Label-efficient learning to see

Hu, T.

Publication date
2023
Document Version
Final published version

Label-Efficient Learning To See

Tao Hu

Label-Efficient Learning To See

Tao Hu

Label-Efficient Learning to See

Tao Hu

This book was typeset by the author using LaTeX 2_ε.
The cover was generated by Mid-Journey by the following prompts:

- Front: "buddha is sitting under the sunset without any anxiety from the outside world, ignoring the birds around him –ar 928:1364 –v 5.1"
- Back: "Nietzsche and Yang Guifei are sitting around a fire stove, laughing, the rusted pendulum as a background –ar 928:1364 –v 5"

Label-Efficient Learning to See

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Aula der Universiteit
op donderdag 13 juli 2023, te 14.00 uur

door Tao Hu

geboren te Hubei

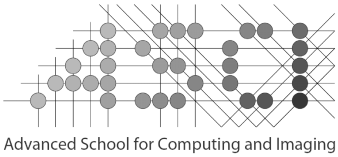
Promotiecommissie

Promotor:	prof. dr. C.G.M. Snoek	Universiteit van Amsterdam
Co-promotor:	dr. P.S.M. Mettes	Universiteit van Amsterdam
Overige leden:	prof. dr. T. Gevers	Universiteit van Amsterdam
	prof. dr. M. Worring	Universiteit van Amsterdam
	dr. E. Gavves	Universiteit van Amsterdam
	prof. dr. B.S. Ghanem	King Abdullah University for Science and Technolog
	dr. T.S. Salimans	Google

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

UNIVERSITEIT VAN AMSTERDAM

The work described in this thesis has been carried out within the graduate school ASCI, at the Video & Image Sense Lab.



CONTENTS

1

INTRODUCTION

Chinese philosopher Lao Tzu stated in his prominent book "Tao Te Ching" [141]: "If you have Tao while no Skill, Skill still can be obtained by Tao. While, if you have no Tao but only Skill, your life will be influenced due to the lack of Tao." If we understand "Tao" and "Skill" as the data and the annotation needed for a modern artificial intelligence system, it suggests prioritization of data over annotations, as the annotation itself can be acquired from the data. This philosophy, furthermore, poses a thought-provoking question to us: can we achieve machine learning with little reliance on annotations?

1.1 LEARNING TO SEE

The story of this thesis begins with the concept of human vision. Human vision is essential for humans to interact with an ever changing world. We continuously learn to visually recognize things from the world around us, such as through watching TV or reading books. Visual concepts are an indispensable part of our environment, and we have managed to learn them without too much supervision per concept. For example, a child can recognize a snake in real life, even when they have only seen the snake's contour in a comic book. These abilities can generally be applied well in life, beyond recognizing various species of snakes. This raises a fundamental question: Can machines that see also learn to generalize in the broad world without too much supervision?

Inspired by the remarkable abilities of human vision, researchers have been exploring ways to build machine vision that imitates human vision. These efforts aim to make machine vision more interpretable [15, 286], improve its accuracy [53, 65], and bridge the gap between machine vision and biological vision [134], among other objectives. Much of the progress in machine vision has been driven by computer science, most notably by the field of computer vision. The current state-of-the-art in computer vision emphasizes on formulating visual recognition problems as deep learning algorithms [53]. Essentially, these neural network algorithms learn associations between images and labels during training, enabling them to predict labels for unseen images during testing. Labels in these images are typically assigned to either the entire image [53, 65], boxes around objects in these images [33, 198] or even individual object pixels [40, 155]. These deep learning techniques have significantly transformed various important application areas, such as security surveillance [223], medical imaging [225], and molecule design [113].

Despite the great success of computer vision, most existing solutions rely on annotated labels [53, 95, 150, 236]. However, label annotation poses several challenges, including the labor-intensive process of annotation [53], the inherent lack of labels in most data [7], and the difficulty of dealing with long-tail label distribution problems [269]. These issues

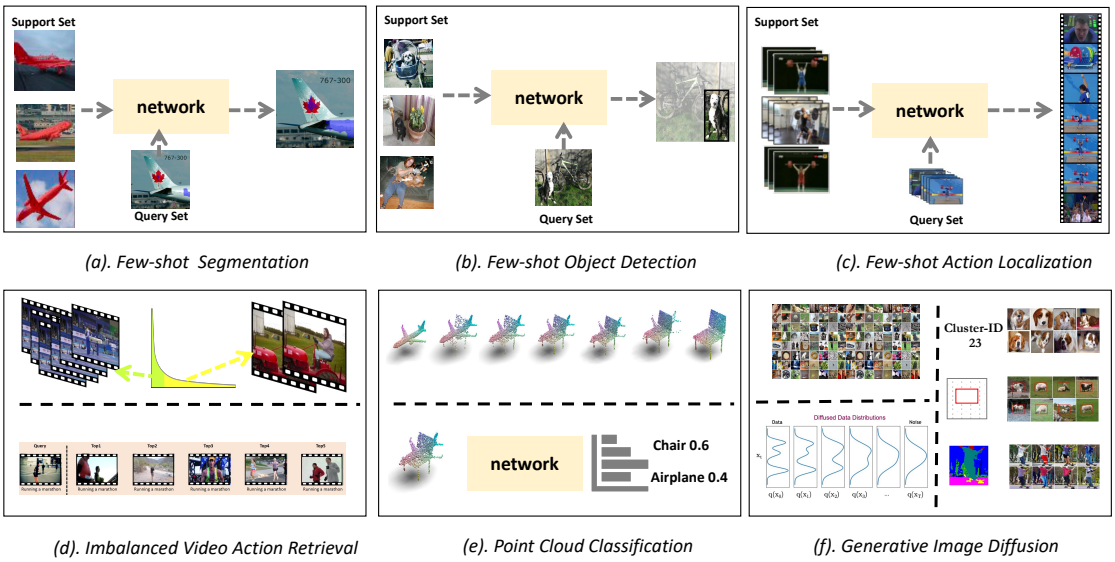


Figure 1: **In this thesis, we address label-efficient learning to see for six computer vision tasks:** (a) few-shot segmentation, (b) few-shot object detection, (c) few-shot video action localization, (d) imbalanced video action retrieval, (e) point cloud classification with mix-up, and (f) self-guided generative image diffusion. Some components of this figure are courtesy of the CVPR2022 diffusion tutorial by Kreis *et al.* and Wikipedia.

are further exacerbated when pixel-level annotation is required [65, 245]. To address these challenges, our approach aims to focus on label-efficiency. The goal is to reduce reliance on supervision signals from label annotation in computer vision and to improve the utilization of the supervision signal to imitate the learning process of a human being. These facts highlight the need to achieve label-efficient learning in the era of computer vision.

1.2 RESEARCH QUESTIONS

To achieve label-efficient learning in computer vision, we need deep learning algorithms that use a considerably reduced set of labels or even zero labels, at the expense of only a small drop in performance compared to using the full set of labels. How can we resolve this problem? How can we enable such label-efficient learning for various computer vision challenges? To answer these questions, we pose the central research question of this thesis:

How to achieve computer vision by label-efficient learning?

To showcase the diversity of our research tasks, we address the main research question across six typical computer vision challenges: image segmentation, object detection, video localization, video retrieval, point cloud classification, and generative image diffusion. A visual depiction of those tasks deployed with the label-efficient solutions proposed in this thesis is provided in Fig. 1.

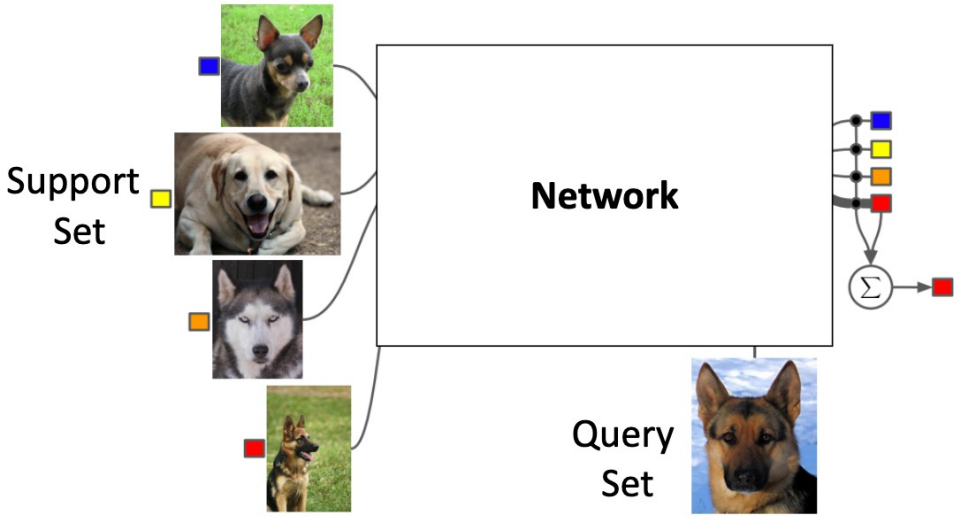


Figure 2: **Few-shot learning.** The support images consist of four classes, each with one instance. For any image in the query set, the network categorizes it into one of the classes from the support set. Figure adapted from [239].

1.2.1 *Image Segmentation*

Image segmentation is a pixel-level prediction task on an image and has been a classical problem in computer vision, see [103, 170] for recent surveys covering the topic more exhaustively. Modern solutions rely on deep neural networks, which expect an image as input and outputs a same-size segmentation mask, which denotes the semantic representation of the input image. The state-of-the-art [40, 155, 170, 259] relies on thousands of image-mask pairs to learn an optimal representation for segmentation. Unfortunately, a dense pixel-level annotation over multiple images is quite labor-consuming, e.g., segmenting 2,500,000 object instances required over 22 worker hours per 1,000 segmentations [150]. The annotation cost grows further as image resolutions tend to become larger and larger or annotation requirement is denser, for example in medical imaging [180], remote sensing [161], or crowd counting [39]. At the same time, too few image-mask annotations could potentially backfire the learning process by overfitting [147]. This dilemma spurs us to pose the question:

How to achieve label-efficient learning for image segmentation?

To answer the question, we take inspiration from the few-shot learning paradigm for image classification [68, 215, 239]. As shown in Figure 2, the goal is to design and optimize a neural network that can generalize well to a dynamic classification hypersphere when conditioned on the guidance signals consisting of only a few image instances per class. The support images serve as guidance and are composed of several classes. The network is designed and pre-trained to be able to generalize and recognize any given query image into one of those classes from the support set. The training

stage of few-shot learning is usually conducted with a closed set of classes, meaning the number of classes is always bounded due to the restrictions of the training dataset. However, the paradigm of few-shot learning enables us to conduct inference on an open set of classes, as long as a few annotations of the open-set classes are supplied in the support set.

We propose a few-shot learning solution for image segmentation. Concretely, we have several binary image-mask pairs acting as support set, accompanied by an extra unlabeled image as query set. Ideally, we hope our network can generalize well to segment any concept based on the guidance signal from the support sets. This can bring us two advantages: 1). A marginal amount of annotation is needed per class during inference. 2). Different from traditional closed-vocabulary image segmentation solutions [40], the network readily generalizes to classes it has not seen during training. In Chapter 2, we present an attention-based multi-context guiding network for few-shot segmentation. It consists of three branches: a support branch, a query branch, and a feature fusion branch. A key feature of our network is the integration of multi-scale context features between support and query branches, enforcing better guidance from the support set. Besides achieving state-of-the-art results on challenging benchmarks, our method reaches label-efficient learning in semantic segmentation for unseen novel classes without annotating many pixels for those classes.

1.2.2 *Object Detection*

As our second computer vision challenge, we consider the task of object detection. Here the goal is to localize an object in an image by its enclosing bounding box and to assign an object class label from a predefined set of object classes. Modern approaches can be separated into two-stage [94, 189, 198] and single-stage [33, 153, 197] neural network architectures. Independent of the architecture design, all approaches depend on numerous triplets of images, bounding boxes and labels to achieve competitive performance [33,94]. This dependence on predefined images, boxes and labels lead to several weaknesses. Firstly, annotators need to annotate at least two points per bounding box for each object class, which is labor-intensive, e.g., it can take nearly 20 minutes for a single image in the well-known COCO dataset to be fully annotated [31, 58, 150]. Secondly, detection with a predefined closed-set of object labels yields the application unrealistic and hazardous as well, especially in the scenario of autonomous driving [21] and medical imaging [76], as we can not sidestep the appearance of the open-set (unseen) classes in those scenarios. We, therefore, pose the research question:

How to achieve label-efficient learning for object detection?

In Chapter 3, we propose a new label-efficient task named few-shot common-localization, that is again inspired by few-shot learning for images. Given a few weakly-supervised support images, we aim to localize the common object in the query image without the need for any box and label annotation. To tackle this new problem, we propose a network that aims to get the most out of the support and query images. To that end, we introduce a spatial similarity module that searches the spatial commonality among the given images. We furthermore introduce a feature reweighting module, which balances the influence of

different support images through graph convolutional networks. Experimental evaluation of the new settings demonstrates the label-efficient ability of our method to achieve object detection in an open-set setting.

So far, we only considered the label-efficient learning problem for spatial 2D images. However, computer vision is not bounded by 2D imagery only and one may think about the label-efficient learning problem for more complex vision data. In the next chapter, we focus on spatio-temporal image data and consider the problem of temporal activity recognition in video.

1.2.3 *Video Action Localization*

Video action localization aims to localize a target action temporally and/or spatially from a long untrimmed video. Like its object detection counterpart, the leading methods can be divided into two-stage [258, 262, 284] and single-stage [118, 265] architectures, and most of the works laboriously rely on the video-annotation pairs, and commonly, those annotations can be a start-end timestamp, spatial start-end tubes, and a label describing the action category. Noticeably, those video annotations are even more complex and demanding than those for image tasks, as the annotators need to scan every frame of the videos manually [32, 139, 166] . How to reduce the annotation complexity renders a serious challenge for video-related tasks. We thus pose the next research question:

How to achieve label-efficient learning for action recognition?

In Chapter 4, we seek to localize the temporal extent of an action in a long untrimmed video to remove the necessity of (temporal) label-annotations. While existing work leverages many examples with their start, their ending, and/or the class of the action during training time, we propose few-shot common action localization. The start and end of an action in a long untrimmed video are determined based on just a handful of trimmed video examples containing the same action, without knowing their common class label. To tackle this task, we introduce a new 3D convolutional network architecture able to align representations from the support videos with the relevant query video segments. Evaluation of few-shot common action localization in untrimmed videos containing single or multiple action instances demonstrates the effectiveness of label-efficient learning for action recognition.

1.2.4 *Video Action Retrieval*

In addition to the problem of temporal action recognition, we also address the issue of video action retrieval. Since the data is inherently characterized by a long-tail distribution [279], the imbalance of action classes in the video data needs to be addressed to facilitate the learning process. A key problem is how to better utilize the labels under the imbalanced scenario. Therefore, we pose the following research question:

How to achieve label-efficient learning for action retrieval?

In Chapter 5, we explore how to conduct label-efficient learning in action retrieval from a video query in an imbalanced scenario. In the query-by-activity-video literature, a common assumption is that all activities have sufficient labeled examples when learning an embedding. This assumption does however practically not hold, as only a portion of activities have many examples, while other activities are only described by few examples. This raises the importance of label-efficient learning of action retrieval in the wild. We propose a visual-semantic embedding network that explicitly deals with the imbalanced scenario for activity retrieval. By matching the activity representations from the perspective of visual and semantic alignment, we no longer ignore infrequent activities during retrieval, consequently improving the label-efficiency under this unbalanced setting. Experiments on a new imbalanced activity retrieval benchmark show the effectiveness of our approach for better label-efficient learning.

1.2.5 *Point Cloud Classification*

So far we focused on label-efficient learning for visual grid structures, i.e., image and video. Here we alter our focus to a 3D point cloud. In a point cloud, an instance-level and point-level annotation is laborious before being able to consider tasks such as point cloud classification [66, 188] and segmentation [250]. Annotating 3D point clouds is extremely difficult due to their unordered, unstructured, and non-uniform nature [105]. For example, it can take over 1700 person-hours to annotate KITTI dataset [16], and around 22.3 minutes for a single indoor scene measuring 5m $\times$ 5m $\times$ 2m [50]. There is a general lack of exploration on how to facilitate label-efficient learning in those tasks. We therefore pose the research question:

How to achieve label-efficient learning for point cloud classification?

In Chapter 6, we introduce a data augmentation technique for point clouds that uses interpolation between examples to increase label-efficiency. Data augmentation by interpolation [276] has proven to be a simple and effective approach in the image domain, but it is not directly transferable to point clouds because there is no one-to-one correspondence between the points of two different objects. Instead, we define data augmentation between point clouds as a shortest path linear interpolation. To achieve this, we introduce PointMixup, an interpolation method that generates new examples through an optimal assignment of the path function between two point clouds. With this interpolation definition, PointMixup allows for the introduction of strong interpolation-based regularizers such as mixup and manifold mixup to the point cloud domain. In our experiments, we demonstrate the label-efficiency potential of PointMixup for point cloud classification, as well as increased robustness to noise and geometric transformations of points.

1.2.6 *Generative Image Diffusion*

In all chapters we have focused on discriminative vision tasks e.g., classification, detection, and segmentation. Here, we consider a computer vision challenge from the

generative perspective, namely diffusion modeling. Diffusion models [99, 130, 216, 220] rely on a fixed noise schedule to gradually corrupt an input image to a standard multivariate Gaussian distribution in a forward process. In a reverse process, the trained network predicts the appropriate amount of noise needed to gradually recover the original image from a multivariate Gaussian distribution. For different aims of the diffusion model, it can be designed and applied in the form of a cascade [203] or in a latent space [200]. The diffusion model can also be categorized into conditional diffusion and unconditional diffusion models according to its dependence on a condition as input. It was recently shown that the conditional diffusion model outperforms the unconditional model in terms of fidelity [100, 200]. Also, the extra condition can empower the model with more controllability on the generation process [54, 100]. However, a large amount of labor-intensive annotations such as labels [100], bounding boxes [200], and segmentation masks [49] are mandated for conditional diffusion models. Thus hindering progress of the diffusion model in terms of fidelity and controllability [99, 100]. Accordingly, we pose the following research question:

How to achieve label-efficient learning for generative image diffusion?

Diffusion models have demonstrated remarkable progress in image generation quality, especially when guidance is used to control the generative process [100]. However, guidance requires a large amount of image-annotation pairs for training, which seriously hinders their label-efficiency. In Chapter 7, we eliminate the need for such annotation by instead exploiting the flexibility of self-supervision signals to design a framework for *self-guided* diffusion models. By leveraging a feature extraction function and a self-annotation function, our method provides guidance signals at various image granularities: from the level of holistic images to object boxes and even segmentation masks. Those guidance signals boost the label-efficient learning in generative diffusion tasks considerably.

The introduction part of the thesis concludes here. In the following section, we present a list of all publications that resulted from the research. Furthermore, each research question introduced in the beginning is expanded upon in the subsequent chapters. Finally, we provide an ultimate conclusion in the last chapter of the thesis.

1.3 LIST OF PUBLICATIONS

- **Chapter 2** is based on "Attention-based Multi-Context Guiding for Few-Shot Semantic Segmentation", *AAAI* 2019 [109], by Tao Hu, Pengwan Yang, Chiliang Zhang, Gang Yu, Yadong Mu, and Cees G. M. Snoek.

  Contribution of authors

  Tao Hu: all aspects,
  Pengwan Yang: guidance and technical advice,
  Chiliang Zhang: guidance and technical advice,
  Gang Yu: guidance and technical advice,
  Yadong Mu: guidance and technical advice,
  Cees G. M. Snoek: Overall discussion.

- **Chapter 3** is based on "SILCO: Show a Few Images, Localize the Common Object", published in *ICCV*, 2019 [107], by Tao Hu, Pascal Mettes, Jia-Hong Huang, Cees G. M. Snoek.

  Contribution of authors

  Tao Hu: all aspects,
  Pascal Mettes: guidance and technical advice,
  Jia-Hong Huang: help with experiments,
  Cees G. M. Snoek: supervision and insight.

- **Chapter 4** is based on "Localizing the Common Action Among a Few Videos", published in *ECCV*, 2020 [270], by Pengwan Yang*, Tao Hu*, Pascal Mettes, Cees G. M. Snoek.

  Contribution of authors

  Pengwan Yang*, Tao Hu*,: all aspects,
  Pascal Mettes: guidance and technical advice,
  Cees G. M. Snoek: supervision and insight.

- **Chapter 5** is based on "Query by Activity Video in the Wild", *submitted for publication*, by Tao Hu, William Thong, Pascal Mettes, Cees G. M. Snoek.

  Contribution of authors

  Tao Hu: all aspects,
  William Thong: guidance and technical advice,
  Pascal Mettes: guidance and technical advice,
  Cees G. M. Snoek: supervision and insight.

- **Chapter 6** is based on "PointMixup: Augmentation for Point Clouds", published in *ECCV*, 2020 [44], by Yunlu Chen*, Tao Hu*, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, Cees G. M. Snoek.

  Contribution of authors

  Yunlu Chen*, Tao Hu*: all aspects,
  Efstratios Gavves: guidance and technical advice,
  Pascal Mettes: guidance and technical advice,
  Thomas Mensink: guidance and technical advice,
  Pengwan Yang: guidance and technical advice,
  Cees G. M. Snoek: supervision and insight.

- **Chapter 7** is based on "Self-Guided Diffusion Models", published in *CVPR* 2023 [110], by Tao Hu*, David W Zhang*, Yuki M. Asano, Gertjan J. Burghouts, Cees G. M. Snoek.

  Contribution of authors

  Tao Hu*, David W Zhang*: all aspects,
  Yuki M. Asano: guidance and technical advice,
  Gertjan J. Burghouts: guidance and technical advice,
  Cees G. M. Snoek: supervision and insight.

More works by the author are provided in the following:

- Self-supervised Video Representation Learning with Cross-Stream Prototypical Contrasting, WACV, 2022 [232], Martine Toering, Ioannis Gatopoulos, Maarten Stol, Tao Hu
- Interactivity Proposals for Surveillance Videos, ICMR, 2020 [41], Shuo Chen, Pascal Mettes, Tao Hu, Cees G. M. Snoek
- Accelerating Convolutional neural networks with dynamic channel pruning, DCC, 2019 [275], Chiliang Zhang, Tao Hu, Yingda Guan, Zuochang Ye
- Dense In Dense: Training Segmentation from Scratch, ACCV, 2018 [106], Tao Hu
- Sobel heuristic kernel for aerial semantic segmentation, ICIP, 2018 [108], Tao Hu, Yao Wang, Yisong Chen, Peng Lu, Heng Wang, G Wang

2

ATTENTION-BASED MULTI-CONTEXT GUIDING FOR FEW-SHOT SEMANTIC SEGMENTATION

2.1 INTRODUCTION

The state-of-the-art of image classification, detection, segmentation techniques have been greatly advanced by the convolution neural network (CNN). Although CNN exhibits superior performances in a variety of tasks, it has the key problem of being data hungry. Typically, gigantic data with annotations are required for achieving high performance. This issue becomes more severe in pixel-level annotations. In recent years, there emerges a new research thrust which learns new concepts from limited data, known as few-shot learning in the literature [196, 215, 239]. Though widely explored in tasks like image classification, few-shot learning is rarely considered in dense pixel prediction problems.

Most of existing methods in few-shot semantic segmentation are based on the framework as shown in the top panel of Figure 3. Conceptually, the framework is composed of the support branch and query branch. The support branch provides discriminative support feature to assist the target segmentation, the query branch is the feature extractor for target segmentation. This paradigm has the following difficulties: (1). *Inefficient support feature utilization.* The feature of the support branch is precious, as it determines the final category the network will segment. However, most of the previous methods only consider the single output from the end of the network, which do not take full advantage of multi-context features. (2). *Lack of attention.* As the amount of support and query data is often very small, optimization based on a large amount of data is impossible, some self-supervised methods such as attention mechanism should be introduced to make the network concentrate on our target class. (3). *Inconvenience in multi-shot learning.* Traditional fusion method for multi-shot semantic segmentation is logical OR operation, this inflexible approach lacks in exploring the inner common feature between various support images.

To attack the above problems in few-shot semantic segmentation, we propose an Attention-based Multi-Context Guiding network (A-MCG) shown in the bottom panel of Figure 3. Our A-MCG tries to fuse small-to-large scale context information to globally guide the query branch to make the right segmentation decision. A multi-context feature will largely facilitate query branch segmentation based on multiple scales of support feature. In addition, we utilize the Residual Attention Module (RAM) [242] to carry out a self-supervised attention mechanism for further improvement of the segmentation.

Published in Proceedings of the AAAI Conference on Artificial Intelligence, 2019 [109].

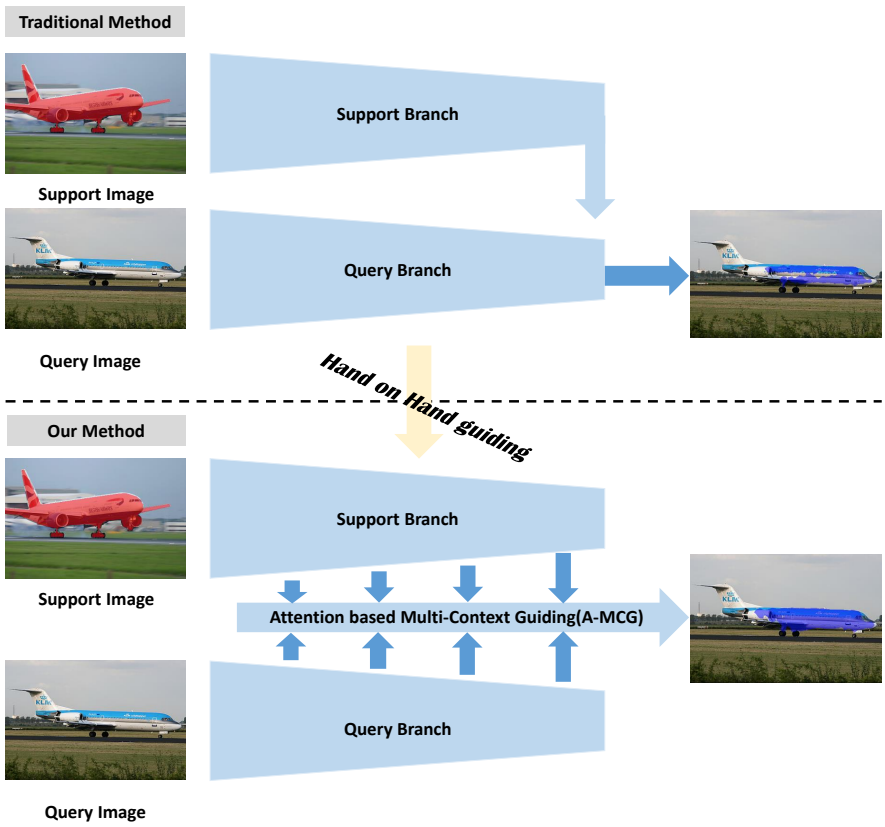


Figure 3: **Our motivation.** The support image mask is overlaid with ground truth in red, and the query image is overlaid with ground truth in blue. The above is the traditional method, and below is our method. Our A-MCG network can help the support branch fuse multi-context information to guide the query branch.

To deal with multi-shot learning, ConvLSTM [260] is incorporated for better fusing multi-shot support features.

Our A-MCG network makes the following contributions: (1). We first propose a Multi-Context Guiding structure to fuse the small-to-large scale context features between the support branch and query branch to globally guide the query branch segmentation. (2). We introduce a Residual Attention Module [242] in our MCG network to realize the attention mechanism in few-shot learning of segmentation. (3). We embed the ConvLSTM [260] module into the end of our network to better merge the feature map from the support set in multi-shot semantic segmentation. (4). Compared with previous methods, our A-MCG reaches state-of-the-art 61.2%, 62.2% measured in mIoU in the 1-shot or 5-shot setting respectively.

2.2 RELATED WORK

Semantic Segmentation. During the early period, the CNN is only employed in the classification tasks, most of them [136, 227] are composed of convolution layers and fully connected layers. Fully Convolutional Network(FCN) [155] firstly applies CNN for the task of image semantic segmentation. FCN's key contribution is building "fully

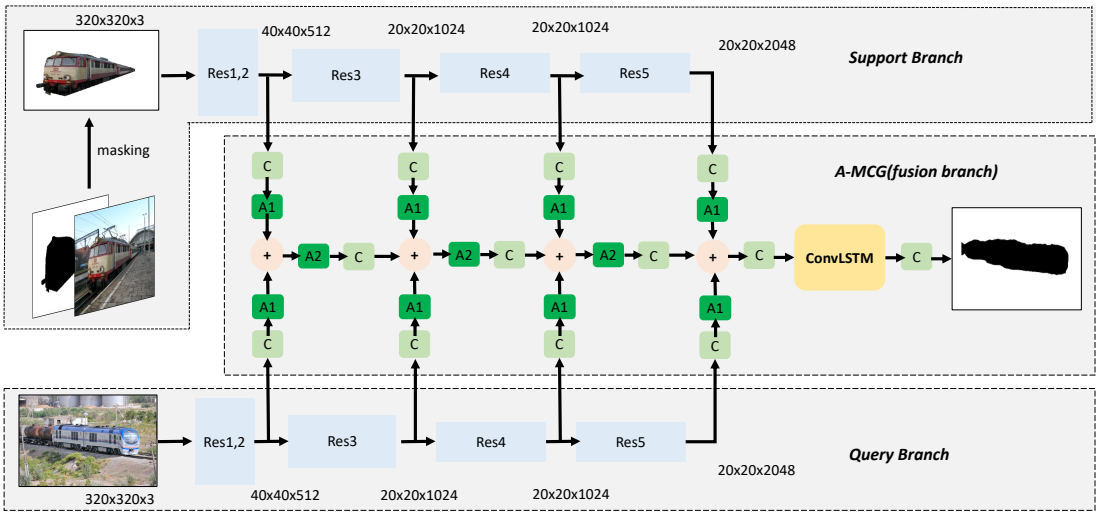


Figure 4: **Attention-based Multi-Context Guiding Network Architecture for One-Shot Segmentation.** It includes three parts: (1). support branch. (2). query branch. (3). A-MCG module. Res1, 2, 3, 4, 5 represent different block in ResNet. C is 1×1 convolution unit, stride=2 convolution is employed when the image size becomes smaller. We design two exclusive location settings of the attention mechanism: A1, A2. The detail illustration of the attention architecture are deferred in Figure 5.

convolutional" network that takes an input of arbitrary size and produces correspondingly-sized output with efficient inference and learning.

In Deeplab [40], Dilated Convolutions are introduced as an alternative to CNN pooling layers in deep part to capture larger context without reducing the image resolution. A module named Atrous Spatial Pyramid Pooling (ASPP) is also included in Deeplab where parallel Dilated Convolution layers with different rates capture multi-scale information. Our method is also illuminated by the multi-context fusion pattern of ASPP and merges the multi-context information by borrowing the multi-scale context from support branch and query branch.

Attention Mechanism. In this chapter, we mainly talk about two types of attention mechanism: (1). *Spatial Attention* such as Residual Attention Module(RAM) [242]. Inside each Attention Module, an Hourglass-like [174] bottom-up and top-down feed-forward structure is used for generating attention map. (2). *Channel Attention* like SENet [104]. "Squeeze-and-Excitation"(SE) block is designed that adaptively recalibrates channel-wise feature responses by explicitly modeling interdependencies between channels. We mainly attempt to employ these two types of attention mechanism into our MCG architecture.

Few-Shot Learning in Semantic Segmentation. The first work in few-shot semantic segmentation is OSLSM [207]. They proposed the basic paradigm in few-shot segmentation. Support branch and query branch are constructed by VGG [213] to supervise training. However, the structure is not fully convolutional, which leads to inefficient utilization of spatial information. Later co-FCN [191] turns both of the support and query branches into FCN architecture, but the exploration of multi-scale context is not thorough as our method.

Table 1: **Problem Formulation Notations.**

notation	meaning
I_S^i	i-th image in support set
$Y_S^i(l)$	i-th image-binary mask for class l
$S = \{(I_S^i, Y_S^i(l))\}_{i=1}^k$	support set
I_q	image in query set
L_{train}	train label set
L_{test}	test label set

On the other side, OSVOS [30] tries to solve the task of semi-supervised video object segmentation. OSVOS is also based on a fully-convolutional network architecture and transfers generic semantic information to the task of foreground segmentation. OSVOS shows the effectiveness of fine-tuning for video object segmentation, but fine-tuning for every test video is too time-consuming.

Convolutional Long Short-Term Memory. Long Short-Term Memory(LSTM) [101] is proposed as a special RNN structure to model long-range dependencies in various previous studies. However, LSTM is not suitable for handling spatiotemporal data because the input-to-state and state-to-state are all full connections thus no spatial information is encoded. To tackle this problem, ConvLSTM [260] is put forward by using a convolution operator in the state-to-state and input-to-state transitions.

In our network, ConvLSTM works as a memory unit to capture and integrate the previous support set feature for better multi-shot learning in semantic segmentation. This method gives us more interpretability and can better smoothen our k-shot learning result compared with traditional fusion method.

2.3 METHOD

2.3.1 *Problem Formulation*

We follow the paradigm and notations in [207], which are detailed in Table 1. The target is to learn a model $f(I_q, S)$ that, when given a support set S and query image I_q, predicts a binary mask $\hat{M}_q$ for the semantic class l. The f function is parameterized by neural networks of support branch and query branch.

During training, the algorithm has access to a large set of image-mask pairs $D = \{(I^j, Y^j)\}_{j=1}^N$ where $Y^j \in L_{train}^{H\times W}$ is the binary mask for training image I^j. At testing, the query images are only annotated for new(unseen) semantic classes i.e. $L_{train} \cap L_{test} = \prec$, which leads us to divide the PASCAL VOC12 dataset like Table 2. This is the key difference between one-shot learning for segmentation and traditional segmentation, what we really care about is the segmentation performance on unseen data. Similar to the extension from one-shot learning to k-shot learning in classification task, k-shot learning can also be applied in semantic segmentation. In OSLSM [207], k-shot learning results are fused by a logical OR operation between the k binary masks.

2.3.2 *Preliminary Knowledge of Attention Mechanism*

Residual Attention Module(RAM) is first proposed by Wang *et al.* [242] for image classification. We here review the structure of RAM in Figure 5. The original RAM proceeds many ablation studies for its setting, we directly use the explored optimal structure. In this chapter, we mainly explore the attention location in our MCG module in Section 2.4.

The RAM actually utilizes a two-scale(down sample 2 times, then up sample 2 times) Hourglass structure to construct a soft attention mask M(x). In the original ResNet [95], residual learning is formulated as:

$$H_{i,c} = x + F_{i,c}(x), \tag{2.1}$$

where $F_{i,c}(x)$ approximates the residual function, i ranges over all spatial positions and $c \in \{1, ..., C\}$ is the index of the channel.

In RAM, the attention module is modified as:

$$H_{i,c} = (1 + M_{i,c}(x)) * F_{i,c}(x), \tag{2.2}$$

where M(x) ranges from [0,1] because of the sigmoid function. With M(x) approximating 0, H(x) will approximate original features F(x). The key of RAM lies in M(x), which works as feature selectors that enhance good features and suppress noises from trunk features. This characteristic of RAM is particularly important for few-shot learning cases.

Except that, we also explored the SE block [104], which is a typical channel attention structure. Our later ablation study in Section 2.4 shows that SE block fails to compete with RAM under the condition of same parameters.

2.3.3 *Attention-based MCG*

We propose an Attention-based Multi-Context Guiding network (A-MCG) illustrated in Figure 4. Our A-MCG network is composed of three parts: (1). support branch. (2). query branch. (3). A-MCG(fusion) branch. The backbones of support branch, query branch are ResNet101. We elaborate our network as input image size 320×320, and their output feature map size is also marked in Figure 4. Notably, the convolutions in Res-4, Res-lass5 blocks are equipped with Dilated Convolution [40] whose dilated rate=2. Therefore, the feature map size no long decreases after Res-3, but the receptive field continues to be enlarged due to Dilated Convolution.

Our A-MCG module tries to utilize multi-context feature from support branch to globally improve the query branch segmentation. We attempt two types of attention location pattern. In the following, we denote the b_s^i as feature maps after Res-i block in support branch, b_q^i as feature maps after Res-i block in query branch, C as naive convolution(without ReLU [173], BN), H as our attention function, F_i as the mixed features after Res-i block of support branch and query branch.

Therefore, we mainly come up with three variants.

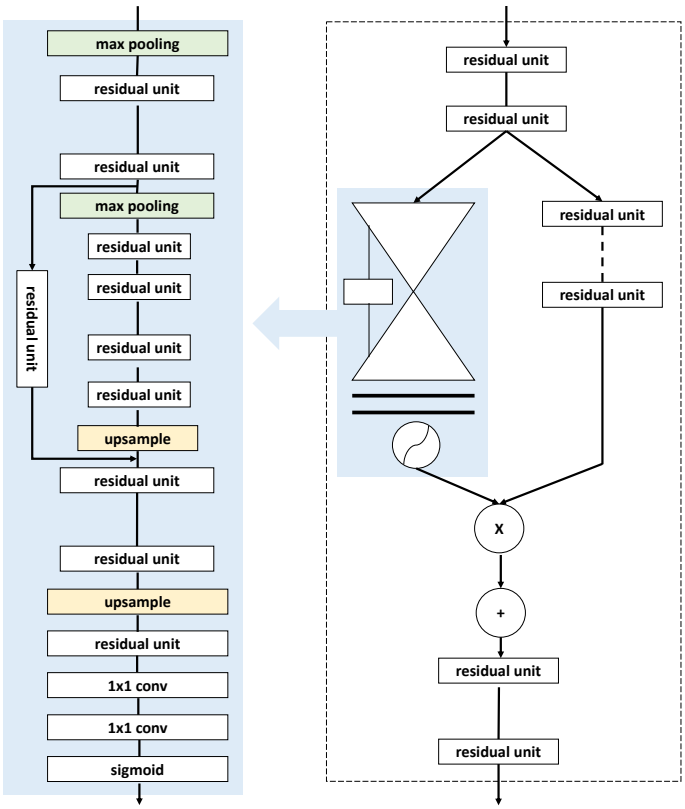


Figure 5: **Residual Attention Module(RAM).** The left side is an hourglass-like Soft Mask Branch, followed by a softmax operation. Right side is the whole structure of RAM. *residual unit* is a typical bottleneck structure which is detailed in ResNet [95].

(1). Multi-context guiding module:

$$F_{i+1} = C\{C(b_s^i) + C(b_q^i) + F_i\} \tag{2.3}$$

Multi-context information from both branches are mixed by convolution operation. Notably, convolution here doesn't include ReLU, BN.

(2). Multi-context guiding with separate attention:

$$F_{i+1} = C\{H(C(b_s^i)) + H(C(b_q^i)) + F_i\}, \tag{2.4}$$

which corresponds to A1 in Figure 4.

Attention mechanism is employed separately both in query branch and support branch.

(3). Multi-context guiding with share attention:

$$F_{i+1} = C\{H(C(b_s^i) + C(b_q^i) + F_i)\}, \tag{2.5}$$

which corresponds to A2 in Figure 4. Attention mechanism is applied after the fusion of query branch and support branch.

2.3.4 *Convolutional LSTM for k-shot Learning*

In previous work, we mainly focus on the circumstance of 1-shot learning in semantic segmentation. How shall we deal with k-shot learning? In OSLSM [207], k-shot learning results are fused by a logical OR operation. However, this straightforward process is unexplainable and fails to utilize the inner relationship between sequential support images.

To better solve this multi-shot learning problem, we attempt to embed ConvLSTM [260] at the end of the fusion branch as illustrated in Figure 4. At last, a 1×1 convolution will be appended to generate segmentation probability map.

ConvLSTM is firstly applied to the precipitation nowcasting task. The key idea of ConvLSTM is to implement all operations, including state-to-state and input-to-state transitions, with kernel-based convolutions. The inner feature of the sequential support image mask can be sustained by ConvLSTM. We here adopt a popular LSTM variant with "peephole connections" [77]. In detail, the three gating functions in ConvLSTM are calculated according to the equations below:

$$i_t = \sigma(\boldsymbol{W}_{x,i} \otimes \boldsymbol{X}_t + \boldsymbol{W}_{h,i} \otimes \boldsymbol{H}_{t-1} + \boldsymbol{W}_{c,i} \otimes \boldsymbol{C}_{t-1}), \tag{2.6}$$

$$o_t = \sigma(\boldsymbol{W}_{x,o} \otimes \boldsymbol{X}_t + \boldsymbol{W}_{h,o} \otimes \boldsymbol{H}_{t-1} + \boldsymbol{W}_{c,o} \otimes \boldsymbol{C}_t), \tag{2.7}$$

$$f_t = \sigma(\boldsymbol{W}_{x,f} \otimes \boldsymbol{X}_t + \boldsymbol{W}_{h,f} \otimes \boldsymbol{H}_{t-1} + \boldsymbol{W}_{c,f} \otimes \boldsymbol{C}_{t-1}), \tag{2.8}$$

where we let $\boldsymbol{X}_t, \boldsymbol{H}_t$ be the input/hidden state at time t respectively. $\otimes$ represents spatio-temporal convolution operator.

Investigating previous $\boldsymbol{H}$ and current $\boldsymbol{X}$, the recurrent model synthesizes a new proposal for the cell state, namely

$$\tilde{\boldsymbol{C}}_t = tanh(\boldsymbol{W}_{x,c} \otimes \boldsymbol{X}_t + \boldsymbol{W}_{h,c} \otimes \boldsymbol{H}_{t-1}) \tag{2.9}$$

The final cell state is obtained by linearly fusing the new proposal $\tilde{\boldsymbol{C}}_t$ and previous state $\boldsymbol{C}_{t-1}$:

$$\boldsymbol{C}_t = f_t \odot \boldsymbol{C}_{t-1} + i_t \odot \tilde{\boldsymbol{C}}_t, \tag{2.10}$$

where $\odot$ denotes the Hadamard product. To continue the recurrent process, it also renders a filtered new H:

$$\boldsymbol{H}_t = o_t \odot tanh(\boldsymbol{C}_t) \tag{2.11}$$

In our previous structure, the network is trained on one-shot support set. Once ConvLSTM is imported into our framework, it will enable us to train with k-shot support set. For every batch(if batch size=1), one query image and k support image masks will be fed into our neural network.

We unroll this procedure in Figure 6 for better understanding the k-shot fusion process. k-shot support image masks enter ConvLSTM in turn. ConvLSTM plays a critical role in summarizing the total features of the k-shot support image masks.

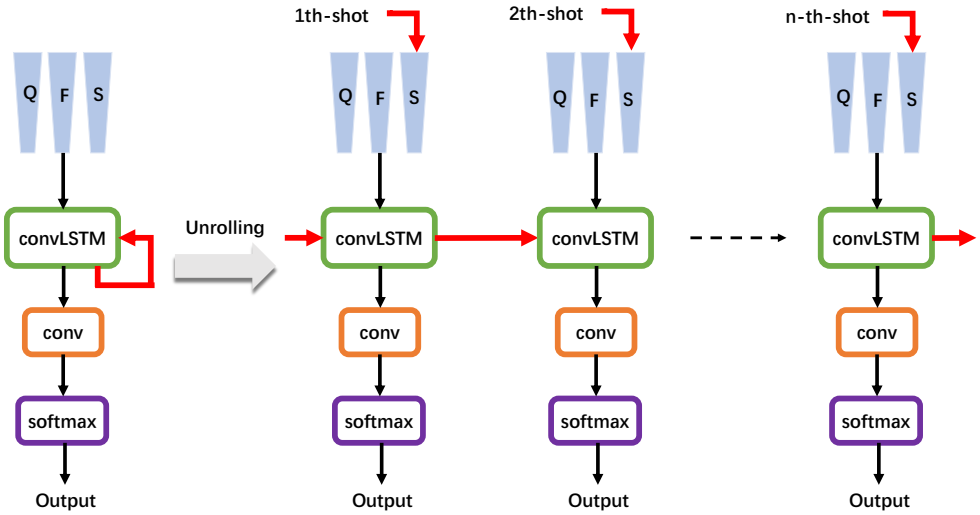


Figure 6: **Unrolling view of ConvLSTM.** Q, F, and S represents the query branch, fusion branch, and support branch accordingly. Best viewed in color.

For better mixing the feature from the support set in k-shot learning, a function loss is designed as follows:

$$L = -\frac{1}{ks^2}\sum_{i=0}^{k}\sum_{m=0,n=0}^{s} Y_{m,n} log X_{m,n}, \tag{2.12}$$

where Y is binary label, X means the neural network output probability, k is shot number, s represents image size.

This loss enforces our A-MCG module to function well on **every** support set image rather than only supervises the segmentation of single support image.

2.4 EXPERIMENTAL RESULT

2.4.1 *Training Details*

We implement our code based on the tensorflow framework [1]. Specially, a scaffold framework named tensorpack [254] is used for quickly setting up our experiment. All our models are trained by Stochastic Gradient Descent(SGD) [25] solver with learning rate=1e-4, momentum=0.99 on one Nvidia Titan XP GPU. To fully fill GPU memory, we set the batch size 12. The weights of support branch and query branch are initialized with ImageNet [53] pre-trained weights. For the weight initialization of A-MCG module, Xavier initialization [82] is adopted. All the images in the support and query branch are resized to 320×320. No further augmentation is employed except the image resizing. For Batch Normalization(BN) [114], we employ current batch statistics at training and use the moving average statistics of BN during validation time.

We use the cross-entropy loss as the object function for training the network. The loss is summed up over all the pixels in a mini-batch.

Table 2: **PASCAL-i^5 group information.** The top table displays 4 groups of label and their semantic classes. The bottom table shows 4 sub-datasets and their training, validation components.

label set index	semantic classes
0	aeroplane, bicycle, bird, boat, bottle
1	bus, car, cat, chair, cow
2	diningtable, dog, horse, motorbike, person
3	potted plant, sheep, sofa, train, tv/monitor

sub-dataset	train label set	val label set
0	1,2,3	0
1	0,2,3	1
2	0,1,3	2
3	0,1,2	3

When we experiment with ConvLSTM for k-shot learning, we set k=5 by default. The max batch size can only be 6 because every time k support image masks will be fed into the support branch. Layer Normalization [10] is utilized in our ConvLSTM for speeding up convergence.

