Learning in Low Data Regimes for Image and Video Understanding

Mihai Marian Puscas

ICT International Doctoral School,
Department of Information Engineering and Computer Science

The University of Trento

Advisor: Prof. Dr. Nicu Sebe
A thesis submitted for the degree of

Doctor of Philosophy (PhD)

May 2019

1. Reviewer:

- Prof. Alberto del Bimbo,
  University of Firenze, Italy
- Prof. Noel O'Connor,
  Dublin City University, Ireland

Day of the defense: 3/5/2019

Signature from head of PhD committee:

Abstract

The use of Deep Neural Networks with their increased representational power has allowed for great progress in core areas of computer vision, and in their applications to our day-to-day life. Unfortunately the performance of these systems rests on the "big data" assumption, where large quantities of annotated data are freely and legally available for use. This assumption may not hold due to a variety of factors: legal restrictions, difficulty in gathering samples, expense of annotations, hindering the broad applicability of deep learning methods.

This thesis studies and provides solutions for different types of data scarcity: (i) the annotation task is prohibitively expensive, (ii) the gathered data is in a long tail distribution, (iii) data storage is restricted.

For the first case, specifically for use in video understanding tasks, we have developed a class agnostic, unsupervised spatio-temporal proposal system learned in a transductive manner, and a more precise pixel-level unsupervised graph based video segmentation method. At the same time, we have developed a cycled, generative, unsupervised depth estimation system that can be further used in image understanding tasks, avoiding the use of expensive depth map annotations.

Further, for use in cases where the gathered data is scarce we have developed two few-shot image classification systems: a method that makes use of category-specific 3D models to generate novel samples, and one that increases novel sample diversity by making use of textual data.

Finally, data collection and annotation can be legally restricted, significantly impacting the function of lifelong learning systems. To overcome catastrophic forgetting exacerbated by data storage limitations, we have developed a deep generative memory network that functions in a strictly class incremental setup.

Contents

[1] "Unsupervised Tube Extraction Using Transductive Learning and Dense Trajectories" Mihai Marian Puscas, Enver Sangineto, Dubravko Culibrk, Nicu Sebe; The IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1653-1661 (1)

[1] "Joint Graph Learning and Video Segmentation via Multiple Cues and Topology Calibration" Jingkuan Song, Lianli Gao, Mihai Marian Puscas, Feiping Nie, Fumin Shen, Nicu Sebe; MM '16 Proceedings of the 24th ACM international conference on Multimedia Pages 831-840 (2)

[2] "Unsupervised Adversarial Depth Estimation using Cycled Generative Networks" Andrea Pilzer*, Dan Xu*, Mihai Puscas*, Elisa Ricci, Nicu Sebe; 2018 International Conference on 3D Vision (3DV)587-595 (3)

[1] "Low-Shot Learning from Imaginary 3D Model" Frederik Pahde, Mihai Marian Puscas, Jannik Wolff, Tassilo Klein, Nicu Sebe, Moin Nabi; WACV 2019 (4)

[2] "Adversarially Learned Feature Generating Network for Low-Shot Learning"; Frederik Pahde, Mihai Marian Puscas, Jannik Wolff, Tassilo Klein, Nicu Sebe, Moin Nabi, Under review, ICCV 2019

[1] "Learning to Remember: A Synaptic Plasticity Driven Framework for Continual Learning"; Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Moin Nabi, CVPR 2019, (5)

1

Introduction

1.1 Motivations and Challenges

Deep Neural Networks have revolutionized machine learning as a whole, both scientifically and practically. Their increased representational power has allowed for progress in core areas of computer vision, such as object detection, semantic segmentation, action recognition, etc. leading to a dissemination and application of these method to our day-to-day lives, from mobile phones apps and search engines to autonomous driving, video surveillance and smart cities.

Even so, most deep learning systems assume having "big data" readily available, more precisely that there exists a sufficient amount of freely and legally available, ideally annotated data fit for consumption for a given deep learning task. As supervised deep learning systems require a significant amount of data - thousands of annotated images per class for object detection (6), hundreds to thousands of spatio-temporally annotated videos for video action detection (7), the assumption that these data are freely available is difficult to justify, both from a practical and a legal standpoint. While indeed the total volume of information stored electronically is constantly increasing, the vast majority of it is not annotated, unsuited or unaccessible, and as such will be expensive to make use of. At the same time, the use and storage of personal information online is becoming more regulated by various governmental bodies (8), and as such many approaches that leverage weak annotations found online may be slipping out of legality at worst or are having the quantity of input data heavily restricted, and their performance reduced, at best.

This thesis explores methods of overcoming limitations in the *quantity* and *quality* of available data in visual image and video understanding tasks. We define the *quality* of a volume

of data as the amount of annotations or low-level features associated to it, and the *quantity* a measure of the amount of gathered data. A common issue in this case arises from the long-tail distribution observed in the wild, where some classes are heavily populated by usable samples, while others have little to no examples available. Beyond accumulating the required number of samples, the evolving legal framework in which we operate has restricted the gathering and storage of usable data, leading to both a quality and quantity deficit.

As a first stage, this thesis tackles the quality issue, where expensive annotations often cannot be economically produced for more complex tasks, specifically for the use of video understanding systems. Unsupervised methods for the automatic annotation of data have been developed, leveraging temporal consistency and other cues (1) (2) to provide a spatio-temporal localization for objects in a video, and stereo image consistency (3) to create depth maps with a minimum of expended resources.

As a second stage we tackle the long-tail quantitative data deficit by developing few-shot learning solutions for categorical image classification firstly through the prediction of a categorical 3D model to further increase generated sample diversity (4), and secondly through the use of available textual information.

Finally, the tightening legal restrictions have increased the difficulty of life-long learning techniques, as in a strict interpretation data cannot be freely stored for further learning. This quantitative deficit exacerbates the catastrophic forgetting of previously learned information, an issue for which we have developed a strictly class-incremental generative image classification system (5, 9).

1.1.1 Unsupervised Video Annotation

Training state of the art supervised detection and classification algorithms requires the gathering and annotation of very large datasets, a practice that is not feasible due to time and resource limitations. While semi-supervised classification and detection methods exist, they often perform significantly worse than purely supervised systems, and are often more data sensitive i.e. difficult to apply on disparate datasets.

Thus, developing robust methods for the automatic extraction of samples from videos is of great use, if not in direct integration to a higher level method, then in greatly simplifying a human annotator's task.

A first method we present as detailed in 2.2 , "Unsupervised Tube Extraction using Transductive Learning and Dense Trajectories" outputs spatio-temporal object proposals in a given

video, that may then be used for video detection or classification. In this system we do not directly deal with category-specific object detection, but focus on extending the objectness property (the likelihood of a sample representing an object) from still images to videos by exploiting the motion information inherent in a video. The output of this system is a set of bounding boxes tracking objects of interest throughout the video, that can then be used by common object detection methods for their training necessities. Other works which deal with automatic tube proposals address this extension of objectness to the temporal domain. However, most similar approaches have the same limitation: they need a large number of tubes (usually hundreds or thousands per video clip) to reach a sufficiently high recall (10, 11) which makes these methods reliable to speed up the testing phase but not sufficiently precise to allow for weakly supervised or unsupervised training. The proposed system requires no manual user input, and is capable of robustly and precisely localizing multiple moving objects in a video.

While (1) outputs the spatio-temporal localization of moving objects accurately, it has a series of drawbacks: background objects are not taken into consideration, and more importantly the algorithm's temporal segmentation capability is limited. As such "Joint graph learning and video segmentation via multiple cues and topology calibration" (2) was developed, as detailed in Section 2.3. This system segments all objects in the video into spatio-temporal voxels by making use of a series of motion and appearence cues and learning their optimal fusion. At the same time, the graph-cutting and segmentation stages of the usual graph-based method are done jointly, greatly simplifying the overall optimization process and increasing overall accuracy.

Both developed methods can be seen as producing spatio-temporal proposals in a class agnostic and unsupervised manner, and as such can be used to inexpensively provide this information to further video understanding systems operating on unfamilliar and weakly annotated data.

Beyond generating annotations for use in further tasks, there is a need for the computation of low-level features. In cases such as the computation of depth maps from RGB images, unsupervised non-deep learning techniques based on stereo-matching exist, but are both slow and are greatly outperformed by deep learning based methods. At the same time, learning depth map computation in a supervised manner require specialized equipment beyond a stereo camera arrangement. Our work detailed in Section 2.4 uses a cyclic generative network to learn depth maps only from stereo image pairs.

1.1.2 Few-Shot Learning

Another distinct low data regime can be observed in data collected in the wild, where the distribution of available samples for a given category is often skewed, with a long 'tail'. This results in gathering few if any samples for a large number of classes of interest and complicating the optimization process of any deep network in use. The most extreme subcase is the complete failure in collecting any samples of a given modality. *Zero-Shot Learning* methods have been developed, often making use of a different modality and learning an embedding space between the data-rich source and the data-scarce task modality (12). A more relaxed subcase, and where the main research effort of this thesis has been expended, is where the number of available samples for a number of novel classes is low, *Low* or *Few* Shot Learning techniques.

Few-shot learning aims to learn a model on novel classes or tasks, where a class C_{novel} contains a small number of n samples. It assumes a subset of classes C_{base} where enough samples have been gathered for regular learning to be used. The main idea of these methods is to leverage the information learned using the base classes for learning the sparsely populated novel classes. Classically, learning both the novel and base classes in the same task, or a simple fine-tuning on the novel classes over a model learned on the base classes will result in gross overfitting and a model unsuited to the task. While this task can be seen as related to transfer learning and domain adaptation techniques, recent advances in methodology has made low-shot learning research distinct. More specifically, recent methods seek to compensate for the lack of samples by optimizing the novel class learner to this sparcity, a 'meta-learning' approach. The learner will thus require a small number of samples for its given task, either by hallucinating unseen samples (13), leveraging dataset statistics (14), or directly preventing overfitting by clustering neurons of interest (15).

As few-shot learning techniques primarily tackle a lack of samples in a given modality, it can be assumed that related information can be found in other modalities. As such we present a few-shot learning image classification that makes use of freely available textual information to empower a strong generative model that is further used in training the system for the novel classes (Section 3.3). At the same time, we have developed a system that makes use of available visual information to learn a prototypical 3D mesh and texture for the given category, that is then used to generate a large number of diverse samples for the novel classes (4), a method detailed in Section 3.2

1.1.3 Overcoming Catastrophic Forgetting

Due to evolving regulations regarding the use of personal data (8), the gathering and storage of useful information can be greatly limited. Restrictions in data storage specifically effect life-long learning systems, as without data associated to already learned knowledge stored, the system is more likely to "forget" older tasks.

These limitations make the use of strictly incremental methodologies for life-long learning tasks necessary, where as a new task arrives, all previously seen data is discarded. Unfortunately strictly incremental systems have an exacerbated catastrophic forgetting problem: older learned knowledge is overwritten by the newly learned information. The system thus forgets and loses capability with each new learning stage, in a more practical sense overfitting on each incoming task or class.

The twin factors of tightening data control and broader use of lifelong learning motivated us to develop a solution for catastrophic forgetting, as detailed in Chapter 4. The proposed method operates in a strictly class-incremental setup, where only the data required to learn the specific class is stored, and makes use of an efficient, dynamically expanding generative replay network. As new tasks are learned, the system masks neurons associated to old knowledge, and expands the generative capacity so that the number of unused, unmasked neurons is kept constant. At the same time, the generative network creates samples representative of the stored knowledge for use in learning the expanded classification task. As a more comprehensive data representation is learned, the network requires less neurons for each new tasks, leading to a saturation point after which the network capacity will not be significantly grown.

1.1.4 Thesis Outline and Contributions

To summarize, Chapter 2 contains the following unsupervised proposal systems:

- A class agnostic spatio-temporal tube proposal system (1) (Section 2.2), where the local objectness proprety is extended with available temporal information. The coarse spatio-temporal tubes are further refined in a transductive manner, using tube-specific, class agnostic detectors.

- An unsupervised pixel-level spatio-temporal segmentation system (2) (Section 2.3). The work uses a novel optimization strategy where the similarity and cutting graphs are jointly optimized. The multiple superpixel cues used have their weights automatically

learned and are then organized in different topologies which are further calibrated such that the weight similarities become comparable.

- An unsupervised depth estimation system (3) (Section 2.4) that makes use of a novel, cyclic and generative network architecture for stereo depth estimation.

Chapter 3 contains our work on few-shot learning, offering differing diversification strategies for generative systems:

- A few-shot classification system (4) (Section 3.2) that predicts a categorical 3D shape and predicted 3D meshes and textures for novel samples to diversify sample generation. The most representative generated samples are then selected through a self-paced learning module.

- Section 3.2 presents a second few-shot learning system that makes use of abundant textual information to generate cross-modal features. A strategy to combine real and generated features is suggested, allowing easy inference using only a simple nearest neighbour approach, the method outperforming competitors by a large margin.

Chapter 4 tackles issues arising in life-long learning systems when data storage is restricted, specifically catastrophic forgetting.

- A system to overcome catastrophic forgetting is presented, where we introduce a deep generative memory network that efficiently learns sparse attention maps, and dynamically expands its network capacity as new tasks arrive.

Finally, Chapter 5 holds the concluding remarks to the thesis, and prospective future works based on it.

1.1.5 Published Works

- "Unsupervised Tube Extraction Using Transductive Learning and Dense Trajectories" Mihai Marian Puscas, Enver Sangineto, Dubravko Culibrk, Nicu Sebe; The IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1653-1661 (1)

- "Joint Graph Learning and Video Segmentation via Multiple Cues and Topology Calibration" Jingkuan Song, Lianli Gao, Mihai Marian Puscas, Feiping Nie, Fumin Shen, Nicu Sebe; MM '16 Proceedings of the 24th ACM international conference on Multimedia Pages 831-840 (2)

- "Unsupervised Adversarial Depth Estimation using Cycled Generative Networks" Andrea Pilzer*, Dan Xu*, Mihai Puscas*, Elisa Ricci, Nicu Sebe; 2018 International Conference on 3D Vision (3DV)587-595 (3)

- "Low-Shot Learning from Imaginary 3D Model" Frederik Pahde, Mihai Marian Puscas, Jannik Wolff, Tassilo Klein, Nicu Sebe, Moin Nabi; WACV 2019 (4)

- "Learning to Remember what to Remember: A Synaptic Plasticity Driven Framework."; Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Moin Nabi, NIPS CL Workshop 2018. (9)

- "Unsupervised Monocular Depth Estimation using Structured Coupled Dual Generative Adversary Networks"; Mihai Marian Puscas, Dan Xu, Andrea Pilzer, Nicu Sebe, Under review, IJCAI 2019

- "Adversarially Learned Feature Generating Network for Low-Shot Learning"; Frederik Pahde, Mihai Marian Puscas, Jannik Wolff, Tassilo Klein, Nicu Sebe, Moin Nabi, Under review, ICCV 2019

- "Learning to Remember: A Synaptic Plasticity Driven Framework for Continual Learning"; Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Moin Nabi, CVPR 2019 (5)

- "Continuous Fusion for Unsupervised StereoDepth Estimation using Cycled Networks" Andrea Pilzer, Stephane Lathuiliere, Dan Xu, Mihai Puscas, Elisa Ricci, Nicu Sebe; Major Revision, IEEE TPAMI

2

Unsupervised proposal systems

In this chapter we explore solutions developed to tackle an annotation scarcity for video and image understanding tasks. Section 2.2 presents a spatio-temporal localization system that extends a locally defined "objectness" proprety to the temporal domain using long term trajectories to create tubes around objects of interest, which are further extended in a transductive manner. The graph-based method expanded upon in section 2.3 provides an unsupervised pixel-level spatio-temporal segmentation. The segmentation and cutting graphs are jointly optimized, and different cues are organized into specific topologies that are automatically calibrated. Finally, Section 2.4 provides a cycled, generative solution for stereo depth that does not require expensive LIDAR depth map annotations during the learning process.

2.1 Background

As a first stage in our thesis, we looked towards core video understanding topics, action recognition and detection (16, 17, 18), and what hinders their applicability. It can be observed that supervised methods can reach high performance on commonly used benchmarks, with datasets such as UCF-101 (7) even becoming saturated for video classification, with the drawback that they require large amounts of annotated data to train. In the case of supervised action recognition, the annotator must label only the videos, a relatively cheap task. For the more complex temporal action detection, the annotator must in addition label the time-frames where the action or actions occur. Finally, for spatio-temporal action localization the prospective annotator must also enclose the actors in bounding boxes, a much slower task than simply labelling a video. Any increase in the complexity of the task causes an increase of annotation task cost, hindering

the broader application of these methods. In the case of spatio-temporal action detection systems, or tasks which require this localization, the qualitative data scarcity can be ameliorated if there exists a system that outputs spatio-temporal proposals in an unsupervised manner, thus greatly simplifying an annotator's workload.

In **Section 2.2** we detail an unsupervised, class agnostic spatio-temporal tube proposal system that outputs tubes covering objects of interest in a given video. This interest is determined through the use of motion cues, i.e. objects the are moving throughout the video are more likely to be of interest in further tasks, and annotating them will be beneficial. As all learning performed in the proposed method is transductive, it is class agnostic and can mine proposals on a broad range of videos.

The proposals are created by extending the local, image-level, 'objectness' property to the temporal domain. 'Objectness' can be defined as the likelyhood of an image window to contain an object instead of uninformative background (19, 20, 21, 22). Further, the goal of the work is to lighten a prospective annotator's workload, leading to the need for high *precision* on the tube proposals in contrast to local, image-level object proposal systems that tend towards a high recall (19). This precision is achieved through a strict pruning of the intermediate tube proposals, followed by an appearance detection step that further filters out noise.

A limitation of the proposed system is a lack of temporal resolution, a consequence of the last stage of tube construction, where intermediate tubes are filtered and expanded using a transductive learning approach. This leads to a very poor temporal localization and limits its applicability to temporally constrained videos. A second limitation is a consequence of the temporal cues used to construct the initial tubes - only objects that are moving are considered to be of interest, an assumption that may be false when applying it broadly.

Section 2.3 presents an unsupervised video segmentation system that offers solutions to both temporal localization, and disregarded, inactive objects. Video segmentation can be defined as partitioning a video into several disjoint spatio-temporal regions such that each region has consistent appearance and motion, a broader definition than used for the previous work. Segmenting general and unconstrained videos is a challenging research problem due to existent scene and scale ambiguities of the segments (23) as well as the temporal-consistency constraints (24). Different types of video segmentation algorithms have been introduced,

from ones based on clustering (25, 26), to graph-based processing (24, 27, 28, 29) and tracking (30, 31, 32).

For the described system, we have chosen to map video elements onto a graph on which superpixels/supervoxels are nodes and edges measure similarity between them. Segmenting such a structure is generally actived in two steps: learning a similarity between nodes, and cutting the graph into semantically significant structures. Learning the similarity requires usable and cues whose weights we automatically learn and which are organized into different topological structures to keep them comparable (local, across one frame, across two frames, long term). Finally, to achieve an optimal segmentation, the graph cut and similarity are jointly optimized. In comparison to the work presented in section 2.2, this system provides a more comprehensive segmentation of a given video - with the caveat that the user must provide the total number of segments for the joint optimization to be achievable, and that any annotator will not receive any cues towards what objects might be of interest or not.

While the systems detailed in sections 2.2 and 2.3 tackle the lack of *quality* in available data through providing annotations for common video understanding tasks, there are cases where low-level features used in more complex learning tasks are expensive or even impossible to learn. This can be caused by a number of factors, such as the need of specialized sensing equipment that can be difficult to acquire and operate, require strict operating conditions, or that is simply too bulky to use in day-to-day activities.

One such case is estimating depth maps, which see broad use in various image and video understanding tasks, from robotics and autonomous driving, to virtual reality and 3D reconstruction. Great progress has been seen over the last years, with supervised deep regression methods significantly improving the accuracy of estimated depth maps (33, 34, 35, 36). The application of these systems is restricted by the need for high quality depth maps for the learning process, usually provided through the use of a LIDAR sensor, and compounding the problem, the large amount of depth maps needed to learn a deep model.

Section 2.4 describes an unsupervised stereo depth estimation system that learns a model through image correspondence between stereo image pairs, thus discarding the use of expensive depth annotations for training. More specifically, we use a novel adversarial network that better learns this correspondence field through the synthesis of the opposing image in a cyclic manner. This ensures stronger constraints between the two views, and results in the networks learning better representations and estimating more accurate depth maps.

2.2 Unsupervised Spatio-Temporal Tube Extraction [1]

2.2.1 Introduction

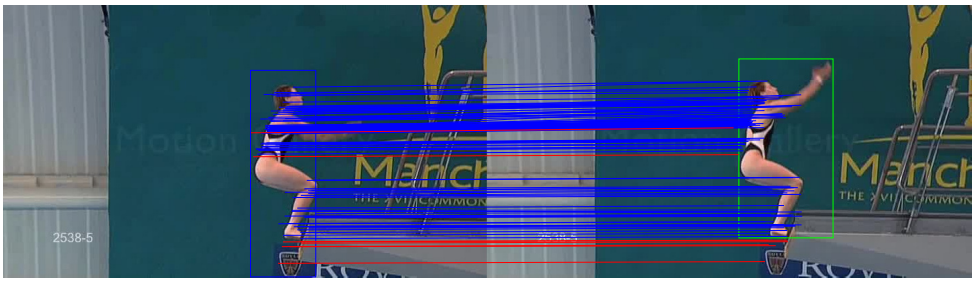


Figure 2.1: Optical Flow between two consecutive frames can be used as a "voting" mechanism for matching Bonding Boxes. The blue lines are dense trajectories in common between the two boxes, while the red lines are trajectory starting from the first box but not included in the second.

In this work we focus on extending the objectness property from still images to videos, to provide class agnostic spatio-temporal proposals for further video understanding tasks. Other works which deal with automatic tube proposals address this extension of objectness to the temporal domain. However, most of the state-of-the-art approaches have the same limitation: they need a lot of tubes (usually hundreds or thousands per video clip) to achieve a sufficiently high recall (10, 11) which makes these methods reliable to speed up the testing phase but not sufficiently precise to allow for weakly supervised or unsupervised training. Using two common benchmarks (UCF Sports and YouTube Objects) we will show that we are able to achieve high recall with few tubes. For instance, in UCF Sports we achieve more than 30% relative improvement with respect to the state-of-the-art when using only one tube (Fig. 2.4b).

These results have been achieved by combining different ideas. First, we use Selective Search in order to produce an initial set of candidate BBs. Then we propose to use Dense Trajectories (37, 38) in order to match BBs in different frames and to discard static BBs. This method allows us to collect initial tubes that we call *optical flow tubes* as they are based on the optical flow computed with Dense Trajectories. In order to avoid drifting (a common problem in all tracking algorithms), optical flow tubes are usually quite short and do not cover the

[1] "Unsupervised Tube Extraction Using Transductive Learning and Dense Trajectories" Mihai Marian Puscas, Enver Sangineto, Dubravko Culibrk, Nicu Sebe; The IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1653-1661 (1)

whole video clip. For this reason we propose using the optical flow tubes in order to collect positive samples of the moving objects and train tube-specific detectors. We highlight that no class labels or other human-provided information is used for training. Conversely, tube-specific object detectors are learned in a *transductive* framework, i.e., we do not need these detectors to generalize to other videos, except the same video in which they have been trained. In fact, once trained, we run the detectors on the input videos in order to extract the final tubes (*detection tubes*). Using this strategy, we are able to extract BBs even in frames in which the object is static, while common tube-proposal approaches usually need movement in all the frames. To summarize, our contributions are:

- We use Dense Trajectories to robustly match BBs pre-selected by means of Selective Search.
- We use tube-specific, *class agnostic* detectors, trained in a transductive learning framework, to extract the final tubes.

The code for the proposed approach is available[2].

The rest of the chapter is organized as follows. In Sec. 2.2.2 we briefly review the literature and in Sec. 2.2.3 we introduce some useful notation which will be used in the other sections. In Sec.s 2.2.4 and 2.2.5 we present our method. Experimental results are shown in Sec. 2.2.6 and we conclude in Sec. 2.2.7.

2.2.2 Related Work

In (39), Prest et al. extract tubes from a video clip exploiting homogeneous clusters of dense point tracks. The tubes are then used to learn a detector, together with video-level-based labels and based on the assumption that there is one dominant moving object per video. It is worth noticing that, in the approach we propose, the detectors are class-agnostic classifiers which are learned for every optical flow tube and then *used to extract the final tubes*. Conversely, in (39) the detectors are class-specific object detectors (fusing the segmentation phase with the final, unsupervised object classification phase). One drawback of this approach is that tubes are selected using inter-tube similarity, which is a fragile assumption when more than one moving object is present in the video clip and/or when a single object has a high variability of appearance.

[2] *https://github.com/mihaipuscas/unsupervised-tube-extraction.git*

Clustering dense tracks, obtained with optical flow, is a strategy adopted by many other authors. For instance in (40) point tracks are clustered using an affinity matrix based on the maximum translational difference between two tracks. Even if encouraging results can be obtained with this technique, articulated motion makes it hard to group tracks belonging to non-homogeneously moving objects. Optical flow is also used in (41), where objects are segmented using motion boundaries and then refined using a dynamic appearance model of the RGB foreground pixels. In (42) and in (43) optical flow and other appearance and saliency cues are used to extract coherent segments corresponding to moving objects.

In (10) the Selective Search (19) criteria for merging pixels in superpixels are extended into the time domain to obtain supervoxels. Supervoxels are used also in (11) with a hierarchical graph-based algorithm and in (44), where, instead of using heuristics, merging is performed using a classifier. In (45) motion boundaries are used in order to generate an initial set of moving object proposals, which is then ranked using a Convolutional Neural Network (CNN), trained using ground truth object BBs. It is worth noticing that both (44) and (45) are *supervised* methods, in which there is an important learning phase based on manually provided examples of ground truth objects and it is not clear what is the cross-dataset generalization capabilities of these systems (when tested on datasets different from the ones used for training), while our approach is *completely unsupervised*. A similar limitation holds in (46, 47), where a CNN is trained in order to regress multiple boxes likely containing objects. The idea behind (46, 47) is that *static* objectness can be learned using ground truth BBs contained in large datasets (Pascal and ILSVRC 2012). However, a dataset bias does exist (48), since the cross-dataset experiments presented by the authors show a drastic drop of performance of the net when trained with Pascal and tested on ILSVRC 2012 and a minor drop vice-versa.

2.2.3 Static Objectness and Notation

Given a video with T frames, we apply Selective Search (19) to each frame F_t in order to extract the set of box candidates $B_t = \{b_1^t, ...b_n^t\}$ (we drop the superscript t when not necessary), and $b_i^t = (ymin_i, xmin_i, ymax_i, xmax_i)$.

Note that we rely on Selective Search to model *static* objectness. In other words, we do not manage pixel-level information, and we leverage on Selective Search for the pixel merging task in a single image. In fact this method is widely adopted and it has been proven to have a high *recall*: for instance, with $n = 2000$, the probability of an object to be highly overlapping with any $b_i^t \in B_t$ is around 0.9 (19). All our efforts will be focused on pruning B_t ($1 \leq t \leq T$)

using movement information in order to end up with a much smaller subset of boxes containing the moving objects of the video.

For simplicity, we also do not explicitly model the dynamics of the tracked boxes (which is difficult especially with "random" movements of biological "objects"). However, we use Intersection-over-Union (IoU) and Intersection-over-Min (IoM) in order to check spatial coherence between boxes of different frames and in the same frame:

$$IoU(b_1, b_2) = A(b_1 \cap b_2)/A(b_1 \cup b_2), \tag{2.1}$$

$$IoM(b_1, b_2) = A(b_1 \cap b_2)/\min\{A(b_1), A(b_2)\}, \tag{2.2}$$

where $A(b)$ is the area of b. Both IoU and IoM are widely adopted metrics in the object detection literature (49, 50, 51, 52) to assess spatial coherence (IoU) and/or to merge small BBs in a larger rectangle (e.g., see the Non-Maxima-Suppression algorithm, NMS, used in (49, 52) and based on IoM).

Finally, we use Dense Trajectories (38) to extract dense trajectories of moving points. In (38) the authors use optical flow in order to track points over different frames. They also improve over (37) by estimating the camera motion and deleting those trajectories whose movement is similar to the camera motion. The final trajectories cluster over the actual moving objects most of the times (but unfortunately camera motion compensation is not able to delete all the noisy trajectories in the background). Trajectories are continuously created and terminated over the video frames and are usually very short (max 15 frames (37)), thus there are no trajectories spanning the whole video. Given two consecutive frames F_t and F_{t+1}, we define the (camera motion compensated) optical flow between F_t and F_{t+1} as:

$$O(t, t+1) = \{o_1, ..., o_m\}, \tag{2.3}$$

where $o_j = (p_j, q_j)$ is a local translational offset belonging to one of the active trajectories between frames F_t and F_{t+1}, p_j is the starting point ($p_j \in F_t$) and q_j the ending point ($q_j \in F_{t+1}$).

For both Selective Search and Improved Trajectories we have used the publicly available code.

2.2.4 Optical Flow Tubes

The first step of our pipeline consists in matching boxes in F_t with boxes in F_{t+1} using optical flow information and spatial coherence. Given B_t and B_{t+1}, for each $b_i \in B_t$ and $b_j \in B_{t+1}$ we define:

$$OV(i,j) := IoU(b_i, bj) \geq 0.5, \tag{2.4}$$

where the threshold 0.5 is commonly adopted in object detection (e.g., in the Pascal and ImageNet detection tasks) to assess the spatial similarity of two BBs. Even if here the context is completely different (we use OV to prune BBs too far apart from each other in two different frames), we adopt the same threshold because it somehow guarantees that b_i and b_j can be matched only when the difference in scale and/or aspect ratio is not that large. This constrains a (possibly articulated) movement of the object between F_t and F_{t+1} to produce a small translational difference and a moderate deformation. If $n_1 = |B_t|$ and $n_2 = |B_{t+1}|$, then OV is an $n_1 \times n_2$ Boolean matrix.

