗	81.8	79.2	84.3	10.2
RMNet† [57]	✗	83.5	81.0	86.0	<11.9
MiVOS†* [11]	✗	84.5	81.7	87.4	11.2
STCN†* [10]	✗	85.3	82.0	88.6	20.2
GSFM [32]	✗	86.2	83.1	89.3	≈ 8.9
BATMAN [66]	✗	86.2	83.2	89.4	-
XMem† [9]	✗	86.2	82.9	89.5	**22.6**
XMem†* [9]	✗	**87.7**	**84.0**	**91.4**	**22.6**
GCNet [28]	✓	71.4	69.3	73.5	<25.0
PReMVOS [35]	✓	77.8	73.9	81.7	0.01
SwiftNet† [54]	✓	81.1	78.3	83.9	<25.0
SST [13]	✓	82.5	79.9	85.1	-
RDE-VOS† [26]	✓	84.2	80.8	87.5	27.0
RDE-VOS†* [26]	✓	**86.1**	82.1	**90.0**	27.0
MobileVOS					
ResNet18†	✓	83.7	80.2	87.1	**90.6**
ResNet18†*	✓	85.0	81.7	88.3	**90.6**
↳ model soup†	✓	85.6	**82.3**	88.9	**90.6**
MobileNetV2†	✓	82.2	78.7	85.7	79.1
↳ wo/ ASPP†	✓	81.8	78.3	85.3	81.3

Table 2. Results on the DAVIS 2017 validation set. CC denotes constant cost during the inference.FPS was averaged over 3 runs.

Method	CC	Overall	$\mathcal{J}_{seen}$	$\mathcal{F}_{seen}$	$\mathcal{J}_{unseen}$	$\mathcal{F}_{unseen}$
STM† [41]	✗	79.2	79.6	83.6	73.0	80.6
MiVOS†* [11]	✗	82.4	80.6	84.7	78.2	85.9
STCN†* [10]	✗	84.2	82.6	**87.0**	79.4	**87.7**
BATMAN† [66]	✓	84.5	**89.3**	79.0	87.2	85.0
XMem†∗ [9]	✓	**84.8**	89.2	80.3	**88.8**	85.8
CFBI† [64]	✓	81.0	80.6	85.1	75.2	83.0
SST† [64]	✓	81.8	80.9	-	76.6	-
SwiftNet† [54]	✓	77.8	77.8	81.8	72.3	79.5
RDE-VOS† [26]	✓	81.9	81.1	85.5	76.2	84.8
RDE-VOS†* [26]	✓	**83.3**	81.9	86.3	**78.0**	**86.9**
MobileVOS						
ResNet18†	✓	82.3	81.6	86.0	76.3	85.2
ResNet18†*	✓	82.8	82.1	86.4	77.0	85.6
↳ model soup†	✓	**83.3**	**83.2**	**87.7**	76.9	85.3
MobileNetV2†	✓	80.3	80.4	84.6	74.0	82.4
↳ wo/ ASPP†	✓	80.1	79.0	83.2	75.1	83.3

Table 3. Results on the YouTube-VOS 2019 validation set.

Qualitative results Figure 3 shows the segmentation of two identical models trained with and without knowledge distillation. In this example, we observe that the distilled model is able to successfully segment the panda despite undergoing drastically different views and occlusions.

Model size and inference latency To conduct a fair comparison of the computational cost of previous works, we jointly evaluate MobileVOS and other methods on the same set of hardware. We consider both a server-grade NVIDIA