2.4.2 *Dataset and Metric*

Dataset: We utilize dataset PASCAL-5^i [207] to conduct our experiment. This dataset is originated from PASCAL VOC12 [65] and extended annotations from SDS [89]. The set of 20 classes in PASCAL VOC12 is divided into four sub-datasets as indicated in Table 2. Three sub-datasets are used as the training label-set L_{train}, the left one sub-dataset is utilized for test label-set L_{test}.

The training set D_{train} is composed of all image-mask pairs from PASCAL VOC12 and SDS training sets that include at least one pixel in the segmentation mask from the label-set L_{train}. The masks in D_{train} are modified into binary masks by setting pixels whose semantic class are not in L_{train} as background class l_ϕ. The test set D_{test} is from PASCAL VOC12 and SDS validation sets, and the processing procedure for test set D_{test} is similar with training set D_{train}. Our evaluation mIoU is the average of 5 sub-dataset mIoUs. For a fair comparison with [207], we take the same random seed and sample N=1000 examples for testing each of our models.

Metric: To compare the quantitative performance of the different models, mean intersection over union(mIoU) over two classes is used for our benchmark evaluation. For binary segmentation in our work, we firstly calculate the 2×2 confusion matrix, then compute the according IoU_l as $\frac{tp_l}{tp_l+fp_l+fn_l}$. tp_l is the number of true positives for class l, fp_l is the number of false positives for class l and fn_l is the number of false negatives for class l. The final mIoU is its average over the set of classes.

Table 3: **Ablation Study for fusion width.** The experiment is conducted on PASCAL-i^5 sub-dataset 0.

fusion width	1-shot	#params(M)
64	63.4	85.6
128	63.3	86.1
256	63.6	87.2
512	63.6	89.8
1024	63.2	96.1

2.4.3 *Ablation Study*

Baseline. Our method is mostly compared with OSLSM [207] and co-FCN [191]. Both of them utilize the VGG [213] as basic model. Different from them, we adopt ResNet101 [95] as our basic model, for ResNet101 owns much less parameter than VGG16, thus it is less prone to over-fitting. Besides, ResNet also enables larger batch size training in our architecture.

After removing the fully connected layers in the end, our ResNet101 baseline becomes a fully-convolutional structure. Support branch and query branch are fused by element-wise Add between the Res-5 output of them, followed by a naive convolution.

MCG. We explore several factors of our Multi-Context Guiding(MCG) architecture such as (1). *fusion width.* Fusion width is the channel number in the MCG branch, all the features in the support branch and query branch will be transformed into features with width of fusion width. Different settings of fusion width are stated in Table 3. (2). *multi-context pattern*, we try to explore what kind of context combination is better for the few-shot learning in Table 4. The number after "context" is the feature we will adopt for fusion. For example, context-45 means that only features from Res-4, Res-5 are used for fusion. For convenience, we only proceed the ablation study in the sub-dataset 0 in this part.

From Table 3, we can conclude that when the fusion width is too small such as 64 ,128, the one-shot mIoU is 0.2% lower than width=256. Meanwhile, larger fusion width like 1024 will make the mIoU worse due to over-fitting. Taking into consideration of the balance between mIoU and parameter cost, we choose fusion width=256 as our default setting. The latter ablation study will also adopt the default value.

As for the multi-context pattern in Table 4, more level context fusion often leads to a better result. Context-2345 outperforms context-5 nearly 2.4% mIoU. This shows that our multi-context guiding strategy works as our motivation. Multi-context information fusion from both support branch and query branch could efficiently "support" the query branch's segmentation.

Attention Mechanism. We set two variants about the attention module: (1). *Spatial Attention.* Residual Attention Module(RAM) is applied here as a representative method. (2). *Channel Attention.* SENet [104] is explored in our ablation study. At the same time, two attention location patterns "separate" and "share" are also explored. "sep" denotes the support branch and query branch adopt separate attention. "share" represents the support branch and query branch share the same attention.

Table 4: **Ablation Study for multi-context pattern.** The experiment is conducted on PASCAL-i^5 sub-dataset 0.

Method	1-shot	#params(M)
context-2345	63.6	87.2
context-45	63.2	86.7
context-5	61.2	86.1

Table 5: **Ablation Study for Attention Mechanism.** ChannelAttention means SENet Block, SpatialAttention means Residual Attention Block. "sep" denotes the support branch and query branch adopt separate attention. "share" represents the support branch and query branch share the same attention.

Method	1-shot	#params(M)
MCG	63.3	87.2
MCG-ChannelAttention-sep[1]	63.6	89.6
MCG-ChannelAttention-share[2]	61.7	89.8
MCG-SpatialAttention-sep	63.3	93.3
MCG-SpatialAttention-share	65.8	89.5

[1] For fair comparison with SpatialAttention method, we change the fusion width to 428 to make #param nearly the same.

[2] For fair comparison with SpatialAttention method, we change the fusion width to 480 to make #param nearly the same.

As shown in Table 5. We can find that Spatial Attention works much better than Channel Attention under the circumstance of same parameters. We conclude that spatial information is more useful in dense pixel task like image segmentation, while channel information plays a more important role in classification task.

It can be obviously figured out that sharing the same attention is basically better than separate attention. We speculate that for support branch, the input has been already masked so that it does not need attention mechanism, while sharing attention mechanism will make the query branch pay more attention to the support branch's input mask.

On the other hand, we demonstrate some images' feature map visualization and feature map histogram in Figure 7. From the visualization, we can qualitatively observe that the feature map becomes more focusing on the target segmentation objects. As for the feature map histogram, we can discovery that the histogram peak move towards small value. We owe this observation to the fact that the Spatial Attention enhances good features and suppresses noises from trunk features.

ConvLSTM for k-shot learning. We mainly contrast two loss variants in ConvLSTM. (1). *1-loss ConvLSTM*. Only output from the last shot is supervised. (2). *5-loss ConvLSTM*. Every output from ConvLSTM is supervised. As indicated in Table 6, the ConvLSTM highly boosts our mIOU both in 1-shot and 5-shot learning. 1-shot result on 1-loss ConvLSTM is 0.7% lower than our baseline, we speculate that 1-loss ConvLSTM fails to fully supervise single-shot learning. Interestingly, both the 1-shot and

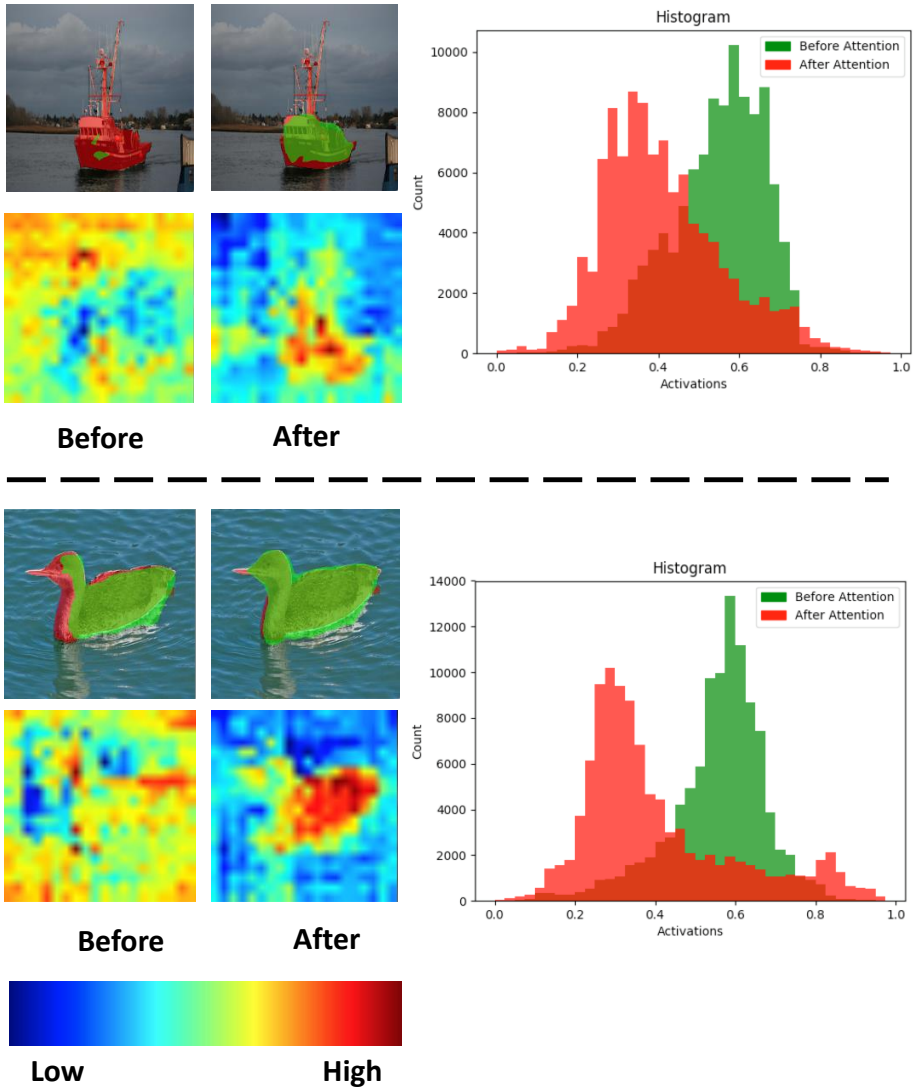


Figure 7: **Attention Mechanism Visualization.** Two images are demonstrated for the comparison between "before attention" and "after attention". The image is overlaid with predicted mask in green. The according histogram of the feature map is displayed in the right side. The feature map's activation value is normalized to 1. The support image is ignored. Best viewed in color.

5-shot result on 5-loss LSTM outperform our baseline, which sufficiently validates our motivation.

Furthermore, we also conduct k-shot learning where k ranges from 1 to 10 in Figure 8. k-loss ConvLSTM fully surpasses the traditional logical or method in all shot number range. When $k \leq 4$, the performance of 1-loss ConvLSTM is less than 5-loss ConvLSTM, while partially larger than our baseline. This proves that our 5-loss ConvLSTM better integrates multi-shot support features than traditional method.

Final result. As shown in Table 7, our A-MCG architecture could outperform nearly 61.2% in 1-shot mIoU, 62.2% in 5-shot mIoU. Based on our baseline, we continue applying the MCG, attention mechanism, ConvLSTM, reach a new state-of-the-art result on PASCAL-i^5 dataset in the end.

Table 6: **Ablation Study for loss function in ConvLSTM.** Baseline is our A-MCG module, we mainly compare the difference between 1-loss ConvLSTM and 5-loss LSTM. The experiment is conducted on PASCAL-i^5 sub-dataset 0.

Method	1-shot	5-shot	#params(M)
baseline	65.8	66.2	89.5
1-loss ConvLSTM	65.1	67.5	90.8
5-loss ConvLSTM	66.1	67.9	90.8

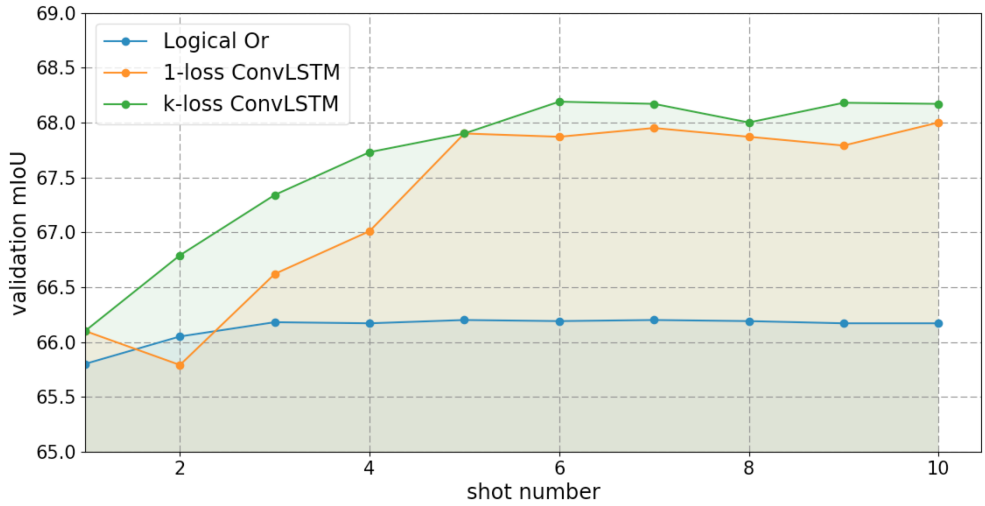


Figure 8: **The relationship between shot number and validation mIoU.** We mainly compare among three multi-shot learning fusion strategies: (1). Logical Or. (2). 1-loss ConvLSTM. (3). k-loss ConvLSTM(k=5 in our experiment). The experiment is conducted on PASCAL-i^5 sub-dataset 0.

Some visualization result can be seen in Figure 9. We also explore the effect of increasing the support set size as shown in Figure 10.

2.5 CONCLUSION

We propose an Attention-based Multi-Context Guiding network (A-MCG) which incorporates multi-level concentrated context. The benefits of our network are in three folds: (1). The shallow part of our network generates low-level semantic features, meanwhile deep part of our network captures high-level semantic features. Context features in equal level are fused by our MCG module, which highly facilitates the support branch to globally "support" the query branch. (2). Spatial Attention is employed along with the whole MCG branch, which makes our network focus on different scales of context information. (3). The import of ConvLSTM enables the network better integrate the feature from support set in multi-shot semantic segmentation. The performance of our model surpasses state-of-the-art methods in few-shot semantic segmentation.

Table 7: **Result on PASCAL-i^5 Dataset.** All results are computed by taking the average of the 5 sub-datasets in PASCAL-i^5. The 5-shot result is obtained by logic or fusion except the method with ConvLSTM.

Method	1-shot	5-shot	#params(M)
OSLSM [207]	40.8	43.9	276.7
co-FCN(Multi-class) [191]	50.9	50.9	–
co-FCN(Overall) [191]	60.1	60.8	–
Baseline	53.0	54.8	85.1
MCG	55.3	56.5	87.2
A-MCG	57.3	57.8	89.5
A-MCG-ConvLSTM	**61.2**	**62.2**	90.8

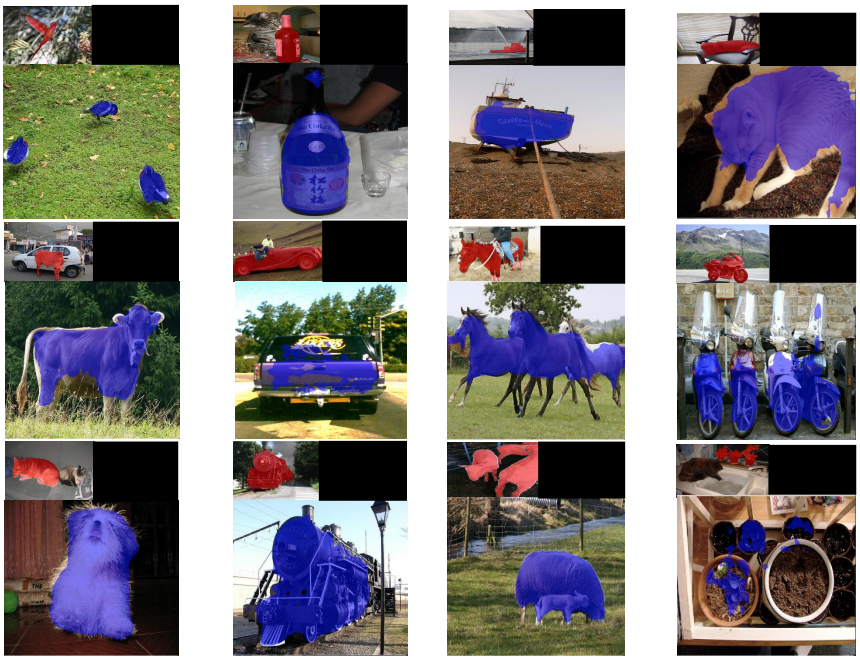


Figure 9: **Qualitative results of our method for 1-shot learning.** Inside each tile, we have the support set at the top and the query image at the bottom. The support is overlaid with the ground truth in red and the query is overlaid with our predicted mask in blue. Best viewed in color.

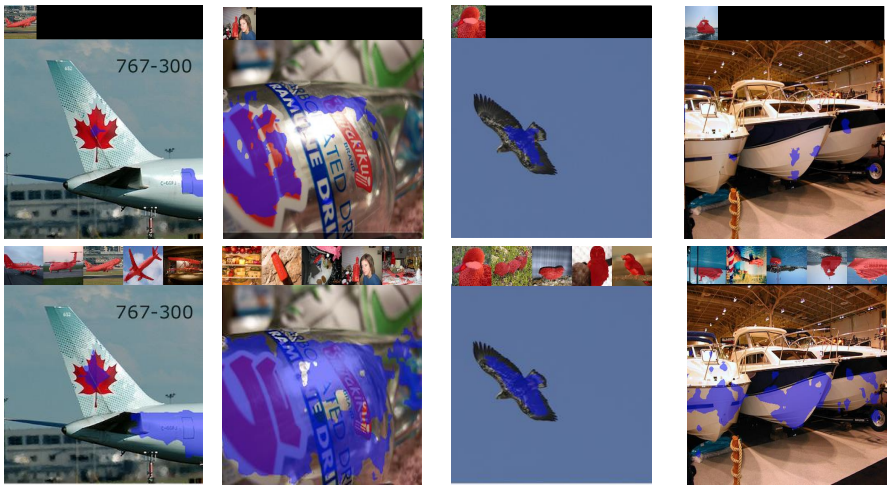


Figure 10: **Effect of increasing the size of the support set.** Results of 1-shot and 5-shot learning on the same query image are in the first and second rows, respectively. The prediction is in blue. Best viewed in color.

3

SILCO: SHOW A FEW IMAGES, LOCALIZE THE COMMON OBJECT

3.1 INTRODUCTION

Convolutional networks exhibit superior accuracy in a wide variety of computer vision challenges, but a key limitation remains their hunger for labeled data [53, 163, 229]. Typically, large amounts of annotated examples are required to achieve a high accuracy. This issue becomes even more severe for localization tasks, which typically require additional localized annotations [153, 197, 198]. In recent years, a new research line has emerged that strives to learn new concepts from limited amounts of data, known as few-shot learning [215, 239]. Though widely explored in tasks like image classification, few-shot learning is rarely considered for object localization problems.

In this chapter, we propose the new task of few-shot common-localization, which takes N support images (without box annotations) and one query image as input and tries to localize the common object in the query image guided by the support images. Our task is demonstrated in Figure 11 for four support images and one query image. Unique to our task is we only know there is a common class among the support and query images, but the class itself and its spatial extent is unknown. In practice, we can obtain query and support images with a common class by leveraging social tags, hash tags, or an off-the-shelf image classification network. Our method adds the bounding box for free.

We investigate this task in an attempt to alleviate the double burden of annotation in visual localization tasks, namely regarding the number of examples and regarding the box annotations for each example. This task is therefore on the intersection between few-shot learning [215, 239] and weakly-supervised detection [19, 230]. We also envision a number of applications that arise from this task. First, few-shot common-localization enables us to search spatially for specific instances in large and complex scenes. Second, we can use this task to enhance other learning tasks. Few-shot common localization can for example be used as a quick annotation tool or as a form of prior annotation for tasks such as active learning [75]. Third, this approach enables easier search in tasks such as remote sensing [108, 161].

At the core of few-shot common-localization is getting the most out of the limited information. To that end, we propose a new deep network made for this task, shown in Figure 11. First, the support and query images are fed into a backbone network to

Published in Proceedings of the International Conference on Computer Vision, 2019 [107].

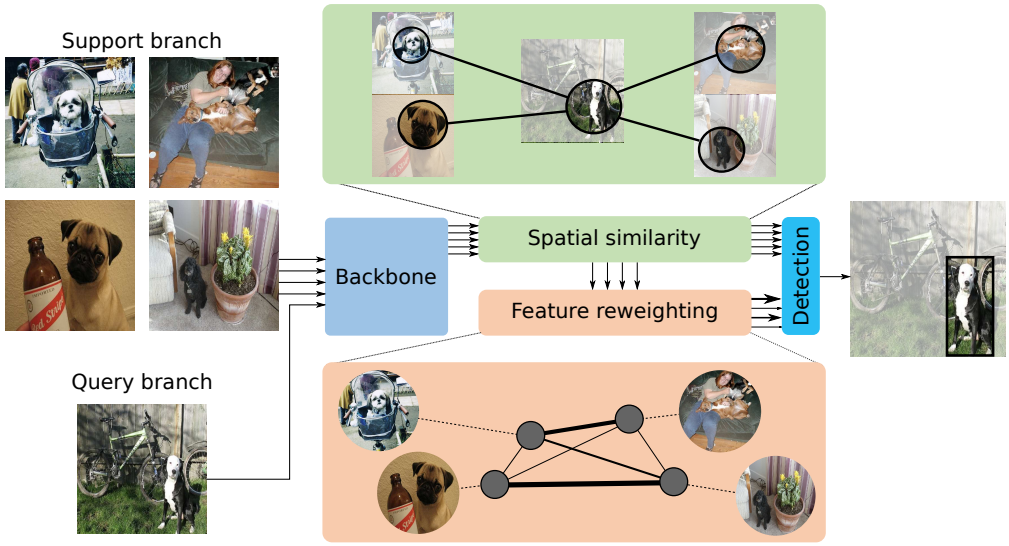


Figure 11: **Few-shot common-localization.** Starting from a few weakly-supervised support images and a query image, we are able to localize the common object in the query image without the need for a single box annotation. We only know that there is a common object, we do not know *where* and *what* the common object is.

obtain spatial features. Second, we discover what is common among the support and query images. We propose a spatial similarity module that learns the spatial regions of commonality through non-local operations. Third, we hypothesize that support images are not equally important and propose a feature reweighting module. This module employs graph convolutional networks to balance the support images. Fourth, we use the spatial and weight information with a class-agnostic localization network to localize the object in the query image.

To experiment on few-shot common-localization, we repurpose and reorganize the well-known Pascal VOC 2007, Pascal VOC 2012, MS-COCO, and ImageNet VID datasets. Experimental results show the importance of spatial similarity and feature reweighting for few-shot common-localization. This results in a system that outperforms baselines from related tasks such as object detection and few-shot learning. The setup and method serve as a catalyst for future work in this task and are all publicly available along with the code of our networks and modules at http://taohu.me/SILCO/.

3.2 RELATED WORK

Object detection. Modern object detectors can be categorized into two categories: one-stage and two-stage detectors. One-stage detectors such as YOLO [197] and SSD [153] directly use the backbone architecture for object instance detection. Two-stage detectors such as Faster R-CNN [198] and FPN [149] first propose many possible object locations and use a sub-network for determining and regressing the best proposals. In this work, we rely on basic components such as SSD [153]. Where standard object detection requires many examples and dense annotations, we utilize such networks to deal with few examples and no box annotations. Weakly-supervised object detection [19, 47, 55, 210, 230] has recently been investigated for the scenario where many examples are given,

but these examples are not annotated with boxes. Compared with our method, both approaches do not require bounding box annotations. Our method localizes arbitrary objects from a few support images only, while weakly-supervised localization requires many examples per class from a pre-defined vocabulary [83].

Object co-detection. More closely connected to the task of few-shot common-localization is object co-detection [12, 85, 91]. Given two images with the same object, the goal of co-detection is to localize the common instance in both two images. This task differs from few-shot common-localization in two aspects. First, co-detection can only handle the scenario with two input images, while we can handle more inputs. Second, our task evaluates few-shots from previously unseen classes, while co-detection uses the same classes for training and evaluation.

Few shot learning. A central task in few-shot learning is global classification [73, 133, 215, 239, 257, 266]. Approaches such as deep siamese networks [133], matching networks [239], and prototypical networks [215] aim to solve this task by learning embedding spaces. The work of Garcia *et al.* [73] leverages graph convolutional networks [132] for few-shot, semi-supervised, and active learning. We are inspired by the success of graph convolutional networks in few-shot settings and incorporate them in the context of common-localization from few examples.

A number of works have investigated few-shot learning beyond classification [38, 56, 109, 122, 123, 167, 168, 192, 207]. HU *et al.* [109] propose a model for image segmentation from few examples. While effective, this work requires dense pixel-wise annotations for the support images, same as [207]. In this work, we relax this constraint by localization without any spatial annotations. Dong *et al.* [56] study object detection using a large pool of unlabeled images and only a few labeled images per category. Pseudo-labels for the unlabeled images are utilized to iteratively refine the detection result. Akin to HU *et al.* [109], Dong *et al.* rely on spatial annotations for the support examples, while we do not utilize any box annotations for our few examples. Chen *et al.* [38] construct a target-domain detector from few target training annotations by leveraging rich source-domain knowledge. Different from their work, our method tries to solve this problem by utilizing weak prior information of common object existence.

Recently, object detection and segmentation have been investigated from a zero-shot perspective [11, 74, 256]. While promising, the results are not yet at the level of supervised tasks, hence we do not compare to zero-shot approaches.

3.3 METHOD

3.3.1 *Problem Formulation*

For our task of few-shot common-localization, the goal is to learn a model $f(S_c^N, Q_c)$ that, when given a support image set S_c^N of N images and query image Q_c, predicts bounding boxes for class c. The function $f(\cdot)$ is parameterized by a deep network containing a support branch and a query branch. During training, the algorithm has access to a set of image tuples $T = (S_c^N, Q_c)$, where $c \in L_{train}$. At testing, we focus on new (unseen) semantic classes, i.e. $c \in L_{test}$ and $L_{train} \cap L_{test} = \emptyset$.

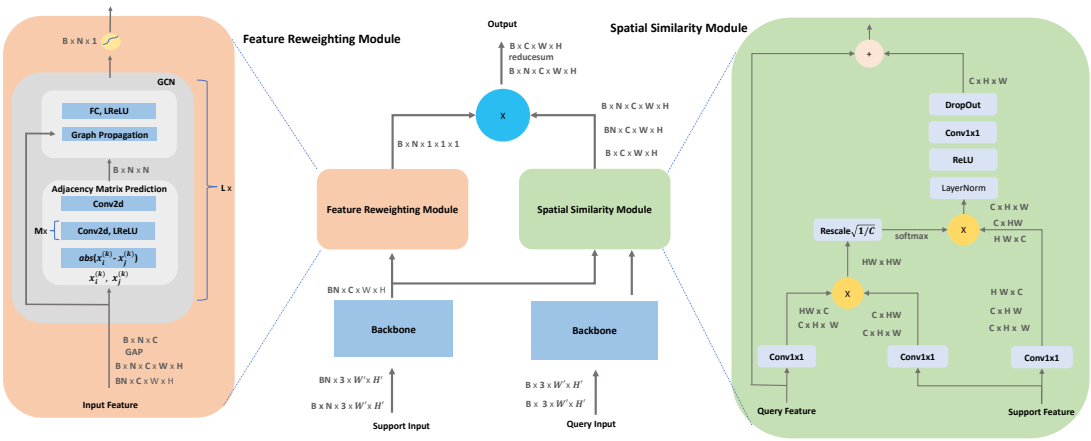


Figure 12: **Overview of the spatial similarity module, feature reweighting module, and the aggregation.** The feature maps are shown as the shape of their tensors . B, N, C denotes batch size, number of support images, and number of channels respectively. The image size is $W^{'} \times H^{'}$, the size after the backbone is $W \times H$. GAP denotes Global Average Pooling. L denotes the number of GCN blocks. M denotes the number of Conv2d-LReLU combinations. Graph Propagation means multiplication between vertex feature and graph adjacency matrix.

3.3.2 *SILCO Network*

For the problem of few-shot common-localization, we propose the SILCO (**S**how a Few **I**mages, **L**ocalize the **C**ommon **O**bject) network. An overview of our approach is shown in Figure 11. Our framework starts from the Single Shot Detector (SSD) architecture [153], using VGG [213] as our backbone. Motivated by multi-scale fusion [149], different scales of features are used to deal with different scales of bounding boxes. The SILCO network facilitates the query image to help the few shot co-localization based for the common class given the weakly-supervised support images at different scales, the key function is to try to integrate support feature and query feature, *i.e.*:

$$\tilde{q}_i = \phi(q_i, S_i), \tag{3.1}$$

for query feature q_i and support feature S_i at scale i. In total, akin to SSD [153], our network contains five scales. Output $\tilde{q}_i \in \mathbb{R}^{B \times C \times W \times H}$ denotes the result of combining the query and support branches. Multiple scales are utilized to help the support branch guide the query branch. The final prediction of SILCO Network is as follows:

$$f(q, S) = \mathrm{DET}(\mathrm{CONCAT}_{i \in \mathcal{S}}(\tilde{q}_i)), \tag{3.2}$$

where CONCAT means concatenation along channel axis, DET is the final detection module used for classification and localization, and $\mathcal{S}$ denotes the set of scales.

There are three choices for function ϕ in our network. We first present a basic way to perform few-shot common-localization with this network. Then we introduce two modules to best leverage the few weakly-supervised support images for common-localization.

A basic version: Global Average Pooling. The common object may exist in different zones in every support image. Therefore, a starting point in the SILCO network is to only consider the channel support and remove spatial information, *i.e.*:

$$\phi(q_i, S_i) = q_i + \frac{1}{N}\sum_{j=1}^{N} \mathrm{GAP}(S_i^{(j)}) \tag{3.3}$$

where GAP denotes global average pooling to remove spatial information, enabling us to directly obtain a representation for localization. Auto Broadcasting is conducted when shape is different. However, this setup does not fully leverage the few support examples we have been given. Therefore, we introduce two new modules.

Spatial Similarity Module. Building upon our starting network and inspired by recent success of the Transformer structure in language processing [236] and non-local blocks [247], we have designed a spatial similarity module, depicted on the right of Figure 12. The main goal of this module is to search for spatial support between the support images and the query image.

The inputs of the spatial similarity module are the query and support features. The outputs are spatially enhanced query features. We investigate two ways to perform spatial similarity. For the first one, the spatial similarity calculates the inner product of the features from support and query branch first, after which softmax is applied to formulate a pixel-wise attention matrix. This matrix then is multiplied with the support features in order to enforce a spatial similarity search between support image and query image. The overall process of this spatial similarity is formulated as:

$$\begin{aligned} \mathrm{SSM}_{im}^{j}(q_i, S_i) =& c_1(soft(c_2(q_i)^T \times c_3(S_i^{(j)})) \\ & \times c_4(S_i^{(j)})) + q_i, \end{aligned} \tag{3.4}$$

where c_1, c_2, c_3, c_4 are convolutional layers, $soft$ means softmax activation, and $\times$ denotes matrix multiplication, $\mathrm{SSM}_{im}^{j}(q_i, S_i)$ denotes the spatial similarity between query image q_i and j-th support image $S_i^{(j)}$ at scale i. For simplicity, some normalization operators such as Dropout [222], Rescaling, Layer Normalization [10] are ignored here. More details of the spatial similarity can be observed in Figure 12. For the final spatial similarity, we can simply take the average over the enhanced features from all support images:

$$\mathrm{SSM}_{im}(q_i, S_i) = \frac{1}{N}\sum_{j=1}^{N} \mathrm{SSM}_{im}^{j}(q_i, S_i), \tag{3.5}$$

where N is the number of support images.

The first image-wise spatial similarity is performed for each support image separately. Another choice of spatial similarity calculation is to consider **all** support images at once (dubbed "global spatial similarity" throughout this work). The global spatial similarity is given as:

$$\begin{aligned}\mathrm{SSM}_g(q_i, S_i) =& c_1(soft(c_2(q_i)^T \times c_3(S_c)_i) \\ & \times c_4(S_c)_i) + q_i,\end{aligned} \tag{3.6}$$

where S_c is the concatenation of all support features. Global spatial similarity considers all support features at once and tries to search for the spatial similarity accordingly. The final formulation of ϕ is given as:

$$\phi(q_i, S_i) = \mathrm{SSM}_{im/g}(q_i, S_i), \tag{3.7}$$

where $\mathrm{SSM}_{im/g}(q_i, S_i)$ denotes the choice for image-wise or global spatial similarity.

Feature Reweighting Module. The spatial similarity module incorporates spatial commonality between support and query images. It assumes that each support image is equally informative for common-localization. Here, we propose a feature reweighting module that reweights the influence of examples in the support branch by interpreting the few-shot images as a connected graph. The weights of this graph are learned through graph convolutional networks (GCNs). The overall structure of the feature reweighting module is demonstrated in Figure 12.

The input of the module are the features of the support images, the output is the weight of each support image. The structure of the module is formulated by a GCN. First we detail how to calculate the weight per support example, then we detail how to conduct the feature reweighting.

Support weights. A GCN is typically fed with an input signal $x \in \mathbb{R}^{N\times d}$ on the vertices of a weighted graph G. We consider an operator family $\mathcal{A}$ of graph intrinsic linear operators that act locally on this signal. The simplest is the *adjacency operator* A. Motivated by ResNet [95], the identity operator is also applied as a form of skip connection in long-ranges. Therefore, we opt for the operator family $\mathcal{A} = \{A, \mathbf{1}\}$ in our work. A GCN receives a feature input $x^{(k)} \in \mathbb{R}^{N\times d_k}$ and produces $x^{(k+1)} \in \mathbb{R}^{N\times d_{k+1}}$, which can be formulated as:

$$x_l^{(k+1)} = gcn(\cdot) = \rho(\sum_{F\in\mathcal{A}} Fx^{(k)}\theta_{F,l}^{(k)}), l = d_1, ..., d_{k+1}, \tag{3.8}$$

where $\Theta = \theta_1^{(k)}, ..., \theta_{|\mathcal{A}|_k}^{(k)}, \theta_{F,l}^{(k)} \in \mathbb{R}^{d_k\times d_{k+1}}$, are trainable parameters and $\rho(\cdot)$ is a point-wise non-linearity, LReLU [261] in our work. Furthermore, the graph adjacency matrix in *adjacency operator* can also be learned from the current node hidden representation [80]:

$$\tilde{A}_{i,j}^{(k)} = \varphi_{\tilde{\theta}}(|\boldsymbol{x}_i^{(k)} - \boldsymbol{x}_j^{(k)}|), \tag{3.9}$$

where φ is a symmetric function that can be parameterized by a neural network, the neural network is stacked after the absolute difference between two vector nodes. To obtain the feature weight, Eq. 3.8 will be cascaded for L times to capture the long-range connection in the graph. In the end, inspired by SENet [104], a sigmoid layer is appended to generate final weight. The detail can be formulated as:

$$\mathrm{FRM}(S) = \sigma(gcn(\cdots gcn(S))), \tag{3.10}$$

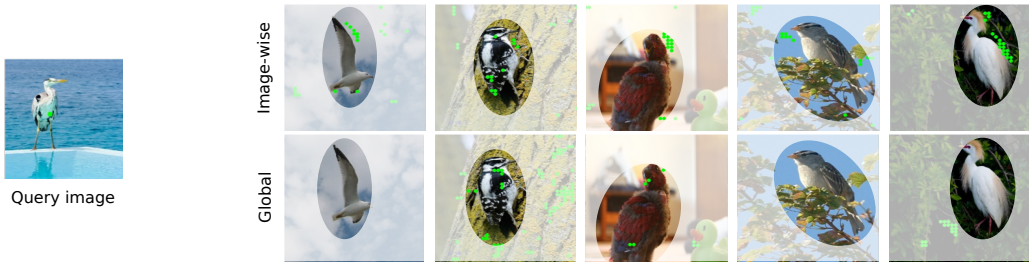


Figure 13: **Spatial similarity module visualization.** Two examples are demonstrated, the left is the query image, the top, bottom images are image-wise similarity visualization and global similarity visualization respectively. For image-wise similarity, the top 20 activations are visualized per image. For global similarity, the top 100 activations are visualized in all 5 images. The green dot in the query image is the reference point. The green dots in the support images are calculated based on the reference point in the query image. Best viewed in color.

where σ is a sigmoid layer, and the output represents feature weight for every support image $\mathrm{FRM}(S) \in \mathbb{R}^{B\times N}$.

Feature Reweighting. To combine the features from the spatial similarity module and the weights from the feature reweighting module, we multiply them both the feature image-wise. In the end, by utilizing the Equation 3.4, ϕ can be further formulated as:

$$\phi(q_i, S_i) = \mathrm{RS}(\mathrm{CONCAT}_{j=1}^{N}(\mathrm{SSM}_{im}^{j}(q_i, S_i)) \otimes \mathrm{FRM}(S_i)), \tag{3.11}$$

where $\mathrm{SSM}_{im}^{j}(q_i, S_i))$ is spatial similarity between query image q_i and j-th support image S_i at scale i, FRM is feature reweighting module, $\otimes$ is hadamard product(broadcasting is ignored if shape mismatches), CONCAT is the concatenation operation, which is a mapping from $\mathbb{R}^{B\times C\times W\times H}$ to $\mathbb{R}^{B\times N\times C\times W\times H}$. The final RS denotes the reduce_sum operation that eliminates the second dimension and leads to $\mathbb{R}^{B\times C\times W\times H}$.

Optimization. Similar to the framework of SSD, our loss function is also composed of a bounding box regression loss and a cross entropy classification loss. The difference is that our classification is class-agnostic, it depends on the common class of the support images and query image. Accordingly, the overall loss function is:

$$L(x, c, l, g) = \frac{1}{BD}\sum_{i,j=1}^{B,D}(\boldsymbol{bce}(c_{ij}, x_{ij}) + \ell_1^s(l_{ij}, g_{ij})), \tag{3.12}$$

where B is the batch size, D is the number of matched default boxes, ***bce*** means binary classification entropy loss function, ℓ_1^s denotes smoothed ℓ_1 norm loss function [81]. c_{ij},x_{ij}, l_{ij} , g_{ij} are the class probability, class ground truth, predicted coordinate, ground truth coordinate of i-th image, j-th bounding box proposal, respectively.

3.4 EXPERIMENTAL SETUP

3.4.1 *Common-localization Datasets*

To accompany the new task of few-shot common-localization, we have prepared a revised setup for three well-known datasets intended for object detection, namely Pascal VOC [65], MS-COCO [150], and ImageNet VID [53].

CL-VOC. We divide the 20 classes of PASCAL VOC into two disjoint groups, one group is used for training, the other for validation/testing. We use both groups for both tasks and report the mean performance of the two runs. We perform experiments both on Pascal VOC 2007 and 2012, dubbed CL-VOC-07 and CL-VOC-12 respectively. The training set D_{train} is composed of all image pairs from the PASCAL VOC training set that include one common class from the label-set L_{train}. The validation set D_{val} and test set D_{test} are both from the PASCAL VOC validation set. For a detailed explanation of our dataset organization procedure, please refer to the supplementary materials.

CL-COCO. We furthermore recompile a common-localization dataset based on the MS-COCO 2014 dataset [150]. The 80 classes in MS-COCO are divided into two disjoint groups. The classes in each group are provided in the supplementary materials.

CL-VID. To evaluate a generalization to videos, we employ the ImageNet VID dataset [53], a benchmark for video object detection. We use the 3,862 video snippets from the training set for evaluation, which includes 30 objects. We employ this dataset to evaluate our approach on open-set (*i.e.*, unseen) classes. We train our model on CL-VOC-12. There are some overlapping classes between Pascal VOC and ImageNet VID. We keep videos which have one target class and no overlap with any Pascal VOC class. For details on the retained classes, please refer to the supplementary materials. The support images are selected from ImageNet DET [53] for evaluation. Each frame of a test video acts as query image.

3.4.2 *Implementation Details*

We use PyTorch [182] for implementation. The network is trained with SGD [25] with a learning rate of 1e-4 and momentum of 0.99 on one Nvidia GTX 1080TI. The weights of the support and query branch are pre-trained on ImageNet [53]. All the images in the support and query branch are resized to 300×300 and the batch size is set to 6. For the query branch we choose photo-metric distortion, random mirror, random sample crop, akin to SSD [153]. M is set to 2 in the feature reweighting. Considering the long-range connection in graphs, L is always equal to the number of support images.

3.4.3 *Evaluation*

For the training tuples, we randomly sample tuples $T = (S_c^N, Q_c)$, such that all tuples contain the common classes $c \in L_{train}$. For evaluation, we randomly sample several tuples $T = (S_c^N, Q_c)$, which contain the common class $c \in L_{test}$. We evaluate on 5000 tuples in CL-VOC and 10000 tuples in CL-COCO. Our training, validation, and test images are always disjunct. The object classes in training are disjunct from those in

Table 8: **Spatial similarity module.** Mean average precision (%) for image-wise versus global spatial similarity on CL-VOC-12. For both groups, image-wise similarity works better and we will use this form of spatial similarity for further experiments.

	Group 1	Group 2	mean
Global	51.71	55.49	53.60
Image-wise	**54.04**	**57.39**	**55.71**

validation/test. The hyperparameter search is done once on Group 1 of CL-VOC-12. We use the same hyperparameters for all experiments on CL-VOC-07, CL-VOC-12, CL-COCO, and CL-VID. On the respective validation sets we choose the best model.

We employ the (mean) Average Precision as evaluation measure throughout our experiments. The overall mAP is averaged on the mAPs of the two groups and computed using the setup of [65]. For evaluation we only consider the top 200 detected bounding boxes, and rank these boxes according to their objectness score. Each prediction that overlaps with the closest ground truth with a value of at least 0.5 will be regarded as a positive detection. After that, a non-maximum suppression with a threshold of 0.45 is applied.

3.5 EXPERIMENTAL RESULTS

3.5.1 *Ablation Study*

Spatial similarity module. In the spatial similarity module, there are two ways to relate features from the support and query branches. The first, image-wise spatial similarity, computes a matrix of size $HW \times HW$ for each support image. The second, global spatial similarity, computes a single matrix of size $HW \times NHW$, which regards all N support images as a whole to the spatial similarity. We compare the two different forms of similarities in Table 8. We observe that image-wise spatial similarity outperforms global spatial similarity. Our hypothesis is that image-wise spatial similarity more explicitly exploits the prior knowledge of the common-localization task that all support images are of the same class. To highlight this ability of image-wise spatial similarity, we visualize the top activation pixels in Figure 13. We find that image-wise spatial similarity balances the attention of every support image, while global spatial similarity exhibits a less uniform attention distribution. For the bird example, the global similarity misses the common object in the first and fourth support image, while many irrelevant areas in the second support images are targeted. Based on this study, image-wise spatial similarity will be adopted for the rest of the experiments.

Feature reweighting module. We also explore the effect of feature reweighting based on the previous results. The ablation result is indicated in Table 9. We first observe that spatial similarity outperforms the global average pooling baseline, further validating its effectiveness. Across all three datasets, adding feature reweighting on top of the spatial similarity benefits the common-localization accuracy. To better understand the inner mechanism of feature reweighting, we visualize the feature heatmaps before and after

Table 9: **Ablation of spatial similarity and feature reweighting.** The metric is mean average precision (%). As we adopt spatial similarity (SSM) and feature reweighting (FRM) the accuracy gradually increases over a simple global average pooling (GAP), indicating the effectiveness of our proposed modules.

dataset	GAP	SSM	FRM	Group 1	Group 2	mean
CL-VOC-07	✓			55.17	52.18	53.67
		✓		56.12	55.52	55.82
		✓	✓	**57.17**	**56.45**	**56.82**
CL-VOC-12	✓			53.55	54.61	54.08
		✓		54.04	57.39	55.71
		✓	✓	**55.11**	**58.62**	**56.86**
CL-COCO	✓			13.37	6.50	9.94
		✓		18.50	7.70	13.10
		✓	✓	**18.62**	**8.20**	**13.40**

Figure 14: **Feature reweighting** heatmap visualization (blue means low, red means high). The first row shows the support images, the second row shows the query image with ground truth (blue box) and prediction (red box), the bottom row shows the heatmap visualization. The heatmaps are the normalized feature map selected from the output of our network. For both examples, the left and right columns show the results with and without feature reweighting. The heatmap results show that feature reweighting can better highlight the areas containing the common object. Label information is only used for illustration here, we don't utilize them in experiment.

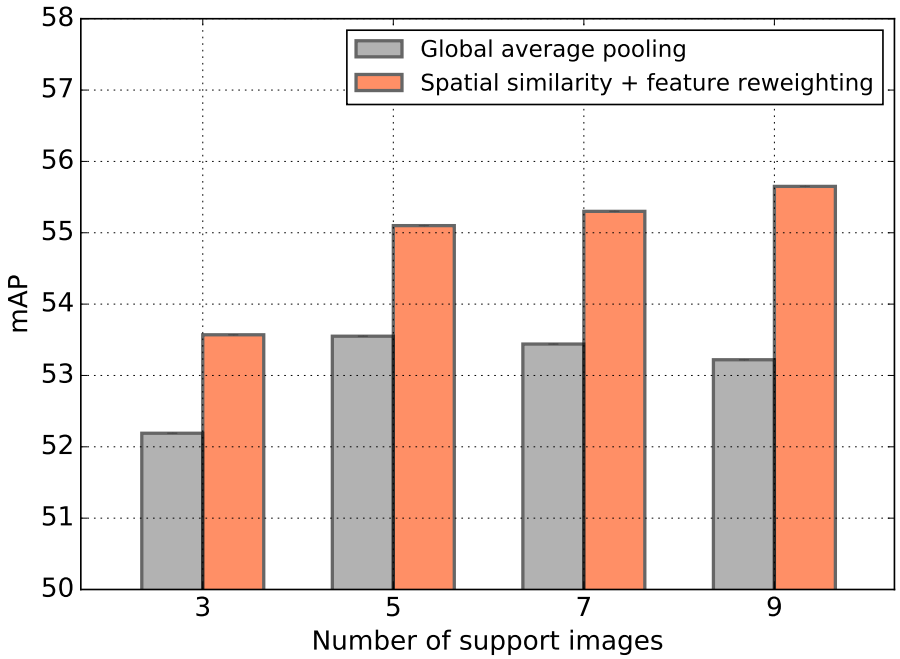


Figure 15: **Effect of support images** on CL-VOC-12. For better visualization results, our y-axis starts from 50% mAP. Compared to a global average pooling (GAP) baseline, our spatial similarity (SSM) and feature reweighting modules (FRM) already works well with little support, and our modules capture the common object even better when support increases.

feature reweighting in Figure 14. The figure shows that the reweighted features better focus on the common class to further enhance the common-localization.

Effect of support images. Our common-localization is optimized to work with few examples as support. To show this capability, we have explored the effect of gradually increasing the number of support images in Figure 15. We have evaluated with 3, 5, 7, and 9 support images. The results show that our approach obtains high accuracy with only a few support images. As the number of support images increases, the gap of our approach with and without spatial similarity and feature reweighting gradually becomes larger, which indicates that our modules can capture the common object even better when the support set grows. We have also investigated the ability to localize more than one common object and we show qualitative examples in the supplementary materials.