For each b_i, b_j such that $OV(i,j) = true$, we compute the optical flow-based matching density between b_i and b_j, defined as:

$$D(i,j) := \frac{m_{ij}}{A(b_i) + A(b_j)}, \tag{2.5}$$

where m_{ij} is the number of optical flow offsets in $O(t, t+1)$ whose starting point is in b_i and ending point in b_j. The intuitive idea behind Eq. (2.5) is straightforward. The nominator represents the number of "votes" that can be accumulated in matching b_i and b_j, being each vote an element in $O(t, t+1)$. The denominator normalizes this number by the sum of the areas of the two BBs. This normalization is necessary because of noisy trajectories (e.g. trajectories laying on the background, despite camera motion compensation). In fact, maximizing m_{ij} without area normalization leads to matching b_i with that b_j in B_{t+1} which is the *largest* possible, i.e., not a BB tight on the moving object but a BB usually including undesired background (e.g., see Fig. 2.1).

Using Eq. 2.5 we match b_i with b_j^* (and we write $M_t(b_i) = b_j^*$) such that:

$$b_j^* = \max_{b_j \in B_{t+1}} D(i,j), \tag{2.6}$$

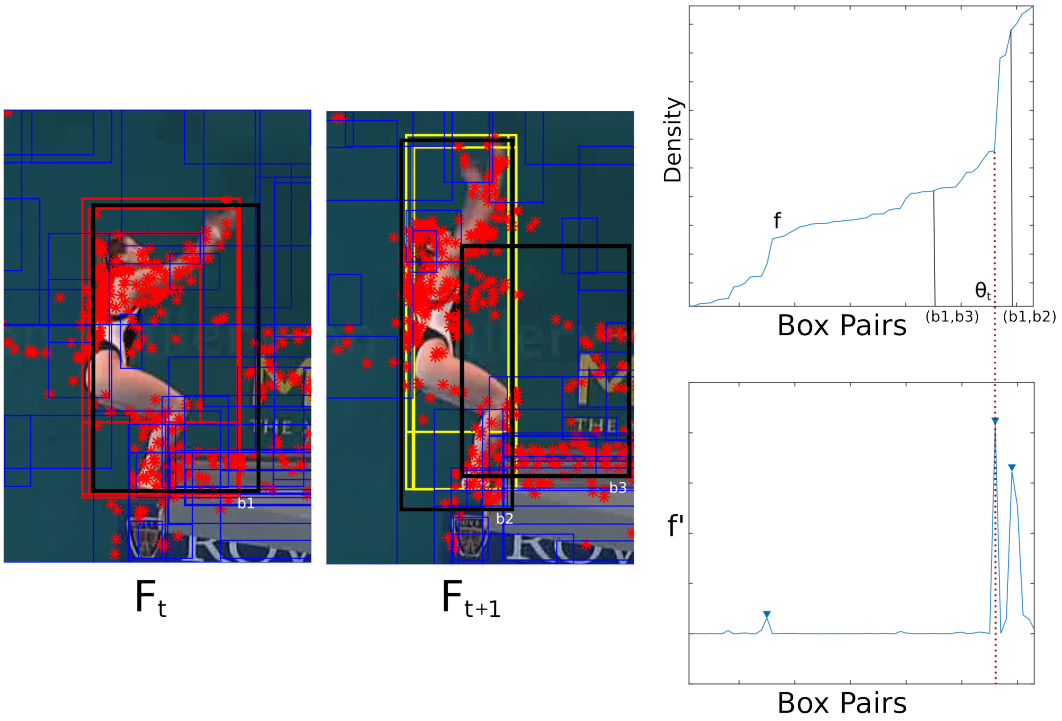


Figure 2.2: Adaptive threshold in matching density. Two consecutive frames with their initial set of BBs. Both optical flow and BBs cluster around the moving object (left). In turn, clusters correspond to plateaus in f, the sorted distribution of D (right-top). The smoothed gradient of f is used in order to detect peaks and to set the density threshold (right-bottom).

subject to:

$$D(i,j) \geq \theta_t. \tag{2.7}$$

In Eq. (2.7) θ_t is a threshold which is used to reduce the risk of drifting in tracking a BB. Instead of using a fixed threshold, which is difficult to set, we adaptively compute θ_t for every pair of frames F_t and F_{t+1}, based on the observation that BBs produced by Selective Search usually cluster around true objects and dense trajectories tend to cluster around moving objects due to the camera motion compensation process. Looking at Fig. 2.2 (left), BBs b_1 and b_2, lying on the moving object, also belong to two corresponding clusters of BBs, respectively in frame F_t and F_{t+1} (depicted with red and yellow). The density value of those BB pairs belonging to these two clusters, computed using Eq. 2.5, is roughly constant for all the possible pairs.

Conversely, matching b_1 with a background BB b_3, the corresponding density value is usually drastically different. In Fig. 2.2 (right-top) we plot the value of D, where the x-axis represents pairs of BBs sorted in ascending order with respect to D. Let $f()$ be the sorted distribution of D. Plateaus in $f()$ correspond to pairs of BBs belonging to clusters in F_t and F_{t+1}, and these clusters usually correspond to moving objects detected by Selective Search. We exploit this observation selecting θ_t as one of the steepest slopes in f. In Fig. 2.2 (right-bottom) we show the (smoothed) gradient of f, where peaks correspond to high variations in f before a plateau. We set θ_t to be the value of f corresponding to the median peak. Preliminary experiments with θ_t equal to the last peak (higher density) gave slightly lower results.

Using Eq.s (2.4)-(2.7) we can compute single frame matchings $M_t()$ for all the BBs in B_t, where $M_t(b_i)$ is not defined ($M_t(b_i) = \varnothing$) when there is no $b_j \in B_{t+1}$ such that (b_i, b_j) satisfies both constraints in Eq.s (2.4) and (2.7). We can then concatenate BBs in different frames forming a set of *chains* $CH = \{ch_1, ch_2, ...\}$, where a chain ch is computed starting from a given BB b_0 in frame t ($b_o \in B_t$) and:

$$ch = (b_0, b_1, ..., b_i, b_{i+1}, ..., b_{n_c}), \tag{2.8}$$

where:

$$b_{i+1} = M_{t+i}(b_i), \tag{2.9}$$

$$M_{t+n_c}(b_{n_c}) = \varnothing. \tag{2.10}$$

Chains are, on average, quite short ($E(n_c) \approx 6$ in our experiments). For this reason we further merge chains in *optical flow tubes*. We deal with the elements in CH as nodes in a graph, where an edge between two chains $ch_1, ch_2 \in CH$ is added when there is at least one frame in common between ch_1 and ch_2 such that the corresponding BBs in the two chains, $b_1 \in ch_1$ and $b_2 \in ch_2$, satisfy: $IoM(b_1, b_2) \geq 0.5$. Using IoM for measuring overlapping (instead of IoU) has the advantage that small BBs lying on subparts of the object of interest are clustered (e.g., (49, 52)) Hence, connected components of this graph correspond to chains with a sufficient spatial overlap in at least one frame. We compute an optical flow tube (ot) for each of these connected components:

$$ot = (r_0, r_1, ..., r_{n_o}), \tag{2.11}$$

where each $r_i \in ot$ is obtained by simply averaging the coordinates of those BBs $b_1, b_2, ...$ corresponding to the same frame F_t (i.e., $b_1, b_2, ... \in B_t$) and respectively belonging to the merged chains $ch_1, ch_2, ...$ (i.e., $b_1 \in ch_1$, $b_2 \in ch_2$, etc. ...).

The final optical flow tubes are relatively accurate. Still they only rely on two elements: the initial set of BBs provided by Selective Search and the matching pipeline described in this section, which is purely based on optical flow information. What is missing is a statistical model of the appearance of the tracked BBs, which can improve the result. We show in the next section how this model is computed.

2.2.5 Transductive learning

Let $OT = \{ot_1, ot_2, ...\}$ be the set of optical flow tubes computed as described in the previous section. For every $ot \in OT$ we build a specialized classifier. We extract positive samples from the BBs in ot and negative samples from other BBs in the video frames in which ot is defined and we train a linear SVM. The classifier obtained is then run on the whole video to obtain a new tube, that we call a *detection tube*.

This is a special case of *transductive learning*, since the training samples are extracted, in an *unsupervised* manner, from the same video in which the classifier is tested. In other words, the aim of each classifier is to model the appearance of a tube and then use this model to refine the tube. We do not need that the classifier is able to generalize to other videos because it is only used for our tube extraction task.

The idea we propose is similar to *tracking by detection* approaches, and it is exploited, for instance, in (53). The main difference of our approach with respect to (53) and other tracking by detection approaches is that our method is completely unsupervised, while in (53) a few positive BBs on the initial video frames need to be provided.

In more detail, given an optical flow tube $ot = (r_0, r_1, ..., r_{n_o})$, we include all of its BBs in the positive set P. Moreover, if $(F_{t_0}, ..., F_{t_{n_o}})$ is the sequence of frames in which ot is defined, we also include in P all those BBs which sufficiently overlap with one of the rectangles $r \in ot$ in one of these frames, using the IoU criterion in Eq. (2.4). The negative set starts with an initial set N_0 which is built including BBs b randomly extracted in the first frame F_{t_0} and such that $IoU(b, r_0) \leq 0.3$. The threshold 0.3 is widely adopted in the object detection literature for collecting negatives (e.g., see (51)). The negative set is iteratively pruned of the "easy negatives" and increased including new "hard negatives" by iteratively testing the current detector on the other frames while learning, following the well known hard negative

mining approach proposed in (50). Specifically, in a given frame $F_t \in (F_{t_0}, ..., F_{t_{n_o}})$, given P (which never changes) and N_t, we train a classifier $\mathbf{c}_t = [\mathbf{w}_t, a_t]$ by minimizing:

$$\mathbf{w}_t, a_t = \arg\min_{\mathbf{w},a} \sum_{r \in P} max(0, 1 - \mathbf{w}\phi(r) - a) + \sum_{r \in N_t} max(0, 1 + \mathbf{w}\phi(r) + a) + \lambda ||\mathbf{w}||_2^2, \tag{2.12}$$

where $\phi(r)$ is a feature representing the BB r. Different kinds of features can be used. For instance, HOG features are quite fast to be extracted from a rectangular patch of an image. In our experiments we used CNN features: $\phi(r)$ is the 4096-dimensional feature vector extracted from the last fully-connected layer (FC_7) of the ImageNet trained net described in (54). Note that we do *not* perform fine tuning of the net's parameters. In principle we could use all the sets of positives P, extracted using all the optical flow tubes, in order to fine-tune the network before extracting our features. However, since the number of these tubes is small (on average, about 3 per video) and they are short, fine-tuning a network with millions of parameters (54) would probably lead to overfitting phenomena. Hence, we just use the net as a feature extractor, relying on the widely proven high discriminative skills of these features (55). Following (51) we also add some padding around each r to include context. Finally, the value of λ, which controls the influence of the regularization term, is chosen according to (51): $\lambda = 10^{-4}$ and the feature values are normalized as suggested in (51). Following a consolidated object detection pipeline and adopting the parameters suggested in (50, 51) allows us to avoid the necessity of tuning the parameters of our classifiers. We believe that this is of primary importance for the success of an unsupervised method because it does not force one to collect data to tune the parameters when the method is applied to a new domain.

Once trained, $\mathbf{c}_t$ is tested on the BBs of the subsequent frames in which ot is defined, new hard negatives are added and training is repeated (Eq. (2.12)). We refer to (50) for details on the hard negative mining procedure. The final classifier is given by the parameters computed in the last frame of the tube: $\mathbf{c} = \mathbf{c}_{t_{n_o}}$.

2.2.5.1 Detection Tubes

Once collected a set of classifiers $C = \{\mathbf{c}^1, ..., \mathbf{c}^k\}$ from a given video, the final part of our pipeline concerns the extraction of detection tubes using these classifiers. Given a frame F_t

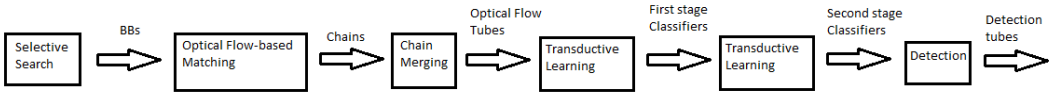


Figure 2.3: Flow chart of the proposed approach.

and a classifier $\mathbf{c}^i = [\mathbf{w}^i, a^i] \in C$, the highest scoring detection BB d_t^i of $\mathbf{c}^i$ in F_t is obtained maximizing:

$$d_t^i = \arg\max_{b \in B_t} \mathbf{w}^i \phi(b) + a^i. \tag{2.13}$$

Note that we use all the BBs in B_t when "testing" the classifier. We then build a detection tube dt_i for each classifier $\mathbf{c}^i$ linking d_t^i over all the T frames of the video:

$$dt_i = (d_1^i, ..., d_t^i, ...d_T^i). \tag{2.14}$$

In this way we collect a set k detection tubes, one per classifier. Note that the cardinality of C, k, is *not* fixed a priori, and it depends on the number of optical flow tubes constructed in the previous phase (see Sec. 2.2.4). In our experiments, k is usually very small ($E(k) \approx 3$).

When many tubes are desired (e.g., to increase recall), we repeat training. More specifically, we *split* a detection tube dt in dt^1, dt^2 using the criteria of the first stage (Sec. 2.2.4). Given two consecutive detections d_t and d_{t+1} in dt, we split dt in $dt^1 = (d_1, ..., d_t)$ and $dt^2 = (d_{t+1}, ..., d_T)$ when: $IoU(d_t, d_{t+1}) < 0.5$ or $D(d_t, d_{t+1}) \geq \theta_t$, where, with a slight abuse of notation, $D(d_t, d_{t+1})$ is the match density defined in Eq. (2.5) and θ_t the adaptive threshold pre-computed in the optical flow tube construction phase. After splitting the optical flow tubes, we use each tube to train a second set of classifiers C' repeating the procedure described in Sec. 2.2.5.

The final set of tubes for a given video is the set of the detection tubes obtained using all the detectors in C and C'. For a given video v, let $DT_v = \{dt_1, dt_2, ...\}$ be the set of all the detection tubes obtained using all the classifiers in C and C'. In Fig 2.3 we show the flow chart of the whole procedure.

2.2.6 Experiments

We evaluate our method using two common benchmarks and evaluation metrics for tube-proposal algorithms and we compare with the state-of-the-art approaches in this field.

2.2.6.1 Experimental Setup

Datasets We use for evaluation the UCF Sports dataset (56) and the YouTube Objects dataset (39). UCF Sports is composed of 150 videos of 10 sports (e.g., diving, running, golf, kicking, etc.). For evaluation we used the ground truth annotation provided in (44). Moreover, in order to allow a comparison with the results reported in (44), we adopted the same train/test split proposed in that article, where 100 videos are used for testing[2]. Note that in (44) the train split is used to train the proposed *supervised* method, while in our *unsupervised* approach we only used this "train" subset of 50 videos in the development stage to do all our design choices. We also do not have dataset-dependent parameters which need to be set (since all our parameter values are set using a consolidated object detection pipeline, see Sec.s 2.2.4-2.2.5), thus there is no training or parameter tuning phase in our approach, which makes the comparison with other supervised methods such as (44) disadvantageous for us since we do not exploit any dataset-specific information.

YouTube Objects is a large dataset composed of 1400 short shots obtained from videos collected on YouTube. As in the case of UCF Sports, many videos have large camera movement, illumination changes and cluttered backgrounds. However, the moving objects in this dataset usually occupy a larger portion of the frame, thus they are easier to detect. Differently from UCF Sports, in YouTube Objects there is only one annotated frame per shot but some frames are annotated with multiple objects. The dataset is split in a "train' and a "test" subset. We used the "test" shots to test our system (346 shots). Note that the "train" shots are usually easier, thus testing on the whole dataset would probably get higher accuracy results.

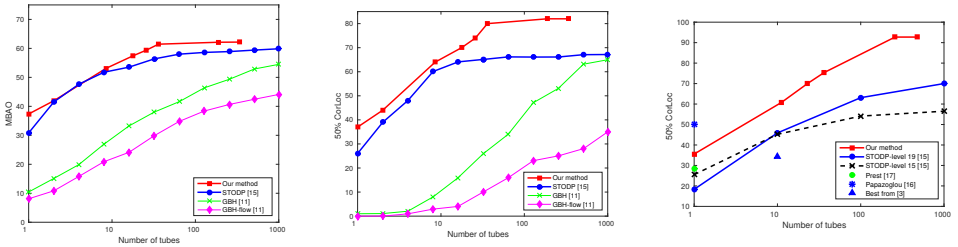
Metrics Following (44) we use two metrics: mBAO and CorLoc. Both metrics are based on the Best Average Overlap (BAO) of a set of tube proposals with ground truth objects. In UCF Sports dataset there is only one moving object annotated per video (but the dataset contains some videos with more than one moving object, being only the predominant object provided of ground truth annotations). Using this assumption, for a given video v and a set DT_v of tube proposals for v, BAO is defined as follows (44):

$$BAO(v) = \max_{dt \in DT_v} \frac{1}{|T_v|} \sum_{t \in T_v} IoU(d_t, g_t), \tag{2.15}$$

[2]The train/test split of the dataset and the annotations are provided at: *http://lear.inrialpes.fr/ËIJoneata/3Dproposals*

where $|T_v|$ is the set of frames of video v with ground truth annotation, d_t is the BB in tube proposal dt at frame t, and g_t is the ground truth at frame t. Note that in case of multiple annotated objects per video (YouTube Objects dataset), Eq. (2.15) is applied separately to each object using the same set of proposals T_v (44). mBAO is the mean BAO across all the videos, while CorLoc is the fraction of videos for which the BAO is greater or equal to 0.5.

2.2.6.2 Comparison with State of the Art



(a) mBAO computed over the UCF Sport dataset.

(b) CorLoc computed over the UCF Sport dataset.

(c) CorLoc computed over the YouTube Objects dataset.

Figure 2.4: Quantitative results on UCF Sports and Youtube datasets

UCF Sports. In Fig.s 2.4a-2.4b we show the experimental results obtained on the "test" part of UCF Sports dataset (100 videos). The methods we compare to are: (1) the Spatio-Temporal Object Detection Proposals (STODP) proposed in (44), (2) The Graph-Based Hierarchical segmentation proposed in (11) and its variant (2) GBH-Flow presented in (44). All the plotted results, except ours, have been obtained from (44).

Fig. 2.4a shows the mBAO plotted with respect to the number of average tube proposals per video and, similarly, Fig. 2.4b shows the CorLoc-based evaluation. In case of one tube per video, we obtain 0.374 mBAO and 0.37 CorLoc versus 0.3 and less than 0.3, respectively, of the state-of-the-art system on UCF Sports (44), with a relative CorLoc improvement of more than 30%. Once more we highlight that (44) is a supervised method, trained on the "train" split of UCF Sports, hence, most likely positively affected by a dataset bias, while our method is completely unsupervised. We achieve the highest reported value of CorLoc (0.82) on UCF Sport with 188 tube proposals (quite close to 0.8, obtained with only 36 tubes). Moreover, our system achieves 0.7 CorLoc with only 18 tubes: a value of recall which is not achieved by the other methods even when using 1000 proposals.

YouTube Objects. Fig. 2.4c shows the results obtained with the YouTube Objects dataset. In this case we compare with two different parameter settings of STODP (we refer the reader to (44) for details), the unsupervised method proposed by Papazoglou et al. (41), the weakly supervised method of Prest et al. (39) and the result for the best tube among the proposals of the unsupervised method proposed by Brox and Malik (40). All the plotted results, except ours, have been obtained from (44) (mBAO is not provided by the other authors).

As Fig. 2.4c clearly shows, we outperform all the competitors, both the supervised and the unsupervised methods. The only approach which achieves a CorLoc value better than our system is (41), which only outputs a single proposal per shot. However, we obtain a CorLoc higher than (41) with only 4 proposals. Compared with Oneata et al. (44), which obtained 0.461 when using 10 proposals, with the same number of tubes we obtain a CorLoc of 0.596, a relative improvement of 29%. Our largest value of CorLoc on this dataset is 0.927, obtained with 258 tubes, a recall much higher than any other published result.

2.2.6.3 Qualitative Results

Figure 2.5: Some examples of detection tubes. In each row we show a tube taken from a different video. 1-st and 2-nd row: UCF Sports dataset, 3-rd and 4-th row: YouTube Objects dataset. Red rectangles are BBs of the tube, while blue rectangles are ground truth annotations. Note that in YouTube Objects, only one frame is provided with ground truth (3-rd column).

In Fig. 2.5 we show some example results of our detection tubes using UCF Sports and

Figure 2.6: Some examples of errors of the proposed method. 1-st and 2-nd row: UCF Sports dataset, 3-rd row: YouTube Objects dataset.

Youtube Objects images. Most of the times our system is able to accurately detect the moving object even when it stops for a while (e.g., the dog, which is still with respect to the background, despite there is camera movement), unlike most of the state-of-the-art methods which require movement in all the frames.

In Fig. 2.6 we show some incorrectly detected tubes. In the middle row the misalignment between the ground truth and the detections is probably due to the difference in speed of the upper part and the lower part of the person, which produced detectors only for the fastest part (the upper body). In the first row our system is actually able to accurately track most of the moving persons but, unfortunately, the UCF Sport dataset contains annotation for only one object (person) per video, penalizing the extraction of multiple-objects.

2.2.7 Conclusions

As a first step, we proposed a method for the extraction of tubes from videos based on a first pipeline in which optical flow obtained with Dense Trajectories is used for matching BBs and a second pipeline in which the initial tubes are used to collect positive training samples for training tube-specific detectors. The final tubes are given by the detections of the trained classifiers, used in a transductive framework. The method was evaluated on UCF Sports and YouTube Objects, showing state-of-the-art results.

Our approach is completely unsupervised and all the critical parameters and thresholds have been set by adopting the values commonly used in a consolidated object detection pipeline

(49, 50, 51, 52) which makes the final system independent of specific datasets. Other important characteristics of the proposed technique are the possibility to detect the object in frames in which there is no movement (thanks to the detection-based approach) and the fact that we do not need to assume that only one moving object is present in a video clip. The major limitations of the system presented in this section are a low temporal resolution, restricting the use to more temporally constrained action videos, and the possibility of ignoring background objects.

2.3 Unsupervised Video Segmentation [2]

Video segmentation can be defined as partitioning a video into several disjoint spatio-temporal regions such that each region has consistent appearance and motion. In contrast to the method addressed in Section 2.2, the commonly accepted definition and benchmarks for this task assume that the segmentation is performed on a pixel level, with the high spatio-temporal segmentation accuracy that it implies. A high performance pixel-level spatio-temporal system mitigates the issues presented in the previous section.

2.3.1 Introduction

Among the existing video segmentation techniques, many successful ones benefit from mapping the video elements onto a graph which pixels/superpixels are nodes and edge weights measure the similarity between nodes. Cutting or merging is then applied on this graph to generate the video segments. Most of the existing graph-based methods focus on (i) what features to extract from each node; (ii) how to define a precise similarity graph and (iii) how to cut/merge the nodes effectively.

Meaningful features are necessary for good video segmentation. Previous work has extracted a variety of features (23, 26) from superpixels. To get the similarity graph, a graph topology is firstly designed according to the spatio-temporal neighborhood of the superpixels and the extracted features are used to weigh their edges. While standard similarity measures on the extracted features provide the basic way to calculate the similarity graph (25, 26), more recent work introduces learning a more precise similarity graph in either a supervised (27) or an unsupervised manner (57). While supervised video segmentation methods (23, 27) can generally achieve better performance, the human annotation is time-consuming and the inherent video object hierarchy may be highly subjective. In contrast, a group of methods improve on cutting techniques (24, 25, 28, 29), which explicitly organize the image elements into mathematically sound structures based on the optimization of the predefined cutting loss function. One representative criterion is the normalized cut (24). By minimizing a cutting cost objective function, the best segmentation can be obtained. This objective function is further proved to be equivalent to the generalized eigenvalue decomposition problem and a number of follow-ups

[2] "Joint Graph Learning and Video Segmentation via Multiple Cues and Topology Calibration" Jingkuan Song, Lianli Gao, Mihai Marian Puscas, Feiping Nie, Fumin Shen, Nicu Sebe; MM '16 Proceedings of the 24th ACM international conference on Multimedia Pages 831-840 (2)

proposed efficient solutions for this problem (58). To reduce the computational cost, in (28, 29), fast partitioning methods that identify and remove between-cluster edges to form node clusters are proposed.

Graph cut methods provide well-defined relationships between the segments, but the problem of finding a cut in an arbitrary graph may be NP-hard. More importantly, because the graph similarity learning (59, 60, 61, 62, 63, 64) and the graph cutting are conducted in two separated steps, the learned graph similarity matrix may not be the optimal one for cutting, leading to suboptimal results. To tackle this problem, in this paper we propose a novel video segmentation framework: *Joint Graph Learning and Video Segmentation* (JGLVS), which learns the similarity graph and segmentations simultaneously. To summarize, the main contributions of this paper are:

- Our unsupervised video segmentation framework learns the similarity graph and cutting structure simultaneously to achieve the optimal segmentation results. We derive a novel and efficient algorithm to solve this challenging problem.
- We utilized multiple cues of the superpixels and the weights of different cues are automatically learned. Furthermore, we calibrate the similarity of different superpixels based on their topology structures to make them comparable.
- The proposed JGLVS achieves up to 11% improvement over the state-of-the-art baselines on the largest public dataset VSB100, which validates the effectiveness and efficiency of our approach.

The remainder of this work is organized as follows. Section 2.3.2 discusses some related works. The details of JGLVS are introduced in section 2.3.3. Section 2.3.5 illustrates the experiments results and we draw a conclusion in section 2.3.6.

2.3.2 Related Work

The relevant state-of-the-art methods on video segmentation are reviewed in this section. The problem definitions for video segmentation have been diverse.

Motion segmentation focuses on separating point trajectories from an image sequence with respect to their motion (65, 66, 67, 68). In (67, 68), the segmentation is based on pairwise affinities, while in (69) third order terms are employed to explain not only translational motion but also in-plane rotation and scaling, and (70) models even more general 3D motions

using group invariants. The actual grouping in these methods is done using spectral clustering. Differently, in (66), they formulate the segmentation of a video sequence based on point trajectories as a minimum cost multicut problem. Unlike the commonly used spectral clustering formulation, the minimum cost multicut formulation gives natural rise to optimize not only for a cluster assignment but also for the number of clusters while allowing for varying cluster sizes. Similarly, in (71), they utilize improved point trajectories to segment moving object in video by a graph-based segmentation method. And in (72), motion trajectory grouping in a setup similar to (68) is used to perform tracking. Although the grouping in (72) is computed using spectral clustering, repulsive weights computed from segmentation topology are used in the affinity matrix. In (65), they introduced minimal supervision, which is shown to be helpful to improve the performance of motion segmentations. In (73), they propose a framework to segment the objects in relative video shots, while discarding the irrelative video shots.

On the other hand, (26, 28, 29, 57) seek to construct full pixelwise segmentation, where every pixel (not only the moving objects) is assigned one of several labels. They can generally be divided into unsupervised and supervised methods.

A large body of literature exists on unsupervised video segmentation, with methods that leverage appearance (24, 30, 31, 74), motion (30, 75), or multiple cues (26, 28, 29, 57). Unsupervised supervoxel generation (26, 76) has been widely accepted as a valuable preprocessing step for various techniques, such as graph-based methods (24, 26, 28, 29), hierarchical methods (24, 74, 77) and streaming methods (28, 57, 74). Graph-based methods map the video elements onto a graph in which pixels/superpixels are nodes, and edge weights measure the similarity between them. Galasso *et al.* (26) proposed a frame-based superpixel segmentation approach (VSS) by extending the ultra-metric contour map (78) to combine with motion-cues and appearance-based affinities for obtaining better video segmentation performance. To deal with the high computational costs of spectral techniques, Galasso *et al.* (28) proposed a spectral graph reduction (SGR) method for video segmentation. They assumed that all pixels within a superpixel are connected by must-link constraints, and then reduced the original graph to a relative small graph such that a density-normalized-cut was preserved. Yu *et al.* (29) proposed an efficient and robust video segmentation framework based on parametric graph partitioning, resulting in a fast and almost parameter free method. On the other hand, hierarchical video segmentation provides a rich multi-scale decomposition of a given video. Grundmann *et al.* (24) proposed a hierarchical graph-based (HGB) video segmentation approach by firstly over-segmenting a volumetric video graph into space-time regions grouped by appearance, and then

constructing a "region graph" over the obtained segmentation. Iteratively repeating this process over multiple levels results in a a tree of spatio-temporal segmentations. In order to process long videos, Xu *et al.* (74) proposed a streaming hierarchical video segmentation framework by integrating a graph-based hierarchical segmentation method with a data streaming algorithm (SHGB). This method leveraged ideas from data streams and enforced a Markovian assumption on the video stream to approximate full video segmentation. Li *et al.* (57) proposed a Sub-Optimal Low-rank Decomposition (SOLD) method, which defines a low-rank model based on very generic assumption that the intra-class supervoxels are drawn from one identical low rank feature subspace, and all supervoxels in a period lie on a union of multiple subspaces, which can be justified by natural statistic and observations of videos. In addition, this method adopts the Normalized-Cut (NCut) algorithm with a solved low-rank representation to segment a video into several spatio-temporal regions. To tackle the lack of a common dataset with sufficient annotation and the lack of an evaluation metric, a united video segmentation benchmark was provided by Galasso *et al.* (79) to effectively evaluate the over- and under-segmentation performance of video segmentation methods.