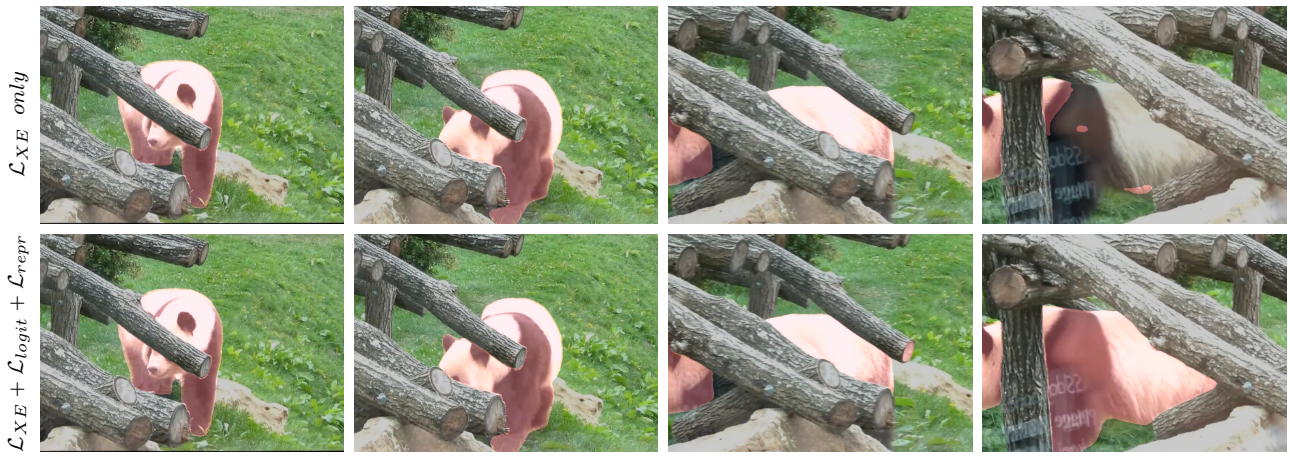

Figure 3. Comparing the segmentation of two models with and without constrastive learning or distillation being applied. Despite both models having finite memory, the cross-entropy only model fails to learn features which are consistent across all frames of the video. This is demonstrated in the last frame above, where the side of the panda is poorly segmented.

A40 and a desktop-grade NVIDIA 1080Ti. In all cases, we use the authors' provided code, but with a modification to accept randomly generated video sequences of two different durations. The results can be seen in Table 4 and show that our MobileVOS models are significantly smaller, while retaining a constant low latency across both the short and long-duration videos. This result is attributed to the smaller backbones and the constant memory costs due to a finite memory queue length. Our largest model with ResNet18 has more than $3\times$ the FPS than RDE-VOS but with $8\times$ fewer parameters. Finally, we observe that the ResNet models perform better on the server-grade GPUs. This is in line with other results in the literature [3] and is related to the trade-off between model depth and GPU compute units available. In the following section, we show that we obtain a different outcome for mobile GPUs.

Method	Params(M)	FPS *NVIDIA A40*		FPS *NVIDIA 1080Ti*	
		short	long (10×)	short	long (10×)
STM [41]	38.9	8.9	4.3	6.8	✗
GSFM [57]	67.0	18.4	4.2	7.6	✗
STCN [10]	54.4	37.4	8.3	18.1	✗
RDE-VOS [57]	64.0	32.0	34.2	14.4	14.1
XMem [9]	62.2	38.6	39.9	12.6	12.7
MobileVOS					
ResNet18	8.1	**144.7**	**145.4**	**76.0**	**76.3**
MobileNetV2	2.5	99.9	99.1	61.6	60.6
↳ wo/ ASPP	**1.9**	105.1	103.4	66.8	67.4

Table 4. Performance evaluation on the same sets of hardware. The FPS metrics were evaluated on two randomly generated short and long video sequences with shape 480×910. The short videos consist of 50 frames, while the long videos consist of 500. In all cases, we use the authors' provided code and ✗ indicates that the models are exceeding the GPU memory limit.

4.4. Mobile phone deployment

For the purpose of mobile deployment, we evaluate the inference latency of our models on the GPU of a Samsung Galaxy S22 device. The models are first converted to tflite format and then benchmarked using the official tflite benchmarking tool [1]. To validate our choice of fixed memory size with 2 entries, we show the model latency against queue sizes between 1 and 20 (see figure 4). MobileNetV2 wo/ ASPP model is the only model that is able to achieve real-time performance up to a memory queue length of 2. Figure 4 also shows a comparison of the $\mathcal{J}\&\mathcal{F}$ performance on the DAVIS 2016 validation set at different memory queue lengths, where there is a minimal accuracy degradation by reducing this length to 2. These two points motivate the choice of a memory queue length of 2.