Support image corruption. Our approach even works when some of the support images do not contain the common object. We did an experiment for the 5-shot setting, where we insert a corrupted support image. On CL-VOC-12 the mAP drops from 56.86 to 56.53 and on CL-COCO from 13.40 to 13.03.

Effect of object size. We explore the effect of different object sizes on CL-VOC-12 in Table 10. The small, medium, large object are defined as area ratio per image ranging from [0, 0.15],[0.15, 0.3],[0.3, 1]. We observe most gain for medium-sized objects, while we observe a gain for all settings. Localizing large objects may be easier, so the gain is modest, explaining their relatively modest improvement.

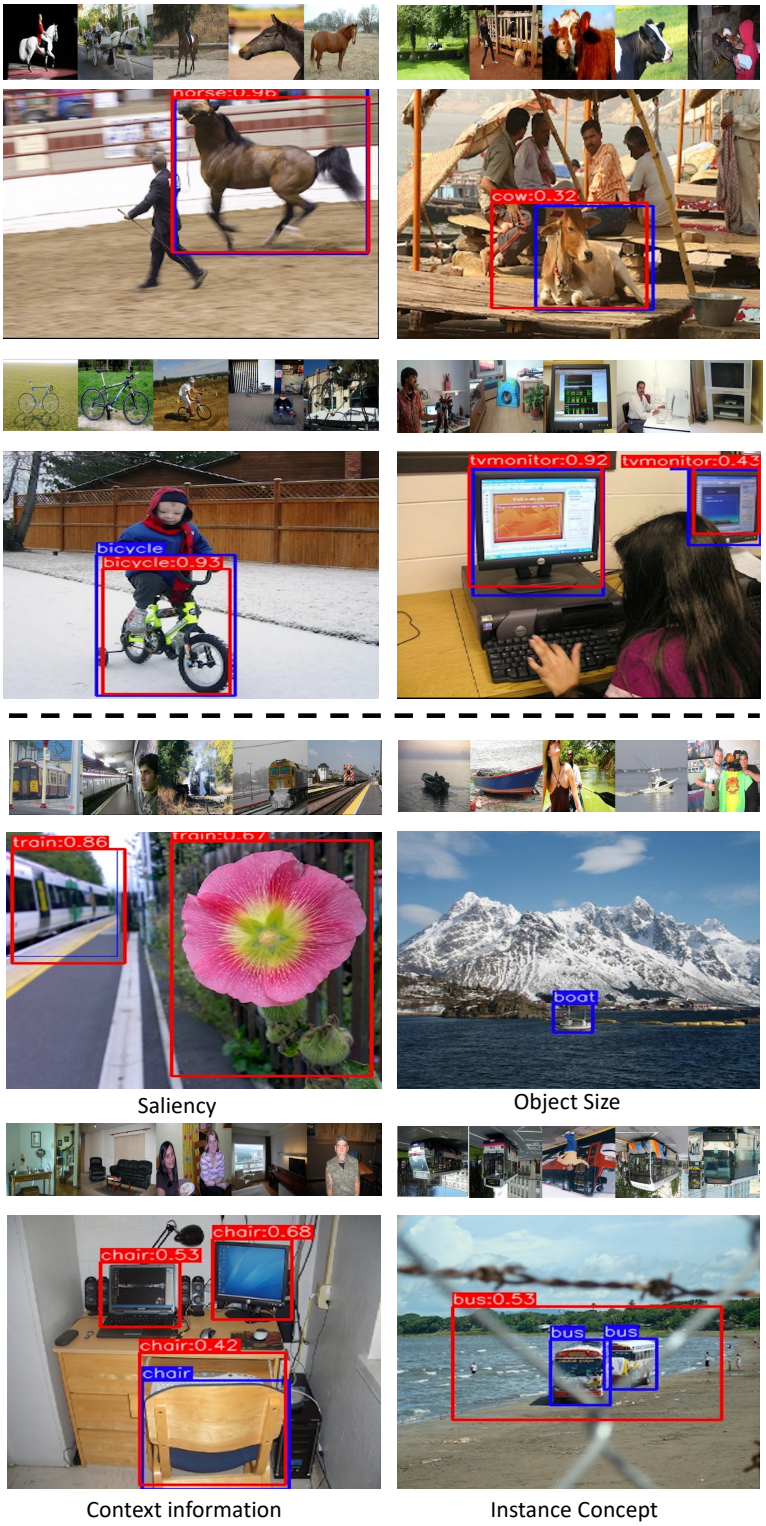


Figure 16: **Success and failure case analysis** on CL-VOC-12. Top two rows: four success cases. Bottom two rows: four common failure cases, where confusion is caused by saliency, object size, lack of context, or instance confusion. For each case, we present 5 support images in the top and the query image in the bottom. Blue indicates ground truth, red means prediction. Label information is only used for illustration here, we don't utilize them in experiment.

Table 10: **Effect of object size** on CL-VOC-12 in mAP(%). Compared to a global average pooling (GAP) baseline, our spatial similarity (SSM) and feature reweighting modules (FRM) especially improves for medium-sized objects.

method	small	medium	large
GAP	9.60	10.87	28.64
SSM, FRM	**10.99**	**13.85**	**29.24**

Table 11: **Comparative evaluation** on CL-VOC-07, CL-VOC-12, and CL-COCO. Across all datasets, our approach outperforms the center box and Region Proposal Network (RPN) [198] baselines by a large margin. Furthermore, our Spatial Similarity (SSM) and Feature Reweighting (FRM) modules are preferred over the ConvLSTM of HU *et al.* [109] for few-shot common-localization.

	CL-VOC-07			**CL-VOC-12**			**CL-COCO**			**CL-VID**
	group 1	group 2	mean	group 1	group 2	mean	group 1	group 2	mean	mean
Center box	5.28	4.05	4.67	6.56	6.90	6.73	0.54	0.51	0.53	-
RPN [198]	18.72	15.13	16.93	20.23	18.54	19.39	3.20	1.93	2.57	-
Contrastive RPN [87]	18.89	15.33	17.21	19.21	18.53	18.87	3.54	2.01	2.77	-
This chapter										
w/ ConvLSTM [109]	53.46	54.90	54.18	51.34	57.42	54.38	16.37	6.72	11.54	53.02
w/ SSM and FRM	**57.17**	**56.10**	**56.64**	**55.12**	**58.62**	**56.87**	**18.62**	**7.47**	**13.04**	**54.40**

Success and failure case analysis. Figure 16 shows that our method can perform common-localization in complex query images, which contain multiple objects. The right example of row two shows that our method even works well when multiple instances exist in a single query image. We also observe several failure cases: 1). Saliency. The most salient object is often mistaken as the true positive. 2). Object size. Our method fails to localize the object that is extremely small. 3). Context information. Our method doesn't consider the context information, for example in this case, that a chair is unlikely to be on a table. 4). Instance concept. Our method may fail if there exists no clear boundary between instances. These all provide interesting avenues for future enhancement of common-localization.

3.5.2 *Comparative Evaluation*

Baselines. We first compare to baselines using a fixed center box or a Region Proposal Network(RPN) [198]. The center box baseline serves as a sanity check to understand the complexity of the few-shot common-localization task. The RPN serves as a state-of-the-art comparison to standard object detection. For the center box baseline, we simply select the center box of the query image as the final object proposal. The optimal size of the center box is determined through grid search per dataset. For the RPN, we first train a class-agnostic RPN, after which we extract both the ROI scores and features from the query image and support images. Second, we generate candidate support features by choosing the ROI with the highest score per support image. Third, we match candidate support features and the query ROI features according to L2 distance. The query ROI with the lowest feature distance is used as the final proposal.

We also extend the RPN baseline by adding a Siamese Network constrained by a contrastive loss [87] to learn a discriminative distance metric. To obtain pairs during training, we sample ground truth boxes with a 1:1 ratio. A distance margin of 0.5 is chosen by cross-validation. The remaining process is the same as the RPN baseline.

To evaluate the spatial similarity and feature reweighting modules, we extend our base approach with the ConvLSTM of HU *et al.* [109], previously used for few-shot image segmentation. ConvLSTM is adopted between support and query features. Because this baseline is GPU-inefficient, we scale down the number of channels to half the original size through a 1×1 convolution. After the ConvLSTM fusion, we scale up the channel number to the original number by another 1×1 convolution.

Results. The results are shown in Table 11. The center box baseline scores lowest across all datasets, indicating that this task is not easy to solve. For the RPN baselines, we obtain a stable gain compared to the center box baseline, illustrating that deep features and their similarity have an important role in our task. Our method, either with ConvLSTM [109] or the proposed spatial similarity and feature reweighting modules outperforms the center box and RPN baseline, which shows that our common-localization structure is more suitable for the few-shot common-localization task. Furthermore, our method with the combination of spatial similarity and feature reweighting works best compared to all other methods in three datasets, which indicates the spatial similarity and feature reweighting modules can function well in a mutually beneficial way.

Time and complexity analysis. SILCO has 37.1M parameters and an inference time of 0.12 seconds with 5 shots. For comparison, ConvLSTM [109] has 56.3M parameters, while the inference time is a bit faster with 0.09 seconds.

3.5.3 *Video Common-localization*

To validate generalization abilities, we also evaluate on video detection dataset ImageNet VID [53]. We use randomly selected images from ImageNet DET as support images, and every frame of a video from VID as query image. The results in Table 11 again confirm our method obtains consistent gains by incorporating spatial similarity and feature reweighting.

3.6 CONCLUSION

This chapter introduces the task of few-shot common-localization. From a few support images without box annotations, we aim to localize the object in the query image that is common among all images. To that end, we introduce a network specific to this problem and propose two modules to improve the common-localization. The first module enhances the spatial similarity among the support and query images. The second module balances the influence of each support image and reweights the features from the spatial similarity accordingly. Experiments show that our approach can robustly localize the common objects from few examples, outperforming baselines from related fields. We see this work as a first step into localized learning from double-weak supervision, where examples are both scarce and without box annotations.

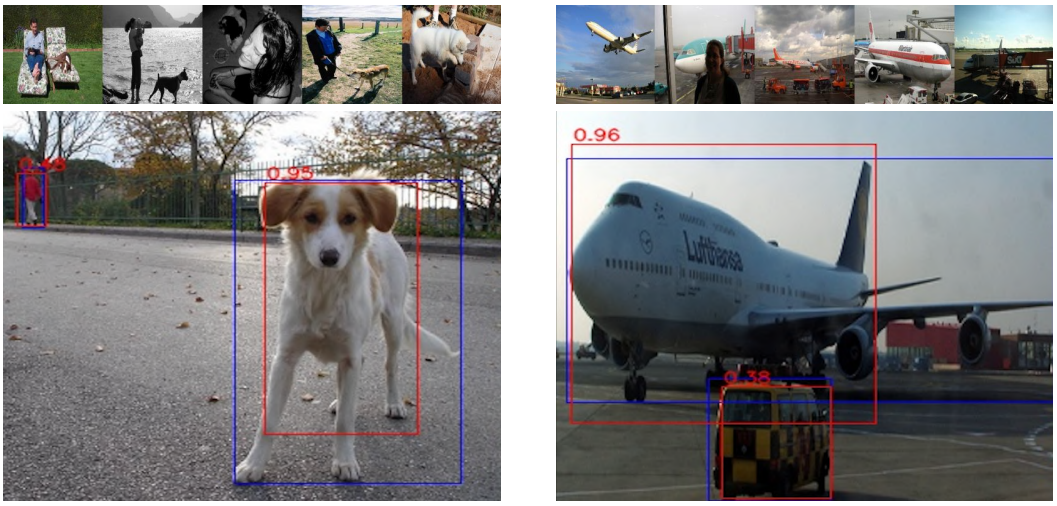


Figure 17: **Localization from support images with multiple objects.** Blue denotes ground truth, red denotes prediction. The two objects common in all support images are localized correctly in the query images.

Table 12: **The #image** of CL-VOC 07, CL-VOC 12, CL-COCO Few-Shot Common Localization Dataset.

dataset	train	val	test
CL-VOC07	2501	1010	1500
CL-VOC12	5717	2623	3200
CL-COCO	62783	20000	40504

3.7 APPENDIX

3.7.1 *Multiple Common Object*

We also assess the ability to localize more than one common object. We have re-trained SILCO on images that also include multiple objects. Figure 17 shows two examples. In both cases, we detect the two objects that appear in all five support images. This further indicates the practical use of SILCO.

3.7.2 *Dataset Details*

We mainly elaborate the image numbers of CL-VOC07, CL-VOC12, CL-COCO in Table 12, the detail classes of all datasets is shown in Table 14, Table 15.

For experiment in VID, we only keeps the classes: bear, elephant, fox, giant panda, hamster, lion, lizard, monkey, rabbit, red panda, snake, squirrel, tiger, turtle, whale.

The data splitting principle of different subset is shown in Figure 18. For train,validation and test, the images are isolated with each other. When training, we random sample tuples(1+N for N-shot common-localization) from the pool.The detail algorithm is shown in Algorithm 1.

Table 13: **The comparison between "before channel attention" and "after channel attention".** A typical channel attention method SENet [104] is deployed in our experiment.

dataset	method	Group 1	Group 2	mAP(%)
CL-VOC07	before	57.17	56.45	56.82
	after	**57.50**	**56.96**	**57.23**
CL-VOC12	before	55.11	58.62	56.86
	after	**56.55**	**59.22**	**57.89**

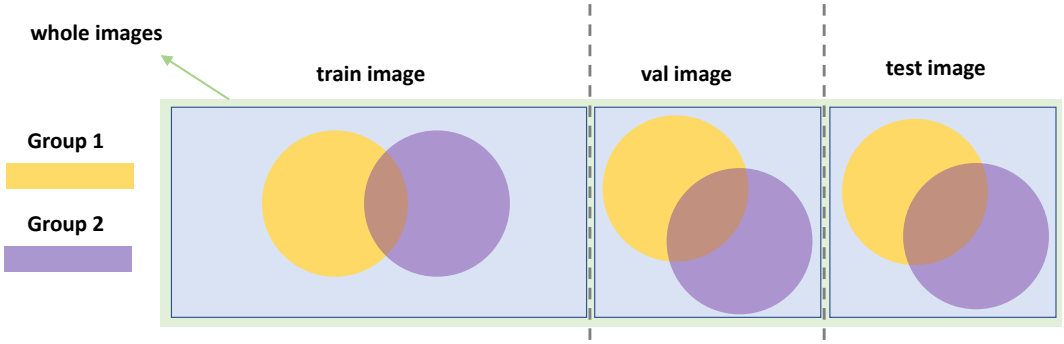


Figure 18: **The dataset splitting demonstration.** The total data are composed of train image, validation image, test image. Our labels are separate between different images.

3.7.3 *Another Perspective of the Method*

From the aspect of space, our spatial similarity can be viewed as a kind of ***spatial*** attention mechanism between query feature and support feature. Furthermore, from the perspective of image, feature reweighting tries to rescale and balance the influence of different support ***images***. A natural question is whether we can introduce ***channel***-level attention into our framework to reach a ***spatial, channel, image***-level attention. Therefore, we try to adopt SENet [104] upon our framework and obtain the result in Table 13. It indicates that the performance can be further improved by channel attention. The result demonstrates that ***spatial, channel, image***-level attention mechanism can be combined in a mutually beneficial way.

Table 14: **The class grouping for CL-VOC.** One group is used to initialize the model, the other group is used to perform the few-shot common-localization.

Group	Semantic Classes
1	aeroplane, bicycle, bird, boat bottle, bus, car, cat, chair, cow
2	dining table, dog, horse, motorbike person, potted plant, sheep, sofa train, tv/monitor

Table 15: **The class grouping for CL-COCO.** One group is used to initialize the model, the other group is used to perform the few-shot common-localization.

Group	Semantic Classes
1	person, bicycle, car, motorcycle, airplane,bus, train, truck, boat, traffic light, fire hydrant, stop sign , parking meter, bench, bird, cat, dog, horse, sheep, cow , elephant, bear, zebra, giraffe, backpack, umbrella, handbag, tie , suitcase, frisbee, skis, snowboard, sports ball, kite, baseballbat, baseball glove, skateboard, surfboard, tennis racket bottle
2	wine glass, cup, fork, knife, spoon, bowl, banana, apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake, chair, couch, potted plant, bed, dining table, toilet, tv, laptop, mouse, remote, keyboard, cell phone, microwave, oven, toaster, sink, refrigerator, book, clock, vase, scissors, teddy bear, hair drier, toothbrush

Algorithm 1 Dataset Preparation Algorithm

Notations:
C: target class set
$\mathcal{D}_i$: the i-th image in the dataset $\mathcal{D}$
d: dictionary for (label, image_list) pairs

for $\mathcal{D}_i$ in $\mathcal{D}$ **do**
  for C_j in C **do**
    if $\mathcal{D}_i$ has the class of C_j and
        d has key C_j **then**
      d [C_j] .append($\mathcal{D}_j$)
    else
      d [C_j] = [$\mathcal{D}_j$]
    end if
  end for
end for
while Not Stop **do**
  random choose 1 item $\tilde{C}$ from C
  random choose k+1 items $\mathcal{R}$ from $d[\tilde{C}]$
  yield $\mathcal{R}$
end while

4

LOCALIZING THE COMMON ACTION AMONG A FEW VIDEOS

4.1 INTRODUCTION

The goal of this chapter is to localize the temporal extent of an action in a long untrimmed video. This challenging problem [61, 179] has witnessed considerable progress thanks to deep learning solutions, *e.g.* [71, 148, 211], fueled by the availability of large-scale video datasets containing the start, the end, and the class of the action [29, 52, 112]. Recently, weakly-supervised alternatives have appeared, *e.g.* [22, 115, 137, 138, 176, 183, 244, 268]. They avoid the need for hard to obtain start and end time annotations, but still require hundreds of videos labeled with their action class. In this chapter, we also aim for a weakly-supervised setup, but we avoid the need for any action class labels. We propose few-shot *common* action localization, which determines the start and end of an action in a long untrimmed video based on just a hand-full of trimmed videos containing the same action, without knowing their common class label.

We are inspired by recent works on few-shot object detection [56, 107, 205, 208]. Dong *et al.* [56] start from a few labeled boxes per object and a large pool of unlabeled images. Pseudo-labels for the unlabeled images are utilized to iteratively refine the object detection result. Both Shaban *et al.* [208] and Hu *et al.* [107] further relax the labeling constraint by only requiring a few examples to contain a common object, without the strict need to know their class name. Hu *et al.* [107] introduce two modules to reweigh the influence of each example and to leverage spatial similarity between support and query images. We also require that our few examples contain a common class and we adopt a reweighting module. Different from Hu *et al.*, we have no module to focus on masking objects spatially in images. Instead, we introduce three alternative modules optimized for localizing actions temporally in long untrimmed videos, as illustrated in Figure 19.

We make three contributions in this work. First, we consider common action localization from the few-shot perspective. All we require is that the few trimmed video examples share a common action, which may be obtained from social tags, hash tags or off-the-shelve action classifiers. Second, we propose a network architecture for few-shot common action localization, along with three modules able to align representations from the support videos with the relevant query video segments. The mutual enhancement

Published in Proceedings of the European Conference on Computer Vision, 2020 [270].

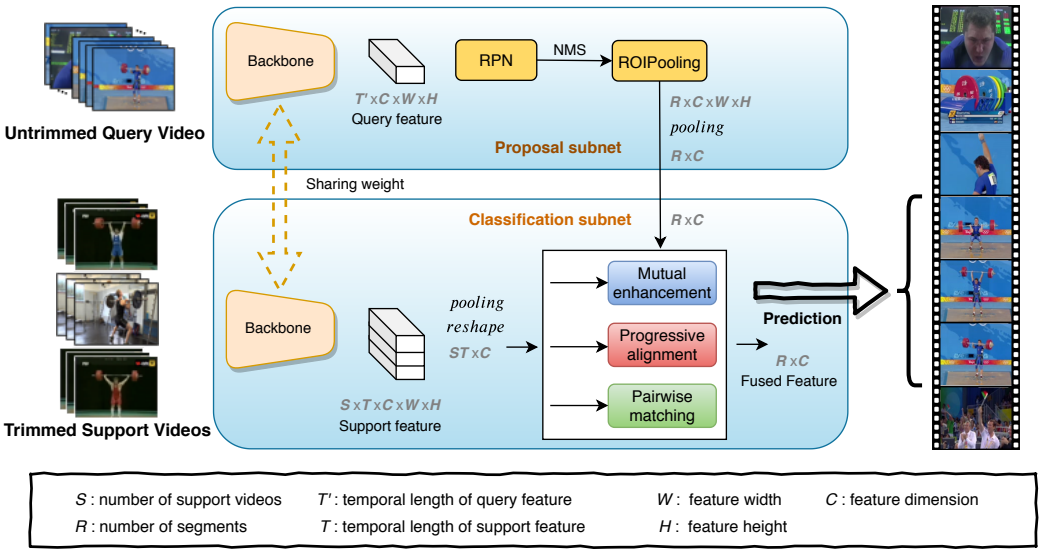


Figure 19: **Common action localization** in an untrimmed query video from three trimmed support videos during inference. The action is localized in the query video based on the common action in the support videos.

module strengthens the representations of the query and support representations simultaneously by building upon non-local blocks [247]. The progressive alignment module iteratively integrates the support branch into the query branch. Lastly, the pairwise matching module learns to weigh the importance of different support videos. As a third contribution, we reorganize the videos in ActivityNet1.3 [29] and Thumos14 [112] to allow for experimental evaluation of few-shot common action localization in long untrimmed videos containing a single or multiple action instances.

4.2 RELATED WORK

Action localization from many examples. Standard action localization is concerned with finding the start and end times of actions in videos from many training videos with labeled temporal boundaries [27, 62, 211]. A common approach is to employ sliding windows to generate segments and subsequently classify them with action classifiers [51, 70, 211, 243, 267]. Due to the computational cost of sliding windows, several approaches model the temporal evolution of actions and predict an action label at each time step [62, 159, 214, 272]. The R-C3D action localization pipeline [262] encodes the frames with fully-convolutional 3D filters, generates action proposals, then classifies and refines them. In this chapter, we adopt the proposal subnet of R-C3D to obtain class-agnostic action proposals. In weakly-supervised localization, the models are learned from training videos without temporal annotations. They only rely on the global action class labels [175, 176, 244]. Different from both standard and weakly-supervised action localization, our *common* action localization focuses on finding the common action in a long untrimmed query video given a few (or just one) trimmed support videos without knowing the common action class label, making our task class-agnostic. Furthermore, the videos used to train our approach contain actions that are not seen during testing.

Action localization from few examples. Yang *et al.* [267] pioneered few-shot *labeled* action localization, where a few (or at least one) positive labeled and several negative labeled videos steer the localization via an end-to-end meta-learning strategy. It relies on sliding windows to swipe over the untrimmed query video to generate fixed boundary proposals. Rather then relying on a few positive and many negative action class labels, our approach does not require any predefined positive nor negative action labels, all we require is that the few support videos have the same action in *common*. Moreover, we propose a network architecture with three modules that predicts proposals of arbitrary length from commonality only.

Action localization from one example. Closest to our work is video re-localization by Feng *et al.* [67], which introduces localization in an untrimmed query video from a *single* unlabeled support video. They propose a bilinear matching module with gating functions for the localization. Compared to video relocalization, we consider a more general and realistic setting, where more than one support video can be used. Furthermore, we consider untrimmed videos of longer temporal extent and we consider action localization from a single frame. To enable action localization under these challenging settings, we introduce modules that learn to enhance and align the query video with one or more support videos, while furthermore learning to weigh individual support videos. We find that our proposed common action localization formulation obtains better results, both in existing and in our new settings.

Action localization from no examples. Localization has also been investigated from a zero-shot perspective by linking actions to relevant objects [116, 121, 165]. Soomro *et al.* [221] tackle action localization in an unsupervised setting, where no annotations are provided overall. While zero-shot and unsupervised action localization show promise, current approaches are not competitive with (weakly-)supervised alternatives, hence we focus on the few-shot setting.

4.3 METHOD

4.3.1 *Problem Description*

For the task of few-shot common action localization, we are given a set of trimmed support videos S_c^N, where N is small, and an untrimmed query video Q_c. Both the support and query videos contain activity class c, although its label is not provided. The goal is to learn a function $f(S_c^N, Q_c)$ that outputs the temporal segments for activity class c in the query video. The function $f(\cdot,\cdot)$ is parametrized by a deep network consisting of a support and query branch. During training, we have access to a set of support-query tuples $T = \{(S_l^N, Q_l)\}_{l \in \mathcal{L}_{\text{train}}}$. During both validation and testing, we are only given a few trimmed support videos with corresponding long untrimmed query video. The data is divided such that $\mathcal{L}_{\text{train}} \cup \mathcal{L}_{\text{val}} \cup \mathcal{L}_{\text{test}} = \emptyset$.

4.3.2 *Architecture*

We propose an end-to-end network to solve the few-shot common action localization problem. A single query video and a few support videos are fed into the backbone, a C3D

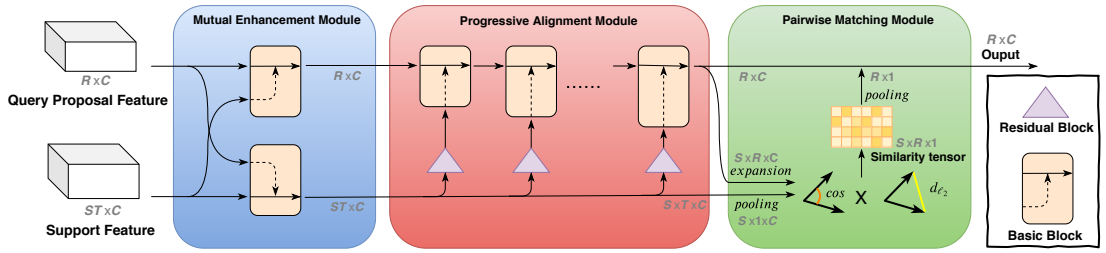


Figure 20: **Modules for aligning representations** from the support videos with the relevant query video segments. The mutual enhancement module augments the support and query representations simultaneously through message passing. Then, the progressive alignment module fuses the support into the query branch through recursive use of the basic block. Finally, the pairwise matching module reweighs the fused features according to the similarity between the enhanced query segments and the enhanced support videos.

network [234], to obtain video representations. The weights of the backbone network are shared between the support and query videos. For the query video, a proposal subnet predicts temporal segments of variable length containing potential activities [262]. Let $\mathcal{F}_Q \in \mathbb{R}^{R \times C}$ denote the feature representation of the query video for R temporal proposal segments, each of dimensionality C. Let $\mathcal{F}_S \in \mathbb{R}^{ST \times C}$ denote the representations of the S support videos, where we split each support video into T fixed temporal parts. The main goal of the network is to align the support representations with the relevant query segment representation:

$$\mathcal{F} = \phi(\mathcal{F}_Q, \mathcal{F}_S), \tag{4.1}$$

In Equation 4.1, $\mathcal{F} \in \mathbb{R}^{R \times C}$ denotes the temporal segment representations after alignment with the support representations through ϕ. In our common localization network, representations $\mathcal{F}$ are fed to fully-connected layers that perform a binary classification to obtain the likelihood that each proposal segment matches with the support actions, which is followed by a temporal regression to refine the activity start- and end-times for all segments.

In our network, we consider the following: *i*) the representations of the support videos need to be aligned with the representations of the activity in the query video, *ii*) not all support videos are equally informative, and *iii*) common action localization is a support-conditioned localization task, where the activityness of different query segments should be guided by the support videos. We propose three modules, namely mutual enhancement module, progressive alignment module, and pairwise matching module to deal with these considerations.

Mutual enhancement module. Building on the recent success of the transformer structure [236] and the non-local block [247], which are forms of self-attention, we propose a module which can simultaneously enhance the representations of the support and query videos from each other. The basic block for this module is given as:

$$m(I_1, I_2) = c_1(soft(c_2(I_1) \times c_3(I_2^T)) \times c_4(I_2)) + I_1, \tag{4.2}$$

where c_1, c_2, c_3, c_4 are fully-connected layers, *soft* denotes the softmax activation, and $\times$ denotes matrix multiplication. I_1 and I_2 denote the two inputs. A detailed overview and illustration of the basic block is provided in the supplementary materials. Based on

the basic block, we design a mutual enhancement module to learn mutually-enforced representations, both for query proposals and support videos, as shown in Figure 20. The mutual enhancement module has two streams $m_{s\to q}$, $m_{q\to s}$ that are responsible for enhancing query proposals and support videos respectively. The inputs to the mutual enhancement module, $\mathcal{F}_Q$ and $\mathcal{F}_S$, will be enhanced by each other:

$$m_{s\to q} = m(\mathcal{F}_Q, \mathcal{F}_S), \tag{4.3}$$

$$m_{q\to s} = m(\mathcal{F}_S, \mathcal{F}_Q). \tag{4.4}$$

Progressive alignment module. We also propose a progressive alignment module to achieve a better fusion of representations from query proposals and support videos. The idea behind this module is to reuse the basic block from the mutual enhancement module to integrate the support branch into the query branch. Inspired by the successful application of residual learning [95, 104], we employ a residual block to make the progressive alignment effective:

$$r(I) = c_1(relu(c_2(I))) + I, \tag{4.5}$$

where c_1, c_2 are fully-connected layers, *relu* denotes the ReLU activation. A detailed overview and illustration of the residual block is provided in the supplementary materials. We first take query proposal representations from the first module $m_{s\to q}$ as 0-depth outcome $\mathcal{P}_0$. On top, we adopt our basic block m to integrate this outcome with $m_{q\to s}$ which has been recalibrated by our residual block r. We perform this operation multiple times in a recursive manner, *i.e.*:

$$\mathcal{P}_0 = m_{s\to q}, \tag{4.6}$$

$$\mathcal{P}_k = m(\mathcal{P}_{k-1}, r(m_{q\to s})), \quad k = 1, 2, \ldots, n. \tag{4.7}$$

where we set $n = 3$ in practice. The advantage of a progressive design is that it strengthens the integration of the support branch into the query branch as we increase the number of basic block iterations. By using the same efficient basic blocks as our first module, the computational overhead is small. An illustration of the progressive alignment module is shown in Figure 20.

Pairwise matching module. In common action localization, a small number of support videos is used. Intuitively, not every support video is equally informative for the query segments. In addition, different query segments should not be treated equally either. To address these intuitions, we add a means to weigh the influence between each support video and each query segment, by introducing a pairwise matching module.

The input for the matching module are all segments of the query video and all support videos. The pair-wise matching is a mapping PMM : $(\mathbb{R}^{R\times C}, \mathbb{R}^{S\times T\times C}) \mapsto \mathbb{R}^{S\times R\times 1}$. To align the two components, we first perform an expansion operation e on the query segments, denoted as $e(\mathcal{P}_n) \in \mathbb{R}^{S\times R\times C}$. Then a pooling p is applied over the support videos along the temporal dimension, denoted as $p(m_{q\to s}) \in \mathbb{R}^{S\times 1\times C}$. Afterwards, we perform an auto broadcasting operation b on $p(m_{q\to s})$, which can broadcast the

dimension of $p(m_{q\to s})$ from $\mathbb{R}^{S\times 1\times C}$ to $\mathbb{R}^{S\times R\times C}$ to align with the dimension of $e(\mathcal{P}_n)$. For query segments $\mathcal{P}_n$ and for support videos $m_{q\to s}$, their match is given by the cosine similarity (cos) and ℓ_2 Euclidean distance (d_{ℓ_2}) along the segment axis:

$$M = cos(\mathcal{P}_n, m_{q\to s}) = \frac{<e(\mathcal{P}_n), b(p(m_{q\to s}))>}{\|e(\mathcal{P}_n)\| \cdot \|b(p(m_{q\to s}))\|}, \tag{4.8}$$

$$N = d_{\ell_2}(\mathcal{P}_n, m_{q\to s}) = \|e(\mathcal{P}_n) - b(p(m_{q\to s}))\|. \tag{4.9}$$

We combine both distance measures:

$$\mathcal{W} = \text{PMM}(\mathcal{P}_n, m_{q\to s}) = M \odot \sigma(-N), \tag{4.10}$$

where σ denotes the Sigmoid operation. Tensor $\mathcal{W} \in \mathbb{R}^{S\times R\times 1}$ can be interpreted as a weight tensor to achieve attention over the R and S dimensions. $\mathcal{W}[i, j]$ is a scalar depicting the similarity between the j-th query segment representation and the i-th support representation. For the j-th query segment representation, $\mathcal{W}[:, j] \in \mathbb{R}^{S\times 1}$ corresponds to the weight for different support videos, while for the i-th support representation, $\mathcal{W}[i, :] \in \mathbb{R}^{R\times 1}$ resembles the weight for different query segments. In the end, we enforce the pairwise matching weight $\mathcal{W}$:

$$\phi(\mathcal{F}_Q, \mathcal{F}_S) = \mathcal{P}_n \odot \text{AP}(\mathcal{W}), \tag{4.11}$$

where AP denotes an average pooling operation along the support dimension, in other words, $\text{AP} : \mathbb{R}^{S\times R\times 1} \mapsto \mathbb{R}^{R\times 1}$.

4.3.3 *Optimization*

To optimize our network on the training set, we employ both a classification loss and a temporal regression loss. Different than *e.g.*, R-C3D [262], our classification task is specifically dependent on the few support videos. Accordingly, the loss function is given as:

$$L = \frac{1}{N_{cls}} \sum_i L_{cls}(a_i, a_i^*) + \frac{1}{N_{reg}} \sum_i a_i^* L_{reg}(t_i, t_i^*), \tag{4.12}$$

where N_{cls} and N_{reg} stand for batch size and the number of proposal segments, while i denotes the proposal segment index in a batch, a_i is the predicted probability of the proposal segment, a_i^* is the ground truth label, and t_i represents predicted relative offset to proposals. In the context of this work, the ground truth label is class-agnostic and hence binary (foreground/background), indicating the presence of an action or not. Lastly, t_i^* represents the coordinate transformation of ground truth segments to proposals.

The above loss function is applied on two parts: the support-agnostic part and the support-conditioned part. All losses for the two parts are optimized jointly. In the support-agnostic part, the foreground/background classification loss L_{cls} predicts whether the proposal contains an activity, or not, and the regression loss L_{reg} optimizes the relative displacement between proposals and ground truths. For the support-conditioned part, the loss L_{cls} predicts whether the proposal has the same common action as the one among the few support videos. The regression loss L_{reg} optimizes the relative displacement

Table 16: **Overview of the common (multi-)instance datasets.** The common instance datasets contain a single target action per video, while the common multi-instance datasets contain more frames and more actions per video, adding to the challenge of few-shot common action localization.

	Common instance		**Common multi-instance**	
	ActivityNet	Thumos	ActivityNet	Thumos
Video statistics				
number of instances	1	1	1.6	14.3
number of frames	266.9	284.6	444.5	5764.2
length (sec)	89.0	11.4	148.2	230.6
number of train videos	10035	3580	6747	1665
number of val+test videos	2483	775	1545	323
Class statistics				
number of train actions	160	16	160	16
number of val+test actions	40	4	40	4

between activities and ground truths. We note explicitly that this is done for the training set only.

During inference, the proposal subnet generates proposals for the query video. The proposals are refined by Non-Maximum Suppression (NMS) with a threshold of 0.7. Then the selected proposals are fused with the support videos through the mutual enhancement, progressive alignment, and pairwise matching modules. The obtained representation is fed to the classification subnet to again perform binary classification and the boundaries of the predicted proposals are further refined by the regression layer. Finally, we conduct NMS based on the confidence scores of the refined proposals to remove redundant ones, and the threshold in NMS is set a little bit smaller than the overlap threshold θ in evaluation ($\theta = 0.1$ in this chapter).

Optimizing for long videos. The longer the untrimmed query video, the larger the need for common localization, as manual searching for the activity becomes problematic. In our setup, the length of the input video is set to 768 frames to fit the GPU memory. When the query video is longer than 768 frames, we employ multi-scale segment generation [211]. We apply temporal sliding windows of 256, 512, and 768 frames with 75% overlap. Consequently, we generate a set of candidates $\Phi = \left\{(s_h, \psi_h, \psi'_h)\right\}_{h=1}^{H}$ as input for the proposal subnet, where H is the total number of sliding windows, and ψ_h and ψ'_h are the starting time and ending time of the h-th segment s_h. All refined proposals of all candidate segments together go through the NMS to remove redundant proposals.

4.4 EXPERIMENTAL SETUP

4.4.1 *Datasets*

Existing video datasets are usually created for classification [112, 128], temporal localization [29], captioning [37], or summarization [86]. To evaluate few-shot common action localization, we have revised two existing datasets, namely ActivityNet1.3 [29]

Table 17: **Module evaluation** on ActivityNet and Thumos in the common instance setting. All three modules have a positive mAP effect on the localization performance with only a slight increase in parameters.

			ActivityNet		**Thumos**	
MEM	PAM	PMM	one-shot	five-shot	one-shot	five-shot
			42.4	42.5	37.5	38.4
	✓		49.7	52.0	42.3	44.5
	✓	✓	51.3	53.6	44.8	46.0
✓	✓		52.5	55.3	47.6	49.6
✓	✓	✓	**53.1**	**56.5**	**48.7**	**51.9**

and Thumos14 [112]. Both datasets come with temporal annotations suitable for our evaluation. We consider both common instance and common multi-instance, where the latter deals with query videos containing multiple instances of the same action.

Common instance. For the revision of ActivityNet1.3, we follow the organization of Feng *et al.* [67]. We divide videos that contain multiple actions into independent videos, with every newly generated video consisting of just one action and background. Next we discard videos longer than 768 frames. We split the remaining videos into three subsets, divided by action classes. We randomly select 80% of the classes for training, 10% of the classes for validation, and the remaining 10% of the classes for testing. Besides ActivityNet, we also revise the Thumos dataset using the same protocol.

Common multi-instance. Query videos in real applications are usually unconstrained and contain multiple action segments. Therefore, we also split the original videos of ActivityNet1.3 and Thumos14 into three subsets according to their action classes without any other video preprocessing. As a result, we obtain long query videos with multiple action instances. The support videos are still trimmed action videos.

During training, the support videos and query video are randomly paired, while the pairs are fixed for validation and testing. The differences between the common instance and common multi-instance video datasets are highlighted in Table 16.

4.4.2 *Experimental Details*

We use PyTorch [182] for implementation. Our network is trained with Adam [131] with a learning rate of 1e-5 on one Nvidia GTX 1080TI. We use 40k training iterations and learning rate is decayed to 1e-6 after 25k iterations. To be consistent with the training process of our baselines [67, 283], we use the same C3D backbone [234]. The backbone is pre-trained on Sports-1M [124] and is fine-tuned with a class-agnostic proposal loss on the training videos for each dataset. The batch size is set to 1. The proposal score threshold is set as 0.7. The proposal number after NMS is 128 in training and 300 in validation and testing.

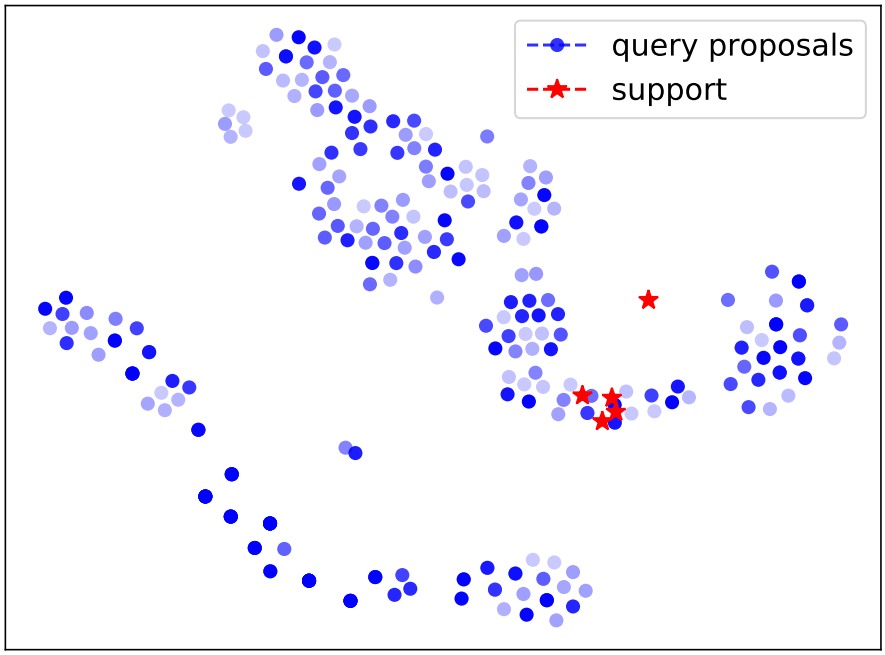


Figure 21: Without our modules.

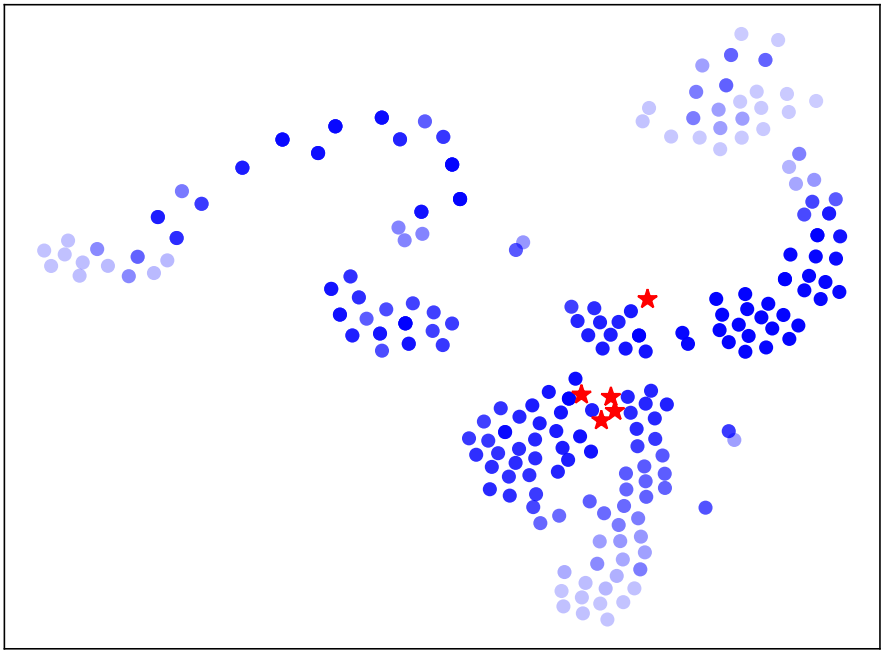


Figure 22: With our modules.

Figure 23: **Module evaluation by t-SNE visualization of support and query representations.** Colors of query proposals indicate their overlap with the ground truth action, the darker, the better. Without our modules (left), both relevant and irrelevant query proposals are near the support videos. Afterwards (right), only relevant proposals remain close to the support videos, highlighting the effectiveness of our modules for localizing common actions among a few videos.

Table 18: **Influence of noisy support videos** on common-instance ActivityNet for the five-shot setting. The result shows that our approach is robust to the inclusion of noisy support videos, whether they come from the same or different classes.

No noise	56.5
1 noisy support video	53.5
2 noisy support videos of different class	51.9
2 noisy support videos of same class	50.6

4.4.3 *Evaluation*

Following [67, 211], we measure the localization performance using (mean) Average Precision. A prediction is correct when it has the correct foreground/background prediction and has a ground truth overlap larger than the overlap threshold. The overlap is set to 0.5 unless specified otherwise.

4.5 EXPERIMENTAL RESULTS

4.5.1 *Ablation Study*

Module evaluation. We evaluate the effect of the mutual enhancement module (MEM), the progressive alignment module (PAM), and the pairwise matching module (PMM) for our task on the common instance datasets. We report results using one and five support videos in Table 17. To validate the effectiveness of our modules, we compare to our baseline system without any modules. Here the support representations are averaged and added to the query representations. We observe that the progressive alignment module increases over the baseline considerably, showing its efficacy. Adding the pairwise matching on top of the progressive alignment or using the mutual enhancement before the progressive alignment further benefits few-shot common action localization. Combining all three modules works best.

To get insight into the workings of our modules for common action localization, we have analysed the feature distribution before and after the use of our modules. In Figure 23, we show the t-SNE embedding [160] before and after we align the five support videos with the 300 proposals in one query video. We observe that after the use of our modules, the proposals with high overlap are closer to the support videos, indicating our ability to properly distill the correct action locations using only a few support videos. Irrelevant proposals are pushed away from the support videos, which results in a more relevant selection of action locations.

Few-shot evaluation. Our common action localization is optimized to work with multiple examples as support. To show this capability, we have measured the effect of gradually increasing the number of support videos, we found that the mAP gradually increases as we enlarge the number of support videos from one to six on common-instance ActivityNet. We obtain an mAP of 53.1 (one shot), 53.8 (two shots), 54.9 (three shots), 55.4 (four shots), 56.5 (five shots), 56.8 (six shots). The results show that our approach obtains high accuracy with only a few support videos. Using more than one

support video is beneficial for common action localization in our approach, showing that we indeed learn from using more than one support video. Results stagnate when using more than six examples.

Effect of support video length. We ablate the effect of the length of the support videos on the localization performance in Figure 24. We sample 16, 32, 48 and 64 frames for each support video respectively. We find that the result gradually increases with longer support videos, which indicates that temporal information in the support videos is beneficial to our modules for common action localization.

Influence of action proportion in query video. Figure 25 shows that for query videos with a dominant action, we can obtain high scores. An open challenge remains localizing very short actions in very long videos.

Influence of noisy support videos. To test the robustness of our approach, we have investigated the effect of including noisy support videos in the five-shot setting. The results are shown in Table 18. When one out of five support videos contains the wrong action, the performance drops only 3% from 56.5 to 53.5. The performance drop remains marginal when replacing two of the five support videos with noisy videos. When two noisy support videos are from the same class, the drop is larger, which is to be expected, as this creates a stronger bias towards a distractor class. Overall, we find that our approach is robust to noise for common action localization.

Qualitative results. To visualize the result of our method, we show three cases in Figure 27. For the first example, we can find the common action location from one support video. Adding more support videos provides further context, resulting in a better fit. For the second one, our method can recover the correct prediction only when five support videos are used. As shown in the third case, our method can also handle the multi-instance scenario. We show a query video with three instances. With only one support video, we miss one instance and have low overlap with another. When more support videos are added, we can recover both misses.