Supervised video segmentations (27, 80) can achieve better performance, but the human annotation is time-consuming and the inherent video object hierarchy may be highly subjective. In (80), they address the problem of integrating object reasoning with supervoxel labeling in multiclass semantic video segmentation. They first propose an object augmented dense CRF in spatio-temporal domain, which captures long-range dependency between supervoxels, and imposes consistency between object and supervoxel labels. Then, they develop an efficient mean field inference algorithm to jointly infer the supervoxel labels, object activations and their occlusion relations for a moderate number of object hypotheses. While in (27), they propose to combine features by means of a classifier, use calibrated classifier outputs as edge weights and define the graph topology by edge selection. Learning the topology provides larger performance gains and benefits efficiency due to a sparser structure of the constructed graph. On the other hand, lots of supervised image segmentations have been proposed (81). In (82), they propose a novel discriminative deep feature learning framework based on stacked autoencoders (SAE) to tackle the problem of weakly supervised semantic segmentation. In (81), they use CNN to train images most only with image-level labels and very few with pixel-level labels for semantic segmentation.

Unsupervised full pixelwise segmentation is the research focus of this paper. A substantial difference between our approach and previous unsupervised work is that, instead of separately

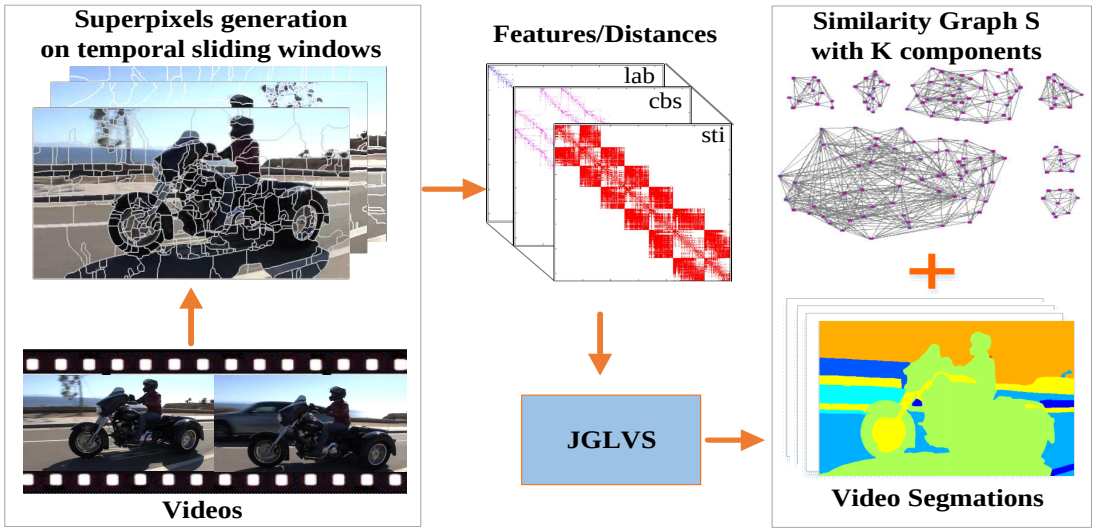


Figure 2.7: The overview of JGLVS. Superpixels are firstly generated from the overlapping sliding windows, based on which the features and distances are computed. Then, JGLVS is applied to learn the similarity matrix and video segmentations.

obtaining a graph and finding a cut in it, we propose a joint graph learning and video segmentation method by assigning adaptive neighbors for each superpixel and imposing a rank constraint on the Laplacian matrix of the similarity graph, such that the learned graph has exactly K connected components, representing K segmentations.

2.3.3 Our Approach

In this section, we first introduce our JGLVS framework, and then elaborate on the details of each component.

2.3.3.1 The framework

In our JGLVS framework (see Fig. 2.7), we propose a novel perspective in solving the graph-based video segmentation problem. Our model makes use of superpixels instead of pixels for two reasons: a great decrease in the number of graph nodes that need to be processed, and an initial, accurate frame-level segmentation.

Firstly, in each temporal sliding window of the video, we extract N superpixels from M successive frames by setting a specific hierarchical level of an image segmentation algorithm (76). Note that a too small value of N leads to large superpixels, and more under-

Topology type	Distances
Within frame	lab, sof, cbs, bof
Across 1 frame	lab, sof, ssd, sti
Across 2 frames	ssd, sti
Across $>$ 2 frames	sti

Table 2.1: The corresponding distances for different topological structures

segmentation errors, while a large value of N is computationally expensive. Then, for each superpixle, a set of features (e.g., appearance, motion and shape features) are extracted. Using these features and the predefined topology structure, our JGLVS framework can learn a similarity graph of superpixels which has exactly K connected components.

2.3.3.2 Feature extraction and graph topology construction

For each superpixel, we follow (26, 76) to extract *LAB*, *boundary*, *motion* and *shape* features, and use them to calculate the distance between two superpixels. However, not all of the superpixels are connected. By allowing different edge connections between neighbors, different graph topologies are constructed. Following (24, 27), edges may connect neighbors: *within frame* (if two superpixels share a common part of their contour or are close by in the spatial domain of the frame); *across 1 frame* (connected by coordinate correspondences over time); *across 2 frames* (connected by across-1 correspondences, further propagated over one more frame) and *across $>$ 2 frames* (linked if overlapping with common long-term point trajectories).

We refer to these four types of neighbours as different topological structures $(1, 2, 3, 4)$ and record the topological structure of each pair of superpixels in a $N \times N$ matrix $\mathbf{W}$. Based on these features and topological structures, we can have the following pairwise distances between superpixels: common boundary strength (*cbs*), LAB (*lab*), boundary optical flow (*bof*), superpixel optical flow (*sof*), superpixel shape distance (*ssd*) and superpixel trajectory intersection (*sti*) (See Section 2.3.5 for details).

As shown in Table 2.1, different topological types have different distances. We further define a set of most-likely-linked superpixels $\mathbb{M}^1, \mathbb{M}^2, \mathbb{M}^3$ and $\mathbb{M}^4$ for each topological structure. More specifically, for the case of within frame, we decrease the number of superpixels by changing the threshold of superpixel generation algorithm, and some similar superpixels will

merge into one superpixel. These similar superpixels will be selected as the within frame most-likely-linked superpixels. For the case of across 1 or 2 frame, if two superpixels' *ssd* distance is less than a threshold, they will be selected as a pair of across 1 or 2 frame most-likely-linked superpixels. Similarly, if two superpixels' *sti* distance is less than a threshold in the case of across > 2 frame, they will be selected as a pair of across > 2 frame most-likely-linked superpixels.

2.3.3.3 Joint graph learning and video segmentation

Let $\mathbf{D}^t = \{\mathbf{D}_{ij}^t\}_{i,j=1}^N$ denote the t-th distance matrix of a set of N superpixels, where $t \in \{1, ..., T\}$. $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N\}$ is the average location information for the superpixels. The goal is to learn the similarity matrix $\mathbf{S}$ between superpixels by using different distances as well as existent spatial information, and that all the superpixels have exact K connected components.

An optimal graph $\mathbf{S}$ should be smooth on different features as well as on the spatial information distribution, which can be formulated as:

$$\min_{\mathbf{S},\alpha} \mathbf{g}(\mathbf{Y}, \mathbf{S}) + \mu \sum\nolimits_{t=1}^{T} \alpha^t \mathbf{h}\left(\mathbf{D}^t, \mathbf{S}\right) + \beta \mathbf{r}(\mathbf{S}, \alpha) \tag{2.16}$$

where $\mathbf{g}(\mathbf{Y}, \mathbf{S})$ is the penalty function that measures the smoothness of $\mathbf{S}$ on the spatial information $\mathbf{Y}$ and $\mathbf{h}\left(\mathbf{D}^t, \mathbf{S}\right)$ is the loss function that measures the smoothness of $\mathbf{S}$ on the feature $\mathbf{D}^t$. $\mathbf{r}(\mathbf{S}, \mathbf{ff})$ is a regularizer defined on the target $\mathbf{S}$ and α. μ and β are balancing parameters, and $\mathbf{ff}^t$ determines the importance of each feature.

The penalty function $\mathbf{g}(\mathbf{F}, \mathbf{S})$ should be defined in a way such that close superpixels have high similarity and vice versa. In this paper, we define it as follows:

$$\mathbf{g}(\mathbf{Y}, \mathbf{S}) = \sum\nolimits_{ij} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2^2 s_{ij} \tag{2.17}$$

where $\mathbf{y}_i$ and $\mathbf{y}_j$ are the locations of the superpixels $\mathbf{x}_i$ and $\mathbf{x}_j$. Similarly, $\mathbf{h}\left(\mathbf{D}^t, \mathbf{S}\right)$ is defined as:

$$\mathbf{h}\left(\mathbf{D}^t, \mathbf{S}\right) = \sum\nolimits_{ij} d_{ij}^t s_{ij} \tag{2.18}$$

The regularizer term $\mathbf{r}(\mathbf{S}, \alpha)$ is defined as:

$$\mathbf{r}(\mathbf{S}, \alpha) = \|\mathbf{S}\|_F^2 + \gamma \|\alpha\|_2^2 \tag{2.19}$$

If there is no regularizer on $\mathbf{S}$ (same for α), $\mathbf{S}$ has a trivial solution. Only the nearest data point can be the neighbor of $\mathbf{x}_i$ with the probability of 1. We further introduce the following

constraints: $\mathbf{S} \geq 0$, $\mathbf{S1} = \mathbf{1}$, $\alpha \geq 0$ and $\alpha^T \mathbf{1} = 1$, where $\mathbf{1}$ is a column vector with all 1s. This is because that the similarity and weights should be positive, and the sum of similarity and weights is set to be 1.

We can then obtain the objective function for learning the optimal graph by replacing $\mathbf{g}(\mathbf{Y}, \mathbf{S})$, $\mathbf{h}\left(\mathbf{D}^t, \mathbf{S}\right)$ and $\mathbf{r}(\mathbf{S}, \alpha)$ in (2.16) using (2.17), (2.18) and (2.19), as follows:

$$\begin{aligned} &\min_{\mathbf{S},\alpha} \sum_{ij} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2^2 s_{ij} + \mu \sum_{tij} \left(\alpha^t d_{ij}^t s_{ij}\right) \\ &+\beta \left\|\mathbf{S}\right\|_F^2 + \beta\gamma \left\|\alpha\right\|_2^2 \\ &s.t.,\ \mathbf{S} \geq 0, \mathbf{S1} = \mathbf{1}, \alpha \geq 0, \alpha^T \mathbf{1} = 1 \end{aligned} \tag{2.20}$$

One limitation for this model is that it assumes that all the superpixels have the same types of distances, which conflicts with the video segmentation application where different topologies have different distances. For example, if superpixels (i, k) are across > 2 frames neighbors and (i, j) are within frame neighbors, the similarity between (i, k) are determined by *sti* but the similarity between (i, j) are determined by *lab, sof, cbs* and *bof*. Their distances are not comparable to each other, and we need to calibrate them. Based on the topology type $w_{ij} \in [1, 2, 3, 4]$ of superpixels i and j, we define a calibration function

$$\mathbf{c}^z(x) = (x - \tau^z)/(max^z - \tau^z), z \in [1, 2, 3, 4], \tag{2.21}$$

where τ^z is the threshold for z-th topology type determined by the mean distance of the set $\mathbb{M}^z$. Then, the objective function becomes:

$$\begin{aligned} &\min_{\mathbf{S},\alpha} \sum_{ij} \left\|y_i - y_j\right\|_2^2 s_{ij} + \mu \sum_{ij} \mathbf{c}^{w_{ij}} \left(\sum_t \alpha^t d_{ij}^t\right) s_{ij} \\ &+\beta \left\|\mathbf{S}\right\|_F^2 + \beta\gamma \left\|\alpha\right\|_2^2 \\ &s.t.,\ \mathbf{S} \geq 0, \mathbf{S1} = \mathbf{1}, \alpha \geq 0, \alpha^T \mathbf{1} = 1 \end{aligned} \tag{2.22}$$

Forcing the number of connected components to be exactly K seems like an impossible goal since this kind of structured constraint on the similarities is fundamental but also very difficult to handle. In this paper, we will propose a novel but very simple method to achieve this goal.

The matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$ obtained in the neighbor assignment can be seen as a similarity matrix of the graph with the N data points as the nodes. For a nonnegative similarity matrix $\mathbf{S}$, there is a Laplacian matrix $\mathbf{L}$ associated with it. According to the definition of Laplacian matrix, for any values of $\mathbf{f}_i \in \mathbb{R}^{K \times 1}$, $\mathbf{L}$ of a similarity matrix $\mathbf{S}$ can be calculated as:

$$\sum\nolimits_{ij} \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_2^2 s_{ij} = 2tr\left(\mathbf{F}^T \mathbf{L} \mathbf{F}\right) \tag{2.23}$$

where $\mathbf{F} \in \mathbb{R}^{N\times K}$ with the i-th row formed by $\mathbf{f}_i$, $\mathbf{L} = \mathbf{D} - \frac{\mathbf{S}^T+\mathbf{S}}{2}$ is called the Laplacian matrix in graph theory, the degree matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ is defined as a diagonal matrix where the i-th diagonal element is $\sum_j (s_{ji} + s_{ij})/2$. The Laplacian matrix $\mathbf{L}$ has the following property.

Theorem 1 *The number K of the eigenvalue 0 of the Laplacian matrix $\boldsymbol{L}$ is equal to the number of connected components in the graph with the similarity matrix $\boldsymbol{S}$ if $\boldsymbol{S}$ is nonnegative.*

Theorem 1 indicates that if $rank(\mathbf{L}) = N - K$, then the superpixels have K connected components based on S. Motivated by Theorem 1, we add an additional constraint $rank(\mathbf{L}) = N - K$ into the (2.22). Thus, our new similarity graph learning model is to solve:

$$\begin{aligned} &\min_{\mathbf{S},\alpha} \sum_{ij} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2^2 s_{ij} + \mu \sum_{ij} \mathbf{c}^{w_{ij}} \left(\sum_t \alpha^t d_{ij}^t\right) s_{ij} \\ &+\beta \left\|\mathbf{S}\right\|_F^2 + \beta\gamma \left\|\alpha\right\|_2^2 \\ &s.t. \begin{cases} \mathbf{S} \geq 0, \mathbf{S1} = \mathbf{1}, \alpha \geq 0, \alpha^T\mathbf{1} = 1 \\ rank\left(\mathbf{L}\right) = N - K \end{cases} \end{aligned} \tag{2.24}$$

It is difficult to solve the problem (2.24). Because $\mathbf{L} = \mathbf{D} - (\mathbf{S}^T + \mathbf{S})/2$ and $\mathbf{D}$ also depends on $\mathbf{S}$, the constraint $rank(\mathbf{L}) = N - K$ is not easy to tackle. In the next subsection, we will propose a novel and efficient algorithm to solve this challenging problem.

2.3.4 Iterative optimization

Suppose e_i is the i-th smallest eigenvalue of $\mathbf{L}$, we know $e_i \geq 0$ since $\mathbf{L}$ is positive semi-definite. It can be seen that the problem (2.24) is equivalent to the following problem for a large enough value of ρ:

$$\begin{aligned} &\min_{\mathbf{S},\alpha} \sum_{ij} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2^2 s_{ij} + \mu \sum_{ij} \mathbf{c}^{w_{ij}} \left(\sum_t \alpha^t d_{ij}^t\right) s_{ij} \\ &+\beta \left\|\mathbf{S}\right\|_F^2 + \beta\gamma \left\|\alpha\right\|_2^2 + 2\rho \sum_{i=1}^{K} e_i \\ &s.t.,\ \mathbf{S} \geq 0, \mathbf{S1} = \mathbf{1}, \alpha \geq 0, \alpha^T\mathbf{1} = 1 \end{aligned} \tag{2.25}$$

When ρ is set to a large enough value [2], $\sum_i^K e_i$ will be imposed to be close to 0, which results in $rank\left(\mathbf{L}\right) = N - K$.

[2] In the real implementation, we initialize ρ with 1000, and increase ρ to $\rho \times 2$ if the current number of connected components is less than K, and decrease rho to ρ/2 if the current number of connected components is larger than K

According to the Ky Fan's Theorem (83), we have:

$$\sum_{i=1}^{K} e_i = \min_{\mathbf{F}\in\mathbb{R}^{N\times K},\mathbf{F}^T\mathbf{F}=\mathbf{I}} tr\left(\mathbf{F}^T\mathbf{L}\mathbf{F}\right) \tag{2.26}$$

Therefore, the problem (2.25) is further equivalent to the following problem:

$$\begin{aligned} &\min_{\mathbf{S},\mathbf{F},\alpha} \sum_{ij} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2^2 s_{ij} + \mu \sum_{ij} \mathbf{c}^{w_{ij}} \left(\sum_t \alpha^t d_{ij}^t\right) s_{ij} \\ &+\beta \left\|\mathbf{S}\right\|_F^2 + \beta\gamma \left\|\alpha\right\|_2^2 + 2\rho tr\left(\mathbf{F}^T\mathbf{L}\mathbf{F}\right) \\ &s.t., \begin{cases} \mathbf{S} \geq 0, \mathbf{S}\mathbf{1} = \mathbf{1}, \alpha \geq 0, \alpha^T\mathbf{1} = 1 \\ \mathbf{F} \in R^{N\times K}, \mathbf{F}^T\mathbf{F} = \mathbf{I} \end{cases} \end{aligned} \tag{2.27}$$

Compared with the original problem (2.24), (2.27) is much easier to solve. We propose an iterative method to minimize the above objective function (2.27).

Firstly, we initialize $\alpha^t = 1/T$ and then $\mathbf{S}$ by the optimal solution to the problem (2.22). Once these initial values are given, in each iteration, we first update $\mathbf{F}$ given $\mathbf{S}$ and α, and then update $\mathbf{S}$ and α by fixing the other parameters. These steps are described below:

Update F: By fixing $\mathbf{S}$ and α, the problem (2.27) is equivalent to optimizing the following objective function:

$$\min_{\mathbf{F}\in R^{N\times K},\mathbf{F}^T\mathbf{F}=\mathbf{I}} tr\left(\mathbf{F}^T\mathbf{L}\mathbf{F}\right) \tag{2.28}$$

The optimal solution $\mathbf{F}$ to the problem (2.28) is formed by the K eigenvectors of $\mathbf{L}$ corresponding to the K smallest eigenvalues.

Update S: By fixing $\mathbf{F}$ and α, we can obtain $\mathbf{S}$ by optimizing (2.27). It is equivalent to optimize the following objective function:

$$\begin{aligned} &\min_{\mathbf{S}\geq 0,\mathbf{S}\mathbf{1}=\mathbf{1}} \sum_{ij} \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2^2 s_{ij} + \rho \sum_{ij} \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_2^2 s_{ij} \\ &+\beta \left\|\mathbf{S}\right\|_F^2 + \mu \sum_{ij} \mathbf{c}^{w_{ij}} \left(\sum_t \alpha^t d_{ij}^t\right) s_{ij} \end{aligned} \tag{2.29}$$

It can be reformulated as:

$$\begin{aligned} &\min_{\mathbf{S}\geq 0,\mathbf{S}\mathbf{1}=\mathbf{1}} \sum_i \left(\beta \mathbf{s}_i \mathbf{s}_i^T + \left(\mathbf{a}_i + \mu\mathbf{b}_i + \rho\mathbf{c}_i\right)\mathbf{s}_i^T\right) \\ &\Rightarrow \min_{\mathbf{S}\geq 0,\mathbf{S}\mathbf{1}=\mathbf{1}} \sum_i \left(\mathbf{s}_i\mathbf{s}_i^T + \frac{\mathbf{a}_i+\mu\mathbf{b}_i+\rho\mathbf{c}_i}{\beta}\mathbf{s}_i^T\right) \end{aligned} \tag{2.30}$$

where $\mathbf{a}_i = \{a_{ij}, 1 \leq j \leq n\}$ with $a_{ij} = \left\|\mathbf{y}_i - \mathbf{y}_j\right\|_2^2$, $\mathbf{b}_i = \{b_{ij}, 1 \leq j \leq n\}$ with $b_{ij} = \sum_t \alpha^t d_{ij}^t$ and $\mathbf{c}_i = \{c_{ij}, 1 \leq j \leq n\} \in \mathbb{R}^{1\times n}$ with $c_{ij} = \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_2^2$. It is further equivalent to:

$$\min_{\mathbf{S}\geq 0,\mathbf{S1}=\mathbf{1}} \sum_i \left(\mathbf{s}_i + \frac{\mathbf{a}_i+\mu\mathbf{b}_i+\rho\mathbf{c}_i}{2\beta}\right)_2^2 \tag{2.31}$$

Then each $\mathbf{s}_i$ can be efficiently solved by using a quadratic programming solver, which will be introduced in the next subsection (solution for problem (2.31)).

Update α: By fixing $\mathbf{F}$ and $\mathbf{S}$, we can obtain α by optimizing (2.27). It is equivalent to optimize the following objective function:

$$\begin{aligned} &\min_{\alpha\geq 0,\alpha^T\mathbf{1}=1} \mu \sum_{ij} \mathbf{c}^{w_{ij}} \left(\sum_t \alpha^t d_{ij}^t\right) s_{ij} + \beta\gamma \left\|\alpha\right\|_2^2 \\ &\Rightarrow \min_{\alpha\geq 0,\alpha^T 1=1} \mu \sum_t \alpha^t \sum_{ij} d_{ij}^t s_{ij} / (\max^{w_{ij}} - \tau^{w_{ij}}) + \beta\gamma \left\|\alpha\right\|_2^2 \\ &\Rightarrow \min_{\alpha\geq 0,\alpha^T 1=1} \mu d\alpha + \beta\gamma \left\|\alpha\right\|_2^2 \end{aligned} \tag{2.32}$$

where $d = \left\{d^t\right\}_{t=1}^T$, $d^t = \sum_{ij} d_{ij}^t s_{ij} / (\max^{w_{ij}} - \tau^{w_{ij}})$ and $\max^{w_{ij}}$ is the max value of $\mathbf{S}$ with the topological structure w_{ij}. Then we can use a quadratic programming solver to obtain α.

We update $\mathbf{F}, \mathbf{S}$ and α iteratively until the objective function (2.22) converges, as shown in Algorithm 1.

Algorithm 1: Solution for JGLVS

Input: Initialized α, segmentation number K, topology structure matrix $\mathbf{W}$, most-likely-linked sets $\mathbb{M}$, parameters β, γ, μ, a large enough ρ;

Output: $\mathbf{S} \in \mathbb{R}^{N\times N}$ with exact K connected components, α;

1: Initialize $\mathbf{c}^z(x)$ using α, $\mathbf{W}$ and $\mathbb{M}$;
2: Initialize $\mathbf{S}$ by the optimal solution of 2.22;
3: **repeat**
4: Fix $\mathbf{S}$ and α, calculate $\mathbf{F}$ according to the solution of problem (2.28);
5: Fix $\mathbf{F}$ and α, update $\mathbf{S}$ by solving the problem (2.31);
6: Fix $\mathbf{F}$ and $\mathbf{S}$, update α by solving the problem (2.32);
7: Update $\mathbf{c}^z(x)$ using α, $\mathbf{W}$ and $\mathbb{M}$;
8: **until** convergence or max iteration is reached.
9: **return** $\mathbf{S}$, α;

Solution for problem (2.31) In this subsection, we introduce an efficient solution for problem (2.31) for determining the regularization parameter β, so that we have fewer parameters to

tune. The Lagrangian function of problem (2.31) is:

$$\begin{aligned}\ell\left(\mathbf{s}_i, \eta, \varepsilon_i\right) = \tfrac{1}{2}\sum_i \left\|\mathbf{s}_i + \tfrac{\mathbf{a}_i+\mu\mathbf{b}_i+\rho\mathbf{c}_i}{2\beta}\right\|_2^2 \\ -\eta(\mathbf{s}_i^T\mathbf{1}-1) - \mathbf{s}_i^T\varepsilon_i\end{aligned} \tag{2.33}$$

where η, $\varepsilon_i \geq 0$ are the Lagrangian multipliers, and β is the regularization parameter for each $\mathbf{s}_i$. Let d_{ij}=$a_{ij}+\mu b_{ij}+\rho c_{ij}$. According to the KKT condition, it can be verified that the optimal solution $\mathbf{s}_i$ should be:

$$s_{ij} = \left(-\frac{a_{ij}+\mu b_{ij}+\rho c_{ij}}{2\beta} + \eta\right)_+ \tag{2.34}$$

By replacing η and ε_i according to the KKT condition, we obtain the optimal $\mathbf{s}_i$. However, in practice, we usually could achieve better performance if $\mathbf{s}_i$ is sparse, i.e., only the P nearest neighbors of $\mathbf{x}_i$ could have chance to connect to $\mathbf{x}_i$. Another benefit of learning a sparse similarity matrix $\mathbf{S}$ is that the computational burden can be largely alleviated for subsequent processing. With this motivation, we determine the parameter β.

Without loss of generality, suppose $d_{i1}, d_{i2}, ..., d_{iN}$ are ordered from small to large. If the optimal $\mathbf{s}_i$ has only P nonzero elements, then according to (2.34), we know $s_{iP} > 0$ and $s_{i,P+1} = 0$. Therefore, we have:

$$-\tfrac{d_{iP}}{2\beta_P} + \eta > 0,\ -\tfrac{d_{i,P+1}}{2\beta_{P+1}} + \eta \leq 0 \tag{2.35}$$

and

$$\begin{aligned}\mathbf{s}_i^T 1 = \sum_{j=1}^{P}(-\tfrac{d_{ij}}{2\beta_i} + \eta) = 1 \\ \Rightarrow \eta = \tfrac{1}{P} + \tfrac{1}{2P\beta_i}\sum_{j=1}^{P} d_{ij}\end{aligned} \tag{2.36}$$

By replacing η in (2.35) using (2.36), we have the following inequality for β_i:

$$\frac{P}{2}d_{ip} - \frac{1}{2}\sum_{j=1}^{P} d_{ij} < \beta_i \leq \frac{P}{2}d_{i,P+1} - \frac{1}{2}\sum_{j=1}^{P} d_{ij} \tag{2.37}$$

Therefore, in order to obtain an optimal solution $\mathbf{s}_i$ to the problem (2.31) that has exact P nonzero values, we set

$$\beta_i = \frac{P}{2}d_{i,P+1} - \frac{1}{2}\sum\nolimits_{j=1}^{P} d_{ij} \tag{2.38}$$

The overall β is set to the mean of $\beta_1, \beta_2, ..., \beta_n$. That is, we set β to be

$$\beta = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{P}{2} d_{i,P+1} - \frac{1}{2} \sum_{j=1}^{P} d_{ij} \right) \tag{2.39}$$

The number of neighbors P is much easier to tune than the regularization parameter β since P is an integer and has explicit meaning.

2.3.4.1 Streaming video segmentation

An effective streaming algorithm can enable us to process an arbitrary long video with limited memory and computational resources, and thus is essential in video segmentation. We propose a simple yet effective clip-based segmentation method that scales well while maintaining temporal coherence, without processing the entire volume at once.

Figure 2.8: The segmentation labels L_1, L_2 of the overlapping frame f. L_1 denotes the segmentation of the overlapping in the previous clip, and provides some constraints to the segmentation of L_2, which is the segmentation in the current clip.

We start by partitioning the video into equally sized clips of n frames (n = 6 in our experiments), and one frame f is overlapped between neighboring clips. The temporal consistent constraints are introduced by properly propagating solutions from previous temporal window to the current window. Given the previous and current segmentation labels L_1, L_2 of the overlapping frame f, we first compute the similarity matrix O of different segments. The similarity of the i-th segment i_{L_1} in the previous segmentation and the j-th segment j_{L_2} in the current segmentation is defined as:

$$o(i_{L_1}, j_{L_2}) = |m_{i_{L_1}} \cap m_{j_{L_2}}| / min\left(|m_{i_{L_1}}|, |m_{j_{L_2}}|\right), \tag{2.40}$$

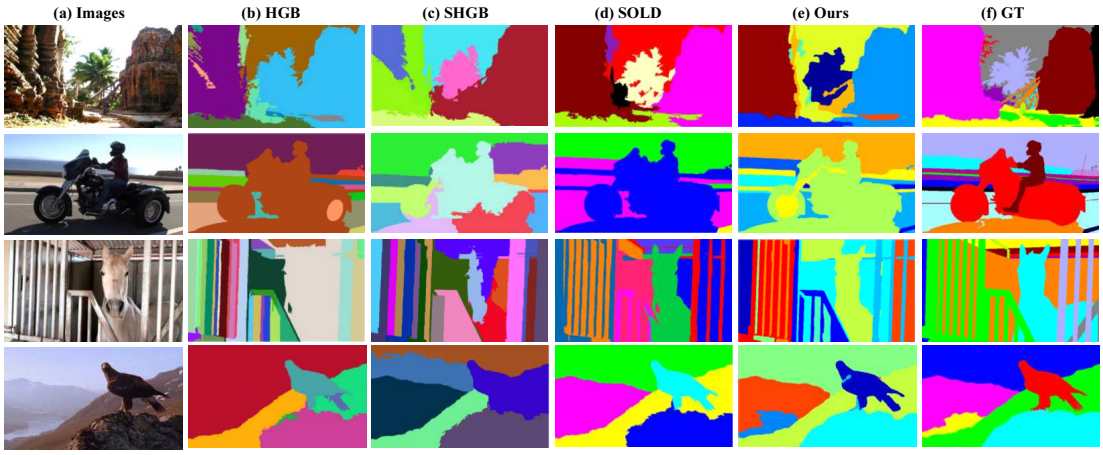


Figure 2.9: Qualitative comparisons with the state-of-the-art video segmentation methods HGB, SHGB and SOLD. We can see that our method substantially outperforms the algorithms of HGB, SHGB and SOLD.

where $m_{i_{L_1}}$ and $m_{j_{L_2}}$ are masks to indicate which pixels belong to the segments i_{L_1} and j_{L_2}. After obtaining the similarity matrix O, we can assign new segmentation ids to the segments in L_2. Intuitively, two segments with the highest similarity should have the same segment id. However, there are three special cases to consider, which are illustrated in Fig. 2.8. The first case is that a segment (e.g., 3_{L_2}) has no overlapping in the previous segment. Then a new segment id should be assigned to 3_{L_2}. The second case is that two segments (e.g., 1_{L_2} and 2_{L_2}) are included by one previous segment (1_{L_1}). Then the one (2_{L_2}) with larger size will be assigned the id (1) of overlapping segment (1_{L_1}), and the one (1_{L_2}) with smaller size will be assigned a new id. Lastly, if one segment (e.g., 4_{L_2}) includes two previous segments (2_{L_1} and 3_{L_1}), this segment will be assigned the id (2) of a larger segment (2_{L_1}). The algorithm for generating temporal consistent constraints is given in Algorithm 2. It takes previous and current segmentation labels L_1 and L_2, and similarity threshold $threshold$ as input. And it calculates which segment id in L_1 corresponds to each segment in L_2. Algorithm 2 outputs the refined segmentation labels L_2' as well as the segment id mapping $mapping$.