Model soups We consider the use of model soups [56] as a scheme for utilising the expressiveness of an ensemble without any additional inference costs. This is in contrast to multi-scale inference, which is typically adopted in the literature [10] and requires 4 forward passes. To do this, we firstly construct a set of models using the checkpoints from previous iterations trained with different values of ω. Afterwards, we run a greedy selection using the training data as the validation metric. Tables 2, 1, and 3 show these results, where we find that the model soup can surpass the performance of all the individual ingredients by at least 0.6 $\mathcal{J}\&\mathcal{F}$ on DAVIS 2017 and 0.5 $\mathcal{J}\&\mathcal{F}$ on YouTube, thus confirming the benefit of adopting this scheme in practice.

5. Ablation Study

We now perform a thorough ablation study to highlight the benefit of our proposed distillation loss, and its extensions with boundary sampling and contrastive learning.

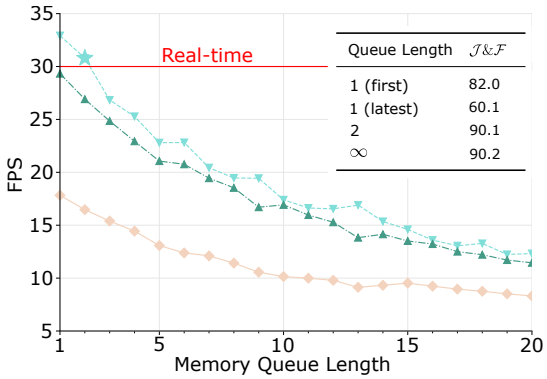

Figure 4. Runtimes of our proposed models with different memory queue lengths were evaluated on a Samsung Galaxy S22 GPU. The line with ▾ markers is the MobileNetV2 wo/ ASPP, ▲ markers is the MobileNetV2, while ♦ markers are for the Resnet18. The table shows $\mathcal{J}\&\mathcal{F}$ on the DAVIS 2016 validation set for MobileNetV2 wo/ ASPP on memory queue lengths of 1, 2 and unbounded. We highlight the candidate that achieves the highest $\mathcal{J}\&\mathcal{F}$, while maintaining real-time performance with the star sign ($\star$). The latency results are averaged across 100 runs.

Loss terms. We train our models with the two distillation losses $\mathcal{L}_{logit}$ and $\mathcal{L}_{repr}$ being incrementally added to the primary loss term (i.e. cross-entropy using ground-truth masks). The results presented in Table 5 show that both $\mathcal{L}_{logit}$ and $\mathcal{L}_{repr}$ introduce improvements.

Losses	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
$\mathcal{L}_{XE}$	80.75	77.68	83.82
$\mathcal{L}_{XE} + \mathcal{L}_{logit}$	81.38	78.13	84.64
$\mathcal{L}_{XE} + \mathcal{L}_{logit} + \mathcal{L}_{repr}$	**82.28**	**79.96**	**85.61**

Table 5. Evaluating the performance improvement on the YouTubeVOS dataset after introducing the two distillation losses. $\mathcal{J}$ and $\mathcal{F}$ metrics are averaged over both the seen and unseen classes.

Interpolating between SCL and KD. We sweep over various ω values in equation 6 to interpolate between the supervised contrastive and distillation training regimes[1]. Table 6 shows the results using a ResNet18-backbone. The results show that the best results are achieved on DAVIS 2016 and 2017 with $\omega = 1$, where the loss is reduced to the distillation loss. We see the opposite trend on YouTube; the best results are achieved with $\omega = 0$, which reduces the loss to a supervised contrastive setting. Our hypothesis is that in the $\omega = 0$ case, it is difficult to distill the unseen class knowledge from the teacher, whereas in $\omega = 1$ scenario, the contrastive loss is robust to unseen classes due to a lowered dependence on object semantics.