4.5.2 *Comparisons*

To evaluate the effectiveness of our proposed approach for common action localization, we perform three comparative evaluations.

One-shot comparison. For the one-shot evaluation, we compare to the one-shot video re-localization of Feng *et al.* [67] and to Hu *et al.* [107], which focuses on few-shot common object detection. We evaluate on the same setting as Feng *et al.* [67], namely the revised ActivityNet dataset using the one-shot setting (common instance). Note that we both use the C3D base network. To evaluate the image-based approach of Hu *et al.* [107], we use their proposed similarity module on the temporal video proposals, rather than spatial proposals based on author provided code [107]. The results in Table 19 show that across all overlap thresholds, our approach is preferred. At an overlap threshold of 0.5, we obtain an mAP of 53.1 compared to 41.0 for [107] and 43.5 for [67]. It is of interest to note that without our three modules, we obtain only 42.4 (Table 17). This demonstrates that a different training setup or a different model architecture by itself does not benefit common action localization. We attribute our improvement to the better alignment between the support and query representations as a result of our

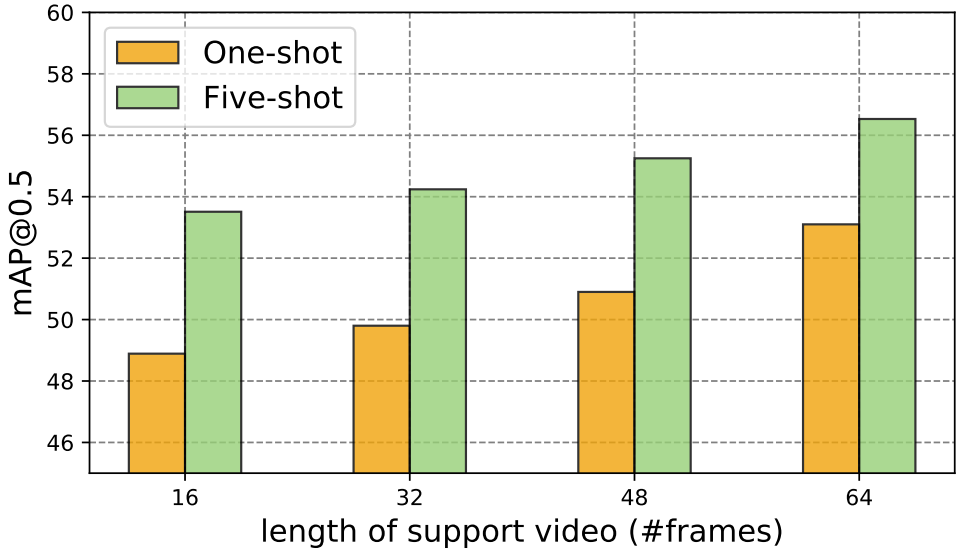


Figure 24: Effect of support video length.

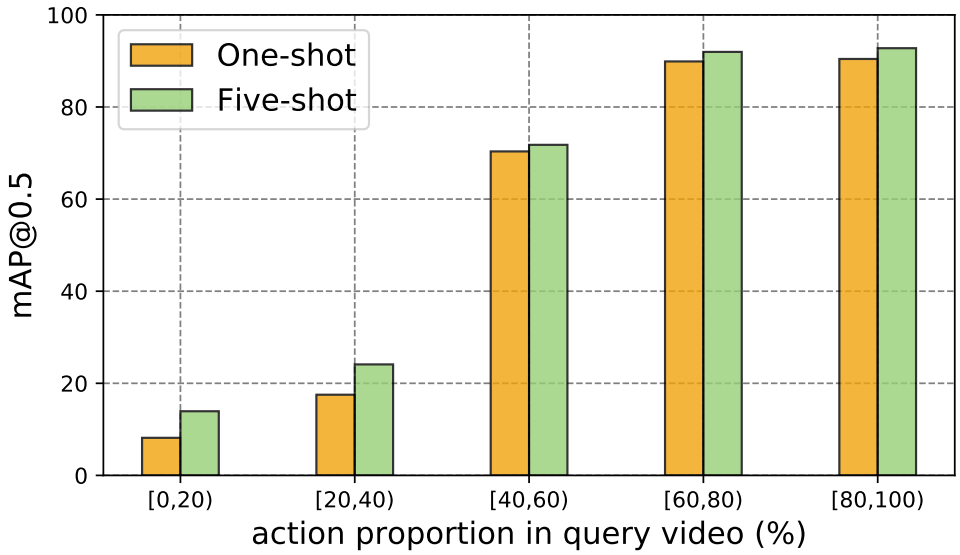


Figure 25: Effect of action ratio in query video.

Figure 26: **Ablation studies** on the length of the support videos and the action proportion in the query video. Both studies are on common-instance ActivityNet. Left: The longer the support videos, the better we perform, as we can distill more knowledge from the limited provided supervision. Right: High scores can be obtained when the common action is dominant, localization of short actions in long videos remains challenging.

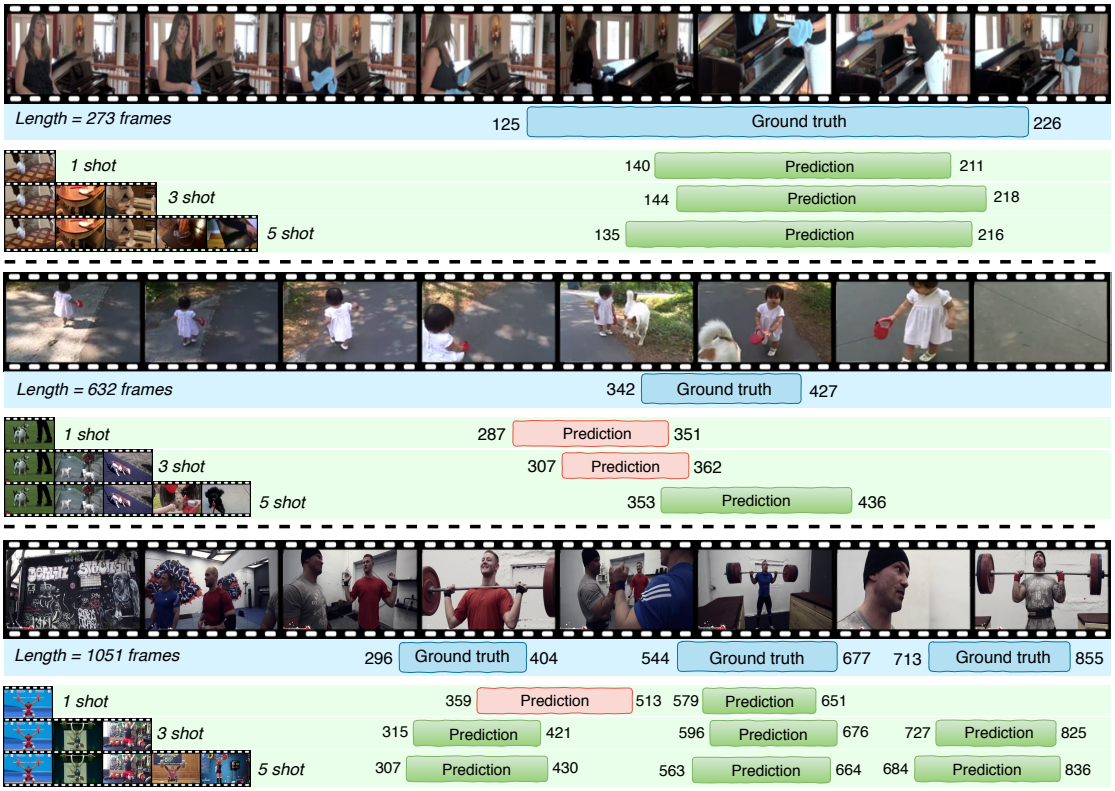


Figure 27: **Qualitative result of predictions by our approach under 1-shot, 3-shot and 5-shot settings.** Correct predictions with an overlap larger than 0.5 are marked in green, and incorrect predictions are marked in red. The length and start-end boundary of segment are indicated in frame numbers.

three modules. Next to a comparison on the common instance dataset, we also perform the same experiment on the longer multi-instance ActivityNet variant. In this more challenging setting, our approach again outperforms the baselines. We note that we are not restricted to the one-shot setting, where the baseline by Feng *et al.* [67] is.

Five-shot comparison. Second, we evaluate the performance of our approach on all datasets in the five-shot setting. We compare to a modified version of SST by Buch *et al.* [27]. We add a fusion layer on top of the original GRU networks in SST to incorporate the support feature, and then choose the proposal with the largest confidence score. SST is used as baseline, because the approach of Feng *et al.* [67] cannot handle more than one support video. We also include another comparison to Hu *et al.* [107]. This time also using their feature reweighting module. The results are shown in Figure 28. We observe that our method performs favorably compared to the two baselines on all datasets, reaffirming the effectiveness of our method. Also note that even when our support videos are noisy (Table 18), we are still better than the baselines without any noise based on Buch *et al.* [27] and Hu *et al.* [107] (39.7 and 45.4 for a threshold of 0.5 on common instance ActivityNet). The large amount of distractor actions in the long videos of common multi-instance Thumos results in lower overall scores, indicating that common action localization is far from a solved problem.

Localization from images. Next to using videos, we can also perform common action localization using images as support. This provides a challenging setting, since any temporal information is lost. We perform localization from support images by inflating

Table 19: **One-shot comparison on common instance ActivityNet**. Results marked with * obtained with author provided code. In both settings, our approach is preferred across all overlaps, highlighting its effectiveness.

	Overlap threshold					
	0.5	0.6	0.7	0.8	0.9	0.5:0.9
Common instance						
Hu *et al.* [107] *	41.0	33.0	27.1	15.9	6.8	24.8
Feng *et al.* [67]	43.5	35.1	27.3	16.2	6.5	25.7
This chapter	**53.1**	**40.9**	**29.8**	**18.2**	**8.4**	**29.5**
Common multi-instance						
Hu *et al.* [107] *	29.6	23.2	12.7	7.4	3.1	15.2
Feng *et al.* [67] *	31.4	25.5	16.1	8.9	3.2	17.0
This chapter	**42.1**	**36.0**	**18.5**	**11.1**	**7.0**	**22.9**

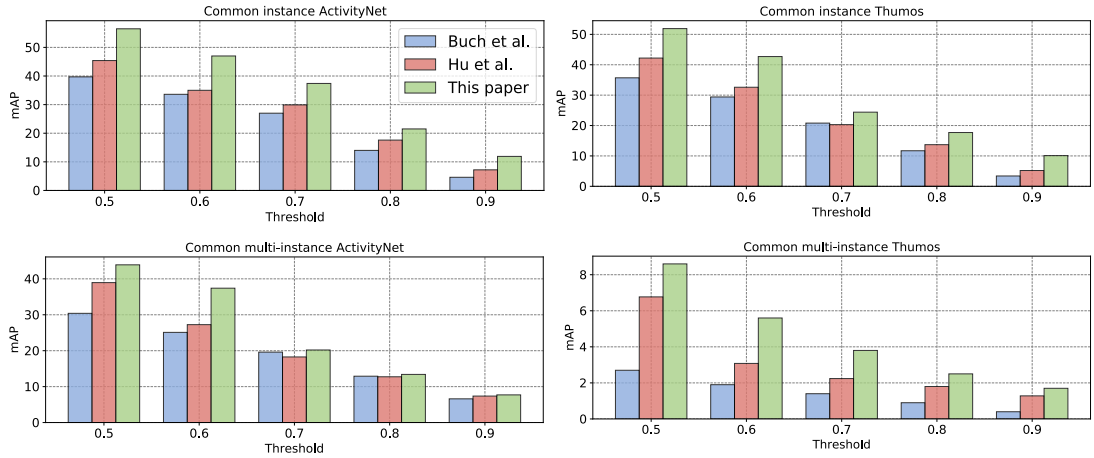


Figure 28: **Five-shot comparison.** We evaluate our method as well as modified versions of Hu *et al.* [107] and Buch *et al.* [27] on all common instance and multi-instance datasets, we obtain favourable results. Detailed numerical results are provided in supplementary file to facilitate the comparison for the follow-up works. Best viewed in color.

the images to create static support videos. We perform a common action localization on common instance ActivityNet and Thumos. We compare to the recent approach of Zhang *et al.* [283], which focuses on video retrieval from images. Results in Table 20 show we obtain favourable results on both datasets, even though our approach is not designed for this setting.

4.6 CONCLUSION

In this chapter we consider action localization in a query video given a few trimmed support videos that contain a common action, without specifying the label of the action. To tackle this challenging problem, we introduce a new network architecture along with three modules optimized for temporal alignment. The first module focuses on enhancing the representations of the query and support representation simultaneously. The second

Table 20: **Localization from images** on the common instance datasets. Our method generalizes beyond videos as support input and outperforms Zhang *et al.* [283]

	ActivityNet		**Thumos**	
	one-shot	five-shot	one-shot	five-shot
Zhang *et al.*	45.2	48.5	36.9	38.9
This chapter	**49.2**	**52.8**	**43.0**	**45.6**

module progressively integrates the representations of the support branch into the query branch, to distill the common action in the query video. The third module weighs the different support videos to deal with non-informative support examples. Experiments on reorganizations of ActivityNet and Thumos dataset, both with settings containing a single and multiple action instances per video, show that our approach can robustly localize the action which is common amongst support videos in both standard and long untrimmed query videos.

4.7 APPENDIX

4.7.1 *Detail of Class Statistics*

We list the class names for the train, validation and test sets of the common instance and multi-instance ActivityNet in Table 23, and for the common instance and multi-instance Thumos in Table 24.

4.7.2 *Overview of the Basic Block and the Residual Block*

The structure of the basic block is illustrated in Figure 29. The main idea of the basic block is to align the features I_2 to the features I_1. Figure 30 depicts the schema of the residual block. Here, the residual block performs recalibration on the input features I.

4.7.3 *More Results*

Numerical results of five-shot comparison. Table 21 displays the numerical results of the five-shot comparison between our method and modified versions of Buch *et al.* [27] and Hu *et al.* [107] on both the common instance and multi-instance datasets. The numerical results are provided here to facilitate the comparison for the follow-up works.

Effect of depth in the progressive alignment module. n in Equation 7 denotes the depth of the progressive alignment module which is set to 3 in practice. We ablate the effect of the depth of the progressive alignment module in Table 22 under the five-shot setting on the common instance ActivityNet dataset. From depth 1 to 3, the mAP improves steadily with only a slight increase in parameters. Results stagnate when the depth is higher than 3.

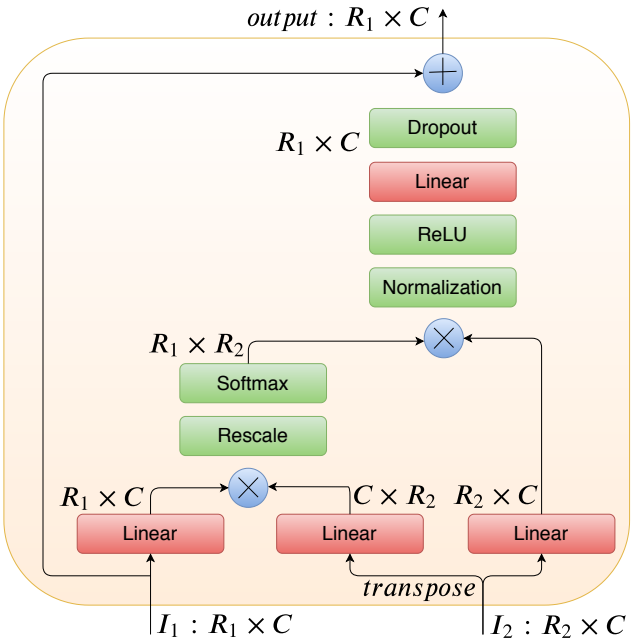


Figure 29: Scheme of basic block.

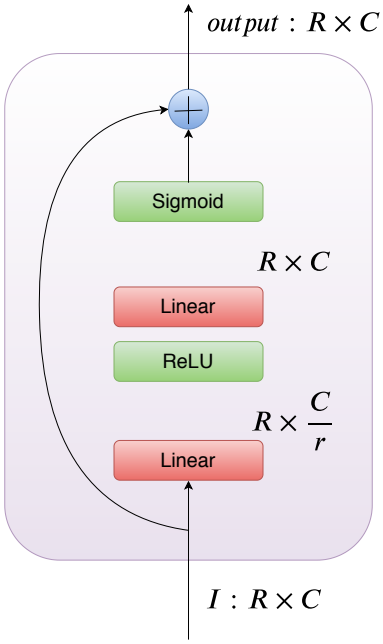


Figure 30: Scheme of residual block.

Figure 31: **Overview of our blocks.** I_1, I_2 denote the inputs of our basic block (left) and I denotes the input of our residual block (right). $\otimes$ denotes matrix multiplication and $\oplus$ is element-wise sum. r is set to 4.

Table 21: **Numerical results of five-shot comparison.**

	ActivityNet						Thumos					
	0.5	0.6	0.7	0.8	0.9	0.5:0.9	0.5	0.6	0.7	0.8	0.9	0.5:0.9
Common instance												
Buch *et al.*	39.7	33.6	27.0	14.0	4.6	23.3	35.7	29.4	20.8	11.7	3.4	20.2
Hu *et al.*	45.4	35.0	29.9	17.6	5.2	27.0	42.2	32.6	20.3	13.7	5.2	22.8
This chapter	**56.5**	**47.0**	**37.4**	**21.5**	**11.9**	**34.9**	**51.9**	**42.7**	**24.4**	**17.7**	**10.1**	**29.3**
Common multi-instance												
Buch *et al.*	30.4	25.1	19.6	12.9	6.6	18.9	2.7	1.9	1.4	0.9	0.4	1.5
Hu *et al.*	38.9	27.2	18.3	12.7	7.3	20.9	6.8	3.1	2.2	1.8	1.3	3.1
This chapter	**43.9**	**37.4**	**20.2**	**13.4**	**7.7**	**24.5**	**8.6**	**5.6**	**3.8**	**2.5**	**1.7**	**4.4**

Table 22: **Ablation study on the depth of the progressive alignment module** under five-shot setting on common instance ActivityNet. From depth 1 to 3, the mAP improves steadily with a slight increase in parameters. Results stagnate when the depth is more than 3.

	n-depth				
	1	2	3	4	5
#param	47.2M	48.2M	49.3M	50.3M	51.4M
mAP	**53.9**	**55.6**	**56.5**	**56.7**	**56.4**

Table 23: **Subset labels on common (multi-)instance ActivityNet.**

Training: Fun sliding down, Beer pong, Getting a piercing, Shoveling snow, Kneeling, Tumbling, Playing water polo, Washing dishes, Blowing leaves, Playing congas, Making a lemonade, Playing kickball, Removing ice from car, Playing racquetball, Swimming, Playing bagpipes, Painting, Assembling bicycle, Playing violin, Surfing, Making a sandwich, Welding, Hopscotch, Gargling mouthwash, Baking cookies, Braiding hair, Capoeira, Slacklining, Plastering, Changing car wheel, Chopping wood, Removing curlers, Horseback riding, Smoking hookah, Doing a powerbomb, Playing ten pins, Getting a haircut, Playing beach volleyball, Making a cake, Clean and jerk, Trimming branches or hedges, Drum corps, Windsurfing, Kite flying, Using parallel bars, Doing kickboxing, Cleaning shoes, Playing field hockey, Playing squash, Rollerblading, Playing drums, Playing rubik cube, Sharpening knives, Zumba, Raking leaves, Bathing dog, Tug of war, Ping-pong, Using the balance beam, Playing lacrosse, Scuba diving, Preparing pasta, Brushing teeth, Playing badminton, Mixing drinks, Discus throw, Playing ice hockey, Doing crunches, Wrapping presents, Hand washing clothes, Rock climbing, Cutting the grass, Wakeboarding, Futsal, Playing piano, Baton twirling, Mooping floor, Triple jump, Longboarding, Polishing shoes, Doing motocross, Arm wrestling, Doing fencing, Hammer throw, Shot put, Playing pool, Blow-drying hair, Cricket, Spinning, Running a marathon, Table soccer, Playing flauta, Ice fishing, Tai chi, Archery, Shaving, Using the monkey bar, Layup drill in basketball, Spread mulch, Skateboarding, Canoeing, Mowing the lawn, Beach soccer, Hanging wallpaper, Tango, Disc dog, Powerbocking, Getting a tattoo, Doing nails, Snowboarding, Putting on shoes, Clipping cat claws, Snow tubing, River tubing, Putting on makeup, Decorating the Christmas tree, Fixing bicycle, Hitting a pinata, High jump, Doing karate, Kayaking, Grooming dog, Bungee jumping, Washing hands, Painting fence, Doing step aerobics, Installing carpet, Playing saxophone, Long jump, Javelin throw, Playing accordion, Smoking a cigarette, Belly dance, Playing polo, Throwing darts, Roof shingle removal, Tennis serve with ball bouncing, Skiing, Peeling potatoes, Elliptical trainer, Building sandcastles, Drinking beer, Rock-paper-scissors, Using the pommel horse, Croquet, Laying tile, Cleaning windows, Fixing the roof, Springboard diving, Waterskiing, Using uneven bars, Having an ice cream, Sailing, Washing face, Knitting, Bullfighting, Applying sunscreen, Painting furniture, Grooming horse, Carving jack-o-lanterns
Validation: Swinging at the playground, Dodgeball, Ballet, Playing harmonica, Paintball, Cumbia, Rafting, Hula hoop, Cheerleading, Vacuuming floor, Playing blackjack, Waxing skis, Curling, Using the rowing machine, Ironing clothes, Playing guitarra, Sumo, Putting in contact lenses, Brushing hair, Volleyball
Testing: Hurling, Polishing forniture, BMX, Riding bumper cars, Starting a campfire, Walking the dog, Preparing salad, Plataform diving, Breakdancing, Camel ride, Hand car wash, Making an omelette, Shuffleboard, Calf roping, Shaving legs, Snatch, Cleaning sink, Rope skipping, Drinking coffee, Pole vault

Table 24: **Subset labels on common (multi-)instance Thumos.**

Training: BaseballPitch, BasketballDunk, Billiards, CleanAndJerk, CliffDiving, CricketBowling, CricketShot, Diving, FrisbeeCatch, GolfSwing, HammerThrow, HighJump, JavelinThrow, LongJump, PoleVault, Shotput
Validation: SoccerPenalty, TennisSwing
Testing: ThrowDiscus, VolleyballSpiking

5

QUERY BY ACTIVITY VIDEO IN THE WILD

5.1 INTRODUCTION

This chapter investigates the problem of activity retrieval given a video as example query. In the literature, activity retrieval is often framed as a classification task [5], a localization task [285], or as a retrieval-by-text problem [63]. For the less common task of activity retrieval by video, several works have shown that activities can be retrieved directly from a user-provided query [218, 249]. A standard assumption however is that the activities form a closed set, *i.e.*, they assume a fixed set of activities, each with many training videos. In practice, most activities, especially the more fine-grained ones, will have only few examples. Without an explicit focus on such activities, they will be ignored in favor of coarse-grained activities with many examples. In this work, we focus on obtaining a balanced video representation of activities for retrieval, regardless of whether they have many or few examples.

Learning with imbalanced data is an active research topics for various visual recognition tasks, including image classification [154], image segmentation [129], and object detection [178]. A central theme in these works is to either make classes with few examples more prominent, or switch to a setting where all classes have an equally-sized representation, *e.g.*, using memory banks [28, 158] or prototypes [126, 288]. In this chapter, we take inspiration from existing imbalanced vision tasks for the problem of imbalanced activity retrieval from video queries. We introduce two alignment modules to match the activity feature regardless of whether they have many or few examples, see Figure 32. Different from existing works, we do so by using both visual and semantic prototypes, where we emphasize the importance of a global alignment with respect to all activities. By this, we can enjoy the advantage of global alignment visually to deal with the imbalance and also owns the ability of interpolation in semantic level [169] to further align the feature.

As a first contribution, we introduce a new task centered around video query by activity *in the wild*, which emphasizes the importance of balancing performance between the activities with many examples and the activities with few examples. Second, we introduce a visual-semantic embedding network for retrieval by a video query. The network extends the standard classification loss in networks with two novel modules. The visual alignment module maintains a visual bank with equal space of feature representation

Submitted for publication.

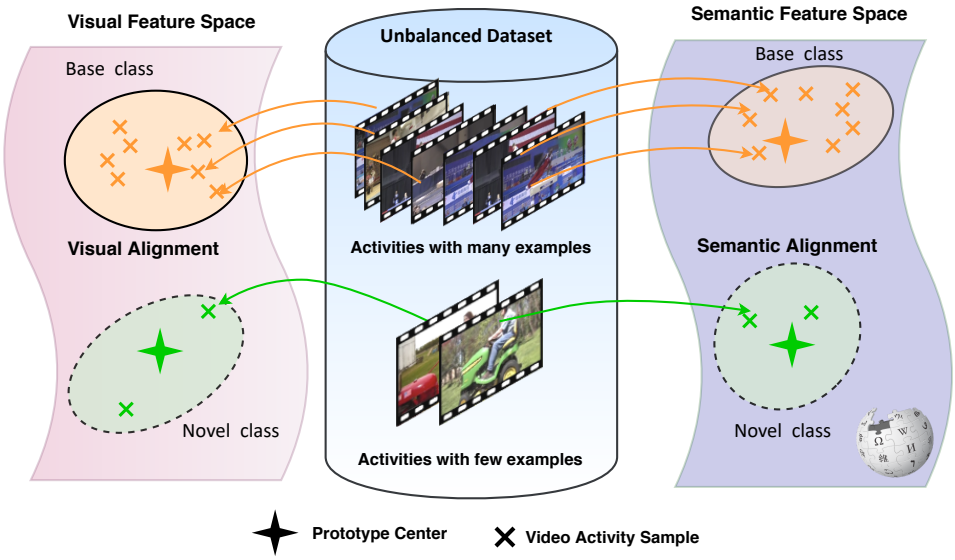


Figure 32: **Our motivation.** In typical activity retrieval, the training set is composed of activities with many examples (base) and activities with few examples (novel). We propose a visual-semantic alignment to balance the retrieval performance between base and novel classes.

for each activity. The representation of an input video is globally aligned with the visual bank representation of all activities to obtain a loss that disregards the amount of examples available for each activity. The semantic alignment module performs a similar global alignment and loss, but between the input video and video-independent semantic representations of activities such as word embeddings. These modules explicitly target the problem of imbalance in video datasets. Third, we perform extensive evaluation and analyses to examine the workings of our approach for imbalanced activity retrieval.

5.2 RELATED WORK

Video retrieval. For video retrieval, a common direction is to retrieve videos by a textual query [59, 263]. Hendricks *et al.* [96] propose a network that localizes text queries in videos using local and global temporal video representations. Hendricks *et al.* [97] further propose to model the context as a latent variable to bridge the gap between videos and textual queries. For activity retrieval by query video, current works are generally concerned with an efficient matching setup between query and test videos. Examples include retrieval using hashing [60] or using quantized video representations [48]. A common starting assumptions is that the activities to retrieve have ample training examples to learn such an efficient matching. In this work, we challenge this assumption and propose a network able to retrieve both activities with many examples and activities with few examples from a query video.

Learning with imbalanced data. When dealing with frequent classes (base classes [90]) and infrequent classes (novel classes [90]), a persistent issue is overfitting to the base classes. Transfer learning to novel classes provides a way to boost the performance on

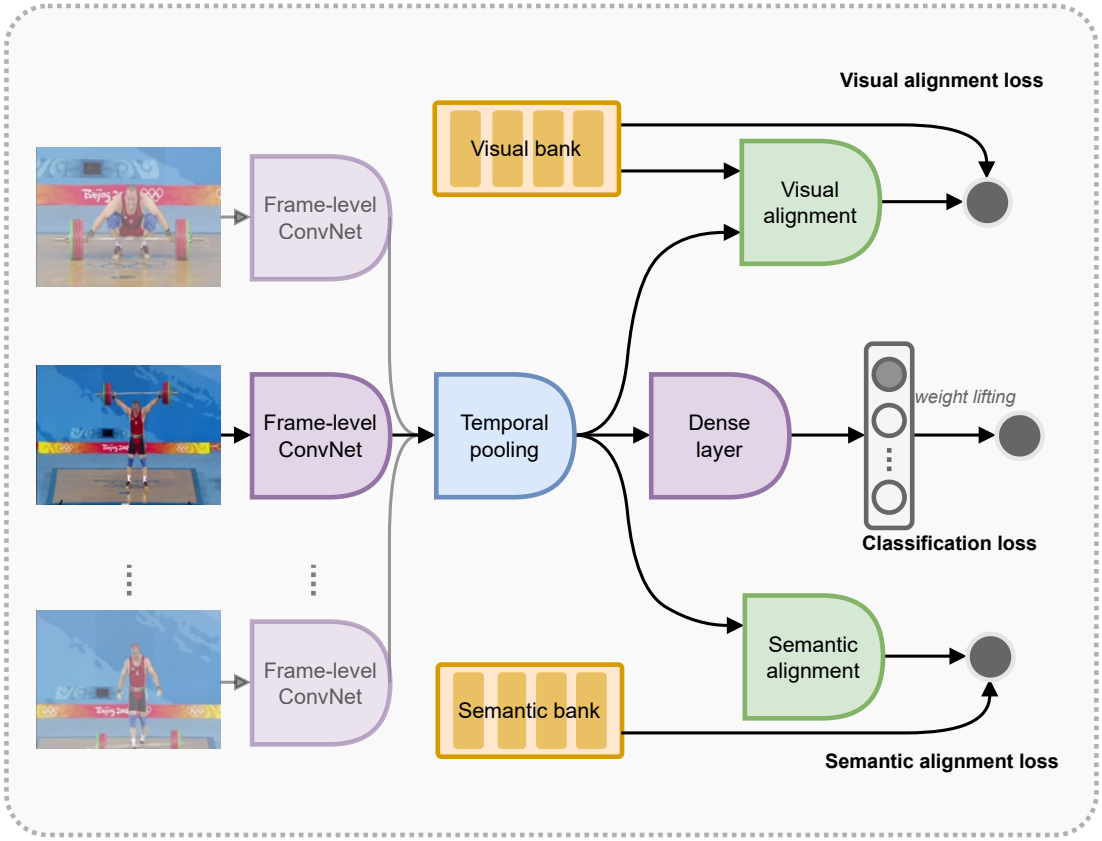


Figure 33: **Visual-Semantic Embedding Network**. During the training phase, we augment the classification loss with visual alignment and semantic alignment modules to preserve the visual and semantic representations of all activities. During testing, we only use the frame-level embedding z.

novel classes, although this is paired with a catastrophic forgetting problem on base classes [206]. Meta-learning is similarly focused on improving generalization to novel classes, *e.g.* through a few steps of fine-tuning [46]. However, these methods only consider the generalization on novel classes while ignoring the performance of base classes [158]. We aim to achieve a balance between both.

To attain such a balance, early work attempted to use a single example of the novel class to adapt classifiers from similar base classes using hand-crafted features [14]. Learning with imbalanced classes has since actively been researched in image classification [280], image segmentation [3], and object detection [143]. Hariharan and Girshick [90], for example, tackle the imbalance problem by hallucinating additional training examples for rare novel classes.

5.3 VISUAL-SEMANTIC EMBEDDING NETWORK

We aim to learn a video representation for activity retrieval, where the task is to retrieve trimmed videos containing the same human activity as the one given in a query video. Let $\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$ be a set of N activity videos, where x is a video of T frames describing an activity $y \in \mathcal{Y}$. Our goal is to learn an embedding function $f(\cdot) \in \mathbb{R}^C$ such that two different videos $x^{(i)}$ and $x^{(j)}$ of the same activity y are close in the embedding space.

In large collections of activities, there usually exists an imbalance in the number of examples per activity. Following Hariharan and Girshick [90], we denote activities with many examples as the *base* classes and activities with few examples as the *novel* classes. Formally, $\mathcal{Y}$ is then split into $\mathcal{Y}_{base}$ and $\mathcal{Y}_{novel}$, with $\mathcal{Y}_{base} \cap \mathcal{Y}_{novel} = \emptyset$. Having an imbalanced training set causes the embedding function $f(\cdot)$ to be geared towards $\mathcal{Y}_{base}$ in the evaluation phase. As a consequence, this induces a poor retrieval performance for the under-represented classes $\mathcal{Y}_{novel}$. To alleviate this issue, we propose two alignment modules to preserve the *visual* and *semantic* representations of all activities.

Action representations. To learn video representations of activities, we opt for a frame-level convolutional neural network (ConvNet) as our embedding function. Working at the frame-level rather than at the video level (*e.g.*, with 3D convolutions) offers more flexibility at the evaluation phase.

We extract the embedding representation for every frame x_t and simply average them over time to obtain a video-level representation $z \in \mathbb{R}^C$:

$$z = \frac{1}{T} \sum_{t=1}^{T} f(x_t). \tag{5.1}$$

The embedding representation is then further projected on a label space for classification. The probability of class $c \in \mathcal{Y}$ given an embedding representation z is:

$$p_{\mathrm{A}}(y = c|z) = \frac{\exp(-W_c \cdot z)}{\sum_{k \in \mathcal{Y}} \exp(-W_k \cdot z)}, \tag{5.2}$$

where W is the learnable parameter of the linear projection.

Visual alignment. While a standard classification embedding uses examples of all activities, the loss is in practice dominated by activities with many examples. In an effort to balance base and novel activity representations, we first focus on a visual alignment between all activities. Let $V \in \mathbb{R}^{K \times C}$ denote a visual bank matrix consisting of features representations of dimension C for every activity $y \in \mathcal{Y}$. The size of the bank then corresponds to $K = |\mathcal{Y}|$ activities. The idea of the visual bank is to obtain a single prototypical representation for every activity. Hence, all activities are treated equally, regardless of the number of examples available for training. For a new activity embedding z of activity y, we update the visual bank V through a convex combination of the current embedding representation z and the corresponding entry in V followed by an ℓ_2 normalization:

$$\begin{aligned} V_y &= \alpha \frac{z}{\|z\|_2} + (1 - \alpha) V_y, \\ V_y &= V_y / \|V_y\|, \end{aligned} \tag{5.3}$$

where α controls the amount of update in the visual bank. The visual bank is initialized to zero when training.

Building upon such visual banks, we propose to align the representations of different activities. For this purpose, we rely on the attention mechanism from the non-local operator [247]. Compared to the original non-local block, we aim to capture the re-

lation between different prototypical representations rather than spatial [107, 247] or temporal [247, 251] relations.

Let GA denote a global alignment operator that reasons the attention between the visual bank representation of one activity in V, in detail, GA : $(\mathbb{R}^{1\times C}, \mathbb{R}^{K\times C}) \mapsto \mathbb{R}^{1\times C}$. When compared to the prototypes in the visual bank, the aligned representation $z^\star$=GA(V_c, z) can then be used to provide the probability of class c:

$$p_{\mathrm{V}}(y = c|z) = \frac{\exp\Big(-d(z^\star, V_c)/\tau\Big)}{\sum_{k\in\mathcal{Y}} \exp\Big(-d(z^\star, V_k)/\tau\Big)}, \tag{5.4}$$

where τ is the temperature of the softmax function and d is the Euclidean distance.

Semantic alignment. We additionally leverage word embeddings of activity names as prior information. A semantic representation of an activity encapsulates relations amongst all pairs of activity classes. We use this information to align activity representations towards such semantic knowledge. We denote $\phi(y) \in \mathbb{R}^W$ as the word embedding of the activity y. Let $S \in \mathbb{R}^{K\times W}$ be the semantic bank which compiles the word embedding $\phi(y)$ of all K activities. For the semantic alignment, we simply opt for a multilayer perceptron $g(\cdot)$. Similar to the visual alignment, a probability for class c can be derived after aligning the representation z with the semantic activity embedding:

$$p_{\mathrm{S}}(y = c|z) = \frac{\exp\Big(-d(g(z), S_c)/\tau\Big)}{\sum_{k\in\mathcal{Y}} \exp\Big(-d(g(z), S_c)/\tau\Big)}. \tag{5.5}$$

Compared to the visual bank, the semantic bank remains fixed during training. We initialize the semantic bank from an existing word embedding (*e.g.*, word2vec [169], ELMo [185] GloVe [184]).

Optimization Training the overall network amounts to minimizing the cross-entropy loss function for all three components over the training set:

$$\mathcal{L} = -\log(p_A) - \lambda_V \log(p_V) - \lambda_S \log(p_S), \tag{5.6}$$

where λ_V and λ_S are trade-off hyper-parameters. Once the network has been trained, we extract the video-level representations z followed by an ℓ_2 normalization for all videos in the gallery set.

5.4 EXPERIMENTAL SETUP

Video Retrieval Dataset. We reorganize ActivityNet1.3 [29] for our video retrieval setting and refer to the reorganization as VR-ActivityNet. As our method aims at evaluating the performance of *base* classes and *novel* classes, we split the 200 activity labels into 100 base classes (C_{0-100}) and 100 novel classes ($C_{100-200}$). We also divide the dataset into a training, validation, and test set. The validation set is trying to evaluate the balanced performance between the C_{0-100} and $C_{100-120}$. Similarly, the test set is designed to evaluate the balanced performance between the C_{0-100} and $C_{120-200}$. The detailed activity splits will be provided on our project website.

Table 25: **Visual module ablation.** Both base and novel classes benefit from the module compared to the baseline.

method	base (mAP)	novel (mAP)	**H**
w/o visual	25.76	16.28	19.95
w/ visual	31.99	18.17	23.18

Table 26: **Scatteredness ablation.** The module works because activities are distinguished well from each other.

method	base	novel	overall
baseline	0.84	0.90	0.87
+visual	**1.19**	**1.17**	**1.18**

Evaluation metrics. For video retrieval, we consider the mean average precision (mAP) both on base classes and novel classes. We also compute the harmonic mean (H) between the mAP of base classes and novel classes to evaluate the balance between base and novel class performance.

Implementation details. We set the learning rate to $1e-4$ with Adam [131] and a weight decay of $1e-5$ for 16k iterations and reduce the learning rate to $1e-5$ after 8k iterations. We use a batch size of 16. The trade-off hyper-parameters λ_V, λ_S are set to 1 by cross-validation and the convex coefficient α of the visual bank update is set to 0.9. Three video frames are extracted per second, resulting in an average of 32 frames per activity video. Every frame is randomly cropped and resized to 112×112. We use Fasttext [120] with 1,024 dimension as our default word embedding method. We use PyTorch for implementation and the Faiss [119] library to measure video similarities. We employ ResNet-18 [95] as a backbone network with weights pre-trained on ImageNet [53]. Code will be made available.

5.5 RESULTS

Ablation: Visual alignment. We first ablate our visual alignment module for imbalanced activity retrieval. In Table 25 we show the results for the baseline setting, which only uses a cross-entropy loss on a linear projection of the video representations, as well as the inclusion of the proposed module. We observe an improvement of 6.2 percent point (p.p.) for base classes and 1.9 p.p. for novel classes. Hence, for both frequent and infrequent activities, the module provides a benefit. We also consider the scatteredness of the visual activity proposals with and without the use of the visual alignment module. The results are in Table 26. The scatteredness is consistently higher with our module, making all activities more unique, which in turn leads to a more discriminative retrieval.

Ablation: Semantic alignment. For the semantic alignment module, we ablate its effect using four different word embedding methods. The results are shown in Table 27 using word2vec [169], ELMo [185], GloVe [184], and fasttext [120]. For all word embedding methods, the multilayer perceptron in the semantic alignment module is kept the same except for the last layer. We find that all word embeddings provide an

Table 27: **Word embeddings ablation.** Adding a semantic prior provides an improvement, regardless of the word embedding.

method	base (mAP)	novel (mAP)	**H**
baseline+visual	31.99	18.17	23.18
+word2vec [169]	33.31	18.73	23.97
+ELMo [185]	32.42	19.26	24.16
+GloVe [184]	32.59	19.28	24.23
+Fasttext [120]	32.36	19.44	24.29

Table 28: **Retrieval result in various activity taxonomy hierarchy**. Level-1 contains 6 super classes, level-2 contains 38 super classes. The mAP is evaluated on the overall classes.

method	level-1(6 -cls) (mAP)	level-2(38-cls) (mAP)
baseline+visual	22.41	20.35
+semantic	**23.14**	**21.76**

Table 29: **Comparison with other methods.** Our approach is preferred over both internal and external baselines, since our modules explicitly give equal importance to base and novel classes.

	base (mAP)	**novel** (mAP)	**H**
ImageNet [53]	9.18	13.02	10.76
Triple loss [102]	24.47	16.48	19.70
Margin loss [252]	25.84	17.36	20.76
Baseline	25.76	16.28	19.95
w/ MSIB [126]	26.00	17.80	21.13
w/ our modules	**32.42**	**19.26**	**24.16**

improvement over the setting with the baseline and our visual alignment module. Overall, fasttext provides the best balance between base and novel classes and we will use this word embedding for our remaining experiments. Having a semantic bank offers another benefit, namely an enhanced retrieval performance for different levels of the activity taxonomy. We show this is the case by utilizing the ActivityNet taxonomy [29] and evaluate the mAP for both the parent classes of the activities and the grandparent classes. Table 28 shows our method can provide improved scores for broader activity categories, highlighting that the proposed alignment results in a semantically more coherent retrieval.

Comparison with other methods. Using our two modules, we perform a comparative evaluation to several baseline retrieval approaches. The first baseline serves as a starting point. We use the network used in this work but only pre-trained on ImageNet [53] to obtain video representations by simply averaging their frames. Query and candidate videos are then matched using the Euclidean distance. As the result in Table 29 shows,

the low scores indicate the difficulty of the task. Interestingly, the off-the-shelf baseline does not suffer from an imbalance performance gap between the base classes and novel classes. This confirms that when fine-tuning, representations of videos tend to be more discriminative towards the base classes as they appear more frequent. We also introduce MSIB [126] as extra baseline, the fixed matrix 199×200 initialized by MSIB is appended to the end of the backbone, we still keep the form of losses unchanged. Table 29 also shows the consequence of imbalanced fine-tuning for two accepted approaches in retrieval, namely the triplet loss [102] and the margin loss [252] optimized on top of the same video representations as for our approach. Both approaches obtain a boost in base mAP and a smaller improvement in novel mAP. Both the sampling-based loss baselines and our baseline setup do not explicitly cater for novel classes, resulting in similar scores for the harmonic mean. Our proposed approach performs favorably compared to all baselines, both for base and for novel classes. Our harmonic mAP is respectively 13.4, 4.5, 3.4, and 4.2 percent point higher than the baselines. Specially, our harmonic mAP also outperforms MSIB [126]. We conclude that our formulation is preferred for activity retrieval regardless of whether they have many or few examples.

5.6 CONCLUSION

In this work we consider the new task of video retrieval by activity in the wild, and emphasize the importance of dealing with imbalanced data when retrieving activities from a video query. We introduce an embedding network that learns to balance frequent base activities and infrequent novel activities. The network contains two novel modules. A visual alignment module matches input videos with visual prototype representations of activities. A semantic alignment module on the other hand matches videos with word embedding representations of activities. Visual and semantic activity representations are of the same length, regardless of the number of examples each activity has. As a result, we arrive at an activity retrieval that better balances both types of activities. Experiments highlight the effectiveness of our approach, a series of ablations and analyses are designed to gain more insight into the problem.

6

POINTMIXUP: AUGMENTATION FOR POINT CLOUDS

6.1 INTRODUCTION

The goal of this chapter is to classify a cloud of points into their semantic category, be it an airplane, a bathtub or a chair. Point cloud classification is challenging, as they are sets and hence invariant to point permutations. Building on the pioneering PointNet by Qi *et al.* [187], multiple works have proposed deep learning solutions to point cloud classification [146, 188, 231, 248, 253, 282]. Given the progress in point cloud network architectures, as well as the importance of data augmentation in improving classification accuracy and robustness, we study how could data augmentation be naturally extended to support also point cloud data, especially considering the often smaller size of point clouds datasets (*e.g.* ModelNet40 [255]). In this work, we propose point cloud data augmentation by interpolation of existing training point clouds.

To perform data augmentation by interpolation, we take inspiration from augmentation in the image domain. Several works have shown that generating new training examples, by interpolating images and their corresponding labels, leads to improved network regularization and generalization, *e.g.*, [84, 233, 237, 276]. Such a mixup is feasible in the image domain, due to the regular structure of images and one-to-one correspondences between pixels. However, this setup does not generalize to the point cloud domain, since there is no one-to-one correspondence and ordering between points. To that end, we seek to find a method to enable interpolation between permutation invariant point sets.

In this work, we make three contributions. First, we introduce data augmentation for point clouds through interpolation and we define the augmentation as a shortest path interpolation. Second, we propose PointMixup, an interpolation between point clouds that computes the optimal assignment as a path function between two point clouds, or the latent representations in terms of point cloud. The proposed interpolation strategy therefore allows usage of successful regularizers of Mixup and Manifold Mixup [237] on point cloud. We prove that (*i*) our PointMixup indeed finds the shortest path between two point clouds; (*ii*) the assignment does not change for any pairs of the mixed point clouds for any interpolation ratio; and (*iii*) our PointMixup is a linear interpolation, an important property since labels are also linearly interpolated. Figure 34 shows two pairs of point clouds, along with our interpolations. Third, we show the empirical benefits of our data augmentation across various tasks, including classification, few-shot learning,

Published in Proceedings of the European Conference on Computer Vision, 2020 [44].

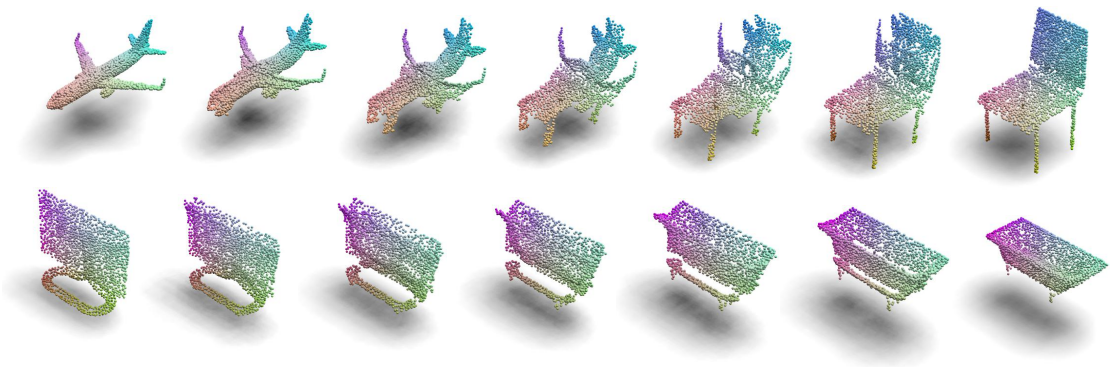


Figure 34: **Interpolation between point clouds.** We show the interpolation between examples from different classes (airplane/chair, and monitor/bathtub) with multiple ratios λ. The interpolants are learned to be classified as $(1-\lambda)$ the first class and λ the second class. The interpolation is not obtained by learning, but induced by solving the optimal bijective correspondence which allows the minimum overall distance that each point in one point cloud moves to the assigned point in the other point cloud.

and semi-supervised learning. We furthermore show that our approach is agnostic to the network used for classification, while we also become more robust to noise and geometric transformations to the points.