2.3.5 Experiments

In this section, we evaluate our JGLVS on the standard benchmark VSB100 (79). First, we compare our method with other state-of-the-art methods. Then, we further analyze the effectiveness of our main components. Finally, we report the efficiency of our method.

Algorithm 2: The algorithm for generation of the temporal consistent constraints between previous and current segmentation labels L_1, L_2 of the overlapping frame f.

Input: Previous and current segmentation labels L_1 and L_2, threshold $threshold$;
Output: Refined segmentation labels L_2', $mapping$;

```
Get number of segments num_1 and num_2 in L_1 and L_2;
Computer O by Eq.(2.40);
Assign mapping the ids with the largest similarity to L_2 by
  [value, mapping] = max(O);
for i = 1 : num_2 do
    if value(i) is smaller than threshold, then
        Updating mapping(i) with a new id;
    end if
    if mapping(i) is used by a previous segment, then
        Assign a new id to the segment with a smaller size;
    end if
    if Another segment in L_1 has the same similarity to segment i as mapping(i), then
        Updating mapping(i) with the id of a larger segment;
    end if
    Refine L_2 to L_2' based on mapping;
end for
return L_2', mapping;
```

2.3.5.1 Experimental Settings

We give the details of dataset selection, feature extraction and evaluation metrics in this subsection.

Dataset: The selected VSB100 (79) is a very challenging dataset used for empirical evaluation. It is the largest video segmentation dataset with high definition frames, and consists of four difficult sub-datasets: general, motion segmentation, non-rigid motion segmentation and camera motion segmentation. Following the setting in (57, 79), we regard the general sub-dataset (60 video sequences) as our test set for all the approaches.

To make the comparison comprehensive, we set $\{\mu, \gamma\} = \{1000, 1\}$, $\{\rho\} = \{1000\}$ and $\{iteration\} = \{30\}$ in the experiment. β is automatically determined by the algorithm. In addition, the number of frames per window is set to be 6, and 1 frame is overlapped between

	BPR			VPR		
Algorithm	**ODS**	**OSS**	**AP**	**ODS**	**OSS**	**AP**
BMC (84)	0.47	0.48	0.32	0.51	0.52	0.38
VSS (26)	0.51	0.56	0.45	0.45	0.51	0.42
HGB (24)	0.47	0.54	0.41	0.52	0.55	**0.52**
SHGB(74)	0.38	0.46	0.32	0.45	0.48	0.44
SOLD (57)	0.54	0.58	0.40	0.53	0.60	0.46
Ours$_{-calibration}$	0.64	0.64	0.45	0.53	0.58	0.49
Ours$_{-consistency}$	0.65	0.65	**0.50**	0.51	0.53	0.47
Ours	**0.65**	**0.65**	0.48	**0.55**	**0.61**	0.51
Human	0.81	0.81	0.67	0.83	0.83	0.70

Table 2.2: Comparison of state-of-the-art video segmentation algorithms with our proposed method on the test set of VSB100.

neighboring windows.

Features and Distances: Common boundary strength [*cbs*]. This measures distance in the close vicinity of the common boundary between two superpixels i_f and j_f by averaging the common boundary strength. We take $\overline{\mathbf{v}}_f^{ij}$ the average UCM of (76) as a measure of the boundary strength between i and j and define: $cbs(i_f, j_f) = \overline{\mathbf{v}}_f^{ij}$.

Lab [*lab*]. This uses the distance between the median brightness and color of a superpixel in Lab-color-space as a measure of the overall distance among two superpixels i and j, from the same or different frames f and f': $lab(i_f, j_{f'}) = \|\overline{LAB}_{i_f} - \overline{LAB}_{j_{f'}}\|_2$.

Boundary optical flow [*bof*]. We consider an optical flow estimation (26). The resulting $u_f(x)$ allows to compute the motion distance in the vicinity of the boundary between two superpixels by averaging their u_f across the common boundary

φ_f^{ij}: $bof(i_f, j_f) = \left(\sum_{(x_i^m, x_i^m) \in \varphi_f^{ij}} \|\overline{u}^f\,(x_i^m) - \overline{u}^f\,(x_j^m)\|_2\right) / |\varphi_f^{ij}|$.

Superpixel optical flow [*sof*]. This measures the overall motion distance between two superpixels i_f and $j_{f'}$ based on their median optical flow u: $sof(i_f, j_{f'}) = \|\overline{u}_{i_f} - \overline{u}_{j_{f'}}\|_2$.

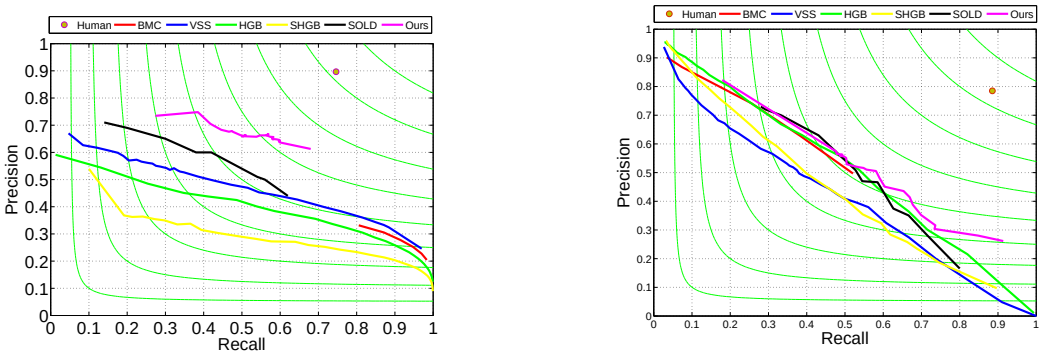
Superpixel shape distance [*ssd*]. We measure the shape distance by comparing $m_{j_{f'}}$ the shape of a superpixel j at frame f' with the shape of i_f propagated with optical flow to frame f' (its projected mask $m_{i_f}^{f'}$). *ssd* is given by the Dice coefficient between the true $m_{j_{f'}}$ and optical-flow-projected $m_{i_f}^{f'}$ binary mask: $ssd(i_f, j_{f'}) = 1 - 2|m_{i_f}^{f'} \cap m_{j_{f'}}| / \left(|m_{i_f}^{f'}| + |m_{j_{f'}}|\right)$.

Superpixel trajectories intersection [*sti*]. It measures the distance between superpixels i_f and $j_{f'}$ which belongs to frames potentially further in time from each other $f' = f + m, m > 2$. We consider the dense point trajectories of (85) as a measure of the shape (binary mask) projection. Let $\phi(i_f)$ be the subset of trajectories intersecting superpixel i_f. The distance is the Dice measure between the intersection sets

$\phi(i_f)$ and $\phi(j'_f)$: $sti(i_f, j_{f'}) = 1 - 2|\phi(i_f) \cap \phi(j'_f)|/\Big(|\phi(i_f)| + |\phi(j'_f)|\Big)$.

Evaluation Metrics: Following (57, 79), we use two evaluation metrics: 1) Boundary Precision-Recall (BPR), which casts the boundary detection problem as one of classifying boundary from nonboundary pixels and measures the quality of a segmentation boundary map in the precision-recall framework; and 2) Volume Precision-Recall (VPR), which optimally assigns spatio-temporal volumes between the computer generated segmentation and the human annotated segmentations and then measures their overlap. For both BPR and VPR, we report average precision (AP), optimal dataset scale (ODS), and optimal segmentation scale (OSS).

2.3.5.2 Comparison with state-of-the-art video segmentation methods



(a) Boundary Global PR Curve

(b) Volume Global PR Curve

Figure 2.10: Comparison curves of our framework with the state-of-the-art video segmentation approaches BMC (84), VSS (26), HGB (24), SHGB (74) and SOLD (57).

We compare our approach with the following five state-of-the-art video segmentation algorithms: BMC (84), VSS (26), HGB (24), SHGB (74) and SOLD (57). We also report our method without calibration (Ours$_{-calibration}$) and our method without temporal consistency (Ours$_{-consistency}$). Table 2.2 illustrates a summary of the aggregate evaluation performance, including ODS, OSS and AP of both BPR and VPR. Fig.2.10 shows the BPR and VPR curves

of the comparisons on the VSB100 dataset. From Table 2.2 and Fig.2.10, we have the following observations:

- Our approach outperforms the state-of-the-art methods (BMC, VSS, HGB, SHGB and SOLD) in both BPR and VPR on the VSB100 dataset. Specifically, our proposed method outperforms the currently best performance (SOLD (57)) on both BPR and VPR by a large margin, as it appears both in the Table 2.2 and Fig.2.10 (11%, 7% and 8% in BPR, 2%, 1% and 5% in VPR). Our AP in VPR is slightly lower than HGB. But we can alleviate it by simply increasing the superpixels number, as shown in Table 2.4.

- Though VSS (26) and SOLD (57) exploits multiple cues as well, our method performs better. This probably owes to the proposed joint graph learning and video segmentation framework, and the automatically learned weights for different cues.

- SOLD (57) is a strong competitor. The superior performances over SOLD in both BPR and VPR demonstrate that our approach can not only effectively infer the spatial similarity between superpixels within a frame, but also preserve the longer-range temporal consistency in a streaming mode.

- Temporal consistency processing plays an important role for VPR, as indicated in Table 2.2. An example is given in Fig.2.11 to illustrate the effect of temporal consistency processing. If we do not constrain that the same object in the close frames to have the same label, the performance on VPR metric will decrease.

- Topology calibration improves the performance, especially for VPR. This is due to that without calibration, the distances of different topological structures (especially for cross frame) are not comparable.

We illustrate qualitative results in Fig.2.9, comparing our proposed method to the state-of-the-art video segmentation algorithms including HGB, SHGB and SOLD. Fig.2.9 shows consistent results to the quantitative results. Our method is able to provide better distinguished visual objects with well-localized boundaries and limited label leakage.

2.3.5.3 Component analysis

In this subsection, we study the effect of *level* and the different types of *distances* on our proposed method.

	BPR			VPR		
Algorithm	**ODS**	**OSS**	**AP**	**ODS**	**OSS**	**AP**
CBS	0.64	0.64	0.46	0.43	0.46	0.38
BOF	0.64	0.64	0.44	0.43	0.44	0.37
LAB	0.64	0.65	0.45	0.51	0.53	0.43
SOF	0.64	0.64	0.47	0.42	0.45	0.37
SSD	0.64	0.64	0.44	0.51	0.56	0.45
STI	0.64	0.64	0.45	0.46	0.49	0.42
All	0.65	0.65	0.48	0.55	0.61	0.51
Human	0.81	0.81	0.67	0.83	0.83	0.70

Table 2.3: The effect of *distances* on our proposed method.

	BPR			VPR		
Algorithm	**ODS**	**OSS**	**AP**	**ODS**	**OSS**	**AP**
$Level = 25$	0.59	0.59	**0.54**	0.54	0.58	**0.56**
$Level = 50$	0.64	0.65	0.53	**0.55**	0.57	0.51
$Level = 75$	**0.65**	**0.65**	0.48	**0.55**	**0.61**	0.51
$Level = 95$	0.64	0.64	0.47	**0.55**	0.59	0.46
$Level = 125$	0.61	0.61	0.41	0.53	0.58	0.40
Human	0.81	0.81	0.67	0.83	0.83	0.70

Table 2.4: The effect of *level* on our proposed method.

As described in Section 2.3.3, our proposed algorithm is imposed on the superpixels which are extracted from (76). The number of superpixels is determined by the value of *level*. From the Table 2.4, we can see that the *level* is important to the performance. In general, when $level = 75$, the overall best performance is achieved for both BPR (65%, 65% and 48%) and VPR (55%, 61% and 51%). In addition, when $level = 25$, the AP for both BPR and VPR reaches the peak values: 54% and 56% respectively. This is due to that when $level = 25$, more superpixels are generated and over-segmentation improves the precision but decreases the recall (76).

The effect of different distances is analyzed and the results are shown in Table 2.3. As described in Section 2.3.3, our method uses different types of pairwise distance between superpxiels for video segmentation. From Table 2.3, we can see that *distances* have different impact

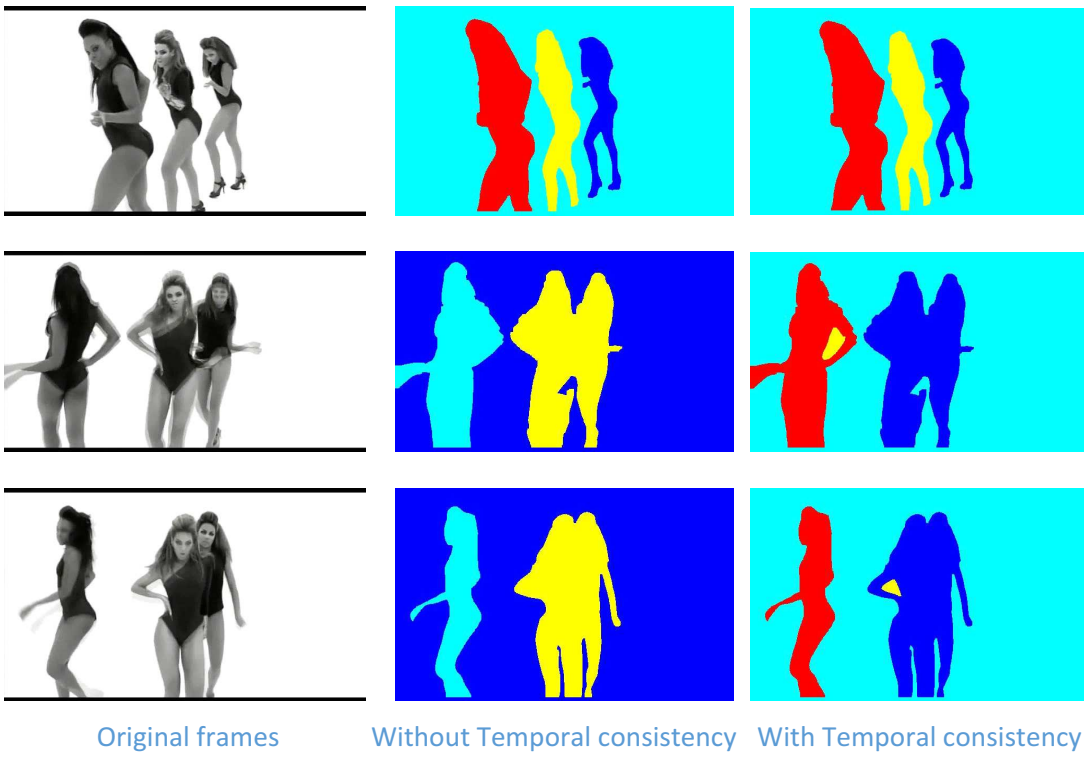


Figure 2.11: Segmentations of a given sequence of video frames. The left are the original frames, the middle lines are the segmentations without temporal consistency, and the right lines are the segmentations with temporal consistency processing. It is clear that if no temporal consistency processing is applied, the same object in the successive frames cannot be guaranteed to have the same segment id (i.e., label). In this case, the VPR will degrade.

on the performance, hence it is important to learn the weights for different distances. Using a combined distances with learned weight achieves better performance than using a single distance.

2.3.5.4 Efficiency Analysis

In this subsection, we evaluate the time efficiency of our proposed method. These experiments are carried out on a desktop with an Intel(R) Core (TM)2 Duo CPU and 8GB RAM. Our method is conducted on the features of the generated superpixels, which are obtained in the preprocessing step. The computational cost for our algorithm is $O(N^2) + O(N^3)$. However, we do segmentation on the N superpixels, which is usually 10-100s. Therefore, our method can segment 1 frame within 0.2s on average, as indicated in Fig. 2.12. Fig. 2.12 reports the running

time corresponding to the number of segmentations. In average, it costs 0.1895 seconds per frame for our proposed method. Large number of segmentations does not necessarily consume more time, because the optimization may terminate in less iterations.

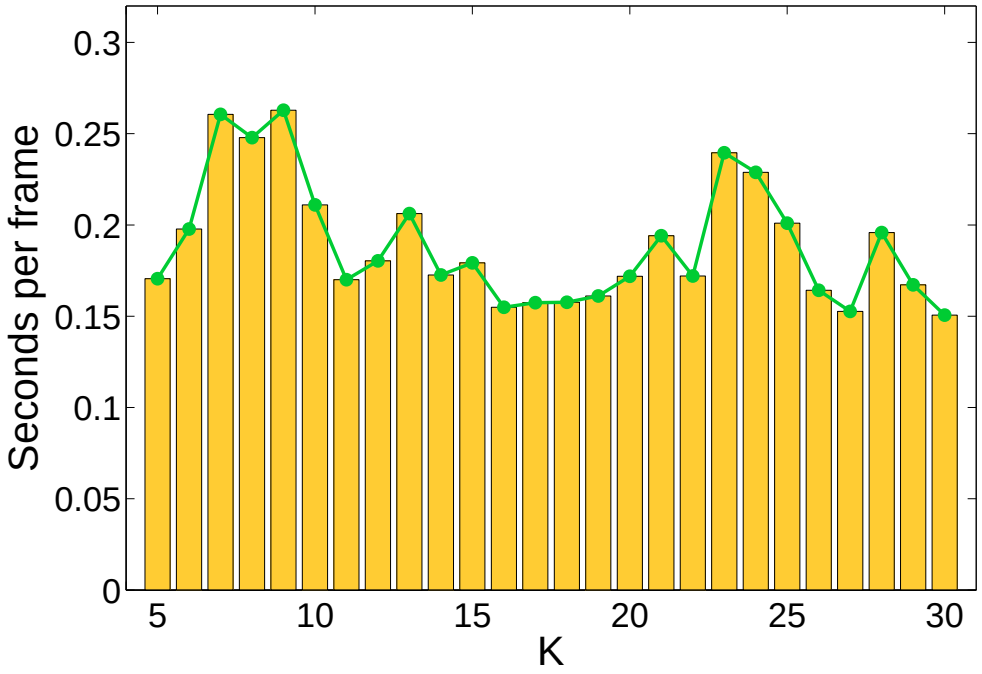


Figure 2.12: The running time (seconds per frame) *vs.* number of segmentations (K) of our proposed method.

2.3.6 Conclusions

This Section proposes a framework that simultaneously learns the graph similarity matrix and video segmentation, instead of first organizing the superpixels into graphs and then cutting the generated graph for segmentation. Based on the spatial and visual information, each vertex is assigned with adaptive and optimal neighbors for graph similarity learning. By imposing rank constraints on the Laplacian matrix, the number of connected components in the generated similarity graph are equal to the number of segmentations, sidestepping the need for separate steps in similarity computation and graph cutting. Experimental results show that the proposed unsupervised system outperforms state-of-the-art video segmentation algorithms by a large margin on the VSB100 dataset.

This method provides a series of advantages over the work presented in Section 2.2, specifically a more precise temporal segmentation, and the guaranteed segmentation of background objects.

2.4 Unsupervised Adversarial Depth Estimation [2]

While Sections 2.2 and 2.3 consider the data quality problem through a lack of annotations, the method detailed in the current section is motivated by the difficulty of computing depth maps for use in further tasks without the use highly expensive LIDAR annotations. The system makes use of a cycled generative network to impose stronger constraints between the stereo image pair, leading to the system learning better representations.

2.4.1 Introduction

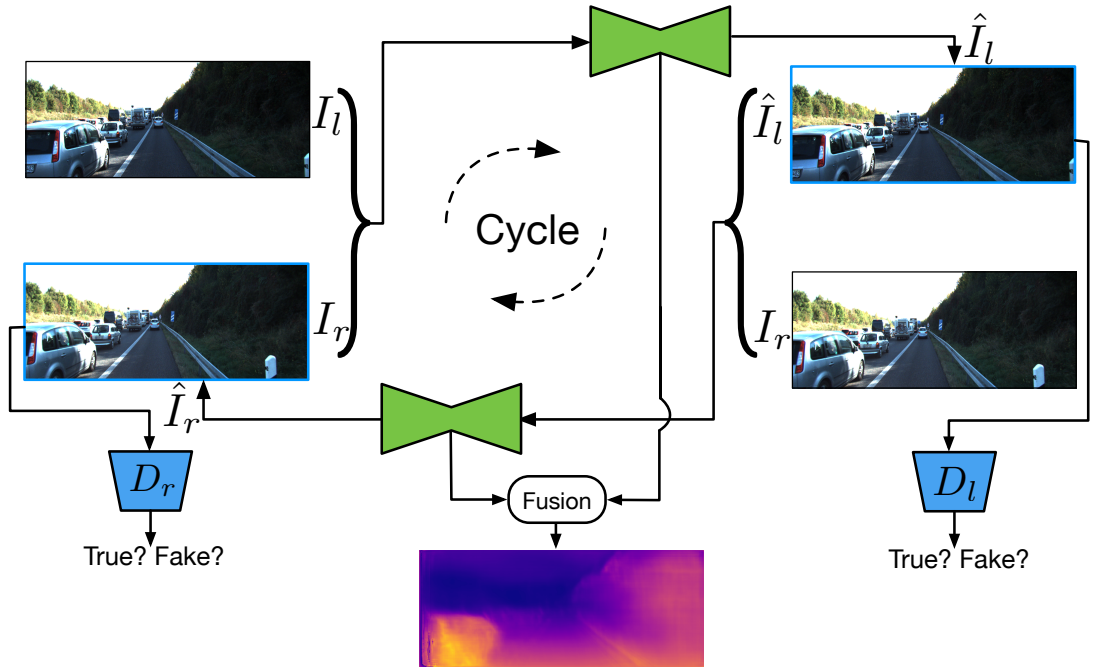


Figure 2.13: Motivation of the proposed unsupervised depth estimation approach using cycled generative networks optimized with adversarial learning. The left and right image synthesis in a cycle provides each other strong constraint and supervision to better optimize both generators. The $\hat{\mathbf{I}}_r$ and $\hat{\mathbf{I}}_l$ are synthesized images. Final depth estimation is obtained by fusing the output from both generators.

Most previous works considering deep architectures for predicting depth maps operate in a supervised learning setting (33, 34, 35, 36) and, specifically, devise powerful deep regression

[2] "Unsupervised Adversarial Depth Estimation using Cycled Generative Networks" Andrea Pilzer*, Dan Xu*, Mihai Puscas*, Elisa Ricci, Nicu Sebe; 2018 International Conference on 3D Vision (3DV)587-595 (3)

models with Convolutional Neural Networks (CNN). These models are used for monocular depth estimation, *i.e.* they are trained to learn the transformation from the RGB image domain to the depth domain in a pixel-to-pixel fashion. In this context, multi-scale CNN models have shown to be especially effective for estimating depth maps (33). Upon these, probabilistic graphical models, such as Conditional Random Fields (CRFs), implemented as neural networks for end-to-end optimization, have proved to be beneficial, boosting the performance of deep regression models (35, 36). However, supervised learning models require ground-truth depth data which are usually costly to acquire. This problem is especially relevant with deep learning architectures, as large amount of data are typically required to produce satisfactory performance. Furthermore, supervised monocular depth estimation can be regarded as an ill-posed problem due to the scale ambiguity issue (86).

To tackle these problems, recently unsupervised learning-based approaches for depth estimation have been introduced (87, 88). These methods operate by learning the correspondence field (*i.e.* the disparity map) between the two different image views of a calibrated stereo camera using only the rectified left and right images. Then, given several camera parameters, the depth maps can be calculated using the predicted disparity maps. Significant progresses have been made along this research line (89, 90, 91). In particular, Godard *et al.* (90) proposed to estimate both the direct and the reverse disparity maps using a single generative network and utilized the consistency between left and right disparity maps to constrain on the model learning. Other works proposed to facilitate the depth estimation by jointly learning the camera pose (92, 93). These works optimized their models relying on the supervision from the image synthesis of an expected view, whose quality plays a direct influence on the performance of the estimated disparity map. However, all of these works only considered a reconstruction loss and none of them have explored using adversarial learning to improve the generation of the synthesized images.

In this paper, we follow the unsupervised learning setting and propose a novel end-to-end trainable deep network model for adversarial learning-based depth estimation given stereo image pairs. The proposed approach consists of two generative sub-networks which predict the disparity map from the left to the right view and vice-versa. The two sub-networks are organized in a cycle (Fig. 2.13), such as to perform the image synthesis of different views in a closed loop. This new network design provides strong constraint and supervision for each image view, facilitating the optimization of both generators from the two sub-networks which

are jointly learned with an adversarial learning strategy. The final disparity map is produced by combining the output from the two generators.

In summary, the main contributions of this work are threefold:

- To the best of our knowledge, we are the first to explore using adversarial learning to facilitate the image synthesis of different views in a unified deep network for improving the unsupervised depth estimation;
- We present a new cycled generative network structure for unsupervised depth estimation which can learn both the forward and the reverse disparity maps, and can synthesize the different image views in a closed loop. Compared with the existing generative network structures, the proposed cycled generative network is able to enforce stronger constraints from each image view and better optimize the network generators.
- Extensive experiments on two large publicly available datasets (*i.e.* KITTI and Cityscapes) demonstrate the effectiveness of both the adversarial image synthesis and the cycled generative network structure.

2.4.2 Related Work

Supervised Depth Estimation. Supervised deep learning greatly improved the performance of depth estimation. Given enough ground-truth depth training data, deep neural networks based approaches have achieved very promising performances in recent years. Multiple large-scale depth-contained datasets (94, 95, 96, 97) have been published. In a single view setting, NYUD (94) presents indoor images while Make3D (95) is recorded in outdoors. Instead KITTI (96) and Cityscapes (97) are collected in outdoors with calibrated stereo cameras. Based on these datasets, a significant effort has been made for the supervised monocular depth estimation task (33, 35, 36, 98, 99). The multi-scale CNN (33) and probabilistic graphical models based deep networks (35, 36, 100) also show an obvious performance boosting on the task. Xu *et al.* (101) first introduce a structured attention mechanism for learning better multi-scale deep representations for the task. However, the supervised-based approaches rely on the expensive ground-truth depth data during training, which are not flexible to deploy crossing application scenarios.

Unsupervised Depth Estimation. A more recent trend is unsupervised-based depth estimation (91, 93, 102, 103). A remarkable advantage of unsupervised estimation lies in avoiding the use of costly ground truth depth annotations in training. Deep stereo matching mod-

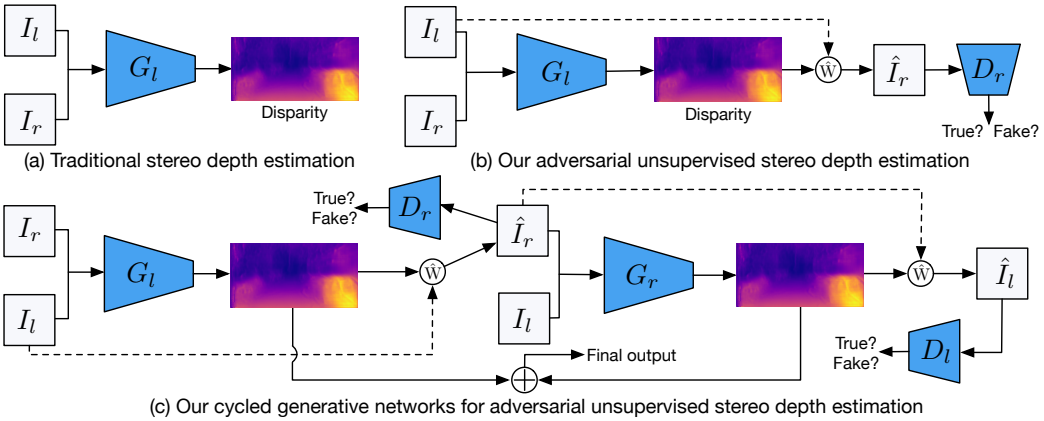


Figure 2.14: An illustrative comparison of different methods for unsupervised stereo depth estimation: (a) traditional stereo-matching-based depth estimation, (b) the proposed unsupervised adversarial depth estimation and (c) the proposed cycled generative networks for unsupervised adversarial depth estimation. The symbols D_l, D_r denote discriminators, and G_l, G_r denote generators. The symbol $\hat{\mathrm{W}}$ denotes a warping operation.

els (87, 88) are proposed for direct disparity estimation. In an indirect means, Garg *et al.* (89) propose a classic approach for unsupervised monocular depth estimation based on image synthesis. Godard *et al.* (90) propose to use forward and backward reconstructions of the different image views, and multiple optimization losses are considered in the model. Zhou *et al.* (92) jointly learn the depth and the camera pose as a reinforcement in a single deep network. There are also works jointly learning the scene depth and ego-motion in monocular videos without using groundtruth data (104, 105). However, none of these works considers the adversarial learning scheme in their models to improve the image generation quality for better depth estimation.