[1]All models are also trained with logit distillation.

ω	DAVIS16	DAVIS17	YoutubeVOS
0.0	**91.0**	83.1	**82.3**
0.2	90.6	83.7	81.3
0.8	90.2	83.4	81.5
1.0	90.9	**84.2**	81.1

Table 6. Interpolating between supervised-contrastive loss, $\omega = 0.0$, and (representation) distillation loss, $\omega = 1.0$. The performance improvement is robust across a range of values for ω, which suggests the models jointly benefit from both training regimes. Different ω values offer flexibility to achieve better results.

Boundary sampling. To demonstrate the performance improvement attributed to sampling boundary pixels, we perform an ablation experiment with a random sampling strategy. More specifically, we use logit distillation on all the pixels, and representation distillation on randomly selected pixels. Note that selecting all the pixels for representation distillation is prohibitively expensive due to the construction of the correlation matrices. The validation curves on DAVIS'16 for both strategies can be seen in figure 5. The curves show that sampling the boundary pixels yields an improved convergence rate. Furthermore, this faster convergence leads to better $\mathcal{J}\&\mathcal{F}$, as highlighted in the figure.

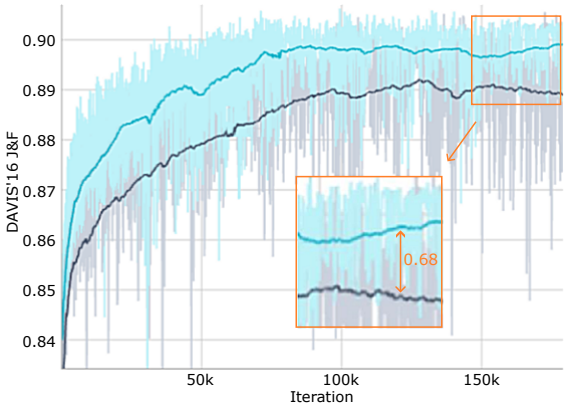

Figure 5. Validation accuracy for two models trained using random pixel and boundary pixel sampling strategies. Random sampling is shown in black and shows much slower convergence than boundary sampling, which is given in blue.

6. Conclusion

In this work, we present a simple loss that unifies knowledge distillation and contrastive learning. By applying this loss in the context of semi-supervised video object segmentation, we achieve competitive results with state-of-the-art at a fraction of the computational cost and model size. These models are compact enough to fit on a standard mobile device, while retaining real-time latency.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 7

[2] Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. 2021. 3

[3] Simone Bianco, Remi Cadene, Luigi Celona, and Paolo Napoletano. Benchmark analysis of representative deep neural network architectures. *IEEE Access*, 2018. 7

[4] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *CVPR*, 2017. 1, 2

[5] Liqun Chen, Dong Wang, Zhe Gan, Jingjing Liu, Ricardo Henao, and Lawrence Carin. Wasserstein Contrastive Representation Distillation. *CVPR*, 2020. 3

[6] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling Knowledge via Knowledge Review. *CVPR*, 2021. 2, 3

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *ICML*, 2020. 3

[8] Yuhua Chen, Jordi Pont-Tuset, Alberto Montes, and Luc Van Gool. Blazingly fast video object segmentation with pixel-wise metric learning. In *CVPR*, 2018. 1, 2

[9] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. *ECCV*, 2022. 1, 2, 6, 7

[10] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. *NeurIPS*. 1, 2, 5, 6, 7

[11] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Modular interactive video object segmentation: Interaction-to-mask, propagation and difference-aware fusion. In *CVPR*, 2021. 6

[12] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *ICCV*, 2017. 1, 2

[13] Brendan Duke, Abdalla Ahmed, Christian Wolf, Parham Aarabi, and Graham W Taylor. Sstvos: Sparse spatiotemporal transformers for video object segmentation. In *CVPR*. 6

[14] Michael H. Fox, Kyungmee Kim, and David Ehrenkrantz. MobileNetV2: Inverted Residuals and Linear Bottlenecks. *CVPR*, 2018. 5