6.2 RELATED WORK

Deep learning for point clouds. Point clouds are unordered sets and hence early works focus on analyzing equivalent symmetric functions which ensures permutation invariance. [187, 195, 274]. The pioneering PointNet work by Qi *et al.* [187] presented the first deep network that operates directly on unordered point sets. It learns the global feature with shared multi-layer perceptions and a max pooling operation to ensure permutation invariance. PointNet++ [188] extends this idea further with hierarchical structure by relying on a heuristic method of farthest point sampling and grouping to build the hierarchy. Likewise, other recent methods follow to learn hierarchical local features either by grouping points in various manners [144, 146, 231, 248, 253, 264, 282]. Li *et al.* [146] propose to learn a transformation from the input points to simultaneously solve the weighting of input point features and permutation of points into a latent and potentially canonical order. Xu *et al.* [264] extends 2D convolution to 3D point clouds by parameterizing a family of convolution filters. Wang *et al.* [248] proposed to leverage neighborhood structures in both point and feature spaces.

In this work, we aim to improve point cloud classification for any point-based approach. To that end, we propose a new model-agnostic data augmentation. We propose a Mixup regularization for point clouds and show that it can build on various architectures to obtain better classification results by reducing the generalization error in classification tasks. A very recent work by Li *et al.* [145] also considers improving point cloud classification by augmentation. They rely on auto-augmentation and a complicated adversarial training procedure, whereas in this work we propose to augment point clouds by interpolation.

Interpolation-based regularization. Employing regularization approaches for training deep neural networks to improve their generalization performances have become

standard practice in deep learning. Recent works consider a regularization by interpolating the example and label pairs, commonly known as Mixup [84, 233, 276]. Manifold Mixup [237] extends Mixup by interpolating the hidden representations at multiple layers. Recently, an effort has been made on applying Mixup to various tasks such as object detection [281] and segmentation [69]. Different from existing works, which are predominantly employed in the image domain, we propose a new optimal assignment Mixup paradigm for point clouds, in order to deal with their permutation-invariant nature.

Recently, Mixup [276] has also been investigated from a semi-supervised learning perspective [17, 18, 238]. Mixmatch [18] guesses low-entropy labels for unlabelled data-augmented examples and mixes labelled and unlabelled data using Mixup [276]. Interpolation Consistency Training [238] utilizes the consistency constraint between the interpolation of unlabelled points with the interpolation of the predictions at those points. In this work, we show that our PointMixup can be integrated in such frameworks to enable semi-supervised learning for point clouds.

6.3 METHOD

6.3.1 *Problem Setting*

In our setting, we are given a training set $\{(S_m, c_m)\}_{m=1}^{M}$ consisting of M point clouds. $S_m = \{p_n^m\}_{n=1}^{N} \in \mathcal{S}$ is a point cloud consisting of N points, $p_n^m \in \mathbb{R}^3$ is the 3D point, $\mathcal{S}$ is the set of such 3D point clouds with N elements. $c_m \in \{0,1\}^C$ is the one-hot class label for a total of C classes. The goal is to train a function $h : \mathcal{S} \mapsto [0,1]^C$ that learns to map a point cloud to a semantic label distribution. Throughout our work, we remain agnostic to the type of function h used for the mapping and we focus on data augmentation to generate new examples.

Data augmentation is an integral part of training deep neural networks, especially when the size of the training data is limited compared to the size of the model parameters. A popular data augmentation strategy is Mixup [276]. Mixup performs augmentation in the image domain by linearly interpolating pixels, as well as labels. Specifically, let $I_1 \in \mathbb{R}^{W \times H \times 3}$ and $I_2 \in \mathbb{R}^{W \times H \times 3}$ denote two images. Then a new image and its label are generated as:

$$I_{\text{mix}}(\lambda) = (1-\lambda) \cdot I_1 + \lambda \cdot I_2, \tag{6.1}$$

$$c_{\text{mix}}(\lambda) = (1-\lambda) \cdot c_1 + \lambda \cdot c_2, \tag{6.2}$$

where $\lambda \in [0,1]$ denotes the mixup ratio. Usually λ is sampled from a beta distribution $\lambda \sim \text{Beta}(\gamma, \gamma)$. Such a direct interpolation is feasible for images as the data is aligned. In point clouds, however, linear interpolation is not straightforward. The reason is that point clouds are sets of points in which the point elements are orderless and permutation-invariant. We must, therefore, seek a definition of interpolation on unordered sets.

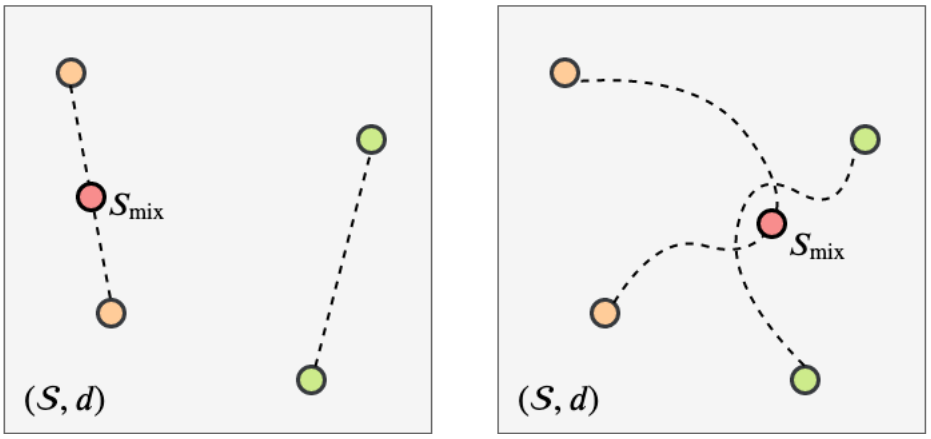


Figure 35: **Intuition of shortest-path interpolation.** The examples lives on a metric space $(\mathcal{S}, d)$ as dots in the figure. The dashed lines are the interpolation paths between different pairs of examples. When the shortest-path property is ensured (left), the interpolation paths from different pairs of source examples are likely to be not intersect in a complicated metric space. While in non-shortest path interpolation (right), the paths can intertwine with each other with a much higher probability, making it hard to tell which pair of source examples does the mixed data come from.

6.3.2 *Interpolation between Point Clouds*

Let $S_1 \in \mathcal{S}$ and $S_2 \in \mathcal{S}$ denote two training examples on which we seek to perform interpolation with ratio λ to generate new training examples. Given a pair of source examples S_1 and S_2, an interpolation function, $f_{S_1 \to S_2} : [0,1] \mapsto \mathcal{S}$ can be *any continuous function*, which forms a curve that joins S_1 and S_2 in a metric space $(\mathcal{S}, d)$ with a proper distance function d. This means that it is up to us to define what makes an interpolation good. We define the concept of shortest-path interpolation in the context of point cloud:

Definition 1 (Shortest-path interpolation) *In a metric space $(\mathcal{S}, d)$, a shortest-path interpolation $f^*_{S_1 \to S_2} : [0,1] \mapsto \mathcal{S}$ is an interpolation between the given pair of source examples $S_1 \in \mathcal{S}$ and $S_2 \in \mathcal{S}$, such that for any $\lambda \in [0,1]$, $d(S_1, S^{(\lambda)}) + d(S^{(\lambda)}, S_2)) = d(S_1, S_2)$ holds for $S^{(\lambda)} = f^*_{S_1 \to S_2}(\lambda)$ being the interpolant.*

We say that Definition 1 ensures the shortest path property because the triangle inequality holds for any properly defined distance d : $d(S_1, S^{(\lambda)}) + d(S^{(\lambda)}, S_2)) \geq d(S_1, S_2)$. The intuition behind this definition is that the shortest path property ensures the uniqueness of the label distribution on the interpolated data. To put it otherwise, when computing interpolants from different sources, the interpolants generated by the shortest-path interpolation is more likely to be discriminative than the ones generated by a non-shortest-path interpolation.

To define an interpolation for point clouds, therefore, we must first select a reasonable distance metric. Then, we opt for the shorterst-path interpolation function based on the selected distance metric. For point clouds a proper distance metric is the Earth Mover's Distance (EMD), as it captures well not only the geometry between two point clouds, but also local details as well as density distributions [2, 66, 151]. EMD measures the least amount of total displacement required for each of the points in the first point cloud,

$x_i \in S_1$, to match a corresponding point in the second point cloud, $y_j \in S_2$. Formally, the EMD for point clouds solves the following assignment problem:

$$\phi^* = \arg\min_{\phi \in \oplus} \sum_i \|x_i - y_{\phi(i)}\|_2, \tag{6.3}$$

where $\oplus = \{\{1, \ldots, N\} \mapsto \{1, \ldots, N\}\}$ is the set of possible bijective assignments, which give one-to-one correspondences between points in the two point clouds. Given the optimal assignment ϕ^*, the EMD is then defined as the average effort to move S_1 points to S_2:

$$d_{\text{EMD}} = \frac{1}{N} \sum_i \|x_i - y_{\phi^*(i)}\|_2. \tag{6.4}$$

6.3.3 *PointMixup: Optimal Assignment Interpolation for Point Clouds*

We propose an interpolation strategy, which can be used for augmentation that is analogous of Mixup [276] but for point clouds. We refer to this proposed PointMixup as *Optimal Assignment (OA) Interpolation*, as it relies on the optimal assignment on the basis of the EMD to define the interpolation between clouds. Given the source pair of point clouds $S_1 = \{x_i\}_{i=1}^N$ and $S_2 = \{y_j\}_{j=1}^N$, the Optimal Assignment (OA) interpolation is a path function $f^*_{S_1 \to S_2} : [0, 1] \mapsto \mathcal{S}$. With $\lambda \in [0, 1]$,

$$f^*_{S_1 \to S_2}(\lambda) = \{u_i\}_{i=1}^N, \quad \text{where} \tag{6.5}$$

$$u_i = (1 - \lambda) \cdot x_i + \lambda \cdot y_{\phi^*(i)}, \tag{6.6}$$

in which ϕ^* is the optimal assignment from S_1 to S_2 defined by Eq. 6.3. Then the interpolant $S_{\mathbf{OA}}^{S_1 \to S_2, (\lambda)}$ (or $S_{\mathbf{OA}}^{(\lambda)}$ when there is no confusion) generated by the OA interpolation path function $f^*_{S_1 \to S_2}(\lambda)$ is the required augmented data for point cloud Mixup.

$$S_{\mathbf{OA}}^{(\lambda)} = \{(1 - \lambda) \cdot x_i + \lambda \cdot y_{\phi^*(i)}\}_{i=1}^N. \tag{6.7}$$

Under the view of $f^*_{S_1 \to S_2}$ being a path function in the metric space $(\mathcal{S}, d_{\text{EMD}})$, f is expected to be the shortest path joining S_1 and S_2 since the definition of the interpolation is induced from the EMD.

6.3.4 *Analysis*

Intuitively we expect that PointMixup is a shortest path linear interpolation. That is, the interpolation lies on the shortest path joining the source pairs, and the interpolation is linear with regard to λ in $(\mathcal{S}, d_{\text{EMD}})$, since the definition of the interpolation is derived from the EMD. However, it is non-trivial to show the optimal assignment interpolation abides to a shortest path linear interpolation, because the optimal assignment between the mixed point cloud and either of the source point cloud is unknown. It is, therefore, not obvious that we can ensure whether there exists a shorter path between the mixed examples and the source examples. To this end, we need to provide an in-depth analysis.

To ensure the uniqueness of the label distribution from the mixed data, we need to show that the *shortest path property* w.r.t. the EMD is fulfilled. Moreover, we need to show that the proposed interpolation is *linear* w.r.t the EMD, in order to ensure that the input interpolation has the same ratio as the label interpolation. Besides, we evaluate the *assignment invariance property* as a prerequisite knowledge for the proof for the linearity. This property implies that there exists no shorter path between interpolants with different λ, *i.e.*, the shortest path between the interpolants is a part of the shortest path between the source examples. Due to space limitation, we sketch the proof for each property. The complete proofs are available in the supplementary material.

We start with the shortest path property. Since the EMD for point cloud is a metric, the triangle inequality $d_{EMD}(A,B) + d_{EMD}(B,C) \geq d_{EMD}(A,C)$ holds (for which a formal proof can be found in [202]). Thus we formalize the shortest path property into the following proposition:

Property 1 (shortest path) *Given the source examples S_1 and S_2, $\forall \lambda \in [0,1]$, $d_{EMD}(S_1, S_{\boldsymbol{OA}}^{(\lambda)}) + d_{EMD}(S_{\boldsymbol{OA}}^{(\lambda)}, S_2) = d_{EMD}(S_1, S_2)$.*

Sketch of Proof From the definition of the EMD we can derive $d_{\text{EMD}}(S_1, S_{\mathbf{OA}}^{(\lambda)}) + d_{\text{EMD}}(S_2, S_{\mathbf{OA}}^{(\lambda)}) \leq d_{\text{EMD}}(S_1, S_2)$. Then from the triangle inequity of the EMD, only the equality remains.

We then introduce the assignment invariance property of the OA Mixup as an intermediate step for the proof of the linearity of OA Mixup. The property shows that the assignment does not change for any pairs of the mixed point clouds with different λ. Moreover, the assignment invariance property is important to imply that the shortest path between the any two mixed point clouds is part of the shortest path between the two source point clouds.

Property 2 (assignment invariance) *$S_{\boldsymbol{OA}}^{(\lambda_1)}$ and $S_{\boldsymbol{OA}}^{(\lambda_2)}$ are two mixed point clouds from the same given source pair of examples S_1 and S_2 as well as the mix ratios λ_1 and λ_2 such that $0 \leq \lambda_1 < \lambda_2 \leq 1$. Let the points in $S_{\boldsymbol{OA}}^{(\lambda_1)}$ and $S_{\boldsymbol{OA}}^{(\lambda_2)}$ be $u_i = (1-\lambda_1) \cdot x_i + \lambda_1 \cdot y_{\phi^*(i)}$ and $v_k = (1-\lambda_2) \cdot x_k + \lambda_2 \cdot y_{\phi^*(k)}$, where ϕ^* is the optimal assignment from S_1 to S_2. Then the identical assignment ϕ_I is the optimal assignment from $S_{\boldsymbol{OA}}^{(\lambda_1)}$ to $S_{\boldsymbol{OA}}^{(\lambda_2)}$.*

Sketch of Proof We first prove that the identical mapping is the optimal assignment from S_1 to $S_{\mathbf{OA}}^{(\lambda_1)}$ from the definition of the EMD. Then we prove that ϕ^* is the optimal assignment from $S_{\mathbf{OA}}^{(\lambda_1)}$ to S_2. Finally we prove that the identical mapping is the optimal assignment from $S_{\mathbf{OA}}^{(\lambda_1)}$ to $S_{\mathbf{OA}}^{(\lambda_2)}$ similarly as the proof for the first intermediate argument. Given the property of assignment invariance, the linearity follows:

Property 3 (linearity) *For any mix ratios λ_1 and λ_2 such that $0 \leq \lambda_1 < \lambda_2 \leq 1$, the mixed point clouds $S_{\boldsymbol{OA}}^{(\lambda_1)}$ and $S_{\boldsymbol{OA}}^{(\lambda_2)}$ satisfies that $d_{EMD}(S_{\boldsymbol{OA}}^{(\lambda_1)}, S_{\boldsymbol{OA}}^{(\lambda_2)}) = (\lambda_2 - \lambda_1) \cdot d_{EMD}(S_1, S_2)$.*

Sketch of Proof The proof can be directly derived from the fact that the identical mapping is the optimal assignment between $S_{\mathbf{OA}}^{(\lambda_1)}$ and $S_{\mathbf{OA}}^{(\lambda_2)}$.

The linear property of our interpolation is important, as we jointly interpolate the point clouds and the labels. By ensuring that the point cloud interpolation is linear, we ensure that the input interpolation has the same ratio as the label interpolation.

On the basis of the properties, we find that PointMixup is a shortest path linear interpolation between point clouds in $(\mathcal{S}, d_{\text{EMD}})$.

6.3.5 *Manifold PointMixup: Interpolate between Latent Point Features*

In standard PointMixup, only the inputs, *i.e.*, the XYZ point cloud coordinates are mixed. The input XYZs are low-level geometry information and sensitive to disturbances and transformations, which in turn limits the robustness of the PointMixup. Inspired by Manifold Mixup [237], we can also use the proposed interpolation solution to mix the latent representations in the hidden layers of point cloud networks, which are trained to capture salient and high-level information that is less sensitive to transformations. PointMixup can be applied for the purpose of Manifold Mixup to mix both at the XYZs and different levels of latent point cloud features and maintain their respective advantages, which is expected to be a stronger regularizer for improved performance and robustness.

We describe how to mix the latent representations. Following [237], at each batch we randomly select a layer l to perform PointMixup from a set of layers L, which includes the input layer. In a point cloud network network, the intermediate latent representation at layer l (before the global aggregation stage such as the max pooling aggregation in PointNet [187] and PointNet++ [188]) is $Z_{(l)} = \{(x_i, z_i^{(x)})\}_{i=1}^{N_z}$, in which x_i is 3D point coordinate and $z_i^{(x)}$ is the corresponding high-dimensional feature. For the mixed latent representation, given the latent representation of two source examples are $Z_{(l),1} = \{(x_i, z_i^{(x)})\}_{i=1}^{N_z}$ and $Z_{(l),2} = \{(y_i, z_i^{(y)})\}_{i=1}^{N_z}$, the optimal assignment ϕ^* is obtained by the 3D point coordinates x_i, and the mixed latent representation then becomes

$$
\begin{aligned}
Z_{(l),\mathbf{OA}}^{(\lambda)} &= \{(x_i^{\text{mix}}, z_i^{\text{mix}})\}, \qquad \text{where} \\
x_i^{\text{mix}} &= (1-\lambda)\cdot x_i + \lambda \cdot y_{\phi^*(i)}, \\
z_i^{\text{mix}} &= (1-\lambda)\cdot z_i^{(x)} + \lambda \cdot z_{\phi^*(i)}^{(y)}.
\end{aligned}
$$

Specifically in PointNet++, three layers of representations are randomly selected to perform Manifold Mixup: the input, and the representations after the first and the second SA modules (See appendix of [188]).

6.4 EXPERIMENTS

6.4.1 *Setup*

Datasets. We focus in our experiments on the **ModelNet40** dataset [255]. This dataset contains 12,311 CAD models from 40 man-made object categories, split into 9,843 for training and 2,468 for testing. We furthermore perform experiments on the **ScanObjectNN** dataset [235]. This dataset consists of real-world point cloud objects, rather than sampled virtual point clouds. The dataset consists of 2,902 objects and 15 categories.

We report on two variants of the dataset, a standard variant OBJ_ONLY and one with heavy permutations from rigid transformations PB_T50_RS [235].

Following [146], we discriminate between settings where each dataset is pre-aligned and unaligned with horizontal rotation on training and test point cloud examples. For the unaligned settings, we randomly rotate the training point cloud along the up-axis. Then, before solving the optimal assignment, we perform a simple additional alignment step to fit and align the symmetry axes between the two point clouds to be mixed. Through this way, the point clouds are better aligned and we obtain more reasonable point correspondences. Last, we also perform experiments using only 20% of the training data.

Network architectures. The main network architecture used throughout the chapter is PointNet++ [188]. We also report results with PointNet [187] and DGCNN [248], to show that our approach is agnostic to the architecture that is employed. PointNet learns a permutation-invariant set function, which does not capture local structures induced by the metric space the points live in. PointNet++ is a hierarchical structure, which segments a point cloud into smaller clusters and applies PointNet locally. DGCNN performs hierarchical operations by selecting a local neighbor in the feature space instead of the point space, resulting in each point having different neighborhoods in different layers.

Experimental details. We uniformly sample 1,024 points on the mesh faces according to the face area and normalize them to be contained in a unit sphere, which is a standard setting [146, 187, 188]. In case of mixing clouds with different number of points, we can simply replicate random elements from the each point set to reach the same cardinality. During training, we augment the point clouds on-the-fly with random jitter for each point using Gaussian noise with zero mean and 0.02 standard deviation. We implement our approach in PyTorch [182]. For network optimization, we use the Adam optimizer with an initial learning rate of 10^{-3}. The model is trained for 300 epochs with a batch size of 16. We follow previous work [237, 276] and draw λ from a beta distribution $\lambda \sim \text{Beta}(\gamma, \gamma)$. We also perform Manifold Mixup [237] in our approach, through interpolation on the transformed and pooled points in intermediate network layers. In this work, we opt to use the efficient algorithm and adapt the open-source implementation from [151] to solve the optimal assignment approximation. Training for 300 epochs takes around 17 hours without augmentation and around 19 hours with PointMixup or Manifold PointMixup on a single NVIDIA GTX 1080 ti.

Baseline interpolations. For our comparisons to baseline point cloud augmentations, we compare to two variants. The first variant is random assignment interpolation, where a random assignment $\phi^{\mathbf{RA}}$ is used, to connect points from both sets, yielding:

$$S^{(\lambda)}_{\mathbf{RA}} = \{(1-\lambda) \cdot x_i + \lambda \cdot y_{\phi^{\mathbf{RA}}(i)}\}.$$

The second variant is point sampling interpolation, where random draws without replacement of points from each set are made according to the sampling frequency λ:

$$S^{(\lambda)}_{\mathbf{PS}} = S_1^{(1-\lambda)} \cup S_2^{(\lambda)},$$

where $S_2^{(\lambda)}$ denotes a randomly sampled subset of S_2, with $\lfloor \lambda N \rfloor$ elements. ($\lfloor \cdot \rfloor$ is the floor function.) And similar for S_1 with $N - \lfloor \lambda N \rfloor$ elements, such that $S^{(\lambda)}_{\mathbf{PS}}$ contains

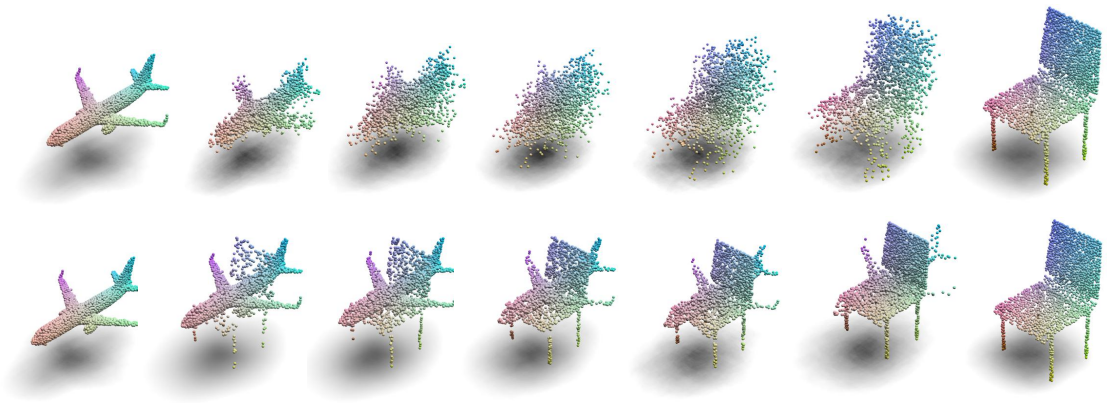


Figure 36: **Baseline interpolation variants.** Top: point cloud interpolation through random assignment. Bottom: interpolation through sampling.

exactly N points. The intuition of the point sampling variant is that for point clouds as unordered sets, one can move one point cloud to another through a set operation such that it removes several random elements from set S_1 and replace them with same amount of elements from S_2.

6.4.2 *Point Cloud Classification Ablations*

We perform four ablation studies to show the workings of our approach with respect to the interpolation ratio, comparison to baseline interpolations and other regularizations, as well robustness to noise.

Effect of interpolation ratio. The first ablation study focuses on the effect of the interpolation ratio in the data augmentation for point cloud classification. We perform this study on ModelNet40 using the PointNet++ architecture. The results are shown in Fig. 37 for the pre-aligned setting. We find that regardless of the interpolation ratio used, our approach provides a boost over the setting without augmentation by interpolation. PointMixup positively influences point cloud classification. The inclusion of manifold mixup adds a further boost to the scores. Throughout further experiments, we use $\gamma = 0.4$ for input mixup and $\gamma = 1.5$ for manifold mixup in unaligned setting, and $\gamma = 1.0$ for input mixup and $\gamma = 2.0$ for manifold mixup in pre-aligned setting.

Comparison to baseline interpolations. In the second ablation study, we investigate the effectiveness of our PointMixup compared to the two interpolation baselines. We again use ModelNet40 and PointNet++. We perform the evaluation on both the pre-aligned and unaligned dataset variants, where for both we also report results with a reduced training set. The results are shown in Table 30. Across both the alignment variants and dataset sizes, our PointMixup obtains favorable results. This result highlights the effectiveness of our approach, which abides to the shortest path linear interpolation definition, while the baselines do not.

PointMixup with other regularizers. Third, we evaluate how well PointMixup works by comparing to multiple existing data regularizers and mixup variants, again on ModelNet40 and PointNet++. We investigate the following augmentations: (*i*) Mixup [276], (*ii*) Manifold Mixup [237], (*iii*) mix input only without target mixup, (*iv*)

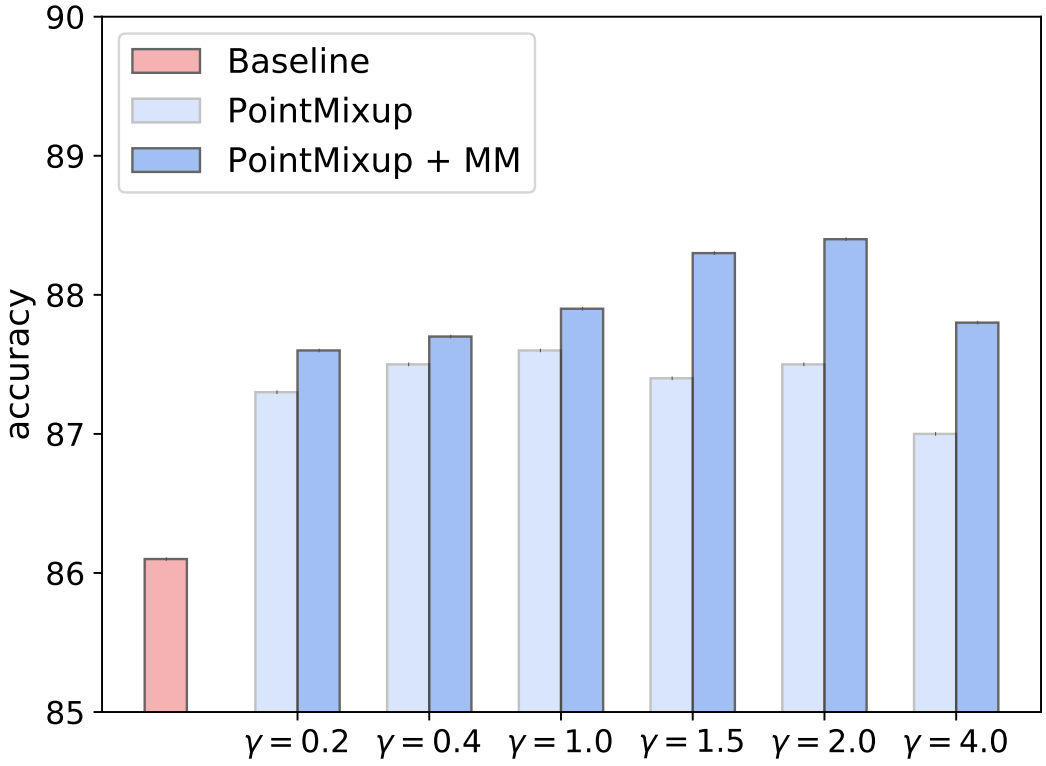


Figure 37: **Effect of interpolation ratios.** MM denotes Manifold Mixup.

mix latent representation at a fixed layer (manifold mixup does so at random layers), and (*v*) label smoothing [228]. Training is performed on the reduced dataset to better highlight their differences. We show the results in Table 31 on the left. Our approach with manifold mixup obtains the highest scores. The label smoothing regularizer is outperformed, while we also obtain better scores than the mixup variants. We conclude that PointMixup is forms an effective data augmentation for point clouds.

Robustness to noise. By adding additional augmented training examples, we enrich the dataset. This enrichment comes with additional robustness with respect to noise in the point clouds. We evaluate the robustness by adding random noise perturbations on point location, scale, translation and different rotations. Note that for evaluation of robustness against up-axis rotation, we use the models which are trained with the pre-aligned setting, in order to test also the performance against rotation along the up-axis as a novel transform. The results are in Table 31 on the right. Overall, our approach including manifold mixup provides more stability across all perturbations. For example, with additional noise ($\sigma = 0.05$), we obtain an accuracy of 56.5, compared to 35.1 for the baseline. We similar trends for scaling (with a factor of two), with an accuracy of 72.9 versus 59.2. We conclude that PointMixup makes point cloud networks such as PointNet++ more stable to noise and rigid transformations.

Qualitative analysis. In Figure 38, we show eight examples of PointMix for point cloud interpolation; four interpolations of cups and tables, four interpolations of chairs and cars. Through our shortest path interpolation, we end up at new training examples

Table 30: **Comparison of PointMixup to baseline interpolations** on ModelNet40 using PointNet++. PointMixup compares favorable to excluding interpolation and to the baselines, highlighting the benefits of our shortest path interpolation solution.

	No mixup	**Random assignment**		**Point sampling**		**PointMixup**	
Manifold mixup	×	×	✓	×	✓	×	✓
Full dataset							
Unaligned	90.7	90.8	91.1	90.9	91.4	91.3	**91.7**
Pre-aligned	91.9	91.6	91.9	92.2	92.5	92.3	**92.7**
Reduced dataset							
Unaligned	84.4	84.8	85.4	85.7	86.5	86.1	**86.6**
Pre-aligned	86.1	85.5	87.3	87.2	87.6	87.6	**88.6**

Table 31: **Evaluating our approach to other data augmentations** (left) **and its robustness to noise and transformations** (right). We find that our approach with manifold mixup (MM) outperforms augmentations such as label smoothing and other variations of mixup. For the robustness evaluation, we find that our approach with strong regularization power from manifold mixup provides more robustness to random noise and geometric transformations.

	PointMixup	
	×	✓
Baseline with no mixing	86.1	–
Mixup	–	87.6
Manifold mixup	–	**88.6**
Mix input, not labels	–	86.6
Mix input from same class	–	86.4
Mixup latent (layer 1)	–	86.9
Mixup latent (layer 2)	–	86.8
Label smoothing (0.1)	87.2	–
Label smoothing (0.2)	87.3	–

Transforms	**PointMixup**		
	×	w/o MM	w/ MM
Noise $\sigma^2 = 0.01$	91.3	91.9	**92.3**
Noise $\sigma^2 = 0.05$	35.1	51.5	**56.5**
Noise $\sigma^2 = 0.1$	4.03	4.27	**7.42**
Z-rotation [-30,30]	74.3	70.9	**77.8**
X-rotation [-30,30]	73.2	70.8	**76.8**
Y-rotation [-30,30]	87.6	87.9	**88.7**
Scale (0.6)	85.8	84.5	**86.3**
Scale (2.0)	59.2	67.7	**72.9**
DropPoint (0.2)	84.9	78.1	**90.9**

that exhibit characteristics of both classes, making for sensible point clouds and mixed labels, which in turn indicate why PointMixup is beneficial for point cloud classification.

6.4.3 *Evaluation on other Networks and Datasets*

With PointMixup, new point clouds are generated by interpolating existing point clouds. As such, we are agnostic to the type of network or dataset. To highlight this ability, we perform additional experiments on extra networks and an additional point cloud dataset.

PointMixup on other network architectures. We show the effect of PointMixup to two other networks, namely PointNet [187] and DGCNN [248]. The experiments are performed on ModelNet40. For PointNet, we perform the evaluation on the unaligned setting and for DGCNN with pre-aligned setting to remain consistent with the alignment choices made in the respective papers. The results are shown in Table 32 on the left. We find improvements when including PointMixup for both network architectures.

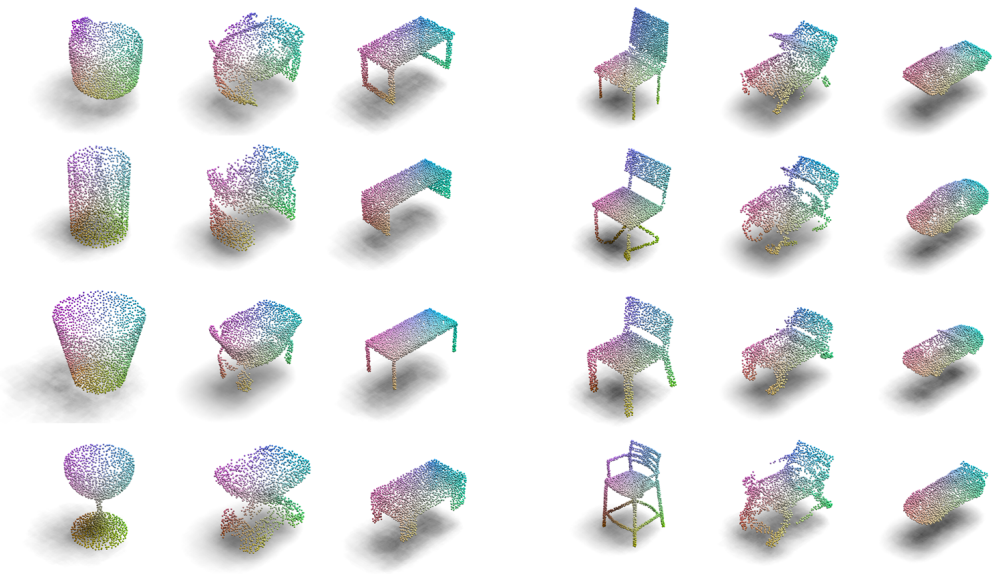


Figure 38: **Qualitative examples of PointMixup.** We provide eight visualizations of our interpolation. The four examples on the left show interpolations for different configurations of cups and tables. The four examples on the right show interpolations for different chairs and cars.

PointMixup on real-world point clouds. We also investigate PointMixup on point clouds from real-world object scans, using ScanObjectNN [235], which collects object from 3D scenes in SceneNN [111] and ScanNet [50]. Here, we rely on PointNet++ as network. The results in Table 32 on the right show that we can adequately deal with real-world point cloud scans, hence we are not restricted to point clouds from virtual scans. This result is in line with experiments on point cloud perturbations.

Table 32: **PointMixup on other networks** (top) **and another dataset** (bottom). We find our approach is beneficial regardless the network or dataset.

	PointNet		**DGCNN**		
	×	w/o MM	×	w/o MM	w/ MM
Full	89.2	**89.9**	92.7	92.9	**93.1**
Reduced	81.3	**83.4**	88.2	88.8	**89.0**

	ScanObjectNN		
	×	w/o MM	w/ MM
Standard	86.6	87.6	**88.5**
Perturbed	79.3	80.2	**80.6**

Table 33: **Evaluating PointMixup in the context of semi-supervised** (top) **and few-shot learning** (bottom). When examples are scarce, as is the case for both settings, using our approach provides a boost to the scores.

	Semi-supervised classification	
	Supervised	[238]+PointMixup
400 examples	69.4	**76.7**
600 examples	72.6	**80.8**
800 examples	73.5	**82.0**

	Few-shot classification	
	[215]	+ PointMixup
5-way 1-shot	72.3	**77.2**
5-way 3-shot	80.2	**82.2**
5-way 5-shot	84.2	**85.9**

6.4.4 *Beyond Standard Classification*

The fewer training examples available, the stronger the need for additional examples through augmentation. Hence, we train PointNet++ on ModelNet40 in both a few-shot and semi-supervised setting.

Semi-supervised learning. Semi-supervised learning learns from a dataset where only a small portion of data is labeled. Here, we show how PointMixup directly enables semi-supervised learning for point clouds. We start from Interpolation Consistency Training [238], a state-of-the-art semi-supervised approach, which utilizes Mixup between unlabeled points. Here, we use our Mixup for point clouds within their semi-supervised approach. We evaluate on ModelNet40 using 400, 600, and 800 labeled point clouds. The result of semi-supervised learning are illustrated in Table 33 on the left. Compared to the supervised baseline, which only uses the available labelled examples, our mixup enables the use of additional unlabelled training examples, resulting in a clear boost in scores. With 800 labelled examples, the accuracy increases from 73.5% to 82.0%, highlighting the effectiveness of PointMixup in a semi-supervised setting.

Few-shot learning. Few-shot classification aims to learn a classifier to recognize unseen classes during training with limited examples. We follow [68, 196, 215, 224, 239] to regard few-shot learning a typical meta-learning method, which learns how to learn from limited labeled data through training from a collection of tasks, *i.e.*, episodes. In an N-way K-shot setting, in each task, N classes are selected and K examples for each class are given as a support set, and the query set consists of the examples to be predicted. We perform few-shot classification on ModelNet40, from which we select 20 classes for training, 10 for validation, and 10 for testing. We utilize PointMixup within ProtoNet [215] by constructing mixed examples from the support set and update the model with the mixed examples before making predictions on the query set. We refer to the supplementary material for the details of our method and the settings. The results in Table 33 on the right show that incorporating our data augmentation provides a boost in

scores, especially in the one-shot setting, where the accuracy increases from 72.3% to 77.2%.

6.5 CONCLUSION

This work proposes PointMixup for data augmentation on point clouds. Given the lack of data augmentation by interpolation on point clouds, we start by defining it as a shortest path linear interpolation. We show how to obtain PointMixup between two point clouds by means of an optimal assignment interpolation between their point sets. As such, we arrive at a Mixup for point clouds, or latent point cloud representations in the sense of Manifold Mixup, that can handle permutation invariant nature. We first prove that PointMixup abides to our shortest path linear interpolation definition. Then, we show through various experiments that PointMixup matters for point cloud classification. We show that our approach outperforms baseline interpolations and regularizers. Moreover, we highlight increased robustness to noise and geometric transformations, as well as its general applicability to point-based networks and datasets. Lastly, we show the potential of our approach in both semi-supervised and few-shot settings. The generic nature of PointMixup allows for a comprehensive embedding in point cloud classification.

6.6 APPENDIX

6.6.1 *Proofs for the Properties of PointMixup Interpolation*

We provide detailed proofs for the shortest path property, the assignment invariance property and the linearity, stated in Section 3.4.

Proof for the shortest path property We denote $x_i \in S_1$ and $y_j \in S_2$ are the points in S_1 and S_2, then the generated $S_{\mathbf{OA}}^{(\lambda)} = \{u_i\}_{i=1}^N$ and $u_i = (1-\lambda)\cdot x_i + \lambda \cdot y_{\phi^*(i)}$, where ϕ^* is the optimal assignment from S_1 to S_2.

Then we suppose an identical one-to-one mapping ϕ_I such that $\phi_I(i) = i$. Then by definition of the EMD as the minimum transportation distance, so

$$d_{\text{EMD}}(S_1, S_{\mathbf{OA}}^{(\lambda)}) \leq \frac{1}{N}\sum_i \|x_i - u_{\phi_I(i)}\|_2, \tag{6.8}$$

where the right term of (6.8) is the transportation distance under identical assignment ϕ_I. Since $\frac{1}{N}\sum_i \|x_i - u_{\phi_I(i)}\|_2 = \frac{1}{N}\sum_i \|x_i - ((1-\lambda)\cdot x_i + \lambda \cdot y_{\phi^*(i)})\|_2 = \lambda\frac{1}{N}\sum_i \|x_i - y_{\phi^*(i)}\|_2 = \lambda \cdot d_{\text{EMD}}(S_1, S_2)$. Thus,

$$d_{\text{EMD}}(S_1, S_{\mathbf{OA}}^{(\lambda)}) \leq \lambda \cdot d_{\text{EMD}}(S_1, S_2). \tag{6.9}$$

Similarly as in (6.8) and (6.9), the following inequality (6.10) can be derived by assigning the correspondence from $S_{\mathbf{OA}}^{(\lambda)}$ to S_2 with ϕ^*:

$$d_{\text{EMD}}(S_{\mathbf{OA}}^{(\lambda)}, S_2) \leq (1-\lambda) \cdot d_{\text{EMD}}(S_1, S_2). \tag{6.10}$$

With (6.9) and (6.10),

$$d_{\mathrm{EMD}}(S_1, S_{\mathbf{OA}}^{(\lambda)}) + d_{\mathrm{EMD}}(S_2, S_{\mathbf{OA}}^{(\lambda)}) \leq d_{\mathrm{EMD}}(S_1, S_2). \tag{6.11}$$

However, as the triangle inequality holds for the EMD, *i.e.*

$$d_{\mathrm{EMD}}(S_1, S_{\mathbf{OA}}^{(\lambda)}) + d_{\mathrm{EMD}}(S_2, S_{\mathbf{OA}}^{(\lambda)}) \geq d_{\mathrm{EMD}}(S_1, S_2). \tag{6.12}$$

Then by summarizing (6.11) and (6.12), $d_{\mathrm{EMD}}(S_1, S_{\mathbf{OA}}^{(\lambda)}) + d_{\mathrm{EMD}}(S_2, S_{\mathbf{OA}}^{(\lambda)})$ $= d_{\mathrm{EMD}}(S_1, S_2)$ is proved.

Proof for the assignment invariance property We introduce two intermediate arguments. We begin with proving the first intermediate argument: ϕ_I is the optimal assignment from S_1 to $S_{\mathbf{OA}}^{(\lambda_1)}$. Similarly as in (6.9) ,(6.10) and (6.12) from the proof for Proposition 1, in order to allow all the three inequalities, the equal signs need to be taken for all of the three inequalities. Consider that the equal sign being taken for (6.9) is equivalent to the equal sign being taken for (6.8), then,

$$d_{\mathrm{EMD}}(S_1, S_{\mathbf{OA}}^{(\lambda_1)}) = \frac{1}{N} \sum_i \|x_i - u_{\phi_I(i)}\|_2, \tag{6.13}$$

which in turn means that ϕ_I is the optimal assignment from S_1 to $S_{\mathbf{OA}}^{(\lambda_1)}$ by the definition of the EMD. So the first intermediate argument is proved.

The second intermediate argument is that ϕ^* is the optimal assignment from $S_{\mathbf{OA}}^{(\lambda_1)}$ to S_2. This argument can be proved samely as the first one. Say the equal sign being taken for (6.10) is equivalent to that

$$d_{\mathrm{EMD}}(S_{\mathbf{OA}}^{(\lambda_1)}, S_2) = \frac{1}{N} \sum_i \|u_i - y_{\phi^*(i)}\|_2. \tag{6.14}$$

Thus, ϕ^* is the optimal assignment from $S_{\mathbf{OA}}^{(\lambda_1)}$ to S_2 is proved.

Then, with the two intermediate arguments, we can reformalize the setup to regard that $S_{\mathbf{OA}}^{(\lambda_2)}$ is interpolated from source pairs $S_{\mathbf{OA}}^{(\lambda_1)}$ and S_2 with the mix ratio $\frac{\lambda_2 - \lambda_1}{1 - \lambda_1}$, because the optimal assignment from $S_{\mathbf{OA}}^{(\lambda_1)}$ to S_2 is the same as the optimal assignment from S_1 to S_2. This argument then becomes an isomorphic with respect to the first intermediate argument. Then we prove that ϕ_I is the optimal assignment from $S_{\mathbf{OA}}^{(\lambda_1)}$ to $S_{\mathbf{OA}}^{(\lambda_2)}$ similarly as the proof for the first intermediate argument.

Proof for linearity We have shown that ϕ_I is optimal assignment between $S_{\mathbf{OA}}^{(\lambda_1)} = \{u_k\} = \{(1-\lambda_1) \cdot x_k + \lambda_1 \cdot y_{\phi^*(k)}\}$ and $S_{\mathbf{OA}}^{(\lambda_2)} = \{v_l\} = \{(1-\lambda_2) \cdot x_l + \lambda_2 \cdot y_{\phi^*(l)}\}$. Thus, $d_{\mathrm{EMD}}(S_{\mathbf{OA}}^{(\lambda_1)}, S_{\mathbf{OA}}^{(\lambda_2)}) = \frac{1}{N} \sum_k \|((1-\lambda_1) \cdot x_k + \lambda_1 \cdot y_{\phi^*(k)}) - ((1-\lambda_2) \cdot x_{\phi_I(k)} + \lambda_2 \cdot y_{\phi^*(\phi_I(k))})\|_2 = \frac{1}{N} \sum_k \|(\lambda_2 - \lambda_1)(x_k - y_{\phi^*(k)})\|_2 = (\lambda_2 - \lambda_1) \frac{1}{N} \sum_k \|(x_k - y_{\phi^*(k)})\|_2 = (\lambda_2 - \lambda_1) \cdot d_{\mathrm{EMD}}(S_1, S_2)$.

6.6.2 *Few-shot Learning with PointMixup*

We test if our PointMixup helps point cloud few-shot classification task, where a classifier must generalize to new classes not seen in the training set, given only a small number of examples of each new class. We take ProtoNet [215] as the baseline method for few-shot learning, and PointNet++ [188] is the feature extractor h_θ.

Episodic learning setup. ProtoNet takes the episodic training for few-shot learning, where an episode is designed to mimic the few-shot task by subsampling classes as well as data. A N_C-way N_S-shot setting is defined as that in each episode, data from N_C classes are sampled and N_S examples for each class is labelled. In the i^{th} episode of training, the dataset $\mathcal{D}_i$ consists of the training example and class pairs from N_C classes sampled from all training classes. Denote $\mathcal{D}_i^S \subset \mathcal{D}_i$ is the support set which consists of labelled data from N_C classes with N_S examples, and $\mathcal{D}_i^Q = \mathcal{D}_i \backslash \mathcal{D}_i^S$ is the query set which consists of unlabelled examples to be predicted.