GANs. Generative-adversarial networks (GANs) have attracted a lot of attention for its advantage in generation problems. Godfellow *et al.* (106) revisit the generative adversarial learning strategy and show interesting results in the image generation task. After that, GANs are applied into various generation applications, and different GAN models are developed, such as CycleGAN (107) and DualGAN (108). There are few works in the literature considering GAN models for the more challenging depth estimation task. Although Kundu *et al.* (109) investigate adversarial learning for the task, they utilize it in a context of domain adaptation in a single-track network, using a semi-supervised setting with an extra synthetic dataset, while ours considers a fully unsupervised setting and the adversarial learning in a cycled generative

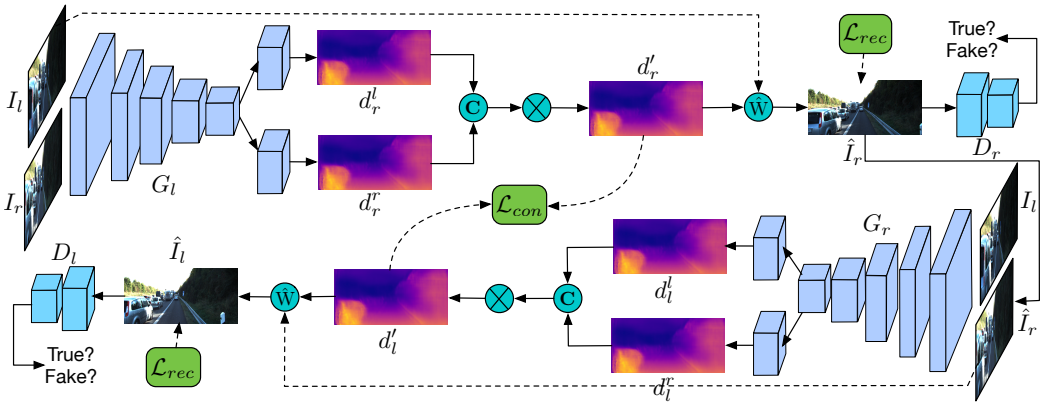


Figure 2.15: Illustration of the detailed framework of the proposed cycled generative networks for unsupervised adversarial depth estimation. The symbol ⓒ denotes a concatenation operation; $\mathcal{L}_{rec}$ represents the reconstruction loss for different generators; $\mathcal{L}_{con}$ denotes a consistence loss between the disparity maps generated from the two generators.

network aims to help the reconstruction of better image views. Both the intuition and the network design are significantly different.

2.4.3 The Proposed Approach

We propose a novel approach for unsupervised adversarial depth estimation using cycled generative networks. An illustrative comparison of different unsupervised depth estimation models is shown in Fig. 2.14. Fig. 2.14a shows traditional stereo matching based depth estimation approaches, which basically learn a stereo matching network for directly predicting the disparity (87). Different from the traditional stereo approaches, we estimate the disparity in an indirect means through image synthesis from different views with the adversarial learning strategy as shown in Fig. 2.14b. Fig. 2.14c shows our full model using the proposed cycled generative networks for the task. In this section we first give the problem statement, and then present the proposed adversarial learning-based unsupervised stereo depth estimation, and finally we illustrate the proposed full model and introduce the overall end-to-end optimization objective and the testing process.

2.4.3.1 Problem Statement

We target at estimating a disparity map given a pair of images from a calibrated stereo camera. The problem can be formally defined as follows: given a left image $\mathbf{I}_l$ and a right image $\mathbf{I}_r$ from the camera, we are interested in predicting a disparity map $\mathbf{d}$ in which each pixel value

represents an offset of the corresponding pixel between the left and the right image. If given the baseline distance b_d between the left and the right camera and the camera focal length f_l, a depth map $\mathbf{D}$ can be calculated with the formula of $\mathbf{D} = (b_d * f_l)/\mathbf{d}$. We indirectly learn the disparity through the image synthesis. Specifically, assume that a left-to-right disparity $\mathbf{d}_r^{(l)}$ is produced from a generative network G_l with the left-view image $\mathbf{I}_l$ as input, and then a warping function $f_w(\cdot)$ is used to perform the synthesis of the right image view by sampling from $\mathbf{I}_l$, *i.e.* $\hat{\mathbf{I}}_r = f_w(\mathbf{d}_r^{(l)}, \mathbf{I}_l)$. A reconstruction loss between $\hat{\mathbf{I}}_r$ and $\mathbf{I}_r$ is thus utilized to provide supervision in optimizing the network G_l.

2.4.3.2 Unsupervised Adversarial Depth Estimation

We now introduce the proposed unsupervised adversarial depth estimation approach. Assuming we have a generative network G_l composed of two sub-networks, a generative sub-network $G_l^{(l)}$ with input $\mathbf{I}_l$ and a generative sub-network $G_l^{(r)}$ with input $\mathbf{I}_r$. These are used to produce two distinct left-to-right disparity maps $\mathbf{d}_r^{(l)}$ and $\mathbf{d}_r^{(r)}$ respectively, *i.e.* $\mathbf{d}_r^{(l)} = G_l^{(l)}(\mathbf{I}_l)$ and $\mathbf{d}_r^{(r)} = G_l^{(r)}(\mathbf{I}_r)$. The sub-network $G_l^{(l)}$ and $G_l^{(r)}$ exploit the same network structure using a convolutional encoder-decoder, where the encoders aim at obtaining compact image representations and could be shared to reduce the network capacity. Since the two disparity maps are produced from different input images, and show complementary characteristics, they are fused using a linear combination implemented as concatenation and 1×1 convolution, and we obtain an enhanced disparity map $\mathbf{d}'_r$, which is used to synthesize a right view image $\hat{\mathbf{I}}_r$ via the warping operation, *i.e.* $\hat{\mathbf{I}}_r = f_w(\mathbf{d}'_r, \mathbf{I}_l)$. Then we use an $L1$-norm reconstruction loss $\mathcal{L}_{rec}$ for optimization as follows:

$$\mathcal{L}_{rec}^{(r)} = \|\mathbf{I}_r - f_w(\mathbf{d}'_r, \mathbf{I}_l)\|_1 \tag{2.41}$$

To improve the generation quality of the image $\hat{\mathbf{I}}_r$ and benefit from the advantage of adversarial learning, we propose to use adversarial learning here for a better optimization due to its demonstrated powerful ability in the image generation task (106). For the synthesized image $\hat{\mathbf{I}}_r$, a discriminators D_r outputting a scalar value which is used to discriminate if the image $\hat{\mathbf{I}}_r$ or $\mathbf{I}_r$ is fake or true, and thus the adversarial objective for the generative network can be formulated as follows:

$$\begin{aligned}\mathcal{L}_{gan}^{(r)}(G_l, D_r, \mathbf{I}_l, \mathbf{I}_r) = \mathbb{E}_{\mathbf{I}_r \sim p(\mathbf{I}_r)}[\log D_r(\mathbf{I}_r)] \\ + \mathbb{E}_{\mathbf{I}_l \sim p(\mathbf{I}_l)}[\log(1 - D_r(f_w(\mathbf{d}'_r, \mathbf{I}_l)))]\end{aligned} \tag{2.42}$$

where we adopt a cross-entropy loss to measure the expectation of the image $\mathbf{I}_l$ and $\mathbf{I}_r$ against the distribution of the left and the right view images $p(\mathbf{I}_l)$ and $p(\mathbf{I}_r)$ respectively. Then the joint optimization loss is the combination of the reconstruction loss and the adversarial loss written as:

$$\mathcal{L}_o^{(r)} = \gamma_1 \mathcal{L}_{rec}^{(r)} + \gamma_2 \mathcal{L}_{gan}^{(r)} \tag{2.43}$$

where γ_1 and γ_2 are the weights for balancing the loss magnitude of the two parts to stabilize the training process. In the testing phase, the inferred $\mathbf{d}'_r$ is the final output.

2.4.3.3 Cycled Generative Networks for Adversarial Depth Estimation

In the previous section, we presented the adversarial learning-based depth estimation approach which reconstructs from one image view to the other one in a straightforward way. In order to make the image reconstruction from different views implicitly constrain on each other, we further propose a cycled generative network structure. An overview of the proposed network structure is shown in Fig. 2.4.2. The network produces two distinct disparity maps from different view directions, and synthesizes different-view images in a closed loop. In our network design, not only the different view reconstruction loss helps for better optimization of the generators, but also the two disparity maps are connected with a consistence loss to provide strong supervision from each half cycle.

We described the half-cycle generative network with adversarial learning in Section 2.4.3.2. The cycled generative network is based on the half-cycle structure. To simplify the description, we follow the notations used in Section 2.4.3.2. Assume we have obtained a synthesized image $\hat{\mathbf{I}}_r$ from the half-cycle network, and then $\hat{\mathbf{I}}_r$ is further used as input of the next cycle generative network. Let us denote the generator as G_r, which we exploit the encoder-decoder network structure similar as G_l in Sec. 2.4.3.2. The encoder part of G_r can be also shared with the encoder of G_l to have a more compact network model (we show the performance difference between using and not using the sharing scheme), and the two distinct decoders are used to produce two right-to-left disparity maps $\mathbf{d}_l^{(l)}$ and $\mathbf{d}_l^{(r)}$ corresponding the left- and the right-view input images respectively. The two maps are also combined with the combination and the convolution operation to have a fused disparity map $\mathbf{d}'_l$. Then we synthesize the left-view image $\hat{\mathbf{I}}_l$ via the warping operation as $\hat{\mathbf{I}}_l = f_w(\mathbf{d}'_l, \mathbf{I}_r)$. An $L1$-norm reconstruction loss is used for optimizing the generator G_r. Then the objective for optimizing the two generators of the

full cycle writes

$$\mathcal{L}_{rec}^{(f)} = \|\mathbf{I}_r - f_w(\mathbf{d}'_r, \mathbf{I}_l)\|_1 + \|\mathbf{I}_l - f_w(\mathbf{d}'_l, \hat{\mathbf{I}}_r)\|_1 \tag{2.44}$$

We add a discriminator D_l for discriminating the synthesized image $\hat{\mathbf{I}}_l$, and then the adversarial learning strategy is used for both the left and the right image views in a closed loop. The adversarial objective for the full cycled model can be formulated as

$$\begin{aligned} &\mathcal{L}_{gan}^{(f)}(G_l, G_r, D_r, \mathbf{I}_l, \mathbf{I}_r) = \mathbb{E}_{\mathbf{I}_r \sim p(\mathbf{I}_r)}[\log D_r(\mathbf{I}_r)] \\ &+\mathbb{E}_{\mathbf{I}_l \sim p(\mathbf{I}_l)}[\log(1 - D_r(f_w(\mathbf{d}'_r, \mathbf{I}_l)))] + \mathbb{E}_{\mathbf{I}_l \sim p(\mathbf{I}_l)}[\log D_l(\mathbf{I}_l)] \\ &+\mathbb{E}_{\mathbf{I}_r \sim p(\mathbf{I}_r)}[\log(1 - D_l(f_w(\mathbf{d}'_l, \hat{\mathbf{I}}_r)))] \end{aligned} \tag{2.45}$$

Each half of the cycle network produces a disparity map corresponding to a different view translation, *i.e.* $\mathbf{d}'_l$ and $\mathbf{d}'_r$. To make them constrain on each other, we add an $L1$-norm consistence loss between these two maps as follows:

$$\mathcal{L}_{con}^{(f)} = ||\mathbf{d}'_l - f_w(\mathbf{d}'_l, \mathbf{d}'_r)||_1 \tag{2.46}$$

where since the two disparity maps are for different views and are not aligned, we use the warping operation to make them pixel-to-pixel matched. The consistence loss put a strong view constraint for each half cycle and thus facilitates the learning of both half cycles.

Full objective. The full optimization objective consists of the reconstruction losses of both generators, the adversarial losses for both view synthesis and the half-cycle consistence loss. It can be written as follows:

$$\mathcal{L}_o^{(f)} = \gamma_1 \mathcal{L}_{rec}^{(f)} + \gamma_2 \mathcal{L}_{gan}^{(f)} + \gamma_3 \mathcal{L}_{con}^{(f)}. \tag{2.47}$$

Where $\{\gamma_i\}_{i=1}^3$ represents a set of weights for controlling the importance of different optimization parts.

Inference. When the optimization is finished, given a testing pair $\{\mathbf{I}_l, \mathbf{I}_r\}$, the testing is performed by combining the output disparity maps $\mathbf{d}'_l$ and $\mathbf{d}'_r$ in a weighted averaging scheme. We treat the two half cycles with equal importance, and the final disparity map $\mathbf{D}$ is obtained as the mean of the two, *i.e.* $D = (\mathbf{d}'_l + f_w(\mathbf{d}'_l, \mathbf{d}'_r))/2$.

2.4.3.4 Network Implement Details

To describe the details of the network implementation, in terms of the generators G_l and G_r, we use a ResNet-50 backbone network for the encoder part, and the decoder part contains

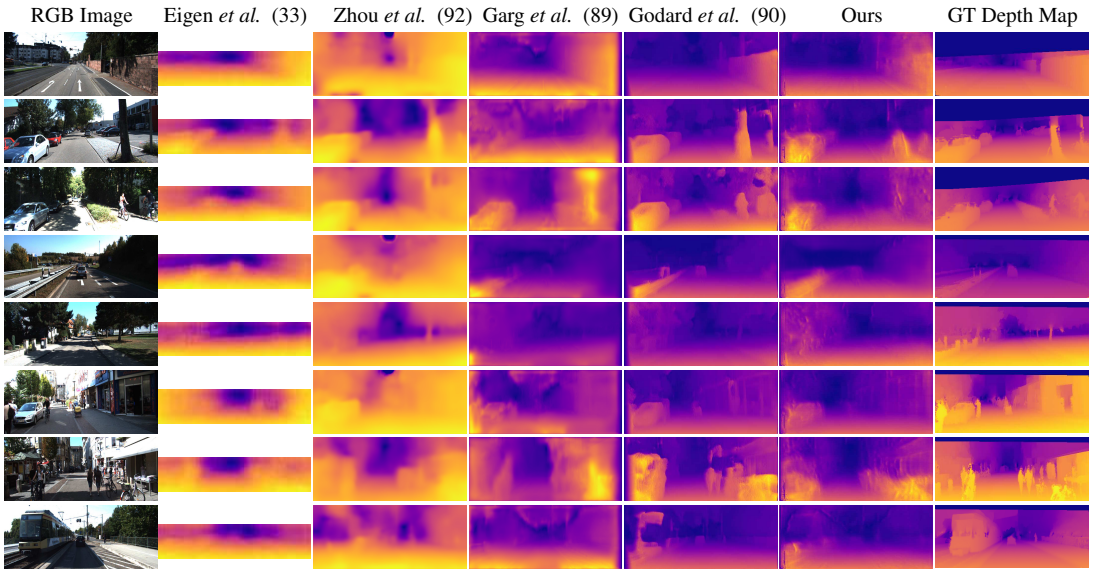


Figure 2.16: Qualitative comparison with different competitive approaches with both supervised and unsupervised settings on the KITTI test set. The sparse groundtruth depth maps are filled with bilinear interpolation for better visualization.

five deconvolution with ReLU operations in which each 2 times up-samples the feature map. The skip connections are also used to pass information from the backbone representations to the deconvolutional feature maps for obtaining more effective feature aggregation. For the discriminators D_l and D_r, we employ the same network structure which has five consecutive convolutional operations with a kernel size of 3, a stride size of 2 and a padding size of 1, and batch normalization (110) is performed after each convolutional operation. Adversarial loss is applied to output patches. For the warping operation, a bilinear sampler is used as in (90).

2.4.4 Experimental Results

We present both qualitative and quantitative results on publicly available datasets to demonstrate the performance of the proposed approach for unsupervised adversarial depth estimation.

2.4.4.1 Experimental Setup

Datasets. We carry out experiments on two large datasets, *i.e.* KITTI (96) and Cityscapes (97) . For the **KITTI** dataset, we use the Eigen split (33) for training and testing. This split contains 22,600 training image pairs, and 697 test pairs. We do data augmentation with online random flipping of the images during training. The **Cityscapes** dataset is collected using a stereo

Method	Sup	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
		lower is better				higher is better		
Half-Cycle Mono	N	0.240	4.264	8.049	0.334	0.710	0.871	0.937
Half-Cycle Stereo	N	0.228	4.277	7.646	0.318	0.748	0.892	0.945
Half-Cycle + D	N	0.211	2.135	6.839	0.314	0.702	0.868	0.939
Full-Cycle + D	N	0.198	**1.990**	**6.655**	**0.292**	0.721	0.884	0.949
Full-Cycle + D + SE	N	**0.190**	2.556	6.927	0.353	**0.751**	**0.895**	**0.951**

Table 2.5: Quantitative evaluation results of different variants of the proposed approach on the KITTI dataset for the ablation study. We do not perform cropping on the depth maps for evaluation and the estimated depth range is from 0 to 80 meters.

camera from a driving vehicle through several German cities, during different times of the day and seasons. It presents higher resolution images and is annotated mainly for semantic segmentation. To train our model we combine the densely and coarse annotated splits to obtain 22,973 image-pairs. For testing we use the 1,525 image-pairs of the densely annotated split. The test set also has pre-computed disparity maps for the evaluation.

Parameter Setup. The proposed model is implemented using the deep learning library *TensorFlow* (111). The input images are down-sampled to a resolution of 512×256 from 1226×370 in the case of the KITTI dataset, while for the Cityscapes dataset, at the bottom one fifth of the image is cropped following (90) and then is resized to 512×256. The output disparity maps from two input images are fused with a learned linear combination to obtain the final disparity map with a size 512×256. The batch size for training is set to 8 and the initial learning rate is 10^{-5} in all the experiments. We use the Adam optimizer for the optimization. The momentum parameter and the weight decay are set to 0.9 and 0.0002, respectively. The final optimization objective has weighed loss parameters $\gamma_1 = 1$, $\gamma_2 = 0.1$ and $\gamma_3 = 0.1$. The learning rate is reduced by half at both $[80k, 100k]$ steps. For our experiments we used an NVIDIA Tesla K80 with 12 GB of memory.

Detailed Training Procedure. We train the half-cycle model with a standard training procedure, *i.e.* initializing the network with random weights and making the network train for a full 50 epochs. For the cycled model we optimize the network with an iterative training procedure. After random weights initialization, we train the first half branch $\{\mathbf{I}_l, \mathbf{I}_r\} \rightarrow \hat{\mathbf{I}}_r$, with generator G_l and discriminator D_r for a 20k iteration steps. After that we train the second half branch $\{\hat{\mathbf{I}}_r, \mathbf{I}_l\} \rightarrow \hat{\mathbf{I}}_l$ with generator G_r and discriminator D_l for another 20k iterations.

Method	Sup	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
		lower is better				higher is better		
Saxena *et al.* (86)	Y	0.280	-	8.734	-	0.601	0.820	0.926
Eigen *et al.* (33)	Y	0.190	1.515	7.156	0.270	0.692	0.899	0.967
Liu *et al.* (35)	Y	0.202	1.614	6.523	0.275	0.678	0.895	0.965
AdaDepth (109), 50m	Y	0.162	1.041	4.344	0.225	0.784	0.930	0.974
Kuznietzov *et al.* (102)	Y	-	-	4.815	0.194	0.845	0.957	0.987
Xu *et al.* (36)	Y	0.132	0.911	-	0.162	0.804	0.945	0.981
Zhou *et al.* (92)	N	0.208	1.768	6.856	0.283	0.678	0.885	0.957
Garg *et al.* (89)	N	0.169	1.08	5.104	0.273	0.740	0.904	0.962
AdaDepth (109), 50m	N	0.203	1.734	6.251	0.284	0.687	0.899	0.958
Godard *et al.* (90)	N	**0.148**	**1.344**	**5.927**	**0.247**	**0.803**	**0.922**	**0.964**
Ours	N	0.166	1.466	6.187	0.259	0.757	0.906	0.961
Ours with shared enc	N	0.152	1.388	6.016	0.247	0.789	0.918	0.965
Ours, 50m	N	0.158	1.108	4.764	0.245	0.771	0.915	0.966
Ours with shared enc, 50m	N	**0.144**	**1.007**	**4.660**	**0.240**	**0.793**	**0.923**	**0.968**

Table 2.6: Comparison with state of the art. Training and testing are performed on the KITTI (96) dataset. Supervised and semi-supervised methods are marked with Y in the supervision column, unsupervised methods with N. Numbers are obtained on Eigen test split with Garg image cropping. Depth predictions are capped at the common threshold of 80 meters, if capped at 50 meters we specify it.

For the training of the first cycle branch, we do not use the cycle consistence loss since the second half branch is not trained yet. Finally we jointly train the whole network with all the losses embedded for a final round of 100k iterations.

Evaluation Metrics. To quantitatively evaluate the proposed approach, we follow several standard evaluation metrics used in previous works (33, 90, 112). Given P the total number of pixels in the test set and $\hat{d}_i$, d_i the estimated depth and ground truth depth values for pixel i, we have (i) the mean relative error (abs rel): $\frac{1}{P}\sum_{i=1}^{P}\frac{\|\hat{d}_i - d_i\|}{d_i}$, (ii) the squared relative error (sq rel): $\frac{1}{P}\sum_{i=1}^{P}\frac{\|\hat{d}_i - d_i\|^2}{d_i}$, (iii) the root mean squared error (rmse): $\sqrt{\frac{1}{P}\sum_{i=1}^{P}(\hat{d}_i - d_i)^2}$, (iv) the mean $\log 10$ error (rmse log): $\sqrt{\frac{1}{P}\sum_{i=1}^{P}\| \log \hat{d}_i - \log d_i \|^2}$ (v) the accuracy with threshold t, *i.e.* the percentage of $\hat{d}_i$ such that $\delta = \max(\frac{d_i}{\hat{d}_i}, \frac{\hat{d}_i}{d_i}) < t$, where $t \in [1.25, 1.25^2, 1.25^3]$.

Method	Sup	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
		lower is better				higher is better		
Half-Cycle Mono	N	0.467	7.399	5.741	0.493	0.735	0.890	0.945
Half-Cycle Stereo	N	0.462	6.097	5.740	0.377	0.708	0.873	0.937
Half-Cycle + D	N	**0.438**	**5.713**	5.745	0.400	0.711	0.877	0.940
Full-Cycle + D	N	0.440	6.036	**5.443**	**0.398**	**0.730**	**0.887**	**0.944**

Table 2.7: Quantitative evaluation results of different variants of the proposed approach on the Cityscapes dataset for the ablation study.

2.4.4.2 Ablation Study

To validate the adversarial learning strategy is beneficial for the unsupervised depth estimation, and the proposed cycled generative network is effective for the task, we present an extensive ablation study on both the KITTI dataset (see Table 2.5) and on the Cityscape dataset (see Table 2.7).

Baseline Models. We have several baseline models for the ablation study, including (i) Half-cycle with a monocular setting (half-cycle mono), which uses a straight forward branch to synthesize from one image view to the other with a single disparity map output and the single RGB image is as input during testing; (ii) half-cycle with a stereo setting (half-cycle stereo), which uses a straight forward branch but with two disparity maps produced and combined; (iii) half-cycle with a discriminator (half-cycle + D), which use a single branch as in (ii) while adds a discriminator for the image synthesis; (iv) full-cycle with two discriminators (full-cycle + D), which is our whole model using a full cycle with two discriminators added; (v) full-cycle with two discriminators and sharing encoders (full-cycle + D + SE), which has the same structure as (iv) while the parameters of the encoders of the generators are shared.

Evaluation on KITTI. As we can see from Table 2.5, the baseline model Half-Cycle Stereo shows significantly better performance on seven out of eight evaluation metrics than the baseline model Half-Cycle Mono, demonstrating that the utilization of the stereo images and the combination of the two estimated complementary disparity maps clearly boosts the performance.

By using the adversarial learning strategy for the image synthesis, the baseline Half-Cycle + D outperforms the baseline Half-Cycle Stereo with around 1.7 points gain on the metric of Abs Rel, which verifies our initial intuition of using the adversarial learning to improve the

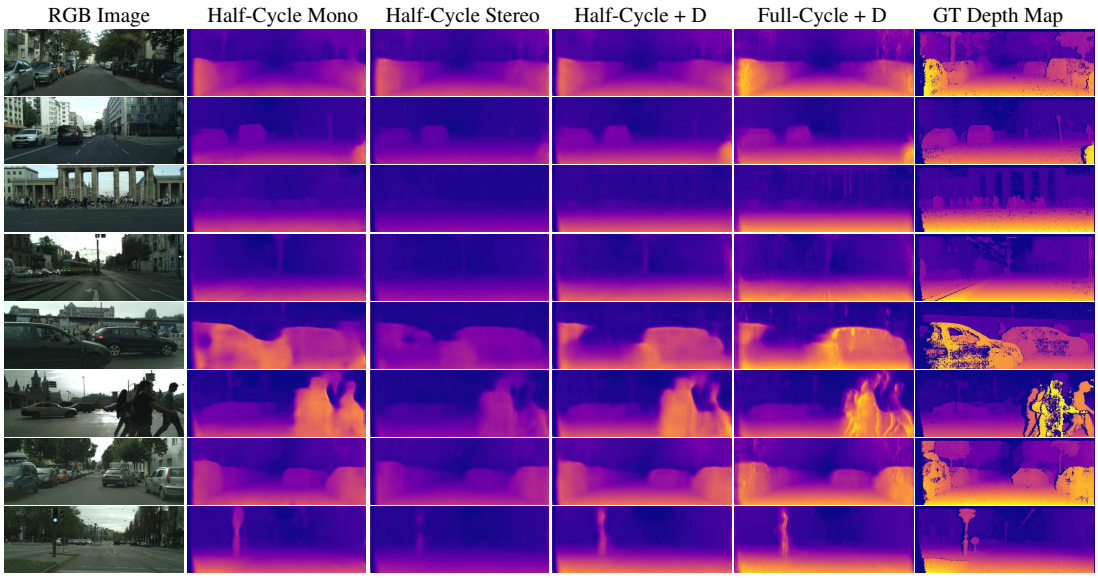


Figure 2.17: Qualitative comparison of different baseline models of the proposed approach on the Cityscapes testing dataset.

quality of the image synthesis, and thus gain the improvement of the disparity prediction. In addition, we also observe in the training process, the adversarial learning helps to maintain a more stable convergence trend with small oscillations in terms of the training loss than the one without it (*i.e.* Half-Cycle Stereo), probably leading to a better optimized model.

It is also clear to observe that the proposed cycled generative network with adversarial learning (Full-Cycle + D) achieved much better results than the models with only half cycle (Half-Cycle + D) on all the metrics. Specifically, the Full-Cycle + D model improves the Abs Rel around 2 points, and also improves the accuracy a1 around 1.9 points over Half-Cycle + D. The significant improvement demonstrates the effectiveness of the proposed network design, confirming that the cycled strategy brings stronger constraint and supervision to optimize the both generators. Finally, we also show that the propose cycled model using a sharing encoder for the generator (Full-Cycle + D + SE). By using the sharing structure, we obtain even better results than the non-sharing model (Full-Cycle + D), which is probably because the shared one has a more compact network structure and thus is relatively easier to optimize with a limited number of training samples.

Evaluation on Cityscapes. We also conduct another ablation study on the Cityscapes dataset and the results are shown in Table 2.7. We can mostly observe similar trend of the performance gain of the different baseline models as we already analyzed on the KITTI dataset.

The performance comparison of the baselines on this challenging dataset further confirms the advantage of the proposed approach. For the comparison of the model Half-Cycle + D and the model Full-Cycle + D, although the latter one achieves slightly worse results on the first two error metrics, it still produces clearly better performance on the remaining six evaluation metrics. Since there is no official evaluation protocol for depth estimation on this dataset, the results are evaluated with the protocol on the KITTI, and are directly evaluated on the disparity maps as they are directly proportional to each other. In Fig. 2.17, some qualitative comparison of the baseline models are presented.

2.4.4.3 State of the Art Comparison

In Table 2.6, we compare the proposed full model with several state-of-the-art methods, including the ones with the supervised setting, *i.e.* Saxena *et al.* (86), Eigen *et al.* (33), Liu *et al.* (35), AdaDepth (109), Kuznietzov *et al.* (102) and Xu *et al.* (36), and the ones with the unsupervised setting, *i.e.* Zhou *et al.* (92), AdaDepth (109), Garg *et al.* (89) and Godard *et al.* (90). Among all the supervised approaches, we have achieved very competitive performance to the best one of them (*i.e.* Xu *et al.* (36)), while ours is totally unsupervised without using any ground-truth depth data in training. For comparison with the unsupervised methods, we are also very close to the best competitor (*i.e.* Godard *et al.* (90)). AdaDepth (109) is the most technically related to our approach, which considers adversarial learning in a context of domain adaptation with extra synthetic training data. Ours significantly outperforms their results with both the supervised and unsupervised setting, further demonstrating the effectiveness of the means we considered and proposed for unsupervised depth estimation with the adversarial learning strategy. As far as we know, there are not quantitative results presented in the existing works on the Cityscapes dataset.

2.4.4.4 Analysis on the Time Aspect.

For the training of the whole network model, on a single Tesla K80 GPU, it takes around 45 hours on KITTI dataset with around 22k training images. For the running time, in our case with the resolution of 512×256, the inference of one image takes around 0.140 seconds, which is a near real-time processing speed.

2.4.5 Conclusions

In the current section we have presented a solution to a separate type of data quality scarcity, where depth map estimation systems are expensive to learn in a supervised manner. As such we present a novel approach for unsupervised deep learning for the depth estimation task using the adversarial learning strategy in a cycled generative network structure. The new approach provides a new insight that shows depth estimation can be effectively tackled via an unsupervised adversarial learning of the stereo image synthesis. More specifically, a generative deep network model is proposed to learn to predict the disparity map between two image views under a calibrated stereo camera setting. Two symmetric generative sub-networks are respectively designed to generate images from different views, and they are further merged to form a closed cycle which is able to provide strong constraint and supervision to optimize better the dual generators of the two sub-networks. Extensive experiments are conducted on two publicly available datasets (*i.e.* KITTI and Cityscapes). The results demonstrate the effectiveness of the proposed model, and show very competitive performance compared to state-of-the-arts on the KITTI dataset.