[15] Tommaso Furlanello, Zachary C. Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born Again Neural Networks. *ICML*, 5 2018. 3

[16] Quentin Garrido, Yubei Chen, Adrien Bardes, Laurent Najman, and Yann Lecun. On the duality between contrastive and non-contrastive self-supervised learning, 2022. 3

[17] Mark Hamilton, Zhoutong Zhang, Bharath Hariharan, Noah Snavely, and William T. Freeman. Unsupervised Semantic Segmentation by Distilling Feature Correspondences. *ICLR*, 3 2022. 3

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. ResNet - Deep Residual Learning for Image Recognition. *CVPR*, 2015. 5

[19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. *NeurIPS*, 2015. 2

[20] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. Maskrnn: Instance level video object segmentation. In *NIPS*, 2017. 1, 2

[21] Yuan-Ting Hu, Jia-Bin Huang, and Alexander G Schwing. Videomatch: Matching based video object segmentation. In *ECCV*, 2018. 1, 2

[22] Xuhua Huang, Jiarui Xu, Yu-Wing Tai, and Chi-Keung Tang. Fast video object segmentation with temporal aggregation network and dynamic template matching. In *CVPR*, 2020. 2

[23] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *NeurIPS*, 2020. 4

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[25] Zhaoqi Leng, Mingxing Tan, Chenxi Liu, Ekin Dogus Cubuk, Xiaojie Shi, Shuyang Cheng, and Dragomir Anguelov. Polyloss: A polynomial expansion perspective of classification loss functions. *arXiv*, 2022. 5

[26] Mingxing Li, Li Hu, Zhiwei Xiong, Bang Zhang, Pan Pan, and Dong Liu. Recurrent dynamic embedding for video object segmentation. In *CVPR*. 1, 2, 3, 5, 6

[27] Xiaoxiao Li and Chen Change Loy. Video object segmentation with joint re-identification and attention-aware mask propagation. In *ECCV*, 2018. 1, 2

[28] Yu Li, Zhuoran Shen, and Ying Shan. Fast video object segmentation using the global context module. In *ECCV*, 2020. 6

[29] Yongqing Liang, Xin Li, Navid Jafari, and Jim Chen. Video object segmentation with adaptive feature bank and uncertain-region refinement. *NeurIPS*. 1, 2

[30] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. 5

[31] Yifan Liu, Changyong Shu, Jingdong Wang, and Chunhua Shen. Structured knowledge distillation for dense prediction. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2, 3

[32] Yong Liu, Ran Yu, Jiahao Wang, Xinyuan Zhao, Yitong Wang, Yansong Tang, and Yujiu Yang. Global spectral filter memory network for video object segmentation. *ECCV*, 2022. 1, 2, 6

[33] Yong Liu, Ran Yu, Fei Yin, Xinyuan Zhao, Wei Zhao, Weihao Xia, and Yujiu Yang. Learning quality-aware dynamic memory for video object segmentation. In *ECCV*, 2022. 2

[34] Xiankai Lu, Wenguan Wang, Martin Danelljan, Tianfei Zhou, Jianbing Shen, and Luc Van Gool. Video object segmentation with episodic graph memory networks. In *ECCV*, 2020. 1, 2

[35] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. In *ACCV*, 2018. 1, 2, 6

[36] Yunyao Mao, Ning Wang, Wengang Zhou, and Houqiang Li. Joint inductive and transductive learning for video object segmentation. In *ICCV*. 1, 2

[37] Bo Miao, Mohammed Bennamoun, Yongsheng Gao, and Ajmal Mian. Region aware video object segmentation with deep motion modeling. *arXiv*, 2022. 1, 2

[38] Roy Miles and Krystian Mikolajczyk. Cascaded channel pruning using hierarchical self-distillation. *BMVC*, 2020. 3

[39] Roy Miles, Adrian Lopez Rodriguez, and Krystian Mikolajczyk. Information Theoretic Representation Distillation. *BMVC*, 2022. 3