Baseline method for few-shot classification: ProtoNet [215]. In each episode $\mathcal{D}_i$, ProtoNet computes a prototype as the mean of embedded support examples $\bar{z}_c$ for each class c, from all examples from the support set $\mathcal{D}_i^S$. The latent embedding is from the network h_θ (for which we use PointNet++ [188] without the last fully-connected layer). Then each example S from the query set $\mathcal{D}_i^Q$ is classified into a label distribution by a softmax over (negative) distance to the class prototypes:

$$p(\hat{y} = c|S) = \frac{\exp(-d(S, \bar{z}_c))}{\sum_{c'} \exp(-d(S, \bar{z}_{c'}))},$$

Algorithm 2 Episodic training of ProtoNet with PointMixUp. From line 3 to line 8 is where PointMixUp takes a role in addition to the ProtoNet baseline. Testing stage is similar as training stage, but without line 13 and line 14 which learn new weight from query examples.

Require: Set of sampled episodes $\{\mathcal{D}_i\}$, where $\mathcal{D}_i = \mathcal{D}_i^S \cup \mathcal{D}_i^Q$ denoting the support and query sets
Require: h_θ: feature extractor network: input $\rightarrow$ latent embedding
1: randomly initialize θ
2: **for** episode i **do**
3: **for** class c **do**
4: calculate prototype $\bar{z}_c$ from $\mathcal{D}_i^S$, with h_θ.
5: **end for**
6: Construct Mixup samples $\mathcal{D}_i^{\text{mix}}$ from support set $\mathcal{D}_i^S$.
7: Predict the label distributions for mixed examples in $\mathcal{D}_i^{\text{mix}}$, with distance to $\bar{z}_c$.
8: Update θ with prediction from mixed examples, as episode-specific weights θ_i.
9: **for** class c **do**
10: calculate new prototype $\bar{z}_c^{(\theta_i)}$ from $\mathcal{D}_i^S$, with h_{θ_i}
11: **end for**
12: Predict the label distributions for query examples in $\mathcal{D}_i^Q$, with distance to $\bar{z}_c^{(\theta_i)}$.
13: Update θ_i with prediction from query examples.
14: $\theta \leftarrow \theta_i$
15: **end for**
16: **return** θ

where $d(\cdot,\cdot)$ is the Eudlidean distance in the embedding space. In training stage, the weights θ for the feature extractor h_θ is updated by the cross-entropy loss for the predicted query label distribution and the ground truth.

Few-shot point cloud classification with PointMixup. We use PointMixup to learn a better embedding space for each episode. Instead of using the h_θ directly to predict examples from query set, we learn a episode-specific weight θ_i from the mixed data, and the query examples are predicted by h_{θ_i}. We use PointMixup to construct a mixed set $\mathcal{D}_i^{\text{mix}}$ from the labelled support set $\mathcal{D}_i^S$, which consists of examples from $\binom{N_c}{2}$ class pairs and for each class pairs N_s mixed examples are constructed from randomly sampling support examples. Then the weight θ is updated as θ_i from backprop the loss from the prediction of mixed examples from $\mathcal{D}_i^{\text{mix}}$. After that, the label of query examples from $\mathcal{D}_i^Q$ is then predicted with the updated feature extractor h_{θ_i}. See Algorithm 2 for an illustration of the learning scheme.

6.6.3 *Further Discussion on Interpolation Variants*

The proposed PointMixUp adopts *Optimal Assignment (OA) interpolation* for point cloud because of its advantages in theory and in practice. To compare Optimal Assignment interpolation with the two alternative strategies, *Random Assignment (RA) interpolation* and *Point Sampling (PS) interpolation*, the proposed PointMixUp with OA interpolation is the best performing strategy, followed by PS interpolation. RA interpolation, which has a non-shortest path definition of interpolation, does not perform well.

Here we extend the discussion on the two alternative interpolation strategies, through which we analyze the possible advantages and limitations under certain conditions, which in turn validates our choice of applying Optimal Assignment interpolation for PointMixup.

Random Assignment interpolation. From our shortest path interpolation hypothesis for Mixup, the inferiority of RA interpolation comes from that it does not obey the shortest path interpolation rule, so that the mixed point clouds from different source examples can easily entangle with each other. From Section 6.4.1 in the main chapter, the Random assignment interpolation produces chaotic mixed examples which can hardly been recognized with the feature from the source class point clouds. Thus, RA interpolation fails especially under heavy Mixup (the value of λ is large).

Point Sampling interpolation: yet another shortest path interpolation. Point Sampling interpolation performs relatively well in PointNet++ and sometimes comparable with the Optimal Assignment interpolation. From Fig. 3 in the main chapter, the PS interpolation produces mixed examples which can be recognized which classes of source data it comes from.

Reviewing the shortest path interpolation hypothesis, We argue that when the number of points N is large enough, or say $N \to \infty$, Point Sampling interpolation also (approximately) defines a shortest path on the metric space $(\mathcal{S}, d_{\text{EMD}})$ (Note that given the initial and the final points, the shortest path in $(\mathcal{S}, d_{\text{EMD}})$ is not unique). This is a bit counter-intuitive, but reasonable.

We show the *shortest path property*. Recall that point sampling interpolation randomly draws without replacement of points from each set are made according to the sampling

Table 34: **Different interpolation strategies on PointNet [187]** Following the original paper [187] we test on unaligned setting. PS interpolation fails with PointNet as a density-invariant model. The numbers are accuracy in percentage.

Baseline	PointMixup	Random Assignment	Point Sampling
89.2	**89.9**	88.2	88.7

frequency λ: $S_{\mathbf{PS}}^{(\lambda)} = S_1^{(1-\lambda)} \cup S_2^{(\lambda)}$, where $S_2^{(\lambda)}$ denotes a randomly sampled subset of S_2, with $\lfloor \lambda N \rfloor$ elements. ($\lfloor \cdot \rfloor$ is the floor function.) And similar for $S_1^{(1-\lambda)}$ with $N - \lfloor \lambda N \rfloor$ elements, such that $S_{\mathbf{PS}}^{(\lambda)}$ contains exactly N points. Imagine that a subset $S_1^{(1-\lambda)}$ with a number of $N - \lfloor \lambda N \rfloor$ points in $S_{\mathbf{PS}}^{(\lambda)}$ are identical with that in S_1. For $d_{\text{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_1)$, the optimal assignment will return these identical points as matched pairs, thus they contribute zero to the overall EMD distance. Thus,

$$
\begin{aligned}
d_{\text{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_1) &= \frac{N - \lfloor \lambda N \rfloor}{N} d_{\text{EMD}}(S_{\mathbf{PS}}^{(\lambda)} \setminus S_1^{(1-\lambda)}, S_1 \setminus S_1^{(1-\lambda)}) \\
&= \frac{N - \lfloor \lambda N \rfloor}{N} d_{\text{EMD}}(S_2^{(\lambda)}, S_1 \setminus S_1^{(1-\lambda)}) \\
&\approx \frac{N - \lfloor \lambda N \rfloor}{N} d_{\text{EMD}}(S_2, S_1) \\
&\approx (1 - \lambda) \cdot d_{\text{EMD}}(S_1, S_2),
\end{aligned}
$$

from which $d_{\text{EMD}}(S_2^{(\lambda)}, S_1 \setminus S_1^{(1-\lambda)}) \approx d_{\text{EMD}}(S_2, S_1)$ is because that S_1 and $S_1 \setminus S_1^{(1-\lambda)}$ are the point clouds representing the same shape but with different density, and the same with S_2 and $S_2^{(\lambda)}$.

Similarly, $d_{\text{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_2) \approx \lambda \cdot d_{\text{EMD}}(S_1, S_2)$, and thus $d_{\text{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_1) + d_{\text{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_2) = d_{\text{EMD}}(S_1, S_2)$, which in turn proves the shortest path property.

We note that the *linearity* of PS interpolation w.r.t. d_{EMD} also holds and the proof can be derived similarly. Thus, although strictly not an ideally continuous interpolation path, PS interpolation is (appoximately) a shortest path linear interpolation in $(\mathcal{S}, d_{\text{EMD}})$, which explains its good performance.

Point Sampling interpolation: limitations. The limitation of PS interpolation is from that the mix ratio λ controls change of local density distribution, but the underlying shape does not vary with λ. So, as shown in Table 34, PS interpolation fails with PointNet [187], which is ideally invariant to the point density, because a max pooling operation aggregates the information from all the points.

A question which may come with PS interpolation is that how it performs relatively well with PointNet++, which is also designed to be density-invariant. This is due to the sampling and grouping stage. PointNet++ takes same operation as PointNet in learning features, but in order to be hierarchical, the sampling and grouping stage, especially the farthest point sampling (fps) operation is not invariant to local density changes such that it samples different groups of farthest points, resulting in different latent point cloud feature representations. Thus, PointNet++ is invariant to global density but not invariant to local density differences, which makes PS interpolation as a working strategy for PointNet++. However, we may still expect that the performance of Mixup based on PS

interpolation is limited, because it does not work well with PointNet as a basic component in PointNet++.

By contrast, the proposed PointMixup with OA interpolation strategy is not limited by the point density invariance. As a well established interpolation, OA interpolation smoothly morphes the underlying shape. So we claim that OA interpolation is a more generalizable strategy.

7

SELF-GUIDED DIFFUSION MODELS

7.1 INTRODUCTION

Diffusion models have recently enabled tremendous advancements in many computer vision fields related to image synthesis, but counterintuitively this often comes with the cost of requiring large annotated datasets [193, 203]. For example, the image fidelity of samples from diffusion models can be spectacularly enhanced by conditioning on class labels [54]. Classifier guidance goes a step further and offers control over the alignment with the class label, by using the classifier gradient to guide the image generation [54]. Classifier-free guidance [100] replaces the dedicated classifier with a diffusion model trained by randomly dropping the condition during training. This has proven a fruitful line of research for several other condition modalities, such as text [194, 203], image layout [200], visual neighbors [8], and image features [78]. However, all these conditioning and guidance methods require ground-truth annotations. In many domains, this is an unrealistic and too costly assumption. For example, medical images require domain experts to annotate very high-resolution data, which is infeasible to do exhaustively [180]. In this paper, we propose to remove the necessity of ground-truth annotation for guided diffusion models.

We are inspired by progress in self-supervised learning [35, 42], which encodes images into semantically meaningful latent vectors without using any label information. It usually does so by solving a pretext task [6, 79, 93, 277] on image-level to remove the necessity of labels. This annotation-free paradigm enables the representation learning to upscale to larger and more diverse image datasets [72]. The holistic image-level self-supervision has recently been extended to more expressive dense representations, including bounding boxes (e.g., [164, 212]) and pixel-precise segmentation masks (e.g., [88, 289]). Some self-supervised learning methods even outperform supervised alternatives [35, 93]. We hypothesize that for diffusion models, self-supervision may also provide a flexible and competitive, possibly even stronger guidance signal than ground-truth labeled guidance.

In this paper, we propose *self-guided diffusion models*, a framework for image generation using guided diffusion without the need for any annotated image-label pairs, the detailed structure is shown in Figure 39. The framework encompasses a feature extraction function and a self-annotation function, that are compatible with recent self-supervised learning advances. Furthermore, we leverage the flexibility of self-supervised

Published in Proceedings of the Computer Vision and Pattern Recognition Conference, 2023 [110].

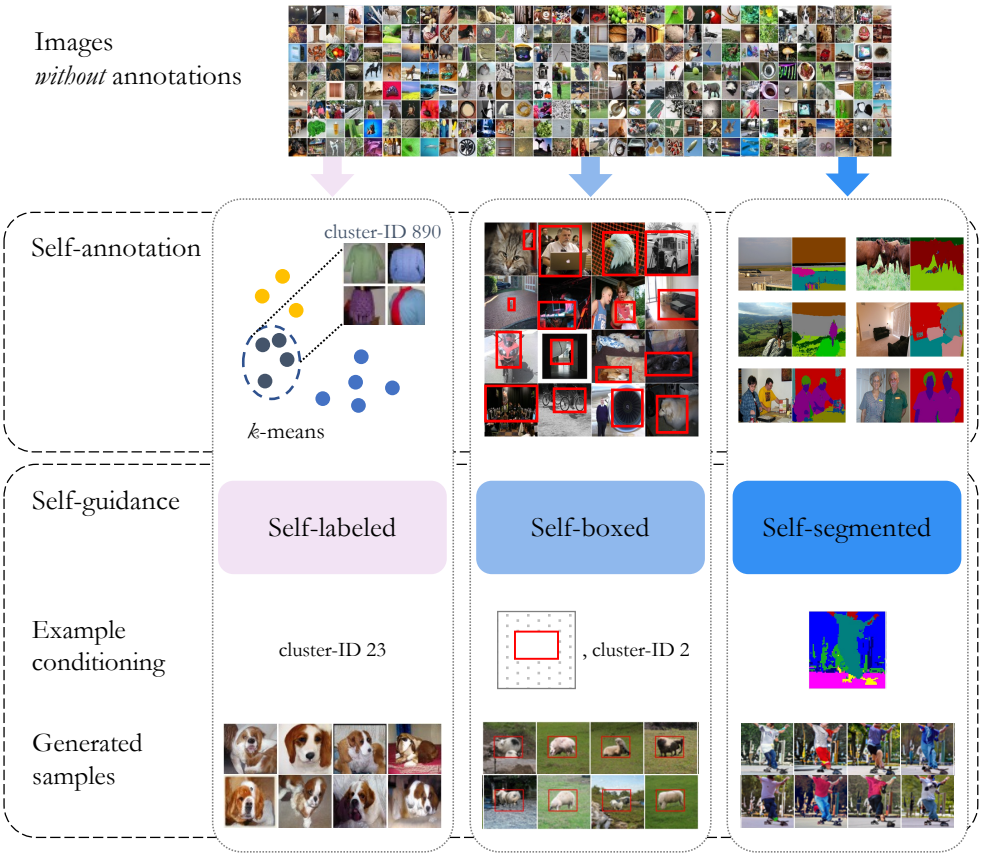


Figure 39: **Self-guided diffusion framework.** Our method can leverage large and diverse image datasets *without* any annotations for training guided diffusion models. Starting from a dataset without ground-truth annotations, we apply a self-supervised feature extractor to create self-annotations. Using these, we train diffusion models with either self-labeled, self-boxed, or self-segmented guidance that enable controlled generation and improved image fidelity.

learning to generalize the guidance signal from the holistic image level to (unsupervised) local bounding boxes and segmentation masks for more fine-grained guidance. We demonstrate the potential of our proposal on single-label and multi-label image datasets, where self-labeled guidance always outperforms diffusion models without guidance and may even surpass guidance based on ground-truth labels. When equipped with self-supervised box or mask proposals, our method further generates visually diverse yet semantically consistent images, without the need for any class, box, or segment label annotation.

7.2 RELATED WORK

CONDITIONAL GENERATIVE MODELS. Earlier works on generative adversarial networks (GANs) have observed improvements in image quality by conditioning on ground-truth labels [26, 36, 171]. Recently, conditional diffusion models have reported similar improvements, while also offering a great amount of controllability via classifier-free guidance by training on images paired with textual descriptions [193, 194, 203], semantic segmentations [246], or other modalities [24, 219, 271]. Our work also aims

to realize the benefits of conditioning and guidance, but instead of relying on additional human-generated supervision signals, we leverage the strength of pretrained self-supervised visual encoders.

Zhou *et al.* [287] train a GAN for text-to-image generation without any image-text pairs, by leveraging the CLIP [190] model that was pretrained on a large collection of paired data. In this work, we do not assume any paired data for the generative models and rely purely on images. Additionally, image layouts are difficult to be expressed by text, thus our self-boxed and self-segmented methods are complementary to text conditioning. Instance-Conditioned GAN [36], Retrieval-augmented Diffusion [20] and KNN-diffusion [8] are three recent methods that utilize nearest neighbors as guidance signals in generative models. Similar to our work, these methods rely on conditional guidance from an unsupervised source, we differ from them by further attempting to provide more diverse *spatial* guidance, including (self-supervised) bounding boxes and segmentation masks.

SELF-SUPERVISED LEARNING IN GENERATIVE MODELS. Self-supervised learning [6, 34, 35, 42] has shown great potential for representation learning in many downstream tasks. As a consequence, it is also commonly explored in GAN for evaluation and analysis [172], conditioning [36, 162], stabilizing training [43], reducing labeling costs [157] and avoiding mode collapse [4]. Our work focuses on translating the benefits of self-supervised methods to the generative domain and providing flexible guidance signals to diffusion models at various image granularities. In order to analyze the feature representation from self-supervised models, Bordes *et al.* [24] condition on self-supervised features in their diffusion model for better visualization in data space. We instead condition on the compact clustering after the self-supervised feature, and further introduce the elasticity of self-supervised learning into diffusion models for multi-granular image generation.

7.3 APPROACH

Before detailing our self-guided diffusion framework, we provide a brief background on diffusion models and the classifier-free guidance technique.

7.3.1 *Background*

DIFFUSION MODELS. Diffusion models [99, 216] gradually add noise to an image $\mathbf{x}_0$ until the original signal is fully diminished. By learning to reverse this process one can turn random noise $\mathbf{x}_T$ into images. This diffusion process is modeled as a Gaussian process with Markovian structure:

$$\begin{aligned} q(\mathbf{x}_t|\mathbf{x}_{t-1}) &:= \mathcal{N}(\mathbf{x}_t;\ \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \\ q(\mathbf{x}_t|\mathbf{x}_0) &:= \mathcal{N}(\mathbf{x}_t;\ \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I}, \end{aligned} \tag{7.1}$$

where $\beta_1, \ldots, \beta_T$ is a fixed variance schedule on which we define $\alpha_t \coloneqq 1 - \beta_t$ and $\overline{\alpha}_t \coloneqq \prod_{s=1}^{t} \alpha_s$. All latent variables have the same dimensionality as the image $\mathbf{x}_0$ and differ by the proportion of the retained signal and added noise.

Learning the reverse process reduces to learning a denoiser $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)$ that recovers the original image as $\left(\mathbf{x}_t - (1-\overline{\alpha}_t)\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right) / \sqrt{\overline{\alpha}_t} \approx \mathbf{x}_0$. Ho *et al.* [99] optimize the parameters $\boldsymbol{\theta}$ of noise prediction network by minimizing:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\epsilon},\mathbf{x},t}\left[\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \boldsymbol{\epsilon}\|_2^2\right], \tag{7.2}$$

in which $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\mathbf{x} \in \mathcal{D}$ is a sample from the training dataset $\mathcal{D}$ and the noise prediction function $\boldsymbol{\epsilon}_\theta(\cdot)$ are encouraged to be as close as possible to $\boldsymbol{\epsilon}$.

The standard sampling [99] requires many neural function evaluations to get good quality samples. Instead, the faster Denoising Diffusion Implicit Models (DDIM) sampler [217] has a non-Markovian sampling process:

$$\mathbf{x}_{t-1} \quad = \quad \sqrt{\overline{\alpha}_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{\overline{\alpha}_t}}\right) + \sqrt{1 - \overline{\alpha}_{t-1} - \sigma_t^2} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) + \sigma_t \boldsymbol{\epsilon}, \tag{7.3}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is Gaussian noise independent of $\mathbf{x}_t$.

CLASSIFIER-FREE GUIDANCE. To trade off mode coverage and sample fidelity in a conditional diffusion model, Dhariwal and Nichol [54] propose to guide the image generation process using the gradients of a classifier, with the additional cost of having to train the classifier on noisy images. Motivated by this drawback, Ho and Salimans [100] introduce label-conditioned guidance that does not require a classifier. They obtain a combination of a conditional and unconditional network in a single model, by randomly dropping the guidance signal $\mathbf{c}$ during training. After training, it empowers the model with progressive control over the degree of alignment between the guidance signal and the sample by varying the guidance strength w:

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t;\ \mathbf{c}, w) = (1-w)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) + w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t;\ \mathbf{c}). \tag{7.4}$$

A larger w leads to greater alignment with the guidance signal, and vice versa. Classifier-free guidance [100] provides progressive control over the specific guidance direction at the expense of labor-consuming data annotation. In this paper, we propose to remove the necessity of data annotation using a self-guided principle based on self-supervised learning.

7.3.2 *Self-Guided Diffusion Models*

The equations describing the diffusion model for classifier-free guidance implicitly assume dataset $\mathcal{D}$ and its images each come with a single manually annotated class label. We prefer to make the label requirement explicit. We denote the human annotation

process as the function $\xi(\mathbf{x}; \mathcal{D}, C) : \mathcal{D} \rightarrow C$, where C defines the annotation taxonomy, and plug this into Equation (7.4):

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t;\ \ \xi(\mathbf{x};\ \ \mathcal{D}, C), w) \quad = \quad (1 - w)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) + w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t;\ \ \xi(\mathbf{x};\ \ \mathcal{D}, C)). \quad (7.5)$$

We propose to replace the supervised labeling process ξ with a self-supervised process that requires *no* human annotation:

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t; f_\psi(g_\phi(\mathbf{x};\ \mathcal{D});\ \mathcal{D}), w) \ = \ (1 - w)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) + w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t;\ f_\psi(g_\phi(\mathbf{x};\ \mathcal{D});\ \mathcal{D}), \quad (7.6)$$

where g is a self-supervised feature extraction function parameterized by ϕ that maps the input data to feature space $\mathcal{H}$, $g : \mathbf{x} \rightarrow g_\phi(\mathbf{x}), \forall \mathbf{x} \in \mathcal{D}$, and f is a self-annotation function parameterized by ψ to map the raw feature representation to the ultimate guidance signal $\mathbf{k}$, $f_\psi : g_\phi(\cdot; \mathcal{D}) \rightarrow \mathbf{k}$. The guidance signal $\mathbf{k}$ can be any form of *annotation*, e.g., label, box, pixel, that can be paired with an image, which we derive by $\mathbf{k}{=}f_\psi(g_\phi(\mathbf{x};\ \mathcal{D});\ \mathcal{D})$. The choice of the self-annotation function f can be non-parametric by heuristically searching over dataset $\mathcal{D}$ based on the extracted feature $g_\phi(\cdot;\ \mathcal{D})$, or parametric by fine-tuning on the feature map $g_\phi(\cdot;\ \mathcal{D})$.

For the noise prediction function $\boldsymbol{\epsilon}_\theta(\cdot)$, we adopt the traditional UNet network architecture [201] due to its superior image generation performance, following [99, 193, 203, 220].

Stemming from this general framework, we present three methods working at different spatial granularities, all without relying on any ground-truth labels. Specifically, we cover image-level, box-level, and pixel-level guidance by setting the feature extraction function $g_\phi(\cdot)$, self-annotation function $f_\psi(\cdot)$, and guidance signal $\mathbf{k}$ to an approximate form.

SELF-LABELED GUIDANCE. To achieve self-labeled guidance, we need a self-annotation function f that produces a representative guidance signal $\mathbf{k} \in \mathbb{R}^K$. Firstly, we need an embedding function $g_\phi(\mathbf{x}), \mathbf{x} \in \mathcal{D}$ which provides semantically meaningful image-level guidance for the model. We obtain $g_\phi(\cdot)$ in a self-supervised manner by mapping from image space, $g_\phi(\cdot) : \mathbb{R}^{W \times H \times 3} \rightarrow \mathbb{R}^C$, where W and H are image width and height and C is the feature dimension. We may use any type of feature for the feature embedding function g, which we will vary and validate in the experiments. As the image-level feature $g_\phi(\cdot;\ \mathcal{D})$ is not compact enough for guidance, we further conduct a non-parametric clustering algorithm, e.g., k-means, as our self-annotation function f. For all features $g_\phi(\cdot)$, we obtain the self-labeled guidance via self-annotation function $f_\psi(\cdot) : \mathbb{R}^C \rightarrow \mathbb{R}^K$. Motivated by [199], we use a one-hot embedding $\mathbf{k} \in \mathbb{R}^K$ for each image to achieve a compact guidance.

We inject the guidance information into the noise prediction function $\boldsymbol{\epsilon}_\theta$ by concatenating it with timestep embedding t and feed the concatenated information $\texttt{concat}[t, \mathbf{k}]$ into every block of the UNet. Thus, the noise prediction function $\boldsymbol{\epsilon}_\theta$ is rewritten as:

$$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t;\ \mathbf{k}) = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \texttt{concat}[t,\ \mathbf{k}]), \quad (7.7)$$

where $\mathbf{k}{=}f_\psi(g_\phi(\mathbf{x};\ \mathcal{D});\ \mathcal{D})$ is the self-annotated image-level guidance signal. For simplicity, we ignore the self-annotation function $f_\psi(\cdot)$ here and in the later text. Self-labeled guidance focuses on image-level global guidance. Next, we consider a more fine-grained spatial guidance.

SELF-BOXED GUIDANCE. Bounding boxes specify the location of an object in an image [33, 198] and complement the content information provided by class labels. Our self-boxed guidance approach aims to attain this signal via self-supervised models. We represent the bounding box as a binary mask $\mathbf{k}_s \in \mathbb{R}^{W\times H}$ rather than coordinates, where 1 indicates that the pixel is inside the box and 0 outside. This design directly aligns the image and mask along the spatial dimensions. We propose the self-annotation function f that obtains bounding box $\mathbf{k}_s$ by mapping from feature space $\mathcal{H}$ to the bounding box space via $f_\psi(\cdot;\ \mathcal{D}) : \mathbb{R}^{W\times H\times C} \rightarrow \mathbb{R}^{W\times H}$, and inject the guidance signal by concatenating in the channel dimension: $\mathbf{x}_t \coloneqq \texttt{concat}[\mathbf{x}_t, \mathbf{k}_s]$. Usually in self-supervised learning, the derived bounding box is class-agnostic [240, 241]. To inject a self-supervised pseudo label to further enhance the guidance signal, we again resort to clustering to obtain $\mathbf{k}$ and concatenate it with the time embedding $t \coloneqq \texttt{concat}[t, \mathbf{k}]$. To incorporate such guidance, we reformulate the noise prediction function $\boldsymbol{\epsilon}_\theta$ as:

$$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t;\ \mathbf{k}_s, \mathbf{k}) = \boldsymbol{\epsilon}_\theta(\texttt{concat}[\mathbf{x}_t, \mathbf{k}_s], \texttt{concat}[t, \mathbf{k}]), \tag{7.8}$$

in which $\mathbf{k}_s$ is the self-supervised box guidance obtained by self-annotation functions f_ψ, $\mathbf{k}$ is the self-supervised image-level guidance from clustering. $\mathbf{k}_s$ and $\mathbf{k}$ denotes the location and class information, respectively. The design of f_ψ is flexible as long as it obtains self-supervised bounding boxes by $f_\psi(\cdot;\ \mathcal{D}) : \mathbb{R}^{W\times H\times C} \rightarrow \mathbb{R}^{W\times H}$. Self-boxed guidance guides the diffusion model by boxes, which specifies the box area in which the object will be generated. Sometimes, we may need an even finer granularity, e.g., pixels, which we detail next.

SELF-SEGMENTED GUIDANCE. Compared to a bounding box, a segmentation mask is a more fine-grained signal. Additionally, a multichannel mask is more expressive than a binary foreground-background mask. Therefore, we propose a self-annotation function f that acts as a plug-in built on feature $g_\phi(\cdot;\ \mathcal{D})$ to extract the segmentation mask $\mathbf{k}_s$ via function mapping $f_\psi(\cdot;\ \mathcal{D}) : \mathbb{R}^{W\times H\times C} \rightarrow \mathbb{R}^{W\times H\times K}$, where K is the number of segmentation clusters.

To inject the self-segmented guidance into the noise prediction function $\boldsymbol{\epsilon}_\theta$, we consider two pathways for injection of such guidance. We first concatenate the segmentation mask to $\mathbf{x}_t$ in the channel dimension, $\mathbf{x}_t \coloneqq \texttt{concat}[\mathbf{x}_t, \mathbf{k}_s]$, to retain the spatial inductive bias of the guidance signal. Secondly, we also incorporate the image-level guidance to further amplify the guidance signal along the channel dimension. As the segmentation mask from the self-annotation function f_ψ already contains image-level information, we do not apply the image-level clustering as before in our self-labeled guidance. Instead, we directly derive the image-level guidance from the self-annotation result $f_\psi(\cdot)$ via spatial maximum pooling: $\mathbb{R}^{W\times H\times K} \rightarrow \mathbb{R}^K$, and feed the image-level guidance $\hat{\mathbf{k}}$ into the noise prediction function via concatenating it with the timestep embedding $t \coloneqq \texttt{concat}[t, \hat{\mathbf{k}}]$.

The concatenated results will be sent to every block of the UNet. In the end, the overall noise prediction function for self-segmented guidance is formulated as:

$$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t;\ \mathbf{k}_s, \hat{\mathbf{k}}) = \boldsymbol{\epsilon}_\theta(\texttt{concat}[\mathbf{x}_t, \mathbf{k}_s],\ \texttt{concat}[t, \hat{\mathbf{k}}]), \tag{7.9}$$

in which $\mathbf{k}_s$ is the spatial mask guidance obtained from self-annotation function f, $\hat{\mathbf{k}}$ is a multi-hot image-level guidance derived from the self-supervised learning mask $\mathbf{k}_s$.

We have described three variants of self-guidances by setting the feature extraction function $g_\phi(\cdot)$, self-annotation function $f_\psi(\cdot)$, guidance signal $\mathbf{k}$ to an approximate form. In the end, we arrive at three noise prediction functions $\boldsymbol{\epsilon}_\theta$, which we utilize for diffusion model training and sampling, following the standard guided [100] diffusion approach as detailed in Section 7.3.1.

7.4 EXPERIMENTS

In this section, we aim to answer the overarching question: Can we substitute ground-truth annotations with self-annotations? First, we consider the image-label setting, in which we examine what kind of self-labeling is required to improve image fidelity. In addition, we explore what semantic concepts are induced by self-labeling approaches that broaden the control over the content beyond the standard ground-truth labels. Next, we look at image-bounding box pairs. Finally, we examine whether it is possible to gain fine-grained control with self-labeled image-segmentation pairs. We first present the general settings relevant for all experiments.

EVALUATION METRIC. We evaluate both diversity and fidelity of the generated images by the Fréchet Inception Distance (FID) [98], as it is the de facto metric for the evaluation of generative methods, e.g., [26,54,125,203]. It provides a symmetric measure of the distance between two distributions in the feature space of Inception-V3 [228]. We use FID as our main metric for the sampling quality.

BASELINES & IMPLEMENTATION DETAILS. As baselines, we compare against both the unconditional diffusion model and a diffusion model trained with classifier-free guidance using ground-truth annotations [100]. We use the same neural network and hyperparameters for the baselines and our method. Note that applying more training steps generally tends to further improve the performance [130, 203], thus to facilitate a fair comparison we use the same computational budget in every experiment when comparing the baselines to our proposed method. We use DDIM [217] samplers with 250 steps, σ_t=0 to efficiently generate samples. For details on the hyperparameters, we refer to Section 7.6.2 . Source code can be found at: `https://taohu.me/sgdm/`.

7.4.1 *Self-Labeled Guidance*

We use ImageNet32/64 [53] and CIFAR100 [135] to validate the efficacy of self-labeled guidance. On ImageNet, we also measure the Inception Score (IS) [204], following common practice [26, 54, 125]. IS measures how well a model fits into the full ImageNet class distribution.

	FID↓	IS↑
Label-supervised		
ResNet50	22.00	8.23
ViT-B/16	22.30	7.81
Self-supervised		
MAE ViTBase	32.58	8.20
SimCLR-v2	23.16	9.35
MSN ViT-B/16	21.16	**10.59**
DINO ViT-B/16	**19.35**	10.41

Table 35: **Choice of feature extraction function** on ImageNet32. DINO and MSN `ViT-B/16` obtain good trade-offs between FID and IS.

Figure 40: **Effect of number of clusters.** Self-labeled guidance outperforms DDPM without any guidance beyond a single cluster, is competitive with classifier-free guidance beyond 1,000 clusters and is even able to outperform guidance by ground-truth (GT) labels for 5,000 clusters. We visualize generated samples from ImageNet64 (middle) and ImageNet32 (right) for ground-truth labels guidance (top) and self-labeled guidance (bottom). More qualitative results in Section 7.6.3 .

CHOICE OF FEATURE EXTRACTION FUNCTION $\mathbf{G}$. We first measure the influence of the feature extraction function g used before clustering. We consider two supervised feature backbones: `ResNet50` [95] and `ViT-B/16` [57], and four self-supervised backbones: `SimCLR` [42], `MAE` [92], `MSN` [9] and `DINO` [35]. To assure a fair comparison, we use 10k clusters for all architectures. From the results in Table 35, we make the following observations. First, features from the supervised `ResNet50`, and `ViT-B/16` lead to a satisfactory FID performance, at the expense of relatively limited diversity (low IS). However, they still require label annotation, which we strive to avoid in our work. Second, among the self-supervised feature extraction functions, the `MSN`- and `DINO`-pretrained ViT backbones have the best trade-off in terms of both FID and IS. They even improve over the label-supervised backbones. This implies that the benefits of guidance is not unique to human annotated labels and self-supervised learning can provide a much more scalable alternative. Since `DINO ViT-B/16` achieves the best FID performance, from now on we pick it as our self-supervised feature extraction function g.

EFFECT OF NUMBER OF CLUSTERS. Next, we ablate the influence of the number of clusters on the overall sampling quality. We consider 1, 10, 100, 500, 1,000, 5,000 and 10,000 clusters on the extracted `CLS` token from the `DINO ViT-B/16` feature. For

efficient, yet uniform, comparison we only run 20 epochs on ImageNet32. To put our sampling results in perspective, we also provide FID and IS results for the no guidance and ground-truth guidance baselines. From the result in Figure 40 we first observe that when the cluster number is ranging from 1 to 5,000, our model's performance monotonously improves and always surpasses the no guidance baseline. At 1,000 clusters, self-labeled guidance is competitive with the baseline trained using ground-truth labels. For 5,000 clusters, there is a sweet spot where our method outperforms the model using ground-truth labels with an FID of 16.4 vs. 17.9 and an IS of 10.35 vs. 9.94. See also the generated images in Figure 40. The FID performance starts to deteriorate between 5,000 and 10,000 clusters. When the cluster number becomes too large the self-supervised clusters tend to pick up on dataset-specific details that no longer correspond to general semantic concepts. We suspect that this deterioration in cluster quality also leads to the observed deterioration in FID performance. On the other hand, we observe that samples generated from the same cluster id have high semantic coherence. This signifies that the self-supervised clusters represent semantically meaningful concepts and can be used to control the generation process. We provide further discussion on how to assign semantic descriptions to clusters in Section 7.6.3.

IMPORTANCE OF SELF-SUPERVISED CLUSTERS. In the previous paragraph, we observed that training a diffusion model with 5,000 clusters can outperform the 1,000 ground-truth labels. Here, we check whether we can reproduce this result on another dataset and examine how the performance varies when we inject different degrees of noise into the cluster assignment. On CIFAR100 [135] we compare the ground-truth 100 labels with 400 self-supervised clusters. We corrupt the cluster assignments at different levels by randomly shuffling the cluster id for 25% to 100% of the images before training. The results in Figure 41 highlight the importance of using self-supervised features for assigning clusters and that assigning cluster ids for a subset of the dataset is already sufficient to see improvements.

SELF-LABELED COMPARISONS ON IMAGENET32/64. We compare our self-labeled guidance method against the baseline trained with ground-truth labels. *For a fair comparison, we use the same compute budget for all runs*. In particular, each model is trained for 100 epochs taking around 6 days on four RTX A5000 GPUs. Results on ImageNet32 and ImageNet64 are in Table 36. Similar to [54], we observe that any guidance setting improves considerably over the unconditional & no-guidance model. Surprisingly, our self-labeled model even outperforms the ground-truth labels by a large gap in terms of FID of 1.9 and 4.7 points respectively. We hypothesize that the ground-truth taxonomy might be suboptimal for learning generative models and the self-supervised clusters offer a better guidance signal due to better alignment with the visual similarity of the images. In Figure 42 we report the FID at different training stages. It is worth noting that the performance advantage of our self-guided method remains consistent over the entire training process. The results suggest that the label-conditioned guidance from [100] can be completely replaced by guidance from self-supervision, which would enable guided diffusion models to learn from even larger (unlabeled) datasets than feasible today.

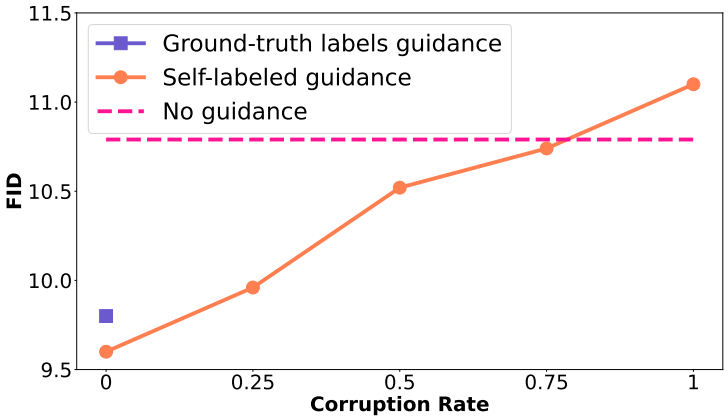


Figure 41: **Corruption of cluster assignments on CIFAR100.** Self-labeled guidance with 400 clusters outperforms the baseline trained with ground-truth labels. The FID performance deteriorates monotonically with the percentage of corrupted cluster assignments, underscoring the importance of assigning cluster ids via the self-supervised features.

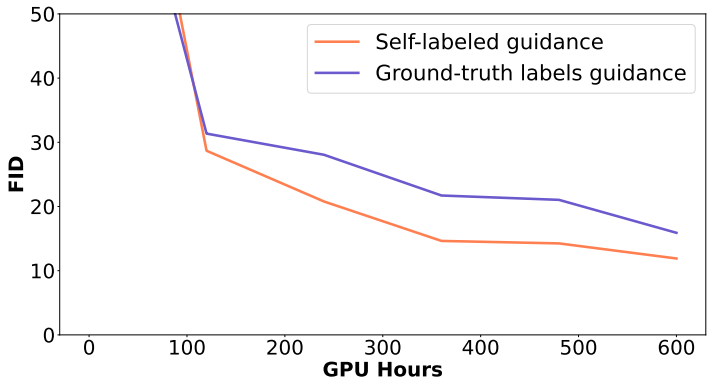


Figure 42: **Performance at different compute budgets.** After training for ~100 GPU hours self-labeled guidance achieves persistent FID reduction over training with ground-truth labels.

Diffusion Method	Annotation free?	ImageNet32		ImageNet64	
		FID↓	IS ↑	FID↓	IS↑
Ground-truth labels guidance	✗	9.2	19.0	16.8	18.6
No guidance	✓	14.3	10.8	36.1	10.4
Self-labeled guidance	✓	**7.3**	**20.3**	**12.1**	**23.1**

Table 36: **Self-labeled comparisons on ImageNet32/64.** Self-labeled guidance surpasses the no-guidance baseline by a large margin on both datasets and even outperforms the guided diffusion model trained using ground-truth class labels.

Diffusion Method	Annotation free?	ImageNet-100		LSUN-Churches	
		FID↓	IS ↑	FID↓	IS↑
Ground-truth labels guidance	✗	11.2	64.1	—	—
No guidance	✓	19.1	41.1	7.2	5.5
Self-labeled guidance	✓	**5.1**	**78.3**	**4.2**	**7.2**

Table 37: **Higher resolution image generation.** ImageNet-100 and LSUN-Churches results for images of size 256×256.

Figure 43: **Generated samples at** 256×256 **resolution using self-labeled guidance on LSUN-Churches.** Samples in each row are based on the same cluster. Self-labeled guidance enables semantically coherent samples, despite the absence of ground-truth annotations.

HIGHER RESOLUTION IMAGE GENERATION. Finally, we verify the effectiveness of self-labeled guidance on larger images. We report on the ImageNet-100 [64] (a subset of ImageNet-1k with 100 classes) and the LSUN-Churches dataset [273], both with images of size 256×256. Notably, the latter does not come with any annotations rendering a ground-truth guided baseline infeasible. Too limit computation, we use the Latent Diffusion Model [200] which is much more efficient at training with large image sizes than directly learning a diffusion model in the pixel space.

Table 37 shows that self-labeled guidance significantly outperforms the baselines, indicating the effectiveness of our method for high-resolution images. Note that the lack of ground-truth labels for LSUN-Churches reflects an advantage of our method since most real-world images are unlabeled. We show generated samples for two different clusters in Figure 43. The samples are diverse and reflect shared characteristics for samples guided by the same cluster. For more qualitative and quantitative results we refer to Section 7.6.1.

Diffusion Method	Annotation free?	Pascal VOC	COCO_20K
Ground-truth labels guidance	✗	23.5	19.3
No guidance	✓	58.6	42.5
Self-boxed guidance	✓	**18.4**	**16.0**

Table 38: **Self-boxed comparisons on Pascal VOC and COCO_20K.** Self-boxed guidance outperforms the no-guidance baseline FID considerably for multi-label datasets and is even better than a label-supervised alternative.

7.4.2 *Self-Boxed Guidance*

We run experiments on Pascal VOC and COCO_20K to validate the efficacy of self-boxed guidance. The self-boxed guidance model takes a bounding box in addition to the cluster-ID as guidance signal. To obtain the class-agnostic object bounding boxes, we use LOST [212] as our self-annotation function f in Equation (7.6). For the clustering, we empirically found k=100 to work well for both datasets, as both are relatively small in scale when compared to ImageNet. We train our diffusion model for 800 epochs with images of size 64×64. We report train FID for Pascal VOC and train/validation FID for COCO_20K. We evaluate the performance on the validation split by extracting the guidance signal from the training dataset to ensure that there is no information leakage. See Section 7.6.2 for more details.

SELF-BOXED COMPARISONS ON PASCAL VOC AND COCO_20K. For the ground-truth labels guidance baseline, we condition on a class embedding. Since there are now multiple objects per image, we represent the ground-truth class with a multi-hot embedding. Aside from the class embedding which is multi-hot in our method, all other settings remain the same for a fair comparison. The results in Table 38, confirm that the multi-hot class embedding is indeed effective for multi-label datasets, improving over the no-guidance model by a large margin. This improvement comes at the cost of manually annotating multiple classes per image. Self-boxed guidance further improves upon this result, by reducing the FID by an additional 5.1 and 3.3 points respectively without using any ground-truth annotation. In Figure 44, we show our method generates diverse and semantically well-aligned images.

7.4.3 *Self-Segmented Guidance*

Finally, we validate the efficacy of self-segmented guidance on Pascal VOC and COCO-Stuff. For COCO-Stuff we follow the split from [45, 88, 117, 278], with a train set of 49,629 images and a validation set of 2,175 images. Classes are merged into 27 (15 stuff and 12 things) categories. For self-segmented guidance, we apply STEGO [88] as our self-annotation function f in Equation (7.6). We set the cluster number to 27 for COCO-Stuff, and 21 for Pascal VOC, following STEGO. We train all models on images of size 64×64, for 800 epochs on Pascal VOC, and for 400 epochs on COCO-Stuff. We

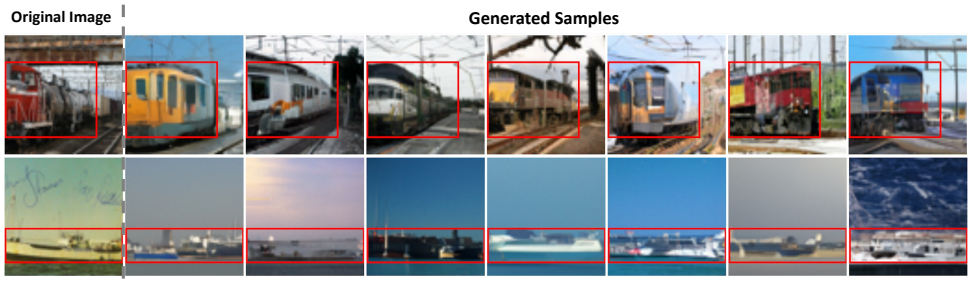


Figure 44: **Self-boxed guided diffusion results on Pascal VOC.** Each column is sampled using different random noise. Our method generates visually diverse and semantically consistent images. The image-level guidance signal successfully puts a limit on the model to create *train station, port* scenarios. The backgrounds are realistic and in harmony with the guidance boxes.

Diffusion Method	Annotation free?	Pascal VOC	COCO-Stuff	
			Train	Val
Ground-truth labels guidance	✗	23.5	16.3	20.5
No guidance	✓	58.6	29.1	34.1
Self-segmented guidance	✓	**17.1**	**12.5**	**17.7**
Ground-truth masks	✗	12.5	8.1	11.2

Table 39: **Self-segmented comparisons on Pascal VOC and COCO-Stuff.** Any form of guidance results in a considerable FID reduction over the no-guidance model. Self-segmented guidance improves over ground-truth multi-hot labels guidance and narrows the gap with guidance by annotation-intensive ground-truth masks.

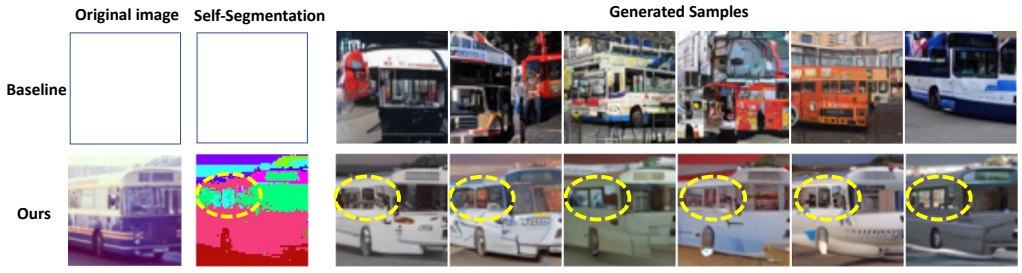


Figure 45: **Self-segmented guided diffusion results on Pascal VOC.** Each column is sampled using different random noise. The visualization indicates our self-segmented guidance provides more fine-grained guidance than ground-truth labels guidance for the generation of bus images. Note how the noisy window-bar in the self-segmented mask (marked by dotted ellipse) still results in plausible window separations in the generated image samples. We provide more examples in Section 7.6.3 .

report the train FID for Pascal VOC and both train and validation FID for COCO-Stuff. More details on the dataset and experimental setup are provided in Section 7.6.2.