The future work would contain using attention mechanism to guide the learning of the feature representations of the generators, and also consider using the graphical models for structured prediction on the output disparity map to have predictions with better scene structures.

3

Low-shot Learning

In this chapter we explore methods of learning in a scenario where the quantity of available data is scarce, specifically when the accessible data is in a long-tail distribution. This scarcity of data cannot be mitigated with the methods described in Chapter 2, requiring techniques that explicitly learn in this data context, i.e. few-shot learning. We extend the generative few-shot learning approach first with learning a category-wise 3D model and sampling different perspectives from predicted instance-specific deformations to further boost generative diversity (Section 3.2), and second we make use of available textual data to condition further sample generation (Section 3.3). While both presented methods make use of additional information during training, it can either be produced using unsupervised tools used on the available samples, or in the case of associated textual information there is likely a much larger volume of it freely available in the wild.

3.1 Background and Related Work

In this section we briefly provide a broader context to the few-shot learning, how it is used to tackle a lack in data quantity, and review previous work considering the related topics of few-shot learning, 3D model learning and inference, self-paced learning, and multimodal learning.

Background Since the successful introduction of deep learning techniques, considerable research has been conducted to reduce the amount of annotated data needed for training such systems. For cases when the data restriction is a lack of quality, i.e. annotations, this problem has been approached systematically by developing algorithms which either require less

expensive annotations such as semi-supervised or weakly supervised approaches, or more rigorously no annotations at all such as unsupervised systems (Chapter 2), trading a usual loss in performance for much wider applicability.

More importantly, there exist situations where the availability of annotated data is heavily skewed, reflecting the tail distribution found in the wild, where previously discussed systems that generate proposals or low-level features are not helpful. Finetuning DNNs has been shown to be effective in a context where the big data assumption holds (113). However, scenarios where access is limited to only very few samples of novel data are extremely susceptible to over-fitting. In consequence, research in the domain of low-shot learning, i.e. learning and generalizing from only few training samples, has gained more and more interest (e.g. (114, 115, 116)).

Originally, few shot learning defined a scenario where the only very few samples per class were accessible (117, 118, 119). With the advent of deep learning the assumption was broadened, into having large amounts of data accessible for a number of *base* classes, with *novel* classes bound by a scarce data regime. This more realistic scenario falls under a meta-learning context, where a representation is learned on the base classes to be employed later on the novel classes.

To leverage the powerful representations that can be learned on the base classes with a DNN, a wide variety of meta-learning methods have been proposed. Santoro et al. (120) make use of a memory network to better assimilate new data and make predictions using it. Edwards et al. (121) aim to make use of learned dataset statistics to better fine-tune on new samples.
In contrast to more *model-driven* methods, (116) learn an embedding of the labelled examples over which an attention mechanism can be utilized, while (115) learns a mapping from the input to an embedding for which its class is represented by a prototype. Upon learning an embedding, both methods make use of a simple k-nearest neighbor approach to infer the class membership of unseen samples, implying that they can leverage the representational power of DNNs in a low data regime.
However, even when optimizing the learning process for the low-shot scenario, the lack of novel samples remains a hindrance. To mitigate this, a series of generative approaches have been developed, increasing the number of novel class samples that can be utilized during training. Hariharan et al. (122) facilitates training the classifier by generating features, disregarding realism or diversity criteria. While this approach provides a stable meta-learning process, and practically generates useful hallucinated samples, the diversity of generated samples is bound

by the samples used to learn the generator.

The current chapter contains two generative few-shot learning methods that increase sample diversity in different manners:
Section 3.2 details a work where a canonical, categorical 3D model is learned, after which predicted instance-wise deformations are used to sample novel viewpoints, increasing overall diversity. In contrast, **Section 3.3** makes use of abundant textual information to further diversify the generative process. The system directly generates cross-modal feature vectors, and features a real and generated feature combination strategy that allows for easy inference.

Few-Shot Learning For learning deep networks using limited amounts of data, different approaches have been developed in recent years. Following Taigman et al. (123), Koch et al. (118) interpreted this task as a verification problem, i.e. given two samples, it has to be verified, whether both samples belong to the same class. Therefore, they employed siamese neural networks (124) to compute the distance between the two samples and perform nearest neighbor classification in the learned embedding space. Some recent works approach few-shot learning by striving to avoid overfitting by modifications to the loss function or the regularization term. Yoo et al. (15) proposed a clustering of neurons on each layer of the network and calculated a single gradient for all members of a cluster during the training to prevent overfitting. The optimal number of clusters per layer is determined by a reinforcement learning algorithm. A more intuitive strategy is to approach few-shot learning on data-level, meaning that the performance of the model can be improved by collecting additional related data. Douze et al. (125) proposed a semi-supervised approach in which a large unlabeled dataset containing similar images was included in addition to the original training set. This large collection of images was exploited to support label propagation in the few-shot learning scenario. Hariharan et al. (122) combined both strategies (data-level and algorithm-level) by defining the squared gradient magnitude loss, that forces models to generalize well from only a few samples, on the one hand and generating new images by hallucinating features on the other hand. For the latter, they trained a model to find common transformations between existing images that can be applied to new images to generate new training data (126). Other recent approaches to few-shot learning have leveraged meta-learning strategies. Ravi et al. (114) trained a long short-term memory (LSTM) network as meta-learner that learns the exact optimization algorithm to train a learner neural network that performs the classification in a few-shot learning setting. This

method was proposed due to the observation that the update function of standard optimization algorithms like SGD is similar to the update of the cell state of a LSTM. Similarly, Finn et al. (127) suggested a model-agnostic meta-learning approach (MAML) that learns a model on base classes during a meta learning phase optimized to perform well when finetuned on a small set of novel classes. Moreover, Bertinetto et al. (14) trained a meta-learner feed-forward neural network that predicts the parameters of another, discriminative feed-forward neural network in a few-shot learning scenario. Another technique that has been applied successfully to few-shot learning recently is attention. (116) introduced matching networks for one-shot learning tasks. This network is able to apply an attention mechanism over embeddings of labeled samples in order to classify unlabeled samples. One further outcome of this work is that it is helpful to mimic the one-shot learning setting already during training by defining mini-batches, called few-shot episodes with subsampled classes. Snell et al. (115) generalize this approach by proposing prototypical networks. Prototypical networks search for a non-linear embedding space (the prototype) in which classes can be represented as the mean of all corresponding samples. Classification is then performed by finding the closest prototype in the embedding space. In the one-shot scenario, prototypical networks and matching networks are equivalent.

3D Shape Learning Inferring the 3D shape of an object from differing viewpoints has long been a topic of interest in computer vision. Based on the idea that there exists a categorical-specific canonical shape, and that class-specific deformations of it can be learned, systems such as SMPL (128) and "Keep it SMPL" (129) model a human 3D shape space, while Zuffi et al. (130) perform a similar task for quadruped animals. However, even though these methods are able to use synthetic training data, they still rely on a 3D shape ground truth. In contrast, Kanazawa et al. (131) make use of much cheaper keypoint and segmentation mask annotations, which allows both 3D mesh and texture inference for images.

Self-Paced Learning Recently, many studies have shown the benefits of organizing the training examples in a meaningful order (e.g., from simple to complex) for model training. Bengio et al. (132) first proposed a general learning strategy: curriculum learning. They show that suitably sorting the training samples, from the easiest to the most difficult, and iteratively training a classifier starting with a subset of easy samples (which is progressively augmented with more and more difficult samples), can be useful to find better local minima. Note that in this and in all the other curriculum-learning-based approaches, the order of the samples is provided

by an external supervisory signal, taking into account human domain-specific expertise.
Curriculum learning was extended to self-paced learning by Kumar et al. (133). They proposed the respective framework, automatically expanding the training pool in an easy-to-hard manner by converting the curriculum mechanism into a concise regularization term. Curriculum learning uses human design to organize the examples, and self-paced learning can automatically choose training examples according to the loss. Supancic et al. (53) adopt a similar framework in a tracking scenario and train a detector using a subset of video frames, showing that this selection is important to avoid drifting. Jiang et al. (134) pre-cluster the training data in order to balance the selection of the easiest samples with a sufficient inter-cluster diversity. Pentina et al. (135) propose a method in which a set of learning tasks is automatically sorted in order to allow a gradual sharing of information among tasks. In Zhang et al.'s (136) model saliency is used to progressively select samples in weakly supervised object detection. In context of visual categorization some of these self-paced learning methods use CNN-based features to represent samples (137) or use a CNN as the classifier directly (138).

Multimodal Learning Kiros et al (139) propose to align visual and semantic information in a joint embedding space using a encoder-decoder pipeline to learn a multimodal representation. Building upon this, Faghri et al (140) improve the mixed representation by incorporating a triplet ranking loss.
Karpathy et al (141) generate textual image descriptions given the visual data. Their model infers latent alignments between regions of images and segments of sentences of their respective descriptions. Reed et al (142) focus on fine-grained visual descriptions. They present an end-to-end trainable deep structured joint embedding trained on two datasets containing fine-grained visual descriptions.
In addition to multimodal embeddings, another related field using data from different modalities is text-to-image generation. Reed et al (143) study image synthesis based on textual information. Zhang et al (144) greatly improve the quality of generated images to a photo-realistic high-resolution level by stacking multiple GANs (StackGANs). Extensions of StackGANs include an end-to-end trainable version (144) and considering an attention mechanism over the textual input (145). Sharma et al. (146) extended the conditioning by involving dialogue data and further improved the image quality. Beside the usage of GANs for conditioned image generation, other work employed Variational Autoencoders (147) to generate images (148). However, they conditioned on attribute vectors instead of text.

Some works have leveraged multimodal data to improve classification results. Elhoseiny et al (149) collect noisy text descriptions and train a model that is able to connect relevant terms to its corresponding visual parts. This allows zero-shot classification for unseen samples, i.e. visual samples for novel classes do not exist. Similarly, Zhu et al. (150) train a classifier with images generated by a GAN given noisy text descriptions and test their approach in a zero-shot setup. Xian et al (12) follow this notion, however, generating feature vectors instead of images. In the context of few-shot learnig, Pahde et al (4) have leveraged textual descriptions to generate additional training images. Along with a self-paced learning strategy for sample selection this method improves few-shot learning accuracies.

3.2 Low-Shot Learning from Imaginary 3D Model[1]

3.2.1 Introduction

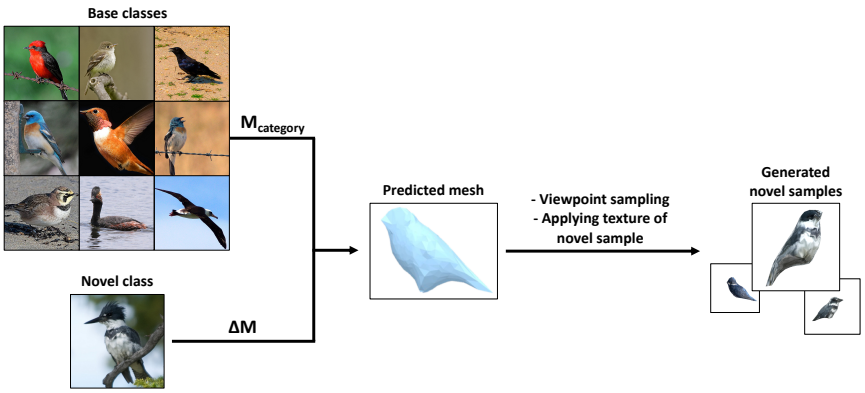


Figure 3.1: Generative method based on (131): We first learn a generic mesh of the bird category. This mesh is then altered to fit the appearance of the target bird. We rotate the predicted 3D mesh to capture various viewpoints resulting in many 2D images that resemble the target bird. Those meshs are then coated with the novel bird's texture. To cope with the varying quality, we subsequently apply a self-paced learning mechanism, which is elaborately outlined in figure 3.2 and in the remainder of the work. For the second approach to sample generation, we exploit the pose variety of the base birds visible on the top left to enhance diversity. This approach is visualized in Figure 3.3.

In this work we propose to maximize the visual generative capabilities to overcome quality and diversity issues commonly seen in generative few-shot learning approaches. Specifically,

[1] "Low-Shot Learning from Imaginary 3D Model" Frederik Pahde, Mihai Marian Puscas, Jannik Wolff, Tassilo Klein, Nicu Sebe, Moin Nabi; WACV 2019 (4)

we assume a scenario where the base classes have a large amount of annotated data whereas the data for novel categories are scarce. To alleviate the data shortage we employ a high quality generation stage by learning a 3D structure (131) of the novel class. A curriculum-based discriminative sample selection method further refines the generated data, which promotes learning more explicit visual classifiers.

Learning the 3D structure of the novel class facilitates low-shot learning by allowing us to hallucinate images from different viewpoints of the same object. Simultaneously, learning the novel objects' texture map allows us for a controlled transfer of the novel objects' appearance to new poses seen in the base class samples. Freely hallucinating w.r.t. different poses and viewpoints of a single novel sample then in turn allows us to guarantee novel class data diversity. The framework by Kanazawa et al. (131) has proven to be very effective for learning both 3D models and texture maps without expensive 3D model annotations. While reconstructing a 3D model from single images in a given category has been achieved in the past (151, 152), these methods lack easy applicability to a hallucinatory setup and specifically miss any kind of texture and appearance reconstruction. The intuition behind our idea is visualized in Fig. 3.1

With a broad range of images generated for varying viewpoints and poses for the novel class, a selection algorithm is applied. To this end, we follow the notion of *self-paced learning* strategy, which is a general concept that has been applied in many other studies (133, 138). It is related to curriculum learning (132), and is biologically inspired by the common human process of gradual learning, starting with the simplest concepts and increasing complexity. We employ this strategy to select a subset of images generated from the imaginary 3D model, which are associated with high confidence w.r.t. "class discriminativeness" by the discriminator.

The self-paced approach allows the method to handle the uncertainty related to the quality of generated samples. Here the notion of "easy" is interpreted as "high quality". Training is then performed using only the subset consisting of images of sufficient quality. This set is then in turn progressively increased in the subsequent iterations when the model becomes more mature and is able to capture more complexity.

The main contributions of this work are: **First**, we massively expand the diversity of generating data from sparse samples of novel classes through learning 3D structure and texture maps. **Second**, we leverage a self-paced learning strategy facilitating reliable sample selection.

Our approach features robustness and outperforms the baseline in the challenging low-shot scenario.

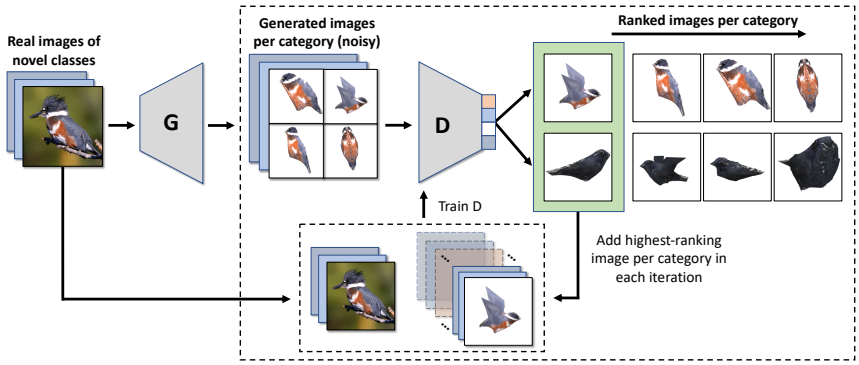


Figure 3.2: Self-paced fine-tuning on novel classes: For each novel class, noisy samples are generated with different viewpoints and poses by G. Those images are ranked by D based on their class-discriminatory power. The highest-ranking images are added to the novel samples and used to update D, which is trained using a simple cross-entropy loss. This process is repeated multiple times. Initially, D has been pre-trained on all base class data.

3.2.2 Method

3.2.2.1 Preliminaries

In this subsection we introduce the necessary notation.

Let $\mathcal{I}$ denote the image space, $\mathcal{T}$ the texture space , $\mathcal{M}$ the 3D mesh space and $\mathcal{C} = \{1, ..., L\}$ the discrete label space. Further, let $x_i \in \mathcal{I}$ be the i-th input data point, and $y_i \in \mathcal{C}$ its label. In the low-shot setting, we consider two subsets of the label space: $\mathcal{C}_{\text{base}}$ for labels for which we have access to a large number of samples, and the novel classes $\mathcal{C}_{\text{novel}}$, which are underrepresented in the data. Note that both subsets exhaust the label space $\mathcal{C}$, i.e. $\mathcal{C} = \mathcal{C}_{\text{base}} \cup \mathcal{C}_{\text{novel}}$. We further assume that in general $|\mathcal{C}_{\text{novel}}| \ll |\mathcal{C}_{\text{base}}|$.

The dataset $\mathcal{S}$ decomposes as follows: $\mathcal{S} = \mathcal{S}_{\text{train}} \cup \mathcal{S}_{\text{test}}$, $\mathcal{S}_{\text{train}} \cap \mathcal{S}_{\text{test}} = \emptyset$. The training data $\mathcal{S}_{\text{train}}$ consists of 2-tuples $\{(x_i, y_i)\}_{i=1}^{N}$ taken from the whole data set containing both image samples and labels. Furthermore, for 3D model prediction we also attach 3-tuples $\{(l_i, k_i, m_i)\}_{i=1}^{N}$, with l_i being a foreground object segmentation mask and k_i a 15-point keypoint vector representing the pose of the object. Additionally, m_i denotes the weak-perspective camera, which is estimated by leveraging structure-from-motion on the training instances' keypoints k_i. The test data is drawn from the novel classes and does not contain any 3D information, but solely images and their labels. Next, there is also $\mathcal{S}_{\text{train}}^{\text{novel}} = \{(x_i, y_i, l_i, k_i, m_i) : (x_i, y_i, l_i, k_i, m_i) \in \mathcal{S}_{\text{train}}, y_i \in \mathcal{C}_{\text{novel}}\}_{i=1}^{M} \subset \mathcal{S}_{\text{train}}$, which denotes the training data for

the novel categories. For each class in $\mathcal{C}_{\text{novel}}$, k samples can be used for training (k-shot), resulting in $\left|\mathcal{S}^{\text{novel}}_{\text{train}}\right| \ll \left|\mathcal{S}_{\text{train}}\right|$

3.2.2.2 3D Model Based Data Generation

The underlying observation on which our method is based on is that increased diversity of generated images directly translates into higher classification performance for novel categories. The proposed work aims at emulating processes in human cognition that allow for reconstructing different viewpoints and poses through conceptualizing a 3D model of an object of interest. Specifically, we aim to learn such a 3D representation for novel samples appearing during training and leverage it to predict different viewpoints and poses of that object.

We use the architecture proposed by Kanazawa et al. (131) to predict a 3D mesh M_i and texture T_i from an image sample x_i. With the assumption that all $x_i \in \mathcal{I}$ represent objects of the same category, the shape of each instance is predicted by deforming a learned category-specific mesh M_{cat}. Note that *category* refers to the entire fine-grained bird dataset, as opposed to *class*. All recovered shapes will share a common underlying 3D mesh structure, $M_i = M_{cat} + \Delta M_i$, with ΔM_i being the predicted mesh deformation for instance x_i. Because the mesh M has the same vertex connectivity as the average categorical mesh M_{cat}, and further as M_{sphere} representing a sphere, a predicted texture map T_i can be easily applied over any generated mesh.

An advantage of (131) over related methods is that learning the 3D representation does not require expensive 3D model or multi-view annotations.
Given (M_i, T_i, Θ_i) and $\Theta = (\alpha, \beta, \gamma)$, where the three camera rotation angles α, β, γ are sampled uniformly from $[0, \pi/6]$, we can project the reconstructed object using $f_{gen}(M_i, T_i, \Theta_i)$ such that $X_i^{view} = \{x_i^0, ..., x_i^L\}$ contains samples of the object seen from different viewpoints.

As X_i^{view} only contains different viewpoints of the novel object, it will not contain any novel poses. This is a concern for non-rigid object categories, where it cannot be guaranteed that the unseen samples in a novel class will have similar poses to the known samples in the novel class. To mitigate this, the diversity of the generated data must be expanded to include new object poses.

All meshes predicted from $x_j \in S_{base}$ obtain the spherical texture map T_i corresponding to $x_i \in \mathcal{S}^{\text{novel}}_{\text{train}}$ using $f_{gen}(M_j, T_i, \Theta_j)$. This transfers the shape from base class objects to novel class instances resulting in $X_i^{pose} = \{x_i^j, ..., x_i^S\}$.

Using poses from images of different labels is an inherently noisy approach through inter-class mesh variance. However, a subsequent sample selection strategy allows the algorithm to make use of the most representative poses. Indeed, as seen in Figure 3.3, meshes $M_j \in S_{base}$ exist for which the predicted images x_i^j are visually similar to samples of the unseen classes.

Thus, for each sample $x_i \in S_{train}^{novel}$, a set of images $S_{gen}^{novel} = X_i^{view} \cup X_i^{pose}$ is generated. This generated data captures both different viewpoints of the novel class and the appearance of the novel class applied to differing poses from the base classes.

3.2.2.3 Pre-Training of Classifier

In the low-shot learning framework proposed by Hariharan and Girshick (13), a representation of the base categorical data must be learned beforehand. This is achieved by learning a classifier on the samples available in the base classes, i.e. $x_i \in S_{train}^{base}$. For this task we make use of an architecture identical to the StackGAN discriminator (144), modified to serve as a classifier. This discriminator D is learned on S_{train}^{base} by minimizing L_{class} defined as a cross-entropy loss.

However, to accommodate for the different amount of classes in base and novel, D has to be adapted. Specifically, the class-aware layer with $|\mathcal{C}_{\text{base}}|$ output neurons is replaced and reduced to $|\mathcal{C}_{\text{novel}}|$ output neurons, which are randomly initialized. We refer to this adapted classifier as D'. Subsequently, the network can be fine-tuned using the available novel class data.

3.2.2.4 Self-Paced Learning

As seen in section 3.2.2.2, for a given novel sample $x_i \in \mathcal{S}_{\text{train}}^{\text{novel}}$ we can generate $S_{gen}^{novel} = X_i^{view} \cup X_i^{pose}$, containing new viewpoints and poses of the given object.

For the self-paced learning stage, we fine-tune with the novel samples, as well as the samples generated through projecting the predicted 3D mesh and texture maps. i.e. with the data given by $S_{train}^{novel} \cup S_{gen}^{novel}$.

Unfortunately, the samples contained in S_{gen}^{novel} can be noisy for a variety of reasons: failure in predicting the 3D mesh deformation due to a too large difference between the categorical mesh and the object mesh, or even viewpoints that are not representative to the novel class.

To mitigate this we propose a self-paced learning strategy ensuring that only the best generated samples within $\mathcal{S}_{\text{gen}}^{\text{novel}}$ are used.

Again taking into account the setting of low-shot learning, we restrict the number of samples per class available to k. Due to the limited amount of samples, the initialized D' will be weak on the classification task, but sufficiently powerful for performing an initial ranking of the generated images. For this task we employ the softmax activation for class-specific confidence scoring. As D' learns to generalize better, more difficult samples will be selected.

This entails iteratively choosing generated images that have highest probability in D' for $\mathcal{C}_{\text{novel}}$, yielding a curated set of generated samples $\mathcal{S}_{\text{gen}}^{\text{novel}}$. An issue in selecting the highest scoring sample in each iteration is the possibility of not making full use of the available data w.r.t. its diversity - the highest scoring images being of a very similar pose and viewpoint to the original sample.

We address this shortcoming by a using a clustering-and-discard strategy: For the novel class training sample x_i, we generate $X_i^{gen} = \{x_i^0, ...x_i^{L+S}\}$ new images, representing new viewpoints and poses of the object. X_i^{gen} is then further associated with $K_i^{gen} = \{k_i^0, ...k_i^Q\}$, representing all the predicted keypoints of the associated generated samples. K_i^{gen} is clustered using a simple k-means implementation (153). On every self-paced iteration, the pose cluster associated to the selected top-ranked sample is discarded to increase data diversity.

Finally, we aggregate original samples and generated images $\mathcal{S}_{\text{train}}^{\text{novel}} \cup \mathcal{S}_{\text{gen}}^{\text{novel}}$ for training, during which we update D'. Doing so yields both a more accurate ranking as well as higher class prediction accuracy as the number of samples increases.

3.2.3 Experiments

3.2.3.1 Datasets

We test the applicability of our method on CUB-200-2011 (154), a fine-grained classification datasets containing 11,788 images of 200 different bird species of size $\mathcal{I} \subset \mathbb{R}^{256\times256}$. The data is split equally into training and test data. As a consequence, samples are roughly equally distributed, with training and test each containing ≈ 30 images per class. Additionally, foreground masks, semantic keypoints and angle predictions are provided by (131). Note that nearly 300 images are removed where the number of visible keypoints is less or equal than 6.

Following Zhang et al. (144), we split the data such that $|C_{base}| = 150$ and $|C_{novel}| = 50$. To simulate low-shot learning, $k \in \{1, 2, 5, 10, 20\}$ images of C_{novel} are used for training, as proposed by (122).

	k				
Model	**1**	**2**	**5**	**10**	**20**
Baseline	27.55	30.75	54.25	58.51	71.62
Views + poses	33.40	43.72	54.81	65.27	74.06
SPL w/ views	33.54	41.49	54.88	65.48	**74.97**
SPL w/ poses	33.82	42.47	54.95	64.85	73.64
SPL w/ poses + clustering	33.40	45.05	57.74	65.69	74.62
SPL w/ poses + views	35.29	41.98	55.37	66.04	71.48
SPL w/ poses + views (balanced)	35.77	44.56	54.60	64.30	74.83
SPL w/ all	**36.96**	**45.40**	**58.09**	**66.53**	74.83

Table 3.1: Ablation study of our model in a top-5, 50-way scenario on the CUB-200-2011 dataset in different k-shot settings, best results are in bolt. We observe that each of the proposed extensions increases the accuracy in at least one setting which justifies their usage. This regards to both, methods for generating additional data and the approach to only select generated samples of sufficient quality for training the classifier.

3.2.3.2 Algorithmic Details

During representation learning, we train an initial classifier on the base classes for 600 epochs and use Adam (155) for optimization. We set the learning rate τ to 10^{-3} and the batch size for D to 32. In the initialization phase for self-paced learning, we construct D' by replacing the last layer of D by a linear softmax layer of size $|C_{novel}|$. The resulting network is then optimized using the cross-entropy loss function and an Adam optimizer with the same parameters. Batch size is set to 32 and training proceeds for 20 epochs. Self-paced learning of D' continues to use the same settings, i.e. the Adam optimizer minimizing a cross-entropy loss. In every iteration we choose exactly one generated image per class and perform training for 10 epochs.

3.2.3.3 Models

In order to asses the performance of individual components, we perform an ablation study.

The simplest transfer learning approach is making use of a pre-trained representation and then fine-tuning that model on the novel data. A first baseline (**Baseline**) uses this strategy: we pre-train a classifier D on the base classes, following by fine-tuning with k novel class

instances $x_i \in \mathbb{S}_{\text{train}}^{\text{novel}}$. This strategy makes use of the fine-grained character of the dataset, learning initial representations on $\mathbb{C}_{\text{base}}$ and performing classification on $\mathbb{C}_{\text{novel}}$.

A second model **views + poses** studies the validity of the generated viewpoint and pose data. For r sampling iterations, a single uniformly sampled $x_i \in S_{gen}^{novel}$ is attached to a novel sample set.

We then introduce sample selection to our method. Note that viewpoint generation is achieved through 3D Mesh M_i and texture T_i of the same sample x_i, while the different poses are generated through applying the novel class instance texture T_i to base class meshes M_j. The **SPL w/ views** and **SPL w/ poses** sample the generated data from the generated viewpoints X^{view} and X^{pose} respectively.

SPL w/ poses + views makes use of the entirety of S_{gen}^{novel}, while **SPL w/ poses + views (balanced)** tackles the data imbalance between different viewpoint samples and different pose samples by ranking the two branches separately, and selecting one sample from each such that for one novel sample, $x_i^{max,pose}$ and $x_i^{max,view}$ are used in fine-tuning.

The clustering-and-dismissal mechanism detailed in 3.2.2.4 is evaluated in the **SPL w/ poses + clustering** model, while **SPL w/ all** makes use of the method in its entirety.

3.2.3.4 Results of Ablation Study

The results of the ablation study outlined in the previous section are shown in Table 3.1, presenting 50-way, top-5 accuracies for k-shot learning with $k \in \{1, 2, 5, 10, 20\}$.

We first evaluate the baseline model, which is trained on the base classes and fine-tuned on the novel classes. Due to using a relatively shallow classification network, and the sparsity of the novel samples, the network rapidly overfits.

Introducing more data diversity to the fine-tuning stage through 3D model inference provides a significant boost in performance in all $k \in \{1, 2, 5, 10, 20\}$ With the generated samples selected randomly, the network does not easily overfit, but this selection method provides no protection against noisy generated samples.

Subsequent models evaluate different selection strategies across the two defined generated data splits for new viewpoints and poses, i.e. X^{view} and X^{pose}. The contribution of the self paced learning strategy can be evaluated directly comparing the top-5 accuracies of the **view + poses** model and the **SPL w/ views + poses** model. The increase of performance when k is small shows that the selection strategy can achieve better performance, but inconsistently across different k values.

One cause of this problem is how the generated data is split, and whether the classifier has access to the most valuable generated samples. In **SPL w/ poses** and **SPL w/ views**, we only select samples from X^{pose} and X^{view} respectively. The experimental results of both models are similar and inferior to **SPL w/ views + poses**, where both sets are used. Even with higher performance, the aggregate model selects from X^{view} almost exclusively, hinting on a type of mode collapse.