[40] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *CVPR*, 2018. 1, 2

[41] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 1, 2, 6, 7

[42] Kwanyong Park, Sanghyun Woo, Seoung Wug Oh, In So Kweon, and Joon-Young Lee. Per-clip video object segmentation. In *CVPR*, 2022. 2

[43] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images. In *CVPR*, 2017. 1, 2

[44] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 5

[45] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbelaez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. 2017. 5

[46] Luis G. Sanchez Giraldo and Jose C. Principe. Information theoretic learning with infinitely divisible kernels. *ICLR*, 2013. 3

[47] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Kernelized memory network for video object segmentation. In *ECCV*, 2020. 1, 2

[48] Changyong Shu, Yifan Liu, Jianfei Gao, Zheng Yan, and Chunhua Shen. Channel-wise knowledge distillation for dense prediction. In *ICCV*, 2021. 2, 3

[49] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. *ICLR*, 2019. 3, 4

[50] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. Video segmentation via object flow. In *CVPR*, 2016. 1, 2

[51] Richard Vock, Alexander Dieckmann, Sebastian Ochmann, and Reinhard Klein. Fast template matching and pose estimation in 3D point clouds. *Computers & Graphics*, 2019. 2

[52] Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. Feelvos: Fast end-to-end embedding learning for video object segmentation. In *CVPR*, 2019. 1, 2

[53] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for the 2017 davis challenge on video object segmentation. In *The 2017 DAVIS Challenge on Video Object Segmentation-CVPR Workshops*, 2017. 1, 2

[54] Haochen Wang, Xiaolong Jiang, Haibing Ren, Yao Hu, and Song Bai. Swiftnet: Real-time video object segmentation. In *CVPR*, 2021. 1, 2, 5, 6

[55] Wenguan Wang, Tianfei Zhou, Fatih Porikli, David Crandall, and Luc Van Gool. A survey on deep learning technique for video segmentation. *arXiv*, 2021. 1

[56] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*. PMLR, 2022. 5, 7

[57] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Shengping Zhang, and Wenxiu Sun. Efficient regional memory network for video object segmentation. In *CVPR*, 2021. 1, 2, 6, 7

[58] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge Distillation Meets Self-supervision. *ECCV*, 2020. 3, 4

[59] Kai Xu and Angela Yao. Accelerating video object segmentation with compressed video. In *CVPR*, 2022. 1

[60] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark, 2018. 5

[61] Yu-Syuan Xu, Tsu-Jui Fu, Hsuan-Kung Yang, and Chun-Yi Lee. Dynamic video segmentation network. In *CVPR*, 2018. 1, 2

[62] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In *ICCV*, 2021. 3

[63] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. In *CVPR*, 2018. 1, 2

[64] Zongxin Yang, Yunchao Wei, and Yi Yang. Collaborative video object segmentation by foreground-background integration. In *ECCV*, 2020. 1, 2, 6

[65] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating objects with transformers for video object segmentation. *NeurIPS*, 2021. 1, 2

[66] Ye Yu, Jialin Yuan, Gaurav Mittal, Li Fuxin, and Mei Chen. Batman: Bilateral attention transformer in motion-appearance neighboring space for video object segmentation. *ECCV*, 2022. 6

[67] Mehmet Kerim Yucel, Valia Dimaridou, Anastasios Drosou, and Albert Saa-Garriga. Real-time monocular depth estimation with sparse supervision on mobile. In *CVPR 2021 Mobile AI (MAI) Workshop*, 2021. 2

[68] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. *ICML*, 2021. 3

[69] Linfeng Zhang, Xin Chen, Xiaobing Tu, Pengfei Wan, Ning Xu, and Kaisheng Ma. Wavelet knowledge distillation: Towards efficient image-to-image translation, 2022. 2, 3

[70] Fangrui Zhu, Li Zhang, Yanwei Fu, Guodong Guo, and Weidi Xie. Self-supervised video object segmentation. *arXiv*, 2020. 3

[71] Wenjun Zhu, Jun Meng, and Li Xu. Self-supervised video object segmentation using integration-augmented attention. *Neurocomputing*, 2021. 3