SELF-SEGMENTED COMPARISONS ON VOC AND COCO-STUFF. We compare against both the ground-truth labels guidance baseline from the previous section and a model trained with ground-truth semantic masks guidance. The results in Table 39 demonstrate that our self-segmented guidance still outperforms the ground-truth labels guidance baseline on both datasets. The comparison between ground-truth labels and segmentation masks reveals an improvement in image quality when using the more fine-grained segmentation mask as the condition signal. But these segmentation masks are one of the most costly types of image annotations that require every pixel to be labeled. Our self-segmented approach avoids the necessity for annotations while narrowing the performance gap, and more importantly offering fine-grained control over the image layout. We demonstrate this controllability with examples in Figure 45 and explain how to assign semantic descriptions to the clusters in Section 7.6.3. These examples further highlight a robustness against noise in the segmentation masks, which our method acquires naturally due to training with noisy segmentations.

7.5 CONCLUSION

We have explored the potential of self-supervision signals for diffusion models and propose a framework for self-guided diffusion models. By leveraging a feature extraction function and a self-annotation function, our framework provides guidance signals at various image granularities: from the level of holistic images to object boxes and even segmentation masks. Our experiments indicate that self-supervision signals are an adequate replacement for existing guidance methods that generate images by relying on annotated image-label pairs during training. Furthermore, both self-boxed and self-segmented approaches demonstrate that we can acquire fine-grained control over the image content, without any ground-truth bounding boxes or segmentation masks. Though in certain cases, clusters can capture visual concepts that are challenging to articulate in everyday language, such as in the case of LSUN-Churches. For future research, it would be interesting to investigate the efficacy of our self-guidance approach on feature extractors trained on larger datasets or with image-text pairs [190]. Ultimately, our goal is to enable the benefits of self-guided diffusion for unlabeled and more diverse datasets at scale, wherein we believe this work is a promising first step.

7.6 APPENDIX

7.6.1 *More Results and Details*

Results on Churches-256

Implementation details. We directly use the official code of Latent Diffusion Model[1] [200], and reduce the base channel number from 192 to 128 and attention

1 https://github.com/CompVis/latent-diffusion

Figure 46: **Generated samples using self-labeled guidance on LSUN-Churches 256×256.** Each row corresponds to a different cluster. Clusters can capture concepts like nighttime, a far shot that includes the city, a close shot of the church, and the church's color.

Diffusion Method	Annotation-free?	ImageNet32				ImageNet64			
		FID↓	IS ↑	P ↑	R ↑	FID↓	IS↑	P↑	R ↑
Ground-truth labels guidance	✗	9.2	19.0	0.71	0.62	16.8	18.6	0.71	**0.62**
No guidance	✓	14.3	10.8	0.49	0.61	36.1	10.4	0.59	0.60
Self-labeled guidance	✓	**7.3**	**20.3**	**0.77**	**0.63**	**12.1**	**23.1**	**0.78**	**0.62**

Table 40: **Comparison with baseline on ImageNet32 and ImageNet64 dataset with FID, IS, Precision (P), Recall (R).**

resolution from [32, 16, 8, 4] to [8, 4] to accelerate training. Note that these changes significantly reduce the number of parameters from 294M to 108M.

Qualitative and Quantitative result. We present more qualitative results in Figure 46. We use the FID metric for quantitative comparison. For non-guidance and our self-labeled guidance, we get an FID of 23.1 and 16.2 respectively. Our self-labeled guidance improves by almost 7 points for free.

Precision and Recall in ImageNet32/64 dataset

We show the extra results of ImageNet on precision and recall in Table 40. We follow the evaluation code of precision and recall from ICGAN [36], our self-labeled guidance also outperforms ground-truth labels in precision and remains competitive in the recall.

Varying Guidance Strength w

We consider the influence of the guidance strength w on our sampling results. We mainly conduct this experiment in ImageNet32, as the validation set of ImageNet32 is strictly balanced, we also consider an unbalanced setting which is more similar to real-world deployment. Under both settings, we compare the FID between our self-labeled guidance and ground-truth guidance. We train both models for 100 epochs. For the standard ImageNet32 validation setting in Figure 47, our method achieves a 17.8% improvement for the respective optimal guidance strength of the two methods. Self-labeled guidance is especially effective for lower values of w. We observe similar trends for the unbalanced setting in Figure 48, be it that the overall FID results are slightly higher for both methods. The improvement increases to 18.7%. We conjecture this is due to the unbalanced nature

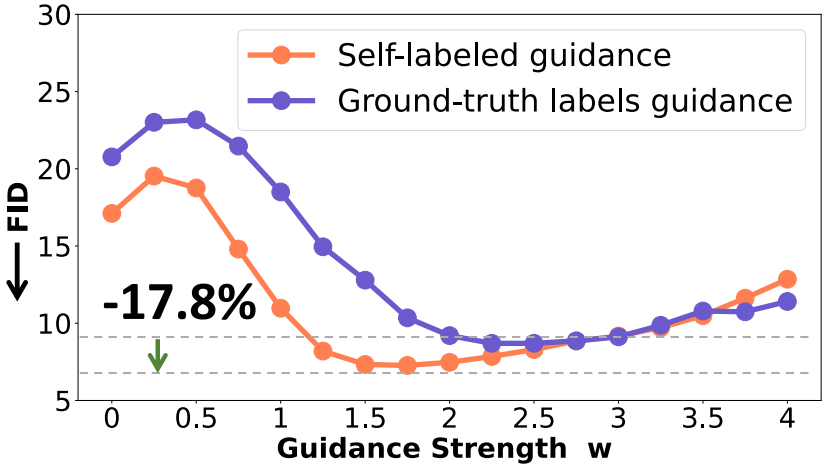


Figure 47: Setting I: ImageNet32 balanced

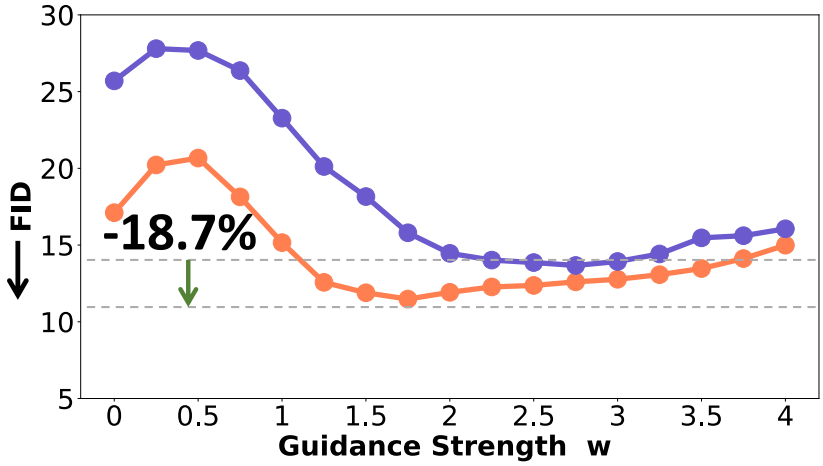


Figure 48: Setting II: ImageNet32 unbalanced

Figure 49: **Varying guidance strength** *w***.** Self-labeled guidance surpasses the guidance based on ground-truth labels for both (a) ImageNet32 balanced and (b) ImageNet32 unbalanced. The dotted gray line indicates the best-achieved performance of both methods under various guidance strengths. The difference between them is slightly more prominent for unbalanced data, we conjecture that this is because our self-labeled guidance is obtained by clustering based on the statistics of the overall dataset, which can potentially lead to more robust performance in unbalanced setting.

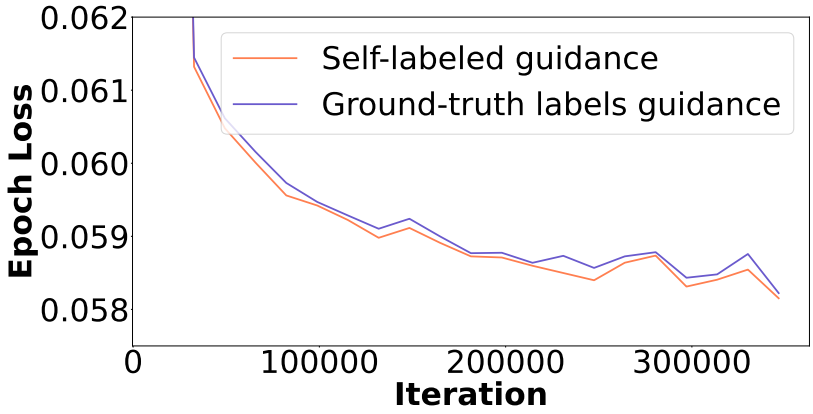


Figure 50: **Epoch loss trend.**

of the k-means algorithm [142], and clustering based on the statistics of the overall dataset can potentially lead to more robust performance in an unbalanced setting.

Cluster Number Ablation in Self-boxed Guidance

In Table 41, we empirically evaluate the performance when we alter the cluster number in our self-boxed guidance. We find the performance will increase from $k = 21$ to $k = 100$, and saturated at $k = 100$.

Cluster number k	**FID** $\downarrow$
21	22.5
50	18.6
100	18.5

Table 41: **Cluster number ablation on Pascal VOC dataset for self-boxed guidance.**

Trend Visualization of Training Loss and Validation FID

We visualize the trend of training loss and validation FID in Figure 50.

7.6.2 *More Experimental Details*

Training details. For our best results, we train 100 epochs on 4 GPUs of A5000 (24G) in ImageNet. We train 800/800/400 epochs on 1GPU of A6000 (48G) in Pascal VOC, COCO_20K, and COCO-Stuff, respectively. All qualitative results in this chapter are trained in the same setting as mentioned above. We train and evaluate the Pascal VOC, COCO_20K, and COCO-Stuff in image size 64, and visualize them by bilinear upsampling to 256, following [152].

Sampling details. We sample the guidance signal from the distribution of training set in our all experiments. For each timestep, we need twice of Number of Forward

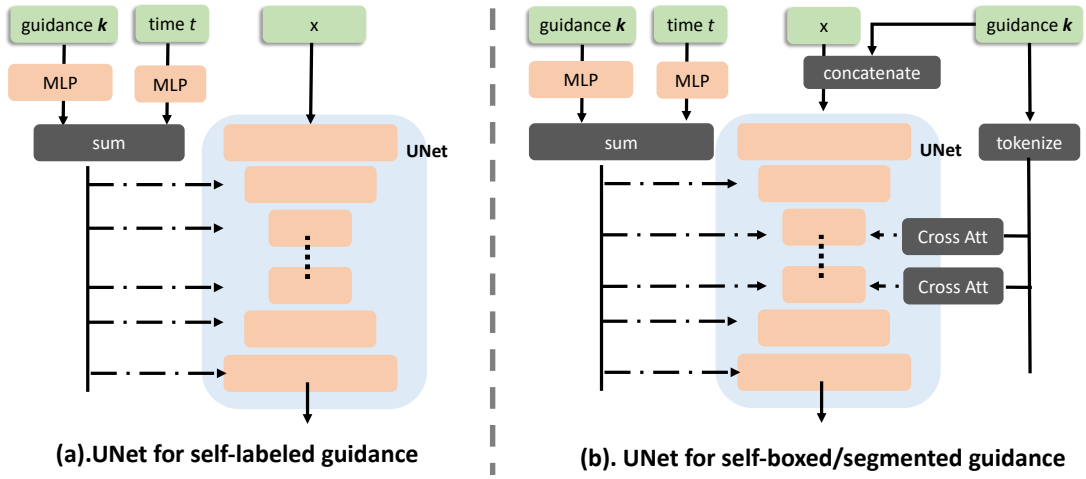


Figure 51: **The structure of UNet module.**

Evaluation (NFE), we optimize them by concatenating the conditional and unconditional signal along the batch dimension so that we only need one time of NFE in every timestep.

Evaluation details. We use the common package Clean-FID [181], torch-fidelity [177] for FID, IS calculation, respectively. For IS, we use the standard 10-split setting, we only report IS on ImageNet, as it might be not an appropriate metric for non object-centric datasets [13]. For the checkpoint, we pick the checking point every 10 epochs by minimal FID between generated sample set and the train set.

UNet structure

Guidance signal injection. We describe the detail of guidance signal injection in Figure 51. The injection of self-labeled guidance and self-boxed/segmented guidance is slightly different. The common part is by concatenation between timestep embedding and noisy input, the concatenated feature will be sent to every block of the UNet. For the self-boxed/segmented guidance, we not only conduct the information fusion as above but also incorporate the spatial inductive-bias by concatenating it with input, the concatenated result will be fed into the UNet.

Timestep embedding. We embed the raw timestep information by two-layer MLP: FC(512, 128)→SiLU→FC(128, 128).

Guidance embedding. The guidance is in the form of one/multi-hot embedding $\mathbb{R}^K$, we feed it into two-layer MLP: FC(K, 256)→SiLU→FC(256, 256), then feed those guidance signal into the UNet following in Figure 51.

Cross-attention. In training for non object-centric dataset, we also tokenize the guidance signal to several tokens following Imagen [203], we concatenate those tokens with image tokens (can be transposed to a token from typical feature map by $\mathbb{R}^{W\times H\times C} \rightarrow \mathbb{R}^{C\times WH}$), the cross-attention [20, 200] is conducted by $\texttt{CA}(\texttt{m}, \texttt{concat}[\mathbf{k}, \texttt{m}])$. Due to the quadratic complexity of transformer [127, 156], we only apply the cross-attention in lower-resolution feature maps.

Base channels: 128	Optimizer: AdamW
Channel multipliers: 1, 2, 4	Learning rate: $3e-4$
Blocks per resolution: 2	Batch size: 128
Attention resolutions: 4	EMA: 0.9999
number of head: 8	Dropout: 0.0
Conditioning embedding dimension: 256	Training hardware: 4 × A5000(24G)
Conditioning embedding MLP layers: 2	Training Epochs: 100
Diffusion noise schedule: linear	Weight decay: 0.01
Sampling timesteps: 256	

Table 42: **3×32×32 model, 4GPU, ImageNet32.**

Base channels: 128	Optimizer: AdamW
Channel multipliers: 1, 2, 4	Learning rate: $1e-4$
Blocks per resolution: 2	Batch size: 48
Attention resolutions: 4	EMA: 0.9999
number of head: 8	Dropout: 0.0
Conditioning embedding dimension: 256	Training hardware: 4 × A5000(24G)
Conditioning embedding MLP layers: 2	Training Epochs: 100
Diffusion noise schedule: linear	Weight decay: 0.01
Sampling timesteps: 256	

Table 43: **3×64×64 model, 4GPU, ImageNet64.**

Base channels: 128	Optimizer: AdamW
Channel multipliers: 1, 2, 4	Learning rate: $1e-4$
Blocks per resolution: 2	Batch size: 80
Attention resolutions: 4	EMA: 0.9999
Number of head: 8	Dropout: 0.0
Conditioning embedding dimension: 256	Training hardware: 1 × A6000(45G)
Conditioning embedding MLP layers: 2	Training Epochs: 800/800/400
Diffusion noise schedule: linear	Weight decay: 0.01
Sampling timesteps: 256	Context token number: 8
Context dim: 32	

Table 44: **3×64×64 model, 1GPU, Pascal VOC, COCO_20K, COCO-Stuff.**

Base channels: 128	Optimizer: AdamW
Channel multipliers: 1, 2,2,3, 4	Learning rate: $5e-5$
Blocks per resolution: 2	Batch size: 48
Attention resolutions: 4,8	EMA: 0.9999
Number of head: 8	Dropout: 0.0
Conditioning embedding dimension: 256	Training hardware: 4 × A5000(24G)
Conditioning embedding MLP layers: 2	Training Steps: 600k
Diffusion noise schedule: linear	Weight decay: 0.01
Sampling timesteps: 200	

Table 45: **3×256×256 model, 4GPU, Churches-256.**

Training Parameter

Dataset Preparation

The preparation of unbalanced dataset. There are 50,000 images in the validation set of ImageNet with 1,000 classes (50 instances for each). We index the class from 0 to 999, for each class c_i, the instance of the class c_i is $\lfloor i \times 50/1000 \rfloor = \lfloor i/200 \rfloor$.

Pascal VOC. We use the standard split from [212]. It has 12,031 training images. As there is no validation set for Pascal VOC dataset, therefore, we only evaluate FID on the train set. We sample 10,000 images and use 10,000 random-cropped 64-sized train images as reference set for FID evaluation.

COCO_20K. We follow the split from [150, 212, 240]. COCO_20k is a subset of the COCO2014 trainval dataset, consisting of 19,817 randomly chosen images, used in unsupervised object discovery [212, 240]. We sample 10,000 images and use 10,000 random-cropped 64-sized train images as reference set for FID evaluation.

COCO-Stuff. It has a train set of 49,629 images, validation set of 2,175 images, where the original classes are merged into 27 (15 stuff and 12 things) high-level categories. We use the dataset split following [45, 88, 117, 278], We sample 10,000 images and use 10,000 train/validation images as reference set for FID evaluation.

LOST, STEGO Algorithms

LOST algorithm details. We conduct padding to make the original image can be patchified to be fed into the `ViT` architecture [57], and feed the original padded image into the `LOST` architecture using official source code [2]. `LOST` can also be utilized in a two-stage approach to provide multi-object, due to its complexity, we opt for only single-object discovery in this chapter.

STEGO algorithm details. We follow the official source code [3], and apply padding to make the original image can be fed into the `ViT` architecture to extract the self-segmented guidance signal.

For COCO-Stuff dataset, we directly use the official pretrained weight. For Pascal VOC, we train `STEGO` ourselves using the official hyperparameters.

In `STEGO`'s pre-processing for the k-NN, the number of neighbors for k-NN is 7. The segmentation head of `STEGO` is composed of a two-layer `MLP` (with `ReLU` activation) and outputs a 70-dimension feature. The learning rate is $5e-4$, the batch size is 64.

7.6.3 *Qualitative Results*

Assigning Semantic Descriptions in Self-labeled/segmented Guidance

In order to control the semantic content of a sample using self-guidance we can assign descriptions to each self-supervised cluster by manually checking a few example images per cluster. This is much more scalable since the total number of training images available are multiple orders of magnitude greater than the number of clusters. Furthermore, images

2 https://github.com/valeoai/LOST
3 https://github.com/mhamilton723/STEGO

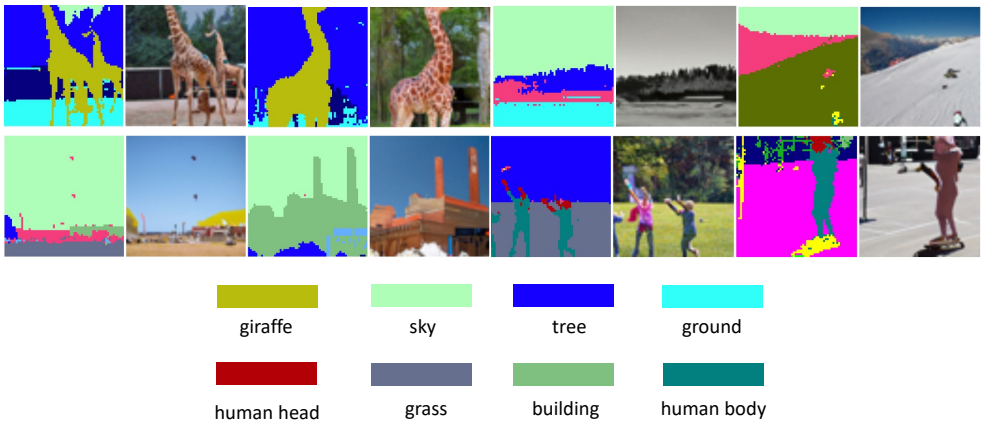


Figure 52: **Self-segmented guidance samples from COCO-Stuff companies with segmentation mask from STEGO [88].** The color map is shared among the overall dataset. The semantic description is deduced based on a few images. Best viewed in color.

in the same self-supervised cluster are highly semantically coherent and humans can easily describe their shared abstract concept [140].

In Figure 53 we show examples of self-labeled guidance that highlight the semantic coherence of samples guided by the same cluster id. In Figure 52 we show how this approach is also extendable to self-segmented guidance.

More Qualitative Results

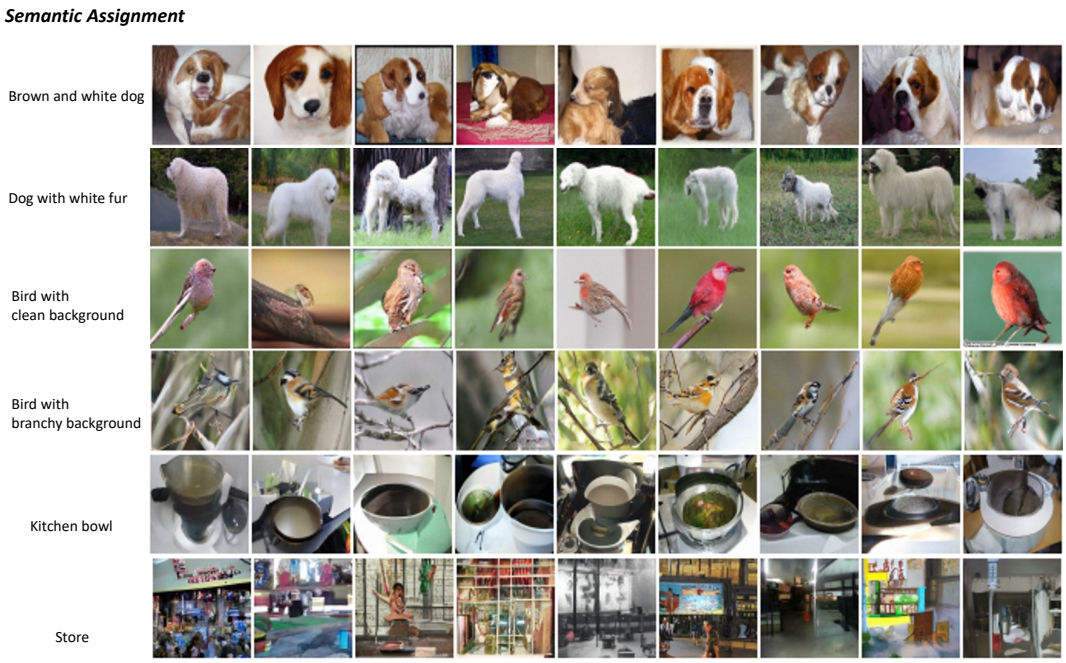


Figure 53: **Self-labeled guidance samples conditioning on the same guidance from ImageNet64.** We assign a cluster description based on a few sample images. Best viewed in color.

Figure 54: **Self-segmented guidance samples from COCO-Stuff.** Best viewed in color.

Figure 55: **Denoising process of self-segmented guidance samples (uncurated) from COCO-Stuff.** The first column is the self-segmented guidance mask from STEGO [88], The remaining columns are from the noisiest period to the less noisy period. Best viewed in color.

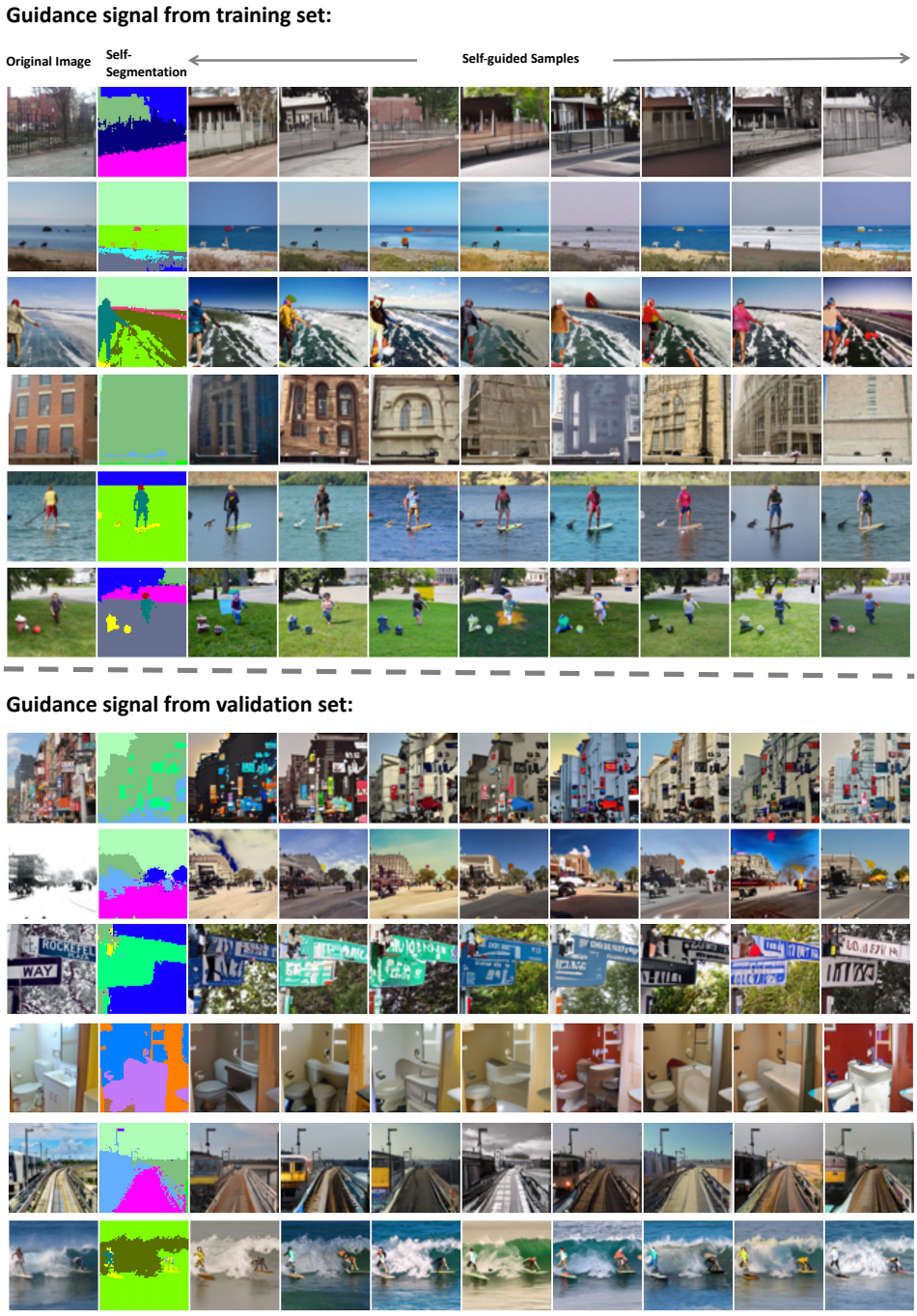


Figure 56: **Self-segmented guidance samples (uncurated) from COCO-Stuff.** The first column is the real image where we attain the conditional mask. The second column is the self-segmented mask we obtain from STEGO [88], The remaining columns are the random samples conditioning on the same self-segmented mask. Best viewed in color.

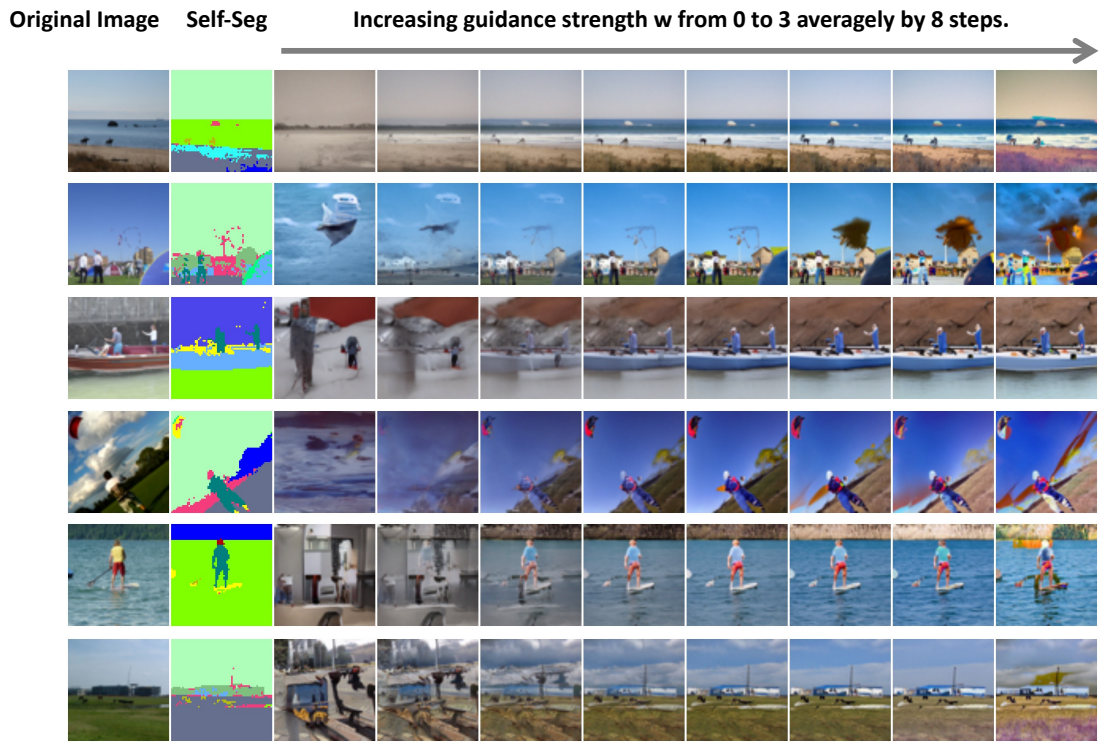


Figure 57: **Self-segmented guidance samples from Pascal VOC.** The first column is the real image where we attain the conditional mask. The second column is the self-segmented mask we obtain from STEGO [88]. The remaining columns are the visualization when we averagely increase guidance strength w from 0 to 3 by 8 steps. Best viewed in color.

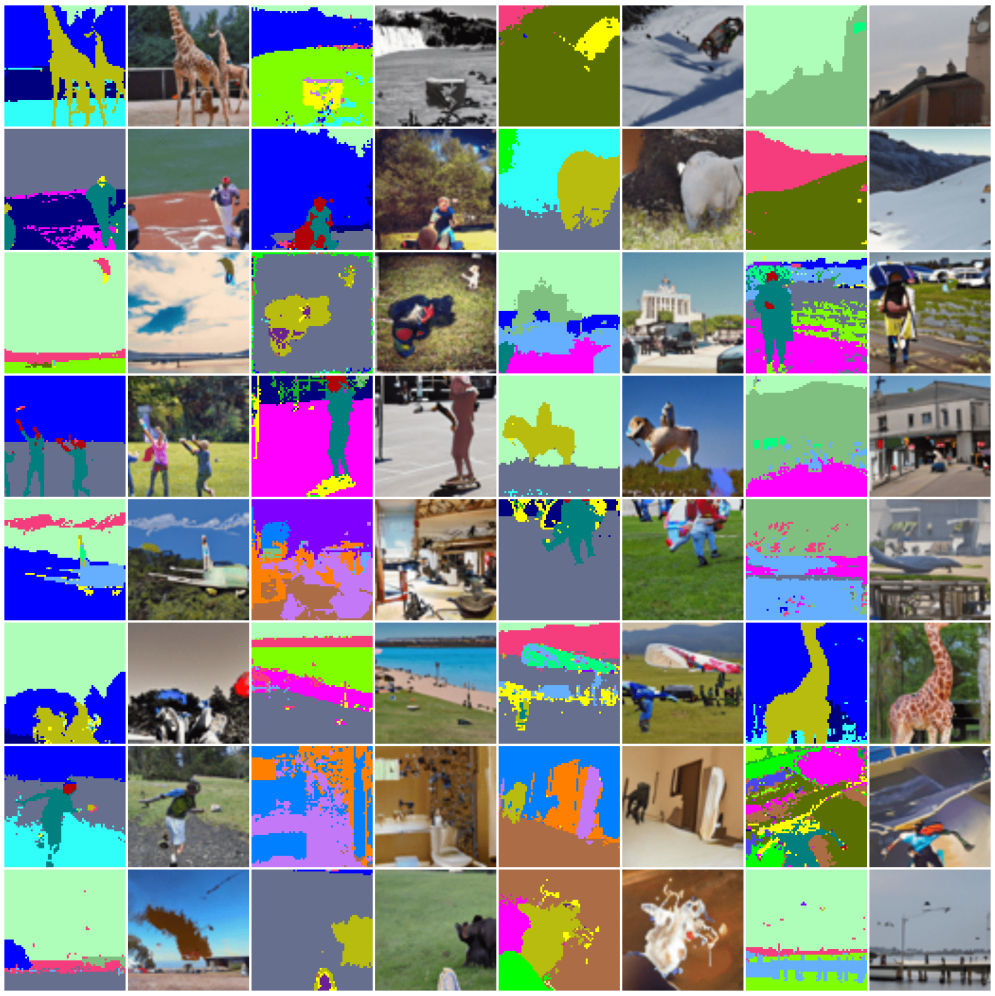


Figure 58: **Self-segmented guidance samples (uncurated)** from COCO-Stuff companies with segmentation mask from STEGO [88]. The color map is shared among the overall dataset. Best viewed in color.

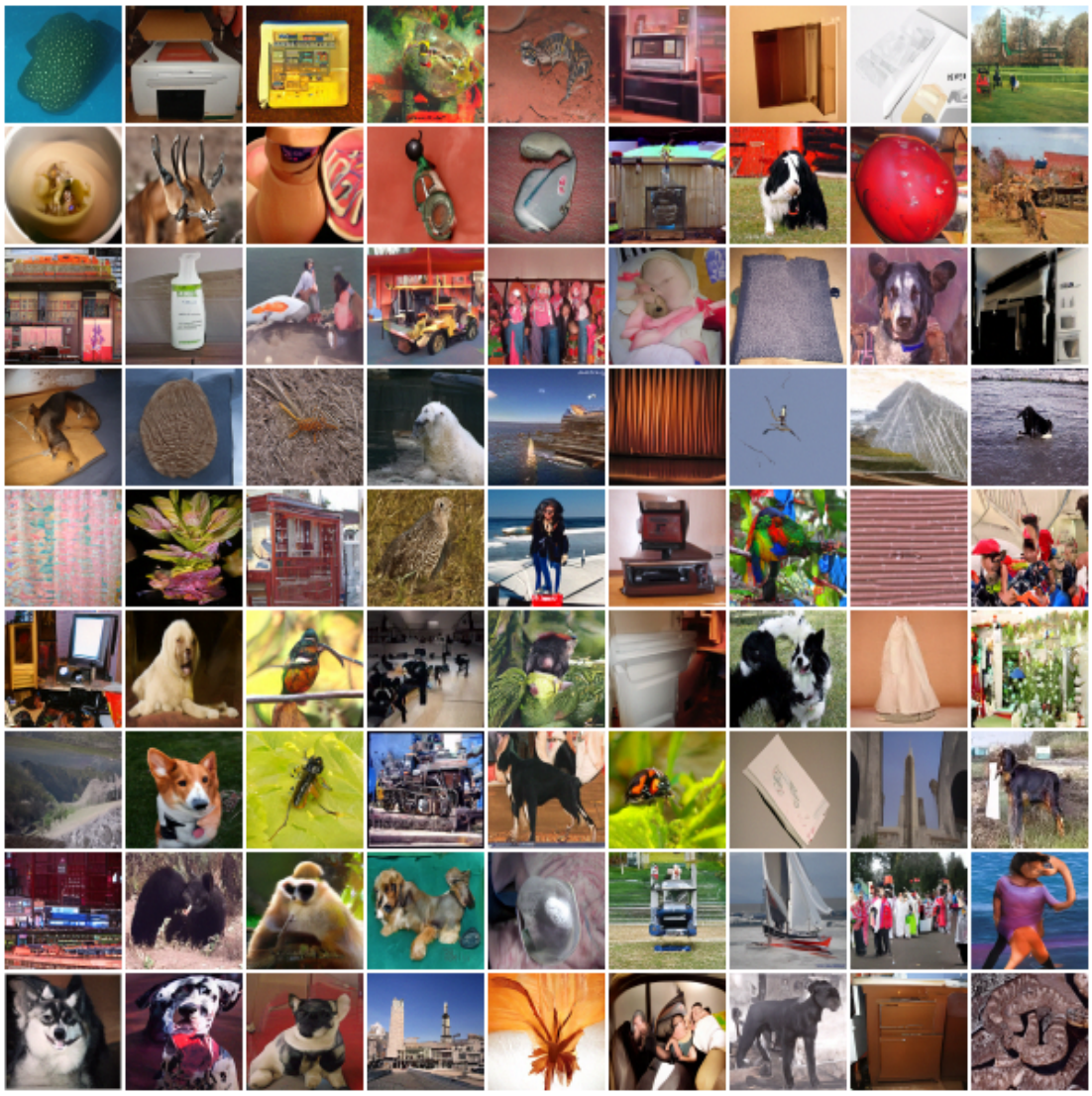


Figure 59: **Self-labeled guidance samples (uncurated) from ImageNet64.** Best viewed in color.

Figure 60: **Self-labeled guidance samples (uncurated) from ImageNet32.** Best viewed in color.

Figure 61: **Denoising process for ImageNet64.**

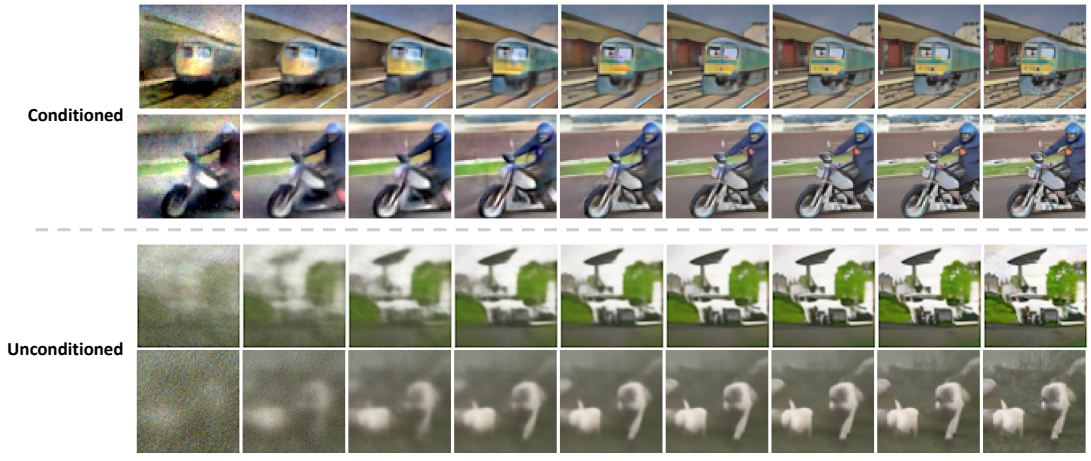


Figure 62: **Denoising process for Pascal VOC.** The first two rows are sampled from guidance strength $w = 2$ using our self-segmented guidance, the last two rows are sampled from guidance strength $w = 0$. By conditioning on our self-segmented guidance, the denoising process becomes easier and faster, this efficient denoising aligns with the observation from [186].

Figure 63: **Sphere interpolation between two random self-labeled guidance signals on ImageNet64.** The sphere interpolation follows the DDIM [217]. Best viewed in color.

Figure 64: cluster625

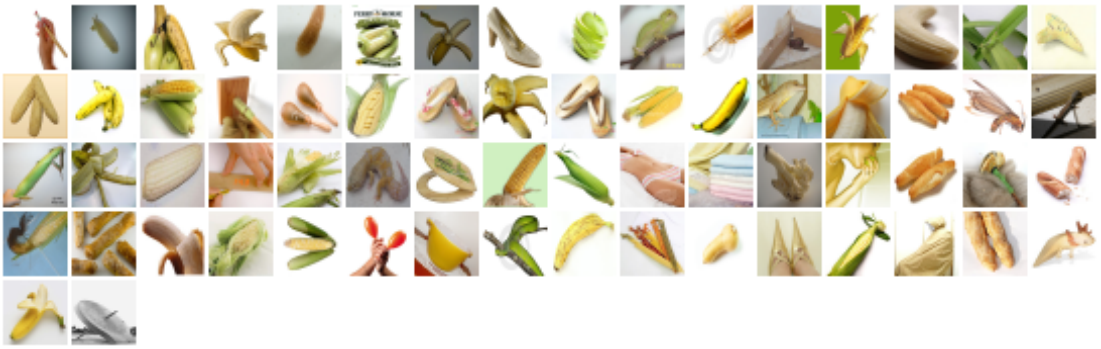


Figure 65: cluster807

Figure 66: cluster890

Figure 67: **Cluster visualization of real images in ImageNet32 after k-means.**

Figure 68: **Bounding box result from LOST on Pascal VOC**. As LOST [212] is an unsupervised-learning method, some flaws in the generated box are expected. Images are resized squarely for better visualization.

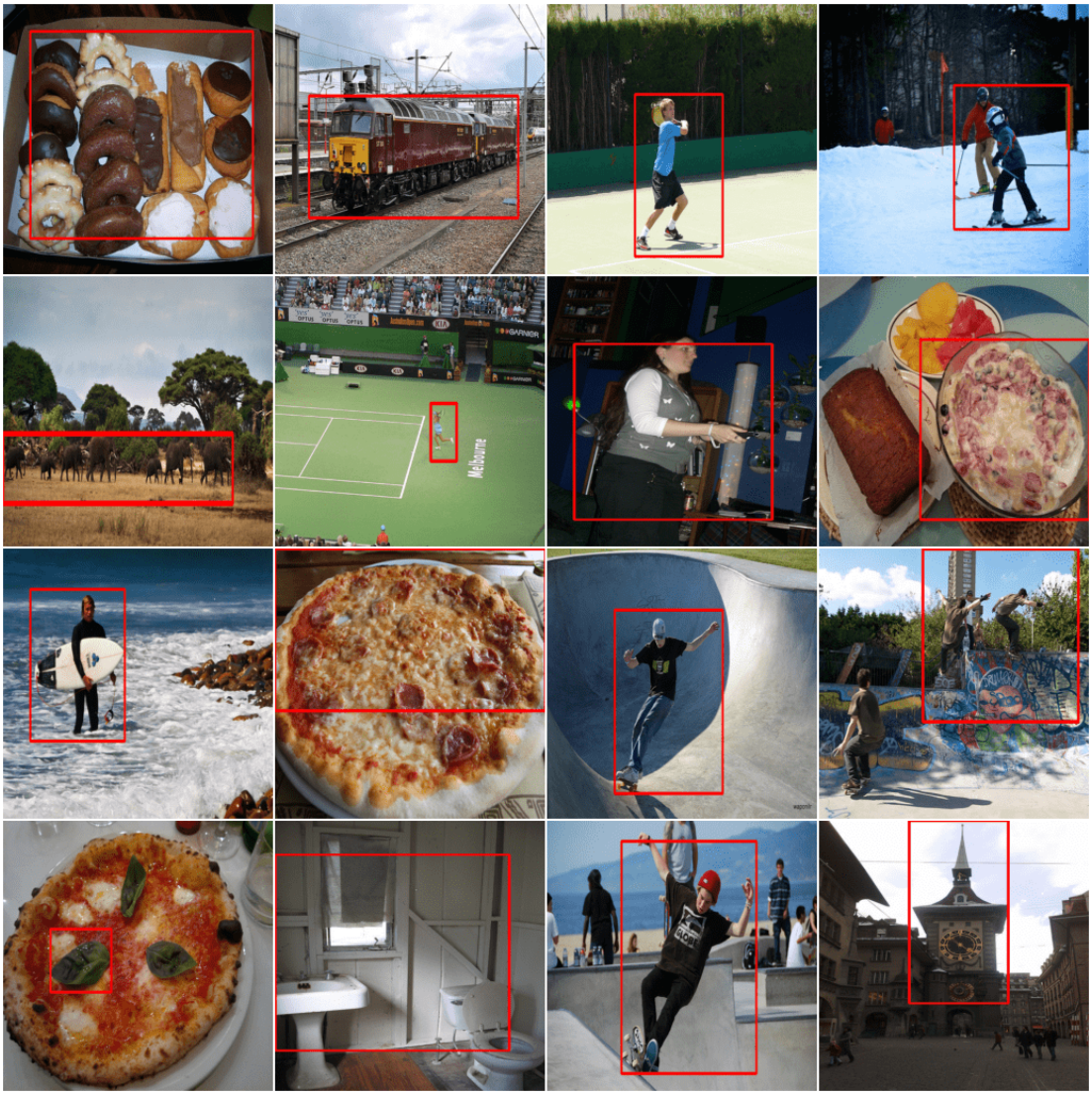


Figure 69: **Bounding box result from LOST [212] on COCO_20K.** Images are resized squarely for better visualization.

Figure 70: **Segmentation mask result from STEGO on Pascal VOC dataset**. Cluster number k is 21. Images are resized squarely for better visualization. The color map is shared among the overall dataset. Best viewed in color.

8

SUMMARY AND CONCLUSIONS

8.1 SUMMARY

This thesis aims to achieve label-efficient learning in computer vision. We demonstrate various algorithms that use a considerably reduced set of labels, or even no labels at all, to achieve label-efficiency. The main research question is addressed across six typical computer vision challenges: image segmentation, object detection, video localization, video retrieval, point cloud classification, and generative image diffusion. A brief summary of each label-efficient computer vision solution per chapter is provided as follows:

CHAPTER 2 IMAGE SEGMENTATION. To tackle label-efficient learning in image segmentation, we propose the Attention-based Multi-Context Guiding (A-MCG) network, which consists of three branches: support branch, query branch, feature fusion branch. A key differentiator of A-MCG is the integration of multi-scale context features between support and query branches, enforcing a better guidance from the support set. In addition, we also adopt a spatial attention mechanism along the fusion branch to highlight context information from several scales, enhancing self-supervision in one-shot learning. To address the fusion problem in multi-shot learning, ConvLSTM is adopted to collaboratively integrate the sequential support features to elevate the final accuracy. Our architecture strikes a state-of-the-art result on unseen classes in a variant of PASCAL VOC12 dataset and performs favorably against the previous work with large gains of 1.1%, 1.4% measured in mIoU in the 1-shot or 5-shot setting respectively.

CHAPTER 3 OBJECT DETECTION. In this chapter, we propose a new task to explore the label-efficient learning in object detection, we call few-shot common-localization. Given a few weakly-supervised support images, we aim to localize the common object in the query image without any box annotation. This task differs from standard few-shot settings, since we aim to address the localization problem, rather than the global classification problem. To tackle this new problem, we propose a network that aims to get the most out of the support and query images. To that end, we introduce a spatial similarity module that searches the spatial commonality among the given images. We furthermore introduce a feature reweighting module, which balances the influence of different support images through graph convolutional networks. To evaluate few-shot common-localization, we repurpose and reorganize the well-known Pascal VOC and MS-COCO datasets, as well as a video dataset from Imagenet VID. Experimental evaluation

on the new settings for few-shot common-localization shows the importance of searching for spatial similarity and feature reweighting, outperforming baselines from related tasks.