To further diversify the possible data picks, we "balance" the two sets: For each sample, $x_i^{max,pose}$ and $x_i^{max,view}$ are selected as the highest scoring samples in their respective sets. This disentangling of pose and viewpoint data offers an across-the-board improvement, as seen in **SPL w/ views + poses (balanced)**.

While normally each sample that was selected in self-paced iteration r is discarded, this will likely leave a number of samples that are similar in pose, such that the classifier may rank them as maximum. This does not add significant new information to the learning process, and as such the clustering-by-pose method guiding the sample dismissal is introduced. Indeed, as observed in **SPL w/ all**, both the sample-discard strategy, and the balancing strategy are similar useful for selections in self-paced learning. With all discussed techniques introduced, the model achieves a significant performance boost compared to the baseline.

3.2.3.5 Analysis of Self-Paced Fine-Tuning

We run several additional experiments to further analyze the behavior of our method. For the those experiments we use the CUB-200-2011 bird dataset, and compare to the method by Hariharan and Girshick (122) in Table 3.2.

Baseline	NN	Our (shallow)	Our (ResNet)	(122)
9.1	9.7	14.4	18.5	19.1

Table 3.2: Top-1, 50-way, 1-shot accuracies on the CUB-200-2011 dataset. We see that our shallow CNN (trained with self-paced learning) exceeds both baselines. The ResNet (not trained with self-paced learning) is within reach of Hariharan and Girshick's model with SGM loss (122), for which we have reproduced respective results.

We first report the baseline model in the top-1, 1-shot scenario. Due to the relative shallowness of the classification network and without any sample selection or hallucination, the performance is quite low.

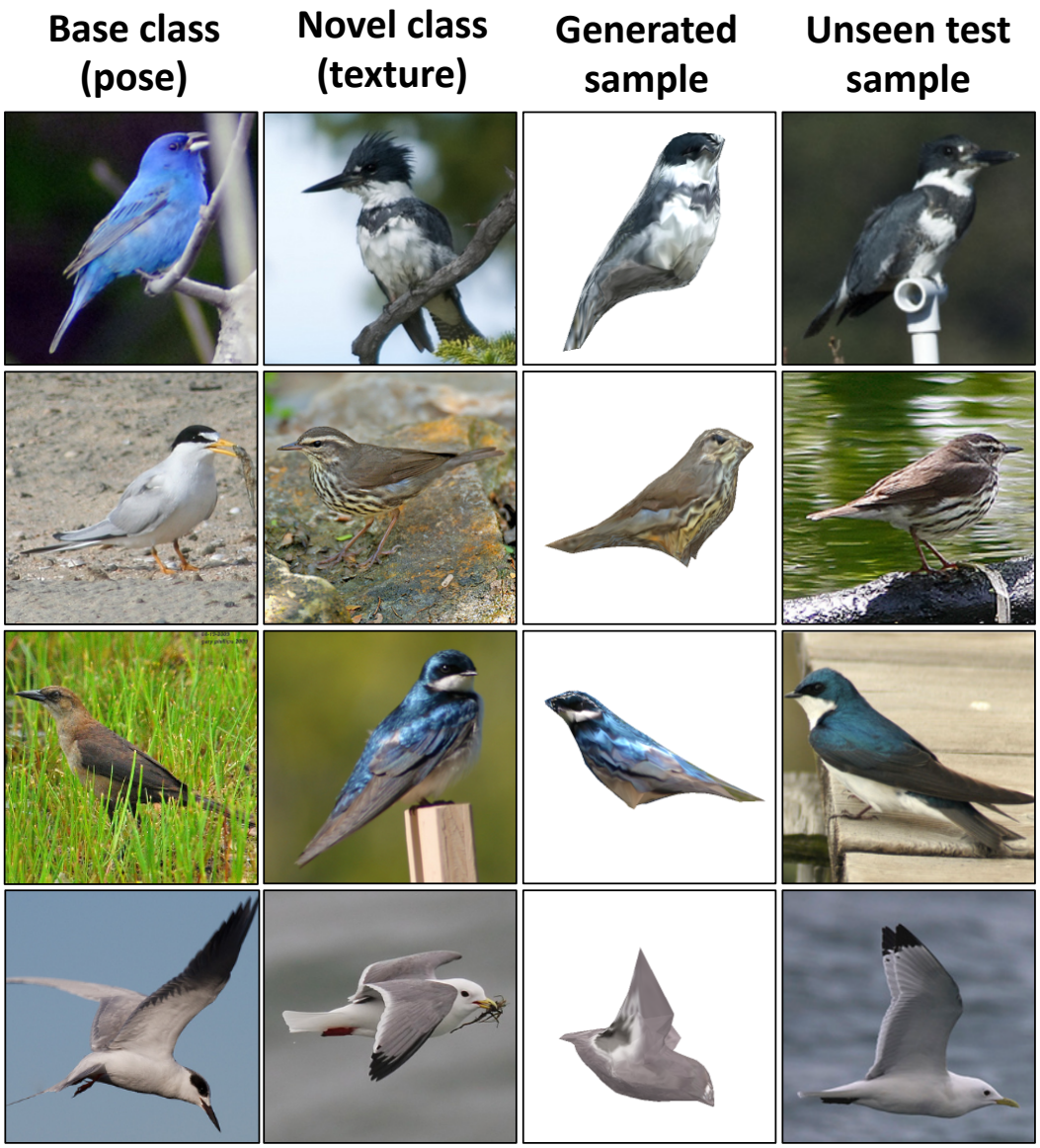


Figure 3.3: Texture from novel class birds is transferred onto poses from base class birds. The generated samples have been previously selected by the discriminator w.r.t. to their class-discriminatory power in the self-paced learning setting. Those hallucinations are visually similar to unseen test samples, indicating their value for training a classifier.

Methods using simple nearest neighbour classifiers can perform well on few-shot learning tasks (156). We implement a simple nearest-neigbour classifier using the representations learned in our baseline on the base class samples, $x_i \in S_{train}^{base}$, specifically making use of the

last hidden layer of the network. This model marginally outperforms the baseline.

Improving the novel class data diversity by using self-paced sample selection and k-means clustering-and-dismissal, the performance rises by 5.3 points to 14.4, which equals more than 50% relative improvement.

So far, we have used a classifier with simple architecture and loss function in order to present the most general possible framework and to allow for a fair comparison with baseline methods. However, we expect a significant boost in accuracy using larger classifiers. To test this hypothesis, we fine-tune a modified ResNet-18 (157). We first reduce the output dimensionality of the last pooling layer from 512 to 256 by lowering the amount of filters. After having trained this model on the base classes, we replace the last, fully-connected layer of size $|\mathcal{C}_{\text{base}}|$ with a smaller one of size $|\mathcal{C}_{\text{novel}}|$ to account for the different amount of classes. Afterwards, we freeze all layers except the final one, and train with $S_{train}^{novel} \cup S_{gen}^{novel}$ after having ranked the existing samples with the best shallow network. We observe comparable results to Hariharan and Girshick (122) despite of neither having used the ResNet-18 as a ranking function for self-paced learning, nor performing iterative sampling. Note that our method provides a general framework to augment the training set with class-discriminative generated samples that can potentially be used in conjunction with more sophisticated methods as the SGM loss (122) to obtain better results.

3.2.4 Conclusions

In this section we proposed to extend few-shot learning by incorporating image hallucination from 3D models in conjunction with a self-paced learning strategy. Experiments on the CUB dataset demonstrate that learning generative methods employing 3D models reaches performance that significantly outperforms our baseline and is competitive to popular methods in the field. Thus the proposed approach allows for an efficient compensation of the lack of data in novel categories. For future work we plan to optimize the pipeline in an end-to-end fashion, discarding the self-paced learning sample selection and replacing it with learnable viewpoint angle parameters.

A disadvantage of the proposed approach is the use keypoint annotations for the categorical 3D mesh generation. While these keypoint annotations are vastly less expensive than complete 3D maps of the object, and can be predicted by existing methods on a large variety of data, it is a direct trade between data quality and data quantity, leveraging existing low-level features to generate novel samples.

3.3 Multimodal feature generation for low-shot learning [2]

The key assumption of this section is that incorporating multimodal data can provide the means to inject diversity into the generation process by using generated cross-modal features to broaden the scope of the sample space, and that textual information in particular is likely to be available for scarce classes. If the method described in Section 3.2 trades data quality for data quantity, the current method trades data quantity in a 'rich' modality for quanitity in a 'poor' modality, a practical assumption when regarding textual information compared to visual data.

3.3.1 Introduction

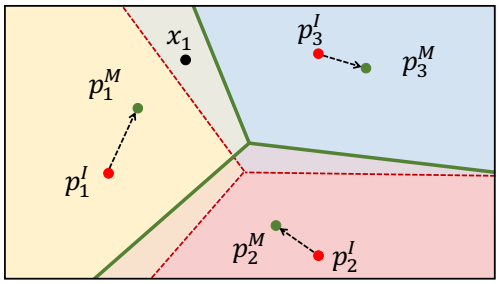


Figure 3.4: By leveraging cross-modally generated feature vectors we can condense the embedding space and move the single-modal prototypes p^I towards more reliable multimodal prototypes p^M improving the classification accuracy of unseen test samples in few-shot scenarios.

The most closely related work to this method is (4), which makes use of additional textual data in an adversarial context, followed by a self-paced selection of the most discriminative samples.

Our method builds upon the observation that the representations learned through DNNs are powerful enough for the use of simple non-parametric classification techniques (158), and that multi-modal data can improve generation diversity. To this end, an image encoder is first trained on the available base classes, after which a text-conditional GAN learns a cross-modal mapping between the textual and visual embedding spaces. This mapping can then be used to generate feature representations that reside in the visual space, conditioned by textual data. Intuitively, our method makes use of the cross-modal feature mapping to shift single-modal prototypes p^I (representing visual data) to p^M, mimicking unseen samples of the novel classes. This process

[2] "Adversarially Learned Feature Generating Network for Low-Shot Learning"; Frederik Pahde, Mihai Marian Puscas, Jannik Wolff, Tassilo Klein, Nicu Sebe, Moin Nabi, Under review, ICCV 2019

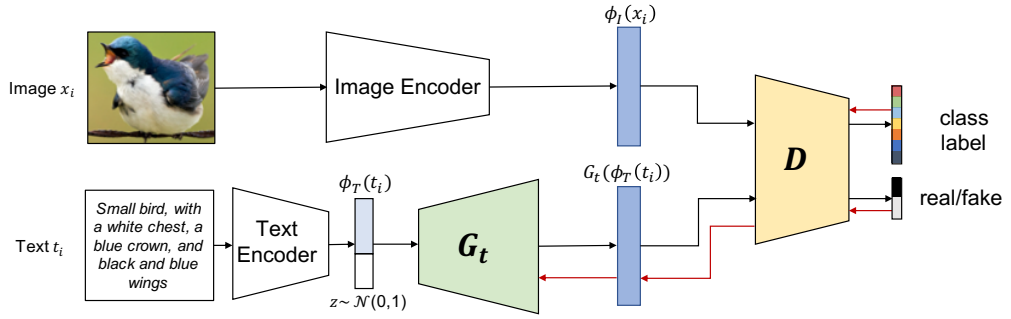


Figure 3.5: Architecture of our proposed feature-generating network for few-shot learning: The GAN framework containing a generator G_t and a discriminator D is optimized to transform a text embedding given a pre-trained text encoder into the visual embedding space φ yielded by a pre-trained image encoder. The discriminator computes a reconstruction loss (real/fake) and an auxiliary classification loss.

can be observed in Fig. 3.4, where a given sample x_i is classified differently though the shift in the prototypes. In a prototypical space, k-NN, a non-parametric classification technique is used, and thus only the representation learning stage requires multi-modal data, the inference stage requiring only visual data.

The main contributions of this work include the use of a cross-modal feature generation network in the context of few-shot learning. Furthermore, we suggest a strategy to combine real and generated features, allowing us to infer the class membership of unseen samples with a simple nearest neighbor approach. Our method outperforms our baselines and the state-of-the-art approaches for multimodal and image-only few-shot learning by a large margin for the CUB-200 and Oxford-102 datasets.

3.3.2 Method

To define our developed method we first introduce the necessary notation and then describe the architecture of our framework.

3.3.2.1 Preliminaries

Let $\mathcal{I}$ denote the image space, $\mathcal{T}$ the text space and $\mathcal{C} = \{1, ..., R\}$ be the discrete label space. Further, let $x_i \in \mathcal{I}$ be the i-th input data point, $t_i \in \mathcal{T}$ its corresponding textual description and $y_i \in \mathcal{C}$ its label. In the few-shot setting, we consider two disjunct subsets of the label space: $\mathcal{C}_{\text{base}}$ - labels for which we have access to sufficient data samples, and $\mathcal{C}_{\text{novel}}$ novel classes,

which are underrepresented in the data. Note that both subsets exhaust the label space $\mathcal{C}$, i.e. $\mathcal{C} = \mathcal{C}_{\text{base}} \cup \mathcal{C}_{\text{novel}}$. We further assume that in general $|\mathcal{C}_{\text{novel}}| < |\mathcal{C}_{\text{base}}|$.
We organize the data set $\mathcal{S}$ as follows. Training data $\mathcal{S}_{\text{train}}$ consists of tuples $\{(x_i, t_i, y_i)\}_{i=1}^{n}$ taken from the whole data set and test data $\mathcal{S}_{\text{test}} = \{(x_i, y_i) : y_i \in \mathcal{C}_{\text{novel}}\}_{i=1}^{m}$ that belongs to novel classes such that $\mathcal{S} = \mathcal{S}_{\text{train}} \cup \mathcal{S}_{\text{test}}$, $\mathcal{S}_{\text{train}} \cap \mathcal{S}_{\text{test}} = \emptyset$. Naturally, we can also consider $\mathcal{S}_{\text{train}}^{\text{novel}} = \{(x_i, t_i, y_i) : (x_i, t_i, y_i) \in \mathcal{S}_{\text{train}}, y_i \in \mathcal{C}_{\text{novel}}\}_{i=1}^{k} \subset \mathcal{S}_{\text{train}}$, where in accordance with a few-shot scenario $k = \left|\mathcal{S}_{\text{train}}^{\text{novel}}\right| \ll \left|\mathcal{S}_{\text{train}}\right| = n$. Additionally, in a few-shot learning scenario, the number of samples per category of $\mathcal{C}_{\text{base}}$ may be limited to g, denoted by $\mathcal{S}_{\text{train}}^{\text{novel}}(g)$. Note that contrary to the benchmark defined by Hariharan et al. (122), the few-shot learning scenario in this work is multimodal in training. However, the testing phase is single-modal on image data of $\mathcal{C}_{\text{novel}}$.

3.3.2.2 Nearest Neighbor in Visual Embedding Space

The classification in the embedding space is performed with a simple nearest neighbor approach. The assumption is that given a powerful feature representation, such as ResNet-18 feature vectors, nearest neighbor is a viable choice as classification model and has proven to outperform more sophisticated few-shot learning approaches (158). Therefore, we use the visual data from base classes $\mathcal{C}_{\text{base}}$ to train an image encoder Φ_I, providing a discriminative visual embedding space φ. For novel visual samples $x_i \in \mathcal{S}_{\text{train}}^{\text{novel}}$, $\Phi_I(x_i)$ then provides the embedding accordingly, featuring discriminativeness given by the pre-trained visual embedding space φ.
Following (115), for every novel class $k \in \mathcal{C}_{\text{novel}}$ we calculate a visual prototype p^k of all encoded training samples:

$$p^k = \frac{1}{|\mathcal{S}_{\text{train}}^{\text{k}}|} \sum_{(x_i, y_i) \in \mathcal{S}_{\text{train}}^{\text{k}}} \Phi_I(x_i), \tag{3.1}$$

where $\mathcal{S}_{\text{train}}^{\text{k}} = \{(x_i, y_i) \in \mathcal{S}_{\text{train}}^{\text{novel}} | y_i = k\}_{i=1}^{n}$ is the set of all training pairs (x_i, y_i) for class k. Classification of test samples is performed by finding the closest prototype given a distance function $d(\cdot)$. Thus, given a sample $x^{\text{test}} \in \mathcal{S}_{\text{test}}$ the class membership is predicted as follows:

$$c = \arg\min_k d(\Phi_I(x_{\text{test}}), p^k) \tag{3.2}$$

This assigns the class label of the closest prototype to an unseen test sample. Given the assumption that φ is a discriminative representation of visual data, Eq. 3.2 provides a powerful classification model. However, due to the few-shot scenario and the intrinsic feature sparsity in training space, $\mathcal{S}_{\text{train}}^{\text{novel}}$ is rather limited such that the computed class prototypes $\{p^k : k \in \mathcal{C}_{\text{novel}}\}$ consequentially yields merely a rough approximation of the true class mean.

Dataset	Method	1-shot	2-shot	5-shot	10-shot	20-shot
CUB	Pahde et al. (4)	57.67	59.83	73.01	78.10	84.24
	Image Only Baseline(Resnet-18+NN)	62.65±0.22	73.52±0.15	82.44±0.09	85.64±0.08	87.27±0.08
	ZSL Baseline (Generated Resnet-18+NN)	58.28±0.22	65.62±0.19	71.79±0.14	74.15±0.11	75.32±0.13
	Our Method (Multimodal Resnet-18 + NN)	**70.39±0.19**	**78.62±0.12**	**84.32±0.06**	**86.23±0.08**	**87.47±0.09**
Oxford-102	Pahde et al. (4)	78.37	91.18	92.21	-	-
	Image Only Baseline (Resnet-18+NN)	84.18±0.48	90.25±0.20	94.18±0.13	95.63±0.14	96.25±0.10
	ZSL Baseline (Generated Resnet-18+NN)	73.35±0.52	77.52±0.34	81.14±0.25	82.95±0.28	83.97±0.21
	Our Method (Multimodal Resnet-18 + NN)	**86.52±0.36**	**91.31±0.18**	**94.57±0.13**	**95.74±0.13**	**96.38±0.10**

Table 3.3: 50-way classification top-5 accuracy in comparison to other multimodal few-shot learning approaches and our baselines for CUB-200 and Oxford-102 datasets with $n \in \{1, 2, 5, 10, 20\}$

3.3.2.3 Cross-modal Feature Generation

A viable solution to enrich the training space to enable the calculation of more reliable estimations of the class prototypes is to leverage the multimodality in $\mathcal{S}_{\text{train}}^{\text{novel}}$. Thus, the core idea of our method is to use textual descriptions provided in the training data to generate additional visual feature vectors compensating the few-shot feature sparsity. Therefore, we propose to train a text-conditional generative network G_t that learns a mapping from the encoded textual description into the pre-trained visual feature space φ for a given training tuple (x_i, t_i, y_i) according to

$$G_t(\Phi_T(t_i)) \approx \Phi_I(x_i). \tag{3.3}$$

For the purpose of cross-modal feature generation we use a modified version of text-condtional generative adversarial networks (tcGAN) (143, 144, 145). The goal of tcGAN is to generate an image given its textual description in the GAN framework (159). More specifically, the tcGAN is provided with an embedding $\phi_T(\cdot)$ of the textual description. Therefore, a common strategy is to define two agents G and D solving the adversarial game of generating images that cannot be distinguished from real samples (G) and detecting the generated images as fake (D). Because our strategy is to perform nearest-neighbor classification in a pre-trained embedding

space φ, we slightly change the purpose of tcGAN. Instead of generating images $x_i \in \mathcal{I}$, we optimize G to generate its feature representation $\Phi_I(x_i)$ in the space φ. Generally, the representation vector in an embedding space has a lower dimensionality than the original image. Consequentially, the generation of feature vectors is a computational cheaper task compared to the generation of images.

To this end, our modified tcGAN can be trained by optimizing the following loss function,

$$\mathcal{L}_{tcGAN}\left(G_t, D\right) = \mathbb{E}_{x_i \sim p_{data}}\left[\log D\left(\Phi_I(x_i)\right)\right] + \mathbb{E}_{t_i \sim p_{data}, z}\left[\log D\left(G_t\left(\Phi_T(t_i), z\right)\right)\right], \quad (3.4)$$

which entails the reconstruction loss that is used for the traditional GAN implementation (159). Moreover, following (4, 150, 160) we define the auxiliary task of class prediction during the training of the tcGAN. This entails augmenting the tcGAN loss given in Eq. 3.4 with a discriminative classification term, which is defined as

$$\mathcal{L}_{class}\left(D\right) = \mathbb{E}_{C,I}\left[\log p\left(C \mid I\right)\right] \quad (3.5)$$

$$\text{and} \quad \mathcal{L}_{class}\left(G_t\right) \triangleq \mathcal{L}_{class}\left(D\right). \quad (3.6)$$

Augmenting the original GAN loss with the defined auxiliary term, the optimization objectives for D and G_t can now be defined as

$$\mathcal{L}\left(D\right) = \mathcal{L}_{tcGAN}\left(G_t, D\right) + \mathcal{L}_{class}\left(D\right) \quad (3.7)$$

$$\mathcal{L}\left(G_t\right) = \mathcal{L}_{tcGAN}\left(G_t, D\right) - \mathcal{L}_{class}\left(G_t\right), \quad (3.8)$$

which are optimized in an adversarial fashion. The adversarial nature of the task forces the generator to focus on the most class-discriminative feature elements. A visualization of our cross-modal feature generating method can be seen in Fig. 3.5.

3.3.2.4 Multimodal Prototype

Having learned a strong text-to-image feature mapping G_t we can employ the conditional network to generate additional visual features $G_t(\Phi_T(t_i))$ given an textual description t_i and a pre-trained text encoder $\Phi_T(\cdot)$. This allows for computing a prototype from generated samples $G_t(t_i)$ according to

$$p_T^k = \frac{1}{|\mathcal{S}^{\mathrm{k}}_{\mathrm{train}}|} \sum_{(t_i, y_i) \in \mathcal{S}^{\mathrm{k}}_{\mathrm{train}}} G_t(\Phi_t(t_i)). \quad (3.9)$$

Next, having both the true visual prototype p^k from Eq. 3.1 and a prototype p_T^k computed from generated feature vectors conditioned on textual descriptions from Eq. 3.9 a new joint prototype can be computed using a weighted average of both representations:

$$p^k = \frac{p^k + \lambda * p_T^k}{1 + \lambda}, \tag{3.10}$$

where λ is a weighting factor and $k \in \mathcal{C}_{\text{novel}}$ represents the class label of the prototype. Note that the step in Eq. 3.10 can be repeated multiple times, because G_t allows for the generation of a potentially infinite number of visual feature vectors in φ. The prediction of the class membership of unseen test samples can now be performed with Eq. 3.2 using the updated prototypes.

3.3.3 Experiments

Method	1-shot	5-shot
MAML (127)	38.43	59.15
Meta-Learn LSTM (114)	40.43	49.65
Matching Networks (116)	49.34	59.31
Prototypical Networks (115)	45.27	56.35
Metric-Agnostic Conditional Embeddings (161)	60.76	74.96
ResNet-18 (162)	66.54 $\pm$ 0.53	82.38 $\pm$ 0.43
ResNet-18 + Gaussian (162)	65.02 $\pm$ 0.60	80.79 $\pm$ 0.49
ResNet-18 + Dual TriNet (162)	69.61 $\pm$ 0.46	80.79 $\pm$ 0.49
Image Only Baseline (ResNet-18 + NN)	68.85 $\pm$ 0.86	83.93 $\pm$ 0.57
Our Full Method (Multimodal ResNet-18 + NN)	**75.01 $\pm$ 0.81**	**85.30 $\pm$ 0.54**

Table 3.4: Top-1 accuracies for the 5-way classification task on the CUB-200 dataset of our approach compared with single-modal state-of-the-art few-shot learning methods. We report the average accuracy of 600 randomly samples few-shot episodes including 95% confidence intervals.

To confirm the general applicability of our method we perform several experiments using two datasets. These experiments include comparisons to existing multimodal and single-modal state-of-the-art approaches for few-shot learning.

3.3.3.1 Datasets

We test our method on two fine-grained multimodal classification datasets. Specifically, we use the CUB-200-2011 (154) with bird data and Oxford-102 (163) containing flower data for our evaluation. The CUB-200 dataset contains 11,788 images of 200 different bird species, with $\mathcal{I} \subset \mathbb{R}^{256\times256}$. The data is split equally into training and test data. As a consequence, samples are roughly equally distributed, with training and test set each containing ≈ 30 images per category. Additionally, 10 short textual descriptions per image are provided by (142). Similar to (144), we use the text-encoder pre-trained by Reed et al. (142), yielding a text embedding space $\mathcal{T} \subset \mathbb{R}^{1024}$ with a CNN-RNN-based encoding function. Following (144), we split the data such that $|C_{base}| = 150$ and $|C_{novel}| = 50$. To simulate few-shot learning, $n \in \{1, 2, 5, 10, 20\}$ images of C_{novel} are used for training, as proposed by (122). We perform 50-way classification, such that during test time, all classes are considered for the classification task. In contrast, the Oxford-102 dataset contains images of 102 different categories of flowers. Similar to the CUB-200 dataset, 10 short textual descriptions per image are available. As for the CUB-200 dataset, we use the text-encoder pre-trained by Reed et al. (142), yielding a text embedding space $\mathcal{T} \subset \mathbb{R}^{1024}$. Following Zhang et al. (144), we split the data such that $|C_{base}| = 82$ and $|C_{novel}| = 20$. To simulate few-shot learning, $n \in \{1, 2, 5, 10, 20\}$ images of C_{novel} are used for training. Again, we perform classification among all available novel classes, yielding a 20-way classification task.

3.3.3.2 Implementation Details

Image Encoding For image encoding we utilize a slightly modified version of the ResNet-18 architecture (157). Specifically, we halve the dimensionality of every layer and add two 256-dimensional fully connected layers with LeakyRelu activation after the last pooling layer, followed by a softmax classification layer with $|\mathcal{C}_{\text{base}}|$ units. This network is trained on base classes using the Adam optimizer (155) for 200 iterations with learning rate 10^{-3}, which is decreased to 5×10^{-4} after 20 iterations. The last fully connected layer is employed as embedding space φ.

Cross-Modal Generation For the text-to-image feature mapping we use a tcGAN architecture inspired by StackGAN++ (144). In the generator G_t, following (144) the text embedding

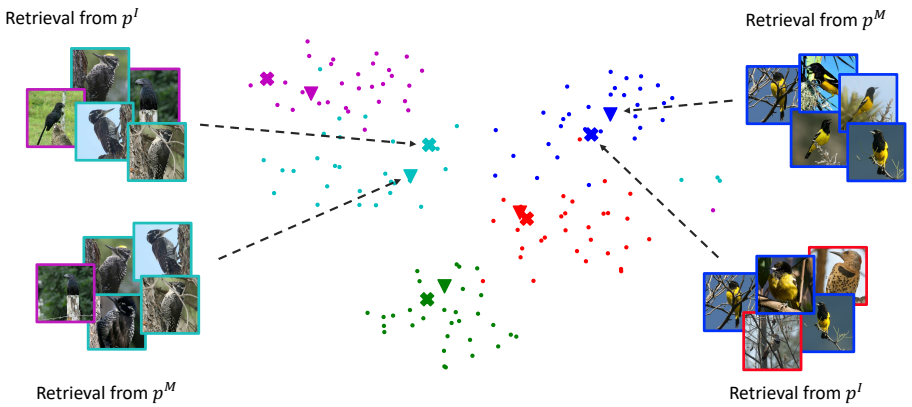


Figure 3.6: tSNE visualization of test samples (dots), prototypes p_I computed from only real image features (crosses) and multimodal prototypes p_M computed from real image features and generated features conditioned on text (triangles) in 5-way 1-shot scenario for CUB-200. The color indicates the class membership. Furthermore, we show the top-5 results for an image retrieval task for unseen images given the image-only prototype p_I and the multimodal prototype p_M. The color of the border indicates the class membership.

$\Phi_T(t_i)$ is first passed into a conditioning augmentation layer to condense the input space during training. This is followed by some upsampling convolutions, yielding a 256-dim output vector, equivalent to the dimensionality of the image feature space φ. Given the calculated feature vector $G_t(\Phi_T(t_i)$ and the original text embedding $\Phi_T(t_i)$, the discriminator D outputs a conditional and an unconditional loss (see (144)) along with the auxiliary classifcation loss. Adam is used to optimize both networks with learning rate 2×10^{-4} for 500 iterations. Having trained a feature generating network G_t, we compute $G_t(\Phi(t_i))$ for all 10 available textual descriptions per image and take the average in φ as its feature representation.

Classification We predict the class membership of test samples by calculating the nearest prototype in the embedding space φ (see Eq. 3.2). As distance function with use the cosine distance. To average visual and textual prototypes we set $\lambda = 1$ (see Eq. 3.10) and repeat this step 10 times, updating G_t in every iteration.

3.3.3.3 Results

For the evaluation, we test our approach in the 50-way classification task for CUB-200, and 20-way classification for Oxford-102. We designed a strong baseline, in which we predict the class label of unseen test samples by finding the nearest prototype in the the embedding space

φ, where the prototype p_I^k is computed exclusively using the limited visual samples (**image only**). Note that nearest neighbor classification is a powerful baseline in the context of few-shot learning, as similarly suggested by other works (158). Furthermore, we evaluate our method in a zero-shot setting, in which we generate feature vectors given the textual descriptions. The class-label of unseen test samples is predicted by computing the nearest prototype p_T^k containing exclusively generated features conditioned on the textual descriptions (**ZSL**). Our full method calculates the average of both prototypes (**multimodal**). We compare our method with (4), which to the best of our knowledge is the only existing work leveraging multimodal data in the context of few-shot learning. Because the classification results highly depend on the choice of samples available in a few-shot scenarios, we run the experiments 600 times following (115) and sample a random few-shot episode, i.e. a random choice of n samples per class in every iteration to cover randomness. We report the average top-5 accuracy including 95% confidence intervals in Tab. 3.3.