CHAPTER 4 VIDEO ACTION LOCALIZATION. To explore label-efficient learning in video action localization, this chapter strives to localize the temporal extent of action in a long untrimmed video. Where existing work leverages many examples with their start, their ending, and/or the class of the action during training time, we propose few-shot common action localization. The start and end of an action in a long untrimmed video are determined based on just a handful of trimmed video examples containing the same action, without knowing their common class label. To address this task, we introduce a new 3D convolutional network architecture able to align representations from the support videos with the relevant query video segments. The network contains: (*i*) a mutual enhancement module to simultaneously complement the representation of the few trimmed support videos and the untrimmed query video; (*ii*) a progressive alignment module that iteratively fuses the support videos into the query branch; and (*iii*) a pairwise matching module to weigh the importance of different support videos. Evaluation of few-shot common action localization in untrimmed videos containing single or multiple action instances demonstrates the effectiveness and general applicability of our proposal.

CHAPTER 5 VIDEO ACTION RETRIEVAL. This chapter considers the retrieval of videos containing human activity from just a video query to explore label-efficiency in imbalanced scenarios. In the literature, a common assumption is that all activities have sufficient labelled examples when learning an embedding for retrieval. However, this assumption does not hold in practice, as only a portion of activities have many examples, while other activities are only described by few examples. In this chapter, we propose a visual-semantic embedding network that explicitly deals with the imbalanced scenario for activity retrieval. Our network contains two novel modules. The visual alignment module performs a global alignment between the input video and visual feature bank representations for all activities. The semantic module aligns between the input video and semantic activity representations. By matching videos with both visual and semantic activity representations of overall activities, we no longer ignore infrequent activities during retrieval. Experiments on a new imbalanced activity retrieval benchmark show the effectiveness of our proposal.

CHAPTER 6 POINT CLOUD CLASSIFICATION. This chapter introduces data augmentation for point clouds by interpolation between examples to arrive label-efficient learning in point cloud. Data augmentation by interpolation has shown to be a simple and effective approach in the image domain. Such a mixup is however not directly transferable to point clouds, as we do not have a one-to-one correspondence between the points of two different objects. In this chapter, we define data augmentation between point clouds as a shortest path linear interpolation. To that end, we introduce PointMixup, an interpolation method that generates new examples through an optimal assignment of the path function between two point clouds. We prove that our PointMixup finds the shortest path between two point clouds and that the interpolation is assignment invariant and linear. With the definition of interpolation, PointMixup allows introducing strong interpolation-based regularizers such as mixup and manifold mixup to the point

cloud domain. Experimentally, we show the potential of PointMixup for point cloud classification, especially when examples are scarce, as well as increased robustness to noise and geometric transformations to points.

CHAPTER 7 GENERATIVE IMAGE DIFFUSION Diffusion models have demonstrated remarkable progress in image generation quality, especially when guidance is used to control the generative process. However, guidance requires a large amount of image-annotation pairs for training and is thus dependent on their availability, correctness and unbiasedness. To achieve better label-efficiency in terms of guidance in diffusion models, in this chapter, we eliminate the need for such annotation by instead leveraging the flexibility of self-supervision signals to design a framework for *self-guided* diffusion models. By leveraging a feature extraction function and a self-annotation function, our method provides guidance signals at various image granularities: from the level of holistic images to object boxes and even segmentation masks. Our experiments on single-label and multi-label image datasets demonstrate that self-labeled guidance always outperforms diffusion models without guidance and may even surpass guidance based on ground-truth labels, especially on unbalanced data. When equipped with self-supervised box or mask proposals, our method further generates visually diverse yet semantically consistent images, without the need for any class, box, or segment label annotation. Self-guided diffusion is simple, flexible and expected to profit from deployment at scale.

8.2 CONCLUSIONS

This thesis presents several novel approaches to achieving label-efficient learning in computer vision. By demonstrating the effectiveness of various algorithms that use a reduced set of labels or even zero labels, we have addressed the main research question across six typical computer vision challenges. Our research highlights the importance of leveraging unlabeled data [110], exploring alternative learning paradigms [107, 109, 270], boosting the label-efficiency by augmentation [44] and carefully designing network structure [107, 109, 270] to achieve label-efficiency. Our results suggest that label-efficient learning has the potential to significantly reduce the amount of labeled data required for training computer vision models, opening up new possibilities for real-world applications. This thesis also presents practical implications for the development of such applications. By reducing the reliance on labeled data, our approach can significantly lower the cost and time required for data annotation. As a result, it becomes easier to develop computer vision solutions for a wide range of industries.

However, further research is needed to explore the full potential of label-efficient learning and to develop new algorithms that can achieve even greater label-efficiency. One interesting direction is about exploring how to utilize the synergy functionality from multi-modalities e.g., image, video, audio, caption, flow, etc. Those explorations can be conducted either in discriminative learning by contrasting views [42] or in generative modeling by learning a joint distribution [226]. What is more, exploration of the potential of Large Language Models [209] and Foundation Models [23] needs to be carried out to check how to achieve an intelligence system that can arrive at self-evolution by constantly consuming data without reliance on labels.

In conclusion, this thesis provides a contribution to the field of computer vision, demonstrating the potential of label-efficient learning and providing insights into how it can be achieved. We hope that our work will inspire further research and development in this area, leading to even more efficient and effective computer vision solutions in the future.

BIBLIOGRAPHY

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, 2016.

[2] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Learning representations and generative models for 3d point clouds. In *ICML*, 2018.

[3] K. P. Alexandridis, J. Deng, A. Nguyen, and S. Luo. Long-tailed instance segmentation using gumbel optimized loss. In *ECCV*, 2022.

[4] M. Armandpour, A. Sadeghian, C. Li, and M. Zhou. Partition-guided gans. In *CVPR*, 2021.

[5] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid. Vivit: A video vision transformer. In *ICCV*, 2021.

[6] Y. M. Asano, C. Rupprecht, and A. Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *ICLR*, 2020.

[7] Y. M. Asano, C. Rupprecht, A. Zisserman, and A. Vedaldi. Pass: An imagenet replacement for self-supervised pretraining without humans. In *NIPS*, 2021.

[8] O. Ashual, S. Sheynin, A. Polyak, U. Singer, O. Gafni, E. Nachmani, and Y. Taigman. Knn-diffusion: Image generation via large-scale retrieval. In *ARXIV*, 2022.

[9] M. Assran, M. Caron, I. Misra, P. Bojanowski, F. Bordes, P. Vincent, A. Joulin, M. Rabbat, and N. Ballas. Masked siamese networks for label-efficient learning. In *ARXIV*, 2022.

[10] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. In *ARXIV*, 2016.

[11] A. Bansal, K. Sikka, G. Sharma, R. Chellappa, and A. Divakaran. Zero-shot object detection. In *ECCV*, 2018.

[12] S. Y. Bao, Y. Xiang, and S. Savarese. Object co-detection. In *ECCV*, 2012.

[13] S. Barratt and R. Sharma. A note on the inception score. In *ARXIV*, 2018.

[14] E. Bart and S. Ullman. Cross-generalization: Learning novel classes from a single example by feature replacement. In *CVPR*, 2005.

[15] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba. Gan dissection: Visualizing and understanding generative adversarial networks. In *ARXIV*, 2018.

[16] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, 2019.

[17] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *CoRR*, 2019.

[18] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019.

[19] H. Bilen and A. Vedaldi. Weakly supervised deep detection networks. In *CVPR*, 2016.

[20] A. Blattmann, R. Rombach, K. Oktay, and B. Ommer. Retrieval-augmented diffusion models. In *ARXIV*, 2022.

[21] D. Bogdoll, M. Nitsche, and J. M. Zöllner. Anomaly detection in autonomous driving: A survey. In *CVPR*, 2022.

[22] P. Bojanowski, R. Lajugie, F. Bach, I. Laptev, J. Ponce, C. Schmid, and J. Sivic. Weakly supervised action labeling in videos under ordering constraints. In *ECCV*, 2014.

[23] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. On the opportunities and risks of foundation models. In *ARXIV*, 2021.

[24] F. Bordes, R. Balestriero, and P. Vincent. High fidelity visualization of what your self-supervised representation knows about. In *TMLR*, 2022.

[25] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*, 2010.

[26] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019.

[27] S. Buch, V. Escorcia, C. Shen, B. Ghanem, and J. Carlos Niebles. Sst: Single-stream temporal action proposals. In *CVPR*, 2017.

[28] A. Bulat, E. Sánchez-Lozano, and G. Tzimiropoulos. Improving memory banks for unsupervised learning with large mini-batch, consistency and hard negative mining. In *ICASSP*, 2021.

[29] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015.

[30] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *CVPR*, 2017.

[31] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.

[32] K. Cao, J. Ji, Z. Cao, C.-Y. Chang, and J. C. Niebles. Few-shot video classification via temporal alignment. In *CVPR*, 2020.

[33] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[34] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

[35] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[36] A. Casanova, M. Careil, J. Verbeek, M. Drozdzal, and A. Romero Soriano. Instance-conditioned gan. In *NeurIPS*, 2021.

[37] D. L. Chen and W. B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *ACL*, 2011.

[38] H. Chen, Y. Wang, G. Wang, and Y. Qiao. LSTD: A low-shot transfer detector for object detection. In *AAAI*, 2018.

[39] J. Chen, W. Su, and Z. Wang. Crowd counting with crowd attention convolutional neural network. In *NeuroComputing*, 2020.

[40] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. In *PAMI*, 2018.

[41] S. Chen, P. Mettes, T. Hu, and C. G. M. Snoek. Interactivity proposals for surveillance videos. In *ICMR*, 2020.

[42] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020.

[43] T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby. Self-supervised gans via auxiliary rotation loss. In *CVPR*, 2019.

[44] Y. Chen, V. T. Hu, E. Gavves, T. Mensink, P. Mettes, P. Yang, and C. G. M. Snoek. Pointmixup: Augmentation for point clouds. In *ECCV*, 2020.

[45] J. H. Cho, U. Mall, K. Bala, and B. Hariharan. Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering. In *CVPR*, 2021.

[46] K. Chua, Q. Lei, and J. D. Lee. How fine-tuning allows for effective meta-learning. In *NeurIPS*, 2021.

[47] R. G. Cinbis, J. Verbeek, and C. Schmid. Weakly supervised object localization with multi-fold multiple instance learning. In *PAMI*, 2017.

[48] A. Ciptadi, M. S. Goodwin, and J. M. Rehg. Movement pattern histogram for action recognition and retrieval. In *ECCV*, 2014.

[49] G. Couairon, J. Verbeek, H. Schwenk, and M. Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. In *ARXIV*, 2022.

[50] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017.

[51] X. Dai, B. Singh, G. Zhang, L. S. Davis, and Y. Qiu Chen. Temporal context network for activity localization in videos. In *ICCV*, 2017.

[52] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling egocentric vision: The epic-kitchens dataset. In *ECCV*, 2018.

[53] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[54] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021.

[55] A. Diba, V. Sharma, A. M. Pazandeh, H. Pirsiavash, and L. Van Gool. Weakly supervised cascaded convolutional networks. In *CVPR*, 2017.

[56] X. Dong, L. Zheng, F. Ma, Y. Yang, and D. Meng. Few-example object detection with model communication. In *PAMI*, 2018.

[57] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[58] X. Du, C. Jiang, H. Xu, G. Zhang, and Z. Li. How to save your annotation cost for panoptic segmentation? In *AAAI*, 2021.

[59] A. Duarte, S. Albanie, X. Giró-i Nieto, and G. Varol. Sign language video retrieval with free-form textual queries. In *CVPR*, 2022.

[60] S. R. Dubey, S. K. Singh, and W.-T. Chu. Vision transformer hashing for image retrieval. In *ICME*, 2022.

[61] O. Duchenne, I. Laptev, J. Sivic, F. Bach, and J. Ponce. Automatic annotation of human actions in video. In *ICCV*, 2009.

[62] V. Escorcia, F. C. Heilbron, J. C. Niebles, and B. Ghanem. Daps: Deep action proposals for action understanding. In *ECCV*, 2016.

[63] V. Escorcia, M. Soldan, J. Sivic, B. Ghanem, and B. Russell. Temporal localization of moments in video collections with natural language. In *ARXIV*, 2019.

[64] T. et al. Contrastive multiview coding. In *ECCV*, 2020.

[65] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. In *IJCV*, 2010.

[66] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017.

[67] Y. Feng, L. Ma, W. Liu, T. Zhang, and J. Luo. Video re-localization. In *ECCV*, 2018.

[68] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

[69] G. French, T. Aila, S. Laine, M. Mackiewicz, and G. Finlayson. Consistency regularization and cutmix for semi-supervised semantic segmentation. In *CoRR*, 2019.

[70] J. Gao, K. Chen, and R. Nevatia. Ctap: Complementary temporal action proposal generation. In *ECCV*, 2018.

[71] J. Gao, Z. Yang, C. Sun, K. Chen, and R. Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *ICCV*, 2017.

[72] S.-H. Gao, Z.-Y. Li, M.-H. Yang, M.-M. Cheng, J. Han, and P. Torr. Large-scale unsupervised semantic segmentation. In *ARXIV*, 2021.

[73] V. Garcia and J. Bruna. Few-shot learning with graph neural networks. In *ICLR*, 2018.

[74] K. Gavrilyuk, A. Ghodrati, Z. Li, and C. G. M. Snoek. Actor and action video segmentation from a sentence. In *CVPR*, 2018.

[75] E. Gavves, T. Mensink, T. Tommasi, C. G. M. Snoek, and T. Tuytelaars. Active transfer learning with zero-shot priors: Reusing past datasets for future tasks. In *ICCV*, 2015.

[76] Z. Ge and X. Wang. Evaluation of various open-set medical imaging tasks with deep neural networks. In *ARXIV*, 2021.

[77] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber. Learning precise timing with lstm recurrent networks. In *JMLR*, 2002.

[78] G. Giannone, D. Nielsen, and O. Winther. Few-shot diffusion models. In *ARXIV*, 2022.

[79] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018.

[80] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *ARXIV*, 2017.

[81] R. Girshick. Fast R-CNN. In *ICCV*, 2015.

[82] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

[83] R. Gokberk Cinbis, J. Verbeek, and C. Schmid. Multi-fold mil training for weakly supervised object localization. In *CVPR*, 2014.

[84] H. Guo, Y. Mao, and R. Zhang. Mixup as locally linear out-of-manifold regularization. In *AAAI*, 2019.

[85] X. Guo, D. Liu, B. Jou, M. Zhu, A. Cai, and S.-F. Chang. Robust object co-detection. In *CVPR*, 2013.

[86] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool. Creating summaries from user videos. In *ECCV*, 2014.

[87] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006.

[88] M. Hamilton, Z. Zhang, B. Hariharan, N. Snavely, and W. T. Freeman. Unsupervised semantic segmentation by distilling feature correspondences. In *ICLR*, 2022.

[89] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.

[90] B. Hariharan and R. Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *ICCV*, 2017.

[91] Z. Hayder, M. Salzmann, and X. He. Object co-detection via efficient inference in a fully-connected crf. In *ECCV*, 2014.

[92] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

[93] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[94] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017.

[95] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[96] L. A. Hendricks, O. Wang, E. Shechtman, J. Sivic, T. Darrell, and B. Russell. Localizing moments in video with natural language. In *ICCV*, 2017.

[97] L. A. Hendricks, O. Wang, E. Shechtman, J. Sivic, T. Darrell, and B. Russell. Localizing moments in video with temporal language. In *ARXIV*, 2018.

[98] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[99] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[100] J. Ho and T. Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop*, 2021.

[101] S. Hochreiter and J. Schmidhuber. Long short-term memory. In *Neural Computation*, 1997.

[102] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *IWPR*, 2015.

[103] C. J. Holder and M. Shafique. On efficient real-time semantic segmentation: a survey. In *ARXIV*, 2022.

[104] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[105] Q. Hu, B. Yang, G. Fang, Y. Guo, A. Leonardis, N. Trigoni, and A. Markham. Sqn: Weakly-supervised semantic segmentation of large-scale 3d point clouds. In *ECCV*, 2022.

[106] T. Hu. Dense in dense: Training segmentation from scratch. In *ACCV*, 2019.

[107] T. Hu, P. Mettes, J.-H. Huang, and C. G. M. Snoek. Silco: Show a few images, localize the common object. In *ICCV*, 2019.

[108] T. Hu, Y. Wang, P. Lu, and H. Wang. Sobel heuristic kernel for aerial semantic segmentation. In *ICIP*, 2018.

[109] T. Hu, P. Yang, C. Zhang, G. Yu, Y. Mu, and C. G. M. Snoek. Attention-based multi-context guiding for few-shot semantic segmentation. In *AAAI*, 2019.

[110] V. T. Hu, D. W. Zhang, Y. M. Asano, G. J. Burghouts, and C. G. M. Snoek. Self-guided diffusion models. In *CVPR*, 2023.

[111] B.-S. Hua, Q.-H. Pham, D. T. Nguyen, M.-K. Tran, L.-F. Yu, and S.-K. Yeung. Scenenn: A scene meshes dataset with annotations. In *3DV*, 2016.

[112] H. Idrees, A. R. Zamir, Y.-G. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah. The thumos challenge on action recognition for videos "in the wild". In *CVIU*, 2017.

[113] I. Igashov, H. Stärk, C. Vignac, V. G. Satorras, P. Frossard, M. Welling, M. Bronstein, and B. Correia. Equivariant 3d-conditional diffusion models for molecular linker design. In *ARXIV*, 2022.

[114] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[115] M. Jain, A. Ghodrati, and C. G. M. Snoek. ActionBytes: Learning from trimmed videos to localize actions. In *CVPR*, 2020.

[116] M. Jain, J. C. van Gemert, T. Mensink, and C. G. M. Snoek. Objects2action: Classifying and localizing actions without any video example. In *ICCV*, 2015.

[117] X. Ji, J. F. Henriques, and A. Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *ICCV*, 2019.

[118] Z. Jia, M. Dong, J. Ru, L. Xue, S. Yang, and C. Li. Stcm-net: A symmetrical one-stage network for temporal language localization in videos. In *Neurocomputing*, 2022.

[119] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with gpus. In *ARXIV*, 2017.

[120] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. In *ARXIV*, 2016.

[121] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid. Joint learning of object and action detectors. In *ICCV*, 2017.

[122] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell. Few-shot object detection via feature reweighting. In *ARXIV*, 2018.

[123] L. Karlinsky, J. Shtok, S. Harary, E. Schwartz, A. Aides, R. Feris, R. Giryes, and A. M. Bronstein. RepMet: Representative-based metric learning for classification and few-shot object detection. In *CVPR*, 2019.

[124] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[125] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[126] T. Kasarla, G. J. Burghouts, M. van Spengler, E. van der Pol, R. Cucchiara, and P. Mettes. Maximum class separation as inductive bias in one matrix. In *NeurIPS*, 2022.

[127] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, 2020.

[128] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman. The kinetics human action video dataset. In *ARXIV*, 2017.

[129] H. Kervadec, J. Bouchtiba, C. Desrosiers, E. Granger, J. Dolz, and I. Ben Ayed. Boundary loss for highly unbalanced segmentation. In *MIDL*, 2019.

[130] D. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models. In *NeurIPS*, 2021.

[131] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *AISTATS*, 2016.

[132] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ARXIV*, 2016.

[133] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Workshop*, 2015.

[134] G. Kreiman. *Biological and Computer Vision*. Cambridge University Press, 2021.

[135] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. In *ARXIV*, 2009.

[136] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[137] H. Kuehne, A. Richard, and J. Gall. A hybrid rnn-hmm approach for weakly supervised temporal action segmentation. In *ARXIV*, 2019.

[138] K. Kumar Singh and Y. Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *ICCV*, 2017.

[139] A. Kuznetsova, A. Talati, Y. Luo, K. Simmons, and V. Ferrari. Efficient video annotation with visual interpolation and frame selection guidance. In *WACV*, 2021.

[140] I. Laina, R. Fong, and A. Vedaldi. Quantifying learnability and describability of visual concepts emerging in representation learning. In *NeurIPS*, 2020.

[141] L. E. Lao Tzu. *Tao te ching*. Wordsworth, 1997.

[142] F. Last, G. Douzas, and F. Bacao. Oversampling for imbalanced learning based on k-means and smote. In *ARXIV*, 2017.

[143] B. Li, Y. Yao, J. Tan, G. Zhang, F. Yu, J. Lu, and Y. Luo. Equalized focal loss for dense long-tailed object detection. In *CVPR*, 2022.

[144] J. Li, B. M. Chen, and G. Hee Lee. So-net: Self-organizing network for point cloud analysis. In *CVPR*, 2018.

[145] R. Li, X. Li, P.-A. Heng, and C.-W. Fu. Pointaugment: an auto-augmentation framework for point cloud classification. In *CVPR*, 2020.

[146] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen. Pointcnn: Convolution on x-transformed points. In *NeurIPS*, 2018.

[147] Z. Li, K. Kamnitsas, and B. Glocker. Analyzing overfitting under class imbalance in neural networks for image segmentation. In *IEEE Transactions on Medical Imaging*, 2020.

[148] T. Lin, X. Zhao, H. Su, C. Wang, and M. Yang. BSN: Boundary sensitive network for temporal action proposal generation. In *ECCV*, 2018.

[149] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[150] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[151] M. Liu, L. Sheng, S. Yang, J. Shao, and S.-M. Hu. Morphing and sampling network for dense point cloud completion. In *CoRR*, 2019.

[152] N. Liu, S. Li, Y. Du, A. Torralba, and J. B. Tenenbaum. Compositional visual generation with composable diffusion models. In *ECCV*, 2022.

[153] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.

[154] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

[155] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[156] J. Lu, J. Yao, J. Zhang, X. Zhu, H. Xu, W. Gao, C. Xu, T. Xiang, and L. Zhang. Soft: softmax-free transformer with linear complexity. In *NeurIPS*, 2021.

[157] M. Lučić, M. Tschannen, M. Ritter, X. Zhai, O. Bachem, and S. Gelly. High-fidelity image generation with fewer labels. In *ICML*, 2019.

[158] T. Luo, A. Li, T. Xiang, W. Huang, and L. Wang. Few-shot learning with global class representations. In *ARXIV*, 2019.

[159] S. Ma, L. Sigal, and S. Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *CVPR*, 2016.

[160] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. In *JMLR*, 2008.

[161] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez. Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark. In *IEEE International Geoscience and Remote Sensing Symposium*, 2017.

[162] P. Mangla, N. Kumari, M. Singh, B. Krishnamurthy, and V. N. Balasubramanian. Data instance prior (disp) in generative adversarial networks. In *WACV*, 2022.

[163] G. Marcus. Deep learning: A critical appraisal. In *ARXIV*, 2018.

[164] L. Melas-Kyriazi, C. Rupprecht, I. Laina, and A. Vedaldi. Deep spectral methods: A surprisingly strong baseline for unsupervised semantic segmentation and localization. In *CVPR*, 2022.

[165] P. Mettes and C. G. M. Snoek. Spatial-aware object embeddings for zero-shot localization and classification of actions. In *ICCV*, 2017.

[166] P. Mettes and C. G. M. Snoek. Pointly-supervised action localization. In *IJCV*, 2019.

[167] C. Michaelis, M. Bethge, and A. S. Ecker. One-shot segmentation in clutter. In *ICML*, 2018.

[168] C. Michaelis, I. Ustyuzhaninov, M. Bethge, and A. S. Ecker. One-shot instance segmentation. In *ARXIV*, 2018.

[169] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, 2013.

[170] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos. Image segmentation using deep learning: A survey. In *PAMI*, 2021.

[171] M. Mirza and S. Osindero. Conditional generative adversarial nets. In *ARXIV*, 2014.

[172] S. Morozov, A. Voynov, and A. Babenko. On self-supervised image representations for gan evaluation. In *ICLR*, 2020.

[173] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[174] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

[175] P. Nguyen, T. Liu, G. Prasad, and B. Han. Weakly supervised action localization by sparse temporal pooling network. In *CVPR*, 2018.

[176] P. X. Nguyen, D. Ramanan, and C. C. Fowlkes. Weakly-supervised action localization with background modeling. In *ICCV*, 2019.

[177] A. Obukhov, M. Seitzer, P.-W. Wu, S. Zhydenko, J. Kyl, and E. Y.-J. Lin. High-fidelity performance metrics for generative models in pytorch, 2020.

[178] K. Oksuz, B. C. Cam, S. Kalkan, and E. Akbas. Imbalance problems in object detection: A review. In *ARXIV*, 2019.

[179] D. Oneata, J. Verbeek, and C. Schmid. Action and event recognition with fisher vectors on a compact feature set. In *ICCV*, 2013.

[180] A. Panteli, J. Teuwen, H. Horlings, and E. Gavves. Sparse-shot learning with exclusive cross-entropy for extremely many localisations. In *ICCV*, 2021.

[181] G. Parmar, R. Zhang, and J.-Y. Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *CVPR*, 2022.

[182] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *ARXIV*, 2017.

[183] S. Paul, S. Roy, and A. K. Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification. In *ECCV*, 2018.

[184] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[185] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *ARXIV*, 2018.

[186] K. Preechakul, N. Chatthee, S. Wizadwongsa, and S. Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *CVPR*, 2022.

[187] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

[188] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017.

[189] S. Qiao, L.-C. Chen, and A. Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *CVPR*, 2021.

[190] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[191] K. Rakelly, E. Shelhamer, T. Darrell, A. Efros, and S. Levine. Conditional networks for few-shot semantic segmentation. In *ICLR Workshop*, 2018.

[192] K. Rakelly, E. Shelhamer, T. Darrell, A. A. Efros, and S. Levine. Few-shot segmentation propagation with guided networks. In *ARXIV*, 2018.

[193] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. In *ARXIV*, 2022.

[194] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021.

[195] S. Ravanbakhsh, J. Schneider, and B. Poczos. Deep learning with sets and point clouds. In *ARXIV*, 2016.

[196] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017.

[197] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[198] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[199] J. T. Rolfe. Discrete variational autoencoders. In *ICLR*, 2017.

[200] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[201] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[202] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. In *IJCV*, 2000.

[203] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. In *ARXIV*, 2022.

[204] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *NeurIPS*, 2016.

[205] J. Sawatzky, M. Garbade, and J. Gall. Ex paucis plura: Learning affordance segmentation from very few examples. In *GCPR*, 2018.

[206] J. Serrà, D. Surís, M. Miron, and A. Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *ARXIV*, 2018.

[207] A. Shaban, S. Bansal, Z. Liu, I. Essa, and B. Boots. One-shot learning for semantic segmentation. In *BMVC*, 2017.

[208] A. Shaban, A. Rahimi, S. Gould, B. Boots, and R. Hartley. Learning to find common objects across image collections. In *ICCV*, 2019.

[209] M. Shanahan. Talking about large language models. In *ARXIV*, 2022.

[210] M. Shi, H. Caesar, and V. Ferrari. Weakly supervised object localization using things and stuff transfer. In *ICCV*, 2017.

[211] Z. Shou, D. Wang, and S.-F. Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, 2016.

[212] O. Siméoni, G. Puy, H. V. Vo, S. Roburin, S. Gidaris, A. Bursuc, P. Pérez, R. Marlet, and J. Ponce. Localizing objects with self-supervised transformers and no labels. In *BMVC*, 2021.

[213] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *NeurIPS*, 2014.

[214] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *CVPR*, 2016.

[215] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017.

[216] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

[217] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[218] J. Song, H. Zhang, X. Li, L. Gao, M. Wang, and R. Hong. Self-supervised video hashing with hierarchical binary auto-encoder. In *TIP*, 2018.

[219] Y. Song, L. Shen, L. Xing, and S. Ermon. Solving inverse problems in medical imaging with score-based generative models. In *ICLR*, 2022.

[220] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[221] K. Soomro and M. Shah. Unsupervised action discovery and localization in videos. In *ICCV*, 2017.

[222] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. In *JMLR*, 2014.

[223] W. Sultani, C. Chen, and M. Shah. Real-world anomaly detection in surveillance videos. In *CVPR*, 2018.

[224] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018.

[225] K. Suzuki. Overview of deep learning in medical imaging. In *Radiological Physics and Technology*, 2017.

[226] M. Suzuki, K. Nakayama, and Y. Matsuo. Joint multimodal learning with deep generative models. In *ICLR*, 2016.

[227] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, et al. Going deeper with convolutions. In *CVPR*, 2015.

[228] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

[229] M. Tang, F. Perazzi, A. Djelouah, I. Ben Ayed, C. Schroers, and Y. Boykov. On regularized losses for weakly-supervised cnn segmentation. In *ECCV*, 2018.

[230] P. Tang, X. Wang, A. Wang, Y. Yan, W. Liu, J. Huang, and A. Yuille. Weakly supervised region proposal network and object detection. In *ECCV*, 2018.

[231] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019.

[232] M. Toering, I. Gatopoulos, M. Stol, and V. T. Hu. Self-supervised video representation learning with cross-stream prototypical contrasting. In *WACV*, 2022.

[233] Y. Tokozume, Y. Ushiku, and T. Harada. Between-class learning for image classification. In *CVPR*, 2018.

[234] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[235] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *ICCV*, 2019.

[236] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[237] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, A. Courville, D. Lopez-Paz, and Y. Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019.

[238] V. Verma, A. Lamb, J. Kannala, Y. Bengio, and D. Lopez-Paz. Interpolation consistency training for semi-supervised learning. In *IJCAI*, 2019.

[239] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016.

[240] H. V. Vo, P. Pérez, and J. Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *ECCV*, 2020.

[241] V. H. Vo, E. Sizikova, C. Schmid, P. Pérez, and J. Ponce. Large-scale unsupervised object discovery. In *NeurIPS*, 2021.

[242] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification. In *CVPR*, 2017.

[243] L. Wang, Y. Qiao, and X. Tang. Action recognition and detection by combining motion and appearance features. In *THUMOS14 Action Recognition Challenge*, 2014.

[244] L. Wang, Y. Xiong, D. Lin, and L. V. Gool. Untrimmednets for weakly supervised action recognition and detection. In *CVPR*, 2017.

[245] T. Wang, Y. Li, B. Kang, J. Li, J. Liew, S. Tang, S. Hoi, and J. Feng. The devil is in classification: A simple framework for long-tail instance segmentation. In *ECCV*, 2020.

[246] W. Wang, J. Bao, W. Zhou, D. Chen, D. Chen, L. Yuan, and H. Li. Semantic image synthesis via diffusion models. In *ARXIV*, 2022.

[247] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, 2018.

[248] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. In *TOG*, 2019.

[249] Z. Wang, Y. Wu, K. Narasimhan, and O. Russakovsky. Multi-query video retrieval. In *ECCV*, 2022.

[250] F. Wirth, J. Quehl, J. Ota, and C. Stiller. Pointatme: efficient 3d point cloud labeling in virtual reality. In *IEEE Intelligent Vehicles Symposium*, 2019.

[251] C.-Y. Wu, C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl, and R. Girshick. Long-term feature banks for detailed video understanding. In *CVPR*, 2019.

[252] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl. Sampling matters in deep embedding learning. In *ICCV*, 2017.

[253] W. Wu, Z. Qi, and L. Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, 2019.

[254] Y. Wu et al. Tensorpack. `https://github.com/tensorpack/`, 2016.

[255] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.

[256] Y. Xian, S. Choudhury, Y. He, B. Schiele, and Z. Akata. Semantic projection network for zero-and few-label semantic segmentation. In *CVPR*, 2019.

[257] Y. Xian, S. Sharma, B. Schiele, and Z. Akata. f-VAEGAN-D2: A feature generating framework for any-shot learning. In *CVPR*, 2019.

[258] S. Xiao, L. Chen, S. Zhang, W. Ji, J. Shao, L. Ye, and J. Xiao. Boundary proposal network for two-stage natural language video localization. In *AAAI*, 2021.

[259] E. Xie, P. Sun, X. Song, W. Wang, X. Liu, D. Liang, C. Shen, and P. Luo. Polarmask: Single shot instance segmentation with polar representation. In *CVPR*, 2020.

[260] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 2015.

[261] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. In *ARXIV*, 2015.

[262] H. Xu, A. Das, and K. Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *ICCV*, 2017.

[263] H. Xu, K. He, B. A. Plummer, L. Sigal, S. Sclaroff, and K. Saenko. Multilevel language and vision integration for text-to-clip retrieval. In *AAAI*, 2019.

[264] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *ECCV*, 2018.

[265] A. Yang, A. Miech, J. Sivic, I. Laptev, and C. Schmid. Tubedetr: Spatio-temporal video grounding with transformers. In *CVPR*, 2022.

[266] F. S. Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018.

[267] H. Yang, X. He, and F. Porikli. One-shot action localization by learning sequence matching network. In *CVPR*, 2018.

[268] J. Yang and J. Yuan. Common action discovery and localization in unconstrained videos. In *ICCV*, 2017.

[269] L. Yang, H. Jiang, Q. Song, and J. Guo. A survey on long-tailed visual recognition. In *IJCV*, 2022.

[270] P. Yang, V. T. Hu, P. Mettes, and C. G. M. Snoek. Localizing the common action among a few videos. In *ECCV*, 2020.

[271] R. Yang, P. Srivastava, and S. Mandt. Diffusion probabilistic modeling for video generation. In *ARXIV*, 2022.

[272] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *CVPR*, 2016.

[273] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. In *ARXIV*, 2015.

[274] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *NeurIPS*, 2017.

[275] C. Zhang, T. Hu, Y. Guan, and Z. Ye. Accelerating convolutional neural networks with dynamic channel pruning. In *DCC*, 2019.

[276] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2017.

[277] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *CVPR*, 2017.

[278] W. Zhang, J. Pang, K. Chen, and C. C. Loy. Dense siamese network. In *ECCV*, 2022.

[279] X. Zhang, Z. Wu, Z. Weng, H. Fu, J. Chen, Y.-G. Jiang, and L. S. Davis. Videolt: large-scale long-tailed video recognition. In *ICCV*, 2021.

[280] Y. Zhang, B. Kang, B. Hooi, S. Yan, and J. Feng. Deep long-tailed learning: A survey. In *ARXIV*, 2021.

[281] Z. Zhang, T. He, H. Zhang, Z. Zhang, J. Xie, and M. Li. Bag of freebies for training object detection neural networks. In *CoRR*, 2019.

[282] Z. Zhang, B.-S. Hua, and S.-K. Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *ICCV*, 2019.

[283] Z. Zhang, Z. Zhao, Z. Lin, J. Song, and D. Cai. Localizing unseen activities in video via image query. In *IJCAI*, 2019.

[284] C. Zhao, M. Ramazanova, M. Xu, and B. Ghanem. Segtad: Precise temporal action detection via semantic segmentation. In *ECCV Workshop*, 2022.

[285] S. Zhao, L. Zhu, X. Wang, and Y. Yang. Centerclip: Token clustering for efficient text-video retrieval. In *SIGIR*, 2022.

[286] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016.

[287] Y. Zhou, R. Zhang, C. Chen, C. Li, C. Tensmeyer, T. Yu, J. Gu, J. Xu, and T. Sun. Lafite: Towards language-free training for text-to-image generation. In *CVPR*, 2022.

[288] J. Zhu, Z. Wang, J. Chen, Y.-P. P. Chen, and Y.-G. Jiang. Balanced contrastive learning for long-tailed visual recognition. In *CVPR*, 2022.

[289] A. Ziegler and Y. M. Asano. Self-supervised learning of object parts for semantic segmentation. In *CVPR*, 2022.

SAMENVATTING

Dit proefschrift heeft als doel label-efficiënt leren te bereiken in computervisie. We laten verschillende algoritmen zien die een aanzienlijk verminderde set van labels gebruiken, of zelfs helemaal geen labels, om label-efficiëntie te bereiken. De belangrijkste onderzoeksvraag wordt behandeld in zes typische uitdagingen op het gebied van computervisie: beeldsegmentatie, objectdetectie, video lokalisatie, video retrieval, puntwolkclassificatie en generatieve beelddiffusie. Een korte samenvatting van elke label-efficiënte computervisie-oplossing per hoofdstuk wordt nu gegeven:

CHAPTER 2 BEELDSEGMENTATIE. Om label-efficiënt leren in beeldsegmentatie aan te pakken, stellen we het Attention-based Multi-Context Guiding (A-MCG) netwerk voor, dat bestaat uit drie takken: de ondersteuningstak, de vraagtak, en de feature-fusietak. Een belangrijk onderscheidend kenmerk van A-MCG is de integratie van multi-schaal contextuele kenmerken tussen ondersteunende en vragende takken, die een betere begeleiding vanuit de ondersteuningsset afdwingen. Daarnaast introduceren we ook een ruimtelijk aandachtsmechanisme langs de fusietak om contextinformatie van verschillende schalen te benadrukken, waardoor zelftoezicht in one-shot leren wordt verbeterd. Om het fusieprobleem in multi-shot leren aan te pakken, wordt ConvLSTM gebruikt om de opeenvolgende ondersteuningskenmerken samen te voegen om de uiteindelijke nauwkeurigheid te verhogen. Onze architectuur behaalt een state-of-the-art resultaat op ongeziene klassen in een variant van de PASCAL VOC12 dataset en presteert gunstig tegenover het vorige werk met grote winsten van 1,1%, 1,4% gemeten in mIoU in respectievelijk de 1-shot of 5-shot setting.

CHAPTER 3 OBJECTDETECTIE. In dit hoofdstuk stellen we een nieuwe taak voor om het label-efficiënt leren in objectdetectie te onderzoeken, we noemen het few-shot gemeenschappelijke lokalisatie. Gegeven een paar zwak gesuperviseerde ondersteuningsafbeeldingen, is ons doel om het gemeenschappelijke object in de vraagafbeelding te lokaliseren zonder enige box-annotatie. Deze taak verschilt van standaard few-shot instellingen, aangezien we het lokalisatieprobleem willen aanpakken, in plaats van het globale classificatieprobleem. Om dit nieuwe probleem aan te pakken, stellen we een netwerk voor dat het meeste uit de ondersteunende en vragende afbeeldingen wil halen. Hiervoor introduceren we een ruimtelijke gelijkheidsmodule die de ruimtelijke gemeenschappelijkheid onder de gegeven afbeeldingen zoekt. We introduceren ook een feature herwegingsmodule, die de invloed van verschillende ondersteuningsbeelden balanceert door middel van graaf-convolutionele netwerken. Om few-shot gemeenschappelijke lokalisatie te evalueren, herformuleren en herorganiseren we de bekende Pascal VOC en MS-COCO datasets, evenals een videodataset van Imagenet VID. Experimentele evaluatie van de nieuwe instellingen voor few-shot gemeenschappelijke lokalisatie toont het belang aan van het zoeken naar ruimtelijke gelijkheid en kenmerkherweging, waarbij baselines van gerelateerde taken worden overtroffen.

CHAPTER 4 VIDEO ACTIE LOKALISATIE. Om label-efficiënt leren in videoactie lokalisatie te onderzoeken, streeft dit hoofdstuk ernaar om de temporele omvang van actie in een lange ongetrimde video te lokaliseren. Waar bestaand werk veel voorbeelden met hun start, hun einde en/of de klasse van de actie tijdens de trainingstijd benut, stellen wij few-shot gemeenschappelijke actie lokalisatie voor. Het begin en einde van een actie in een lange ongetrimde video worden

bepaald op basis van slechts een handvol getrimde video-exemplaren die dezelfde actie bevatten, zonder hun gemeenschappelijke klasse label te kennen. Om deze taak aan te pakken, introduceren we een nieuwe 3D convolutionele netwerk architectuur die in staat is om representaties van de ondersteuningsvideo's af te stemmen op de relevante query video segmenten. Het netwerk bevat: (*i*) een gemeenschappelijke verbeteringsmodule om gelijktijdig de representatie van de weinige getrimde ondersteuningsvideo's en de ongetrimde query video te complementeren; (*ii*) een progressieve uitlijningsmodule die iteratief de ondersteuningsvideo's in de query tak fuseert; en (*iii*) een paarsgewijze matchingsmodule om het belang van verschillende ondersteuningsvideo's te wegen. Evaluatie van few-shot gemeenschappelijke actie lokalisatie in ongetrimde video's met enkele of meerdere actie-instanties toont de effectiviteit en algemene toepasbaarheid van ons voorstel aan.

CHAPTER 5 VIDEO ACTIE OPHALEN. Dit hoofdstuk beschouwt het ophalen van video's die menselijke activiteit bevatten van slechts een video query om label-efficiëntie in onevenwichtige scenario's te onderzoeken. In de literatuur is een veelvoorkomende aanname dat alle activiteiten voldoende gelabelde voorbeelden hebben bij het leren van een embedding voor het ophalen. Deze aanname geldt echter niet in de praktijk, aangezien slechts een deel van de activiteiten veel voorbeelden heeft, terwijl andere activiteiten slechts door een paar voorbeelden worden beschreven. In dit hoofdstuk stellen we een visueel-semantisch embedding netwerk voor dat expliciet omgaat met het onevenwichtige scenario voor activiteiten ophalen. Ons netwerk bevat twee nieuwe modules. De visuele afstemmingsmodule voert een globale afstemming uit tussen de inputvideo en visuele feature bank representaties voor alle activiteiten. De semantische module stemt af tussen de inputvideo en semantische activiteitsrepresentaties. Door video's te matchen met zowel visuele als semantische activiteitsrepresentaties van totale activiteiten, negeren we niet langer zeldzame activiteiten tijdens het ophalen. Experimenten op een nieuwe onevenwichtige activiteitenophalingsbenchmark tonen de effectiviteit van ons voorstel aan.

CHAPTER 6 PUNTWOLK CLASSIFICATIE. Dit hoofdstuk introduceert data-vermeerdering voor puntwolken door interpolatie tussen voorbeelden om label-efficiënt leren in puntwolk te bereiken. Data-vermeerdering door interpolatie heeft zich bewezen als een eenvoudige en effectieve aanpak in het beelddomein. Een dergelijke mixup techniek is echter niet direct overdraagbaar op puntwolken, aangezien we geen één-op-één correspondentie hebben tussen de punten van twee verschillende objecten. In dit hoofdstuk definiëren we data-vermeerdering tussen puntwolken als een kortste pad lineaire interpolatie. Daartoe introduceren we PointMixup, een interpolatiemethode die nieuwe voorbeelden genereert door middel van een optimale toewijzing van de padfunctie tussen twee puntwolken. We bewijzen dat onze PointMixup het kortste pad vindt tussen twee puntwolken en dat de interpolatie toewijzingsmatrices globaal optimaal zijn. Experimenten tonen aan dat PointMixup het label-efficiënte leren van 3D objecten kan bevorderen.

CHAPTER 7 GENERATIEVE BEELDDIFFUSIE. Dit hoofdstuk onderzoekt label-efficiënt leren door het genereren van ongelabelde afbeeldingen. We presenteren een framework dat ongelabelde afbeeldingen genereert door diffusieprocessen en probeert label-efficiënt leren te bereiken door zelfsupervisie. Het door ons voorgestelde framework, Self-supervised Generative Diffusion Model (SGDM), bestaat uit twee componenten: een generatief diffusiemodel en een paar diffusiegerichte zelfsupervisie-modules. Het generatieve diffusiemodel is verantwoordelijk voor het genereren van ongelabelde beelden, terwijl de zelfsupervisie-modules geleidende signalen bieden om het leren van de ongelabelde beelden te begeleiden. Onze SGDM streeft ernaar om het label-efficiënte leren in ongelabelde gegevens te maximaliseren, en toont veelbelovende resultaten op diverse beeldgeneratiebenchmarks.

8.3 CONCLUSIE EN TOEKOMSTIG WERK

Dit proefschrift onderzoekt label-efficiënt leren in zes typische uitdagingen in de computervisie. Het blijkt dat label-efficiënt leren kan worden bereikt door het ontwerp van gespecialiseerde algoritmen voor elke taak, zoals beeldsegmentatie, objectdetectie, videoactie lokalisatie en ophalen, puntwolk classificatie, en generatieve beelddiffusie. Terwijl we specifieke oplossingen bieden voor elk van deze taken, merken we ook dat sommige principes over alle taken heen gedeeld worden. In het bijzonder vinden we dat de keuze van de begeleiding (zoals ondersteuningsbeelden in few-shot taken, of door mensen geannoteerde beelden in puntwolk classificatie en generatieve beelddiffusie) een cruciale rol speelt bij het bereiken van label-efficiënt leren.

In de toekomst kunnen we overwegen om de concepten en principes die we in dit proefschrift hebben geïdentificeerd, toe te passen op nieuwe domeinen of taken. Er zijn bijvoorbeeld veel andere taken in de computervisie die kunnen profiteren van label-efficiënt leren, zoals actieherkenning, gezichtsherkenning, en beeldsynthese. Bovendien kan de combinatie van de verschillende principes en technieken die we hebben ontdekt, wellicht leiden tot nieuwe benaderingen of theorieën voor label-efficiënt leren.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Cees Snoek and Pascal Mettes, for their invaluable guidance throughout my PhD journey. Their unwavering support have been instrumental in my academic success. I would like to thank my committee members, Theo Gevers, Marcel Worring, Efstratios Gavves, Bernard Ghanem, and Tim Salimans, for their efforts in making my PhD defense possible. I would also like to thank Sarah Ibrahimi for translating the summary of the thesis. Finally, I would like to thank Teng Long and Di Wu for accompanying me during the defense.

Thanks David, Yunlu, Yuki, Pengwan for the coorporation. Thanks Teng and Di for the organizing of the defense. Thanks to everyone from VISLab, CVLab, and other groups in UvA. Thanks to my supervised students, Andrea Rigo, Martine Toering, Hinrik Snær, and Paulien Rouwendaal. A special thanks to Dennis, Petra, and Virgine for setting everything up for our research life. I am also grateful to Zhihua Zhang, Yadong Mu, Gang Yu, Heng Wang, and Deli Zhao for writing reference letters to help me set up my research journey. Thanks to Davide, Joseph Tighe, and other team members at Amazon during my internship. I learned a lot from all of you.

Finally, I would like to thank my family, and especially my wife, Sisi Chen, for their unwavering support and encouragement throughout my PhD journey. Their love and guidance have been a constant source of inspiration, and have helped me overcome the challenges along the way.