It can be observed that in every n-shot scenario we outperform our strong baselines and the other existing approach for multimodal few-shot learning. In the CUB-200 dataset, we outperform the baselines by a large margin, confirming our assumption that multimodal data in training is beneficial. For Oxford-102 the margins are lower, however, we still increase the classification results and outperform state-of-the-art results. Interestingly, our approach also stabilizes the results as the confidence intervals decrease compared to the baselines.

3.3.3.4 Comparison to Single-modal Methods

Due to the lack of existing approaches leveraging multimodal data for few-shot learning, we additionally compare our approach to existing methods using only image data during training. Outperforming these state-of-the-art image-only few-shot learning proves the beneficial impact of additional text data during training. Specifically, we compare our method with MAML (127), meta-learning LSTM (114), matching networks (116), prototypical networks (115) and metric-agnostic conditional embeddings (161). The results for CUB-200 for these methods are provided in (162). We also include theirs results in our comparison. However, their experimental setup differs slightly from our evaluation protocol. Instead of performing 50-way classification, the results in (162) are reported for 5-way classification in the 1- and 5-shot scenarios. This implies that in every few-shot learning episode, 5 random classes are sampled for which a classification task has to be solved, followed by the choice of n samples that are available per class. For the sake of comparability, we also evaluated our approach in

the same experimental setup. We repeat our experiment for 600 episodes and report average top-1 accuracy and 95% confidence intervals in Tab. 3.4.

It can be observed that even our image-only baseline, which performs nearest neighbor classification using prototypes in our modified ResNet-18 feature representation reaches state-of-the-art accuracies. Including multimodal data during training outperforms the other approaches in both 1- and 5-shot learning scenarios. This proves the strength of our nearest neighbor baseline and shows that enriching the embedding space φ with generated features conditioned on data from other modalites further improves the classification accuracies. In Fig. 3.6 we show a tSNE visualization of the embedding space φ including the image-only and multimodal prototypes p_I and p_M respectively in the 5-way classification task. The graph clearly shows some clusters indicating the class membership. It can be observed that the generated feature vectors shift the prototypes into regions where more unseen test samples can be classified correctly. Moreover, Fig. 3.6 shows retrieval results of unseen classes for p_I and p_M.

3.3.4 Analysis

In order to get a further in-depth understanding of certain aspects of our method, we performed some additional experiments analyzing its behavior. To this end, we use the CUB-200 dataset for the experiments in this section.

3.3.4.1 Reducing Textual Data

In a first experiment we want to analyze the importance of the amount of available textual descriptions. Note that for the experiments in Tab. 3.3 we used all 10 textual descriptions per image to generate a feature vector $G_t(\Phi(t_i)$. In this experiment we want to understand how the model behaves at reduced text availability. Therefore, in addition to limiting the amount of available images per novel class to n, we limit the amount of textual descriptions per image to $k \in \{1, 2, 5, 10\}$. We evaluate the classification accuracy for $n \in \{1, 2, 5\}$ with reduced number of textual descriptions. In Fig. 3.7 we show the relative accuracy gains for the different amount of texts compared to the image-only baseline. The x-axis shows the amount of texts and the y-axis the relative accuracy gain. It can be observed that the lower the amount of images n the higher is the accuracy gain given the text. The graphs show an increasing trend which indicates that the more texts are available the more the classification accuracies can be increased. This proves our assumption that enriching the embedding space φ is crucial to reach

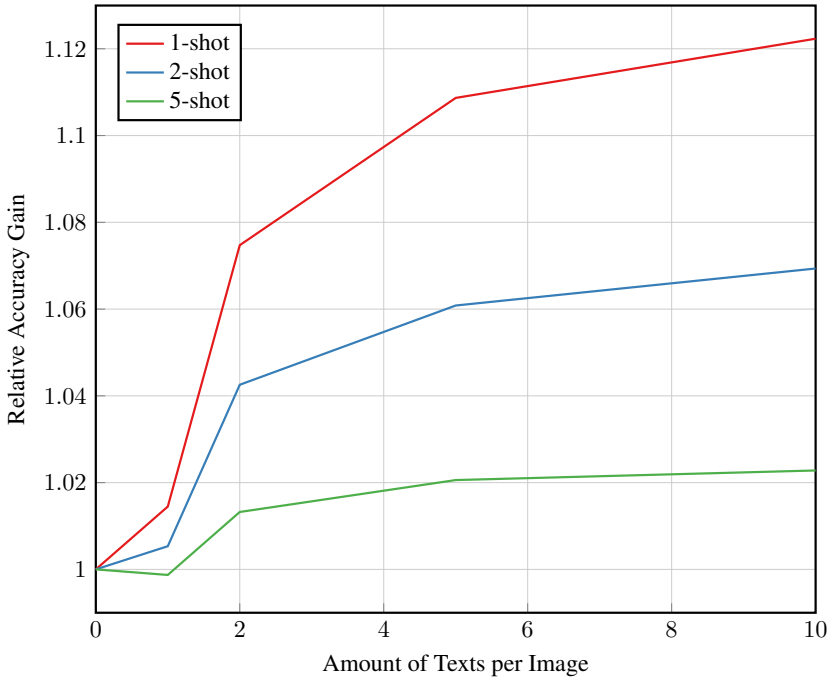


Figure 3.7: Relative top-5-accuracy gain for different amounts of available texts. The y-axis shows the accuracy gain in relation to the image-only baseline and the x-axis the amount of available texts per image k.

high classification results. Interestingly, in every n-shot scenario the second text leads to the highest accuracy gain. However, adding more text constantly improves the results and is never harmful to the model.

3.3.4.2 Impact of Prototype Shift

We investigate how the adjustment of a certain prototype impacts the classification performance. Therefore, we analyze the per-class accuracy gain in correlation with the shift of the prototype when exposed to multimodal data. The assumption we want to confirm whether large adjustments to the prototype go along with higher accuracy gain compared to classes for which the prototype remains almost unchanged. To this end, we measure change in prototype between the original image-only prototype p_I and the updated multimodal prototype p_M using the cosine distance denoted by $d(p_I, p_M)$. For every novel class, we analyze the correlation of the prototype update to the accuracy gain compared to the image-only baseline. In Fig. 3.8 we show the per-class accuracy gain for all prototypes in the 1-shot scenario. The x-axis shows the

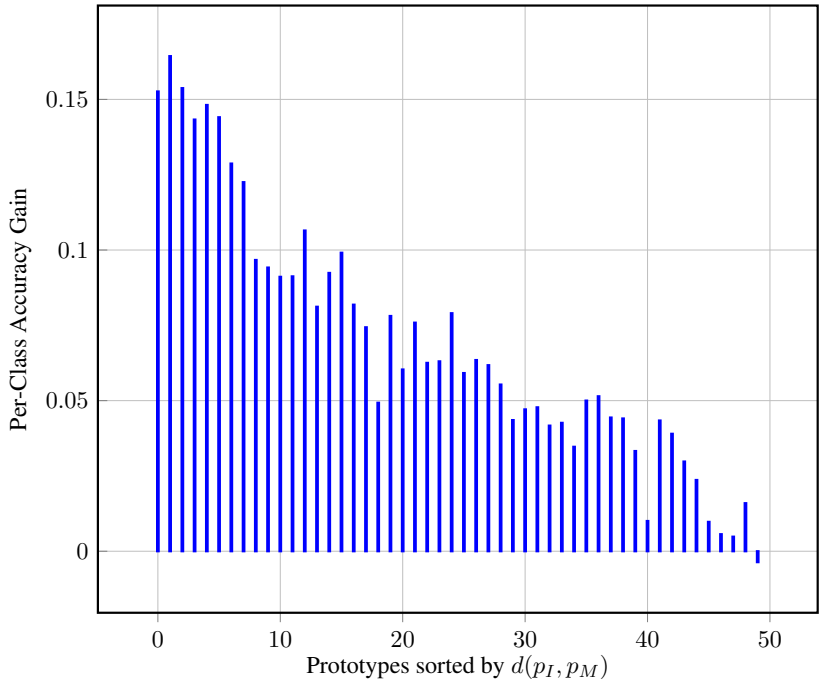


Figure 3.8: Per-class accuracy gain for prototypes after the adjustment with generated feature vectors. The x-axis shows the rank of the prototype sorted by $d(p_I, p_M)$ and the y-axis the top-5-accuracy gain for that particular class.

rank of the prototypes for all 50 novel classes of the CUB-200 dataset sorted by $d(p_M, p_I)$ in a descending order. The y-axis represents the accuracy gain for the certain prototype. We report top-5 accuracy and show the average of the result for 100 few-shot episodes. It can be observed that the more the prototype is changed (low rank) the higher is the accuracy gain for this particular class. On average, the most changed prototype leads to a per-class top-5 accuracy gain of ca. 15%. Smaller changes have a smaller impact on the classification performance and on average, adjusting the prototype with multimodal data is not harmful for the accuracy. This suggests that the multimodal features carry complimentary information that is used to simulate unseen novel class samples. At the same time it shows that the text-to-image feature mapping is well learned, as the most diverse, or farthest multimodal features net the largest performance gains.

3.3.5 Conclusions

In this section we tackled the few-shot learning problem from a multimodal perspective. We leveraged a nearest neighbor classifier in a powerful representation space. To mitigate the low population problem caused by the few-shot scenario we developed a cross-modal generation framework that is capable of enriching the visual feature space given data in another modality.

In contrast to section 3.2, this system does not require better quality data, but simply the use of a modality where there exists abundant information regarding the task at hand. In the case of textual data, this assumption can be justified when considering the volume of freely, and legally available information available on most if not all topics.

4

Catastrophic Forgetting

Previous chapters tackle cases where the data is restricted mechanically, where the required data simply does not exist (Chapter 3) or the annotations needed for learning are prohibitively expensive when broadly applying deep learning methods (Chapter 2). With evolving legislation regarding the access and storage of private data, any combination of quality and quantity restrictions can occur. Lifelong learning systems are particularly vulnerable to such restrictions - in the case where only data storage in restricted, operating in the strictly incremental "no look-back" framework amplifies the network's memory loss. In the following chapter we explore a framework designed to mitigate catastrophic forgetting for such an image classification system operating in a strictly class-incremental fashion.

4.1 Dynamic Generative Memory Network [1]

4.1.1 Introduction

Several recent approaches try to mitigate forgetting by simulating synaptic plasticity in DNNs (164, 165, 166, 167). Common to all these methods is the idea of discouraging updates of the network parameters that keep old knowledge when learning new tasks. In this regard, Serrà et al. (168) propose to rely on a hard attention to the task (HAT) mechanism. HAT finds parameter subspaces for all tasks while allowing them to mutually overlap. The optimal solution is then found in the corresponding parameter subspace of each task. It is noteworthy that all methods above tackle the task-incremental scenario, i.e. a separate classifier (with a separate output

[1] "Learning to Remember: A Synaptic Plasticity Driven Framework for Continual Learning"; Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Moin Nabi, CVPR 2019, (5)

layer per task) is learned to make predictions about each task. This further implies the availability of oracle knowledge of the task label at inference time.Such evaluation is often refereed to as multi-head evaluation in which the task label is associated with a dedicated output head. Alternatively, other approaches rely on single-head evaluation (166, 169). Here, the model is evaluated on all classes observed during the training, no matter which task they belong to. While single-head evaluation does not require oracle knowledge of the task label, it also does not reduce the output space of the model to the output space of the task. Thus single-head evaluation represents a harder, yet more realistic setup. Single-head evaluation is predominantly used in class-incremental setup, in which every newly introduced data batch contains examples of one to many new classes.

As opposed to task-incremental setup, models in class incremental setup typically require the replay of samples from previously seen categories when learning new ones. (166, 169, 170) show that such a replay based on real samples of previous tasks significantly alleviates the problem of catastrophic forgetting in a class-incremental situation. Yet, retaining samples has several intrinsic implications. First, it is very much against the notion of bio inspired design as the brain by no means features the retrieval of raw information identical to originally exposed impressions (171). Second, as pointed out by (172) and (169) storing raw samples of previous data can violate data privacy and memory restrictions of real world applications. Such restrictions are particularly relevant for the vision domain with its continuously growing dataset sizes and rigorous privacy constraints.

In this work, we address the "strict" class incremental setup. We demand a classifier to learn from a stream of data with different classes accruing at different times with no access to previously seen data, i.e. no storing of real samples is allowed. Such a scenario is solely addressed by methods that rely on generative memory - a generative network is used to memorize previously seen data distributions, samples of which can be replayed to the classifier at any time. This largely transfers the forgetting problem from the class discriminator to the generator. Several strategies exist to avoid catastrophic forgetting in generative networks. The most successful approaches make use of deep generative replay (DGR) (173) - repetitive training from-scratch of the generator on a mix of synthesized samples (of previous tasks) and new real samples every time a new task or class is introduced.

Another important factor in the continual learning setting is the ability to scale, i.e. to maintain sufficient capacity to accommodate for a continuously growing number of tasks. Given invariant resource constraints, it is inevitable that with a growing number of tasks to learn,

the model capacity is depleted at some point in time. This issue is again exacerbated when simulating neural plasticity with hard attention mechanisms such as parameter level attention masking. That is because weights blocked for previous tasks can be reused but not changed during subsequent learning, continuously reducing the degree of freedom of the network. In order to ensure sufficient degrees of freedom for every new task, we keep the number of "free" weights (i.e. weights that can be changed) constant by expanding the network with exactly the number of parameters what were blocked for the previous task.

Our contribution is twofold: **(a)** we introduce Deep Generative Memory (DGM) - an adversarially trainable generative network that features neuronal plasticity through efficient learning of a sparse attention masks for the network weights (DGMw) or layer activations (DGMa); To best of our knowledge we are the first to introduce weight level masks that are learned simultaneously with the base network; Furthermore, we conduct it in an adversarial context of a generative model; DGM is able to incrementally learn new information during normal adversarial training *without the need to replay previous knowledge*. **(b)** We propose an adaptive network expansion mechanism, facilitating resource efficient continual learning. In this context, we compare the proposed method to state-of-the-art approaches for continual learning. Finally, we demonstrate that DGMw accommodates for higher efficiency, better parameter re-usability and slower network growth then DGMa.

4.1.2 Related Work

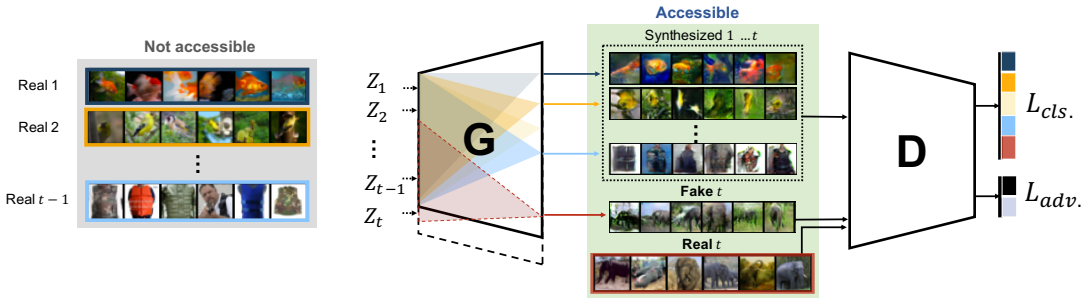


Figure 4.1: Dynamic Generative Memory: classification output of D is trained on the real samples of the current task t and synthesized sample of previously seen tasks $1...t-1$. Adversarial training is accomplished with real and fake samples of current task. A connection plasticity in the generator is learned simultaneously with the weights of the base network.

Among the first works dealing with catastrophic forgetting in the context of lifelong learn-

ing are (174, 175, 176). In contrast to the proposed approach, these methods tackle this problem by employing shallow neural networks, whereas our method makes use of modern deep architectures. In this regard, lately a wealth of works dealing with catastrophic forgetting in context of deep neural networks have appeared in the literature, see e.g., (164, 165, 167, 168, 177). However, all of these methods have been proposed for a "task-incremental learning" setup, where a sequence of disjoint tasks is learned one after the other by a single network. In our work we specifically propose a method to overcome catastrophic forgetting within the "class-incremental learning" setup. The key difference in the task-incremental setup is that the model learns a separate classifier for each task (i.e., multi-head), whereas in the latter the model learns only a single classifier for all of the observed classes of all tasks (i.e., single-head). Notably, a method designed for class-incremental learning can be generally applied in a task-incremental setup, whereas a task incremental learner is generally limited to task incremental situations.

Several continuous learning approaches (169, 170, 178), address catastrophic forgetting in the class-incremental setting, i.e. by storing raw samples of previously seen data and making use of them during the training of subsequent tasks. Thus, iCarl (169) proposes to find m most representative samples of each class whose mean feature space most closely approximates the entire feature space of the class. The final classification task is done by the means of the nearest mean-of-exemplars classifier.

Recently, there has been a growing interest in employing deep generative models for memorizing previously seen data distributions instead of storing old samples. Thus (173, 179) rely on the idea of generative replay, which requires retraining the generator from scratch at each time step on a mixture of synthesized images of previous classes and real samples from currently available data. However, this approach suffers from a number of shortcomings. Apart from being inefficient for training, it is severely prone to "semantic drifting". Namely, the quality of images generated at every memory replay point highly depends on the images generated during previous replays, which can result in loss of quality over time. In contrast to the methods described above, we propose to utilize a single generator that is able to incrementally learn new information during the normal adversarial training without the need to replay previous knowledge. This is achieved through efficiently learning a sparse mask for the units of the generator network.

Similar to our method, (180) proposed to avoid retraining the generator at every time-step on the previous classes by applying EWC (164), e.g. selective per-parameter regularization in the generative network. We pursue a similar goal with the key difference of utilizing a hard

attention mechanism similar to the one described by (168, 181, 182). All three approaches make use of the techniques originally proposed in the context of binary-valued networks (183). Herein, binary weights are specifically learned from a real values embedding matrices that are passed through a binarization function (e.g. sigmoid). To this end, (181, 182) learn to mask a pre-trained network without changing the weights of the base networks, whereas (168) (HAT) features binary mask-learning simultaneously with training the main network. While DGMa features HAT-like layer activation masking, DGMw accomplishes binary mask learning directly on the weights of the generator network simultaneously to the adversarial training.

Similarly to (184), we propose to expand the capacity of the employed network, in our case the samples generator. Expansion is performed dynamically with increasing amount of attained knowledge. However, (184) propose to keep track of the *semantic drift* in every neuron, and then expand the network by duplicating neurons that are subject to sharp changes. In contrast, we compute weightsâĂŹ importance concurrently during the course of network training by modeling the neuron behavior using an explicit binary mask. As a result, our method explicitly does not require any further network retraining after adding new capacity.

Other approaches like (178, 185) try to explicitly model short and long term memory with separate networks while transferring the knowledge from the former one to the later in a separate "sleeping" phase. In contrast to these methods our approach does not explicitly keep two separate memory locations, but rather incorporates it implicitly in a single memory network. Thus, the memory transfer occurs during the binary mask learning from non-binary (short term) to completely binary (long term) values.

4.1.3 Preliminaries

Adopting the notation of (166), let $S_t = \{(x_i^t, y_i^t)\}_{i=1}^{n^t}$ denote a collection of data belonging to the task $t \in T$, where $x_i^t \in \mathcal{X}$ is the input data and $y_{\boldsymbol{i}}^{\boldsymbol{t}} \in y^{\boldsymbol{t}}$ are the ground truth labels. While in the (standard) non-incremental setup the entire dataset $S = \cup_{t=0}^{|T|} S_t$ is available at once, in an incremental setup it becomes available to the model in chunks S_t specifically only during the learning of task t. Thereby, S_t can be composed of a collection of items from different classes, or even from a single class only. Furthermore, during test time the output space covers all the labels observed so far featuring the single head evaluation: $\mathcal{Y}^t = \cup_{j=1}^{t} y^j$.

4.1.4 Dynamic Generative Memory

Rationale. We consider a continual learning setup, in which a task solving model D has to learn its parameters θ^D from the data S_t being available at the learning time of task t. Task solver D should be able to maintain good performance on all classes $\mathcal{Y}^t$ seen so far during the training. A conventional ANN, while being trained on S_t, would adapt its parameters in a way that exhibits good performance solely on the labels of the current task y^t, the previous tasks would be forgotten. To overcome this, we introduce a Generative Memory component G, who's task is to memorize previously seen data distributions. As visualized in Fig. 4.1, samples of the previously seen classes are synthesized by G and replayed to the task solver D at each step of continual learning to maintain good performance on the entire $\mathcal{Y}^t$. We train a generative adversarial network (GAN)(186) *and* a sparse mask for the weights of its generator simultaneously. The learned masks model connection plasticity of neurons, thus avoiding overwriting of important units by restricting SGD updates to the parameter segments of G that exhibit free capacity.

Learning a binary mask. We consider a generator network G_{θ^G} consisting of L layers, and a discriminator network D_{θ^D}. In our approach, D_{θ^D} serves as both: a discriminator for generated fake samples of the currently learned task and as a classifier for the actual learning problem (AC-GAN (160) architecture). The system has to continually learn T tasks. During the SGD based training of task t, we learn a set of binary masks $M^t = [m_1^t, ..., m_L^t]$ for the weights of each layer. Output of layer l is obtained by combining the binary mask m_l^t with the layer weights, e.g.

$$y_l^t = \sigma_{act}[(m_l^t \circ W_l)^\top x], \quad W_l \in \mathbb{R}^{p \times n}, \tag{4.1}$$

for l being fully connected and σ_{act} some activation function. W_l is the weight matrix for connections between layer l and $l-1$, and $\cdot \circ \cdot$ corresponds to the Hadamard product of matrices. In DGMw m_l^t is shaped identically to W_l, whereas in case of DGMa the mask m_l^t is shaped as $1 \times n$ and should be expanded to the size of W_l. Extension to more complex models such as e.g. CNNs is straightforward.

A single binary mask for the generator's layer l and task t is given by:

$$m_l^t = \sigma(s \cdot e_l^t), \tag{4.2}$$

where e_l^t is a real-valued mask embeddings matrix, s is a positive scaling parameter $s \in \mathbb{R}_+$, and σ a thresholding function $\sigma : \mathbb{R} \to [0, 1]$. We use the sigmoid function as a pseudo step-function in order to ensure gradient flow to the embeddings e. In training of DGMw, we anneal the scaling parameter s incrementally during epoch i from 0 to s^i_{max} (local annealing). s^i_{max} is similarly adjusted over the course of I epochs from 0 to s_{max} (global annealing with s_{max} being a fixed meta-parameter) following the scheme largely adopted from (168):

$$s^i_{max} = \frac{1}{s_{max}} + (s_{max} - \frac{1}{s_{max}})\frac{i-1}{I-1} \tag{4.3}$$

$$s = \frac{1}{s^i_{max}} + (s^i_{max} - \frac{1}{s^i_{max}})\frac{b-1}{B-1}. \tag{4.4}$$

Here $b \in \{1, \ldots, B\}$ is the batch index and B the number of batches in each epoch of SGD training. DGMa only features global annealing of s as it showed better performance.

In order to prevent the overwriting of the knowledge related to previously seen classes in the generator network, gradients g_l w.r.t. the weights of each layer l are multiplied by the reverse of the accumulated binary masks for all previous tasks:

$$g'_l = [1 - m_l^{\leq t}]g_l, \quad m_l^{\leq t} = \max(m_l^t, m_l^{t-1}), \tag{4.5}$$

where g'_l corresponds to the new weights gradient and $m_l^{\leq t}$ is the accumulated mask.

Similarly to (168), we promote sparsity of the binary mask by adding a regularization term R^t to the loss function L_G of the AC-GAN(160) generator:

$$R^t(M^t, M^{t-1}) = \frac{\sum_{l=1}^{L-1}\sum_{i=1}^{N_l} m^t_{l,i}(1 - m^{<t}_{l,i})}{\sum_{l=1}^{L-1}\sum_{i=1}^{N_l} 1 - m^{<t}_{l,i}}, \tag{4.6}$$

where N_l is the number of parameters of layer l. Here, parameters that were reserved previously are not penalized, promoting reuse of weight units over reserving new ones as new tasks are learned.

Dynamic network expansion. As discussed by (182), significant domain shift between tasks leads to rapid network capacity exhaustion, ultimately manifesting in catastrophic forgetting. This can be explained by decreasing sparsity of the accumulated mask $m_l^{\leq t}$ over the course of training. To avoid this effect, we take measures to ensure stationary sparsity of the masks in each training cycle t. Consider network layer l with input vector of size p and output vector of size n. At the beginning of the initial training cycle, the binary mask $m_l^1 \in [0, 1]^{p \times n}$ is initialized with zero sparsity. Thus, all neurons of the layer will be used, with all values of the mask m_l^1 set to 0.5 (real-valued embeddings e_l^1 are initialized with 0).

	Method	MNIST (%)		SVHN(%)		CIFAR10(%)		ImageNet-50(%)	
		A_5	A_{10}	A_5	A_{10}	A_5	A_{10}	A_{30}	A_{50}
	JT	97.66	98.10	85.30	84.82	82.20	64.20	57.35	49.88
Episodic memory	iCarl-S (169)	-	55.8	-	-	-	-	29.38	28.98
	EWC-S(164)	-	79.7	-	-	-	-	-	-
	RWalk-S(166)	-	82.5	-	-	-	-	-	-
	PI-S (165)	-	78.7	-	-	-	-	-	-
Generat. memory	EWC-M (180)	70.62	77.03	39.84	33.02	-	-	-	-
	DGR (173)	90.39	85.40	61.29	47.28	-	-	-	-
	MeRGAN (179)	98.19	97.00	80.90	66.78	-	-	-	-
	DGMw (ours)	98.26	96.33	80.37	67.05	**64.94**	**51.7**	**29.67**	**17.32**
	DGMa (ours)	**99.17**	**98.14**	**84.18**	**68.36**	62.50	50.80	25.93	15.16

Table 4.1: Comparison to the benchmark presented by (166) (episodic memory with real samples) and (179) (generative memory) of approaches evaluated in class incremental setup. Joint training (JT) represents the the performance of the discriminator trained in non-incremental fashion. Both variants of our method are evaluated.

After the initial training cycle with the sparsity regularizer R^1, the number of free weight parameters not reserved by the mask will decrease to $np - \delta_1$, with δ_t corresponding to the number of parameters reserved for the generation task t ($t = 1$ here). After training cycle t of DGMw, we expand layer l's number of output weights n by δ_t/p. The free capacity of the layer is kept constant: $(n + \delta_t/p)p - \delta_t = np$.

In practice we extend the number of output units n by $\lceil \delta_t/p \rceil$. The number of free parameters is thus either np, if $\delta_t/p \in \mathbb{Z}$, or $np + p$, otherwise.

Joint training. The proposed system combines learning of three tasks that have to be learned jointly: A generative, a discriminative and finally a classification task in the strictly incremental class setup.

As such, using task labels as conditions, the generator network must learn from a training set $X_t = \{X_1^t, ..., X_N^t\}$ to generate images for task t. To this end, AC-GAN's conditional generator synthesizes images $x_t = G_{\theta^G}(t, z, M_t)$, where θ^G represents the parameters of the generator network, z denotes a random noise vector.

The discriminator network is used to perform two tasks. First, a discriminative task, determining whether sample x_t is real or fake. Second, a classification task, indicating which of the

labels $\mathcal{Y}^t = \cup_{j=1}^{t} y^j$ seen so far can be associated with sample x_t. To achieve both, the final layer of the network has two branches corresponding to each task. The parameters corresponding to each task are optimized in an alternating fashion. As such, the generator optimization problem can be seen as minimizing $\mathcal{L}_{\mathcal{G}} = \mathcal{L}_s^t - \mathcal{L}_c^t + \lambda_{RU} R^t$, with $\mathcal{L}_c$ a classification error on the auxiliary output, $\mathcal{L}_s$ a discriminative loss function used on the binary output layer of the network, and R^t the regularizer term expanded upon in Equation 4.6.

To promote efficient parameter utilization, taking into consideration the amount of the network already in use, the regularization weight λ_{RU} is multiplied by the ratio $\alpha = \frac{S_t}{S_{free}}$, where S_t is the size of the network before training the task t, and S_{free} is the number of free neurons. This ensures that less parameters are reused during early stages of training, and more during the later stages when the model already has gained a certain level of maturity. The sensitivity of λ_{RU} is investigated in Sec. 4.1.5.4.

The discriminator is optimized similarly through minimizing $\mathcal{L}_{\mathcal{D}} = \mathcal{L}_c^t - \mathcal{L}_s^t + \lambda_{GP} \mathcal{L}_{gp}^t$, where $\mathcal{L}_{gp}^t$ represents a gradient penalty term implemented as in (187), to ensure a more stable training process.

4.1.5 Experimental Results

In the following section we provide a qualitative and quantitative evaluation of our method on a number of publicly available datasets. Furthermore, we provide a discussion upon the performance of the different components.

4.1.5.1 Experiments

We perform experiments measuring the classification accuracy of our system in a strictly class incremental setup on the following benchmark datasets: MNIST (188), SVHN (189), CIFAR-10 (190), and ImageNet-50 (191). Similarly to (166, 169, 179) we report an average accuracy (A_t) over the held-out test sets of classes $0...t$ seen so far during the training.

Datasets: The MNIST and SVHN datasets are composed of 60000 and 99289 images respectively, containing digits. The main difference is in the complexity and variance of the data used. SVHN's images are cropped photos containing house numbers and as such present varying viewpoints, illuminations, etc. CIFAR10 contains 60000 labelled images, split in 10 classes, roughly 6k images per class. Finally, we use a subset of the iILSVRC-2012 dataset

containing 50 classes with 1300 images per category. All images are further resized to 32 x 32 before use.

Architectures: We make use of the same architecture for the MNIST and SVHN experiments, a 3-layer DCGAN (192), with the generator's number of parameters modified to be proportionally smaller (approx. 50%) than the architecture reported in (179). The projection and reshape operation is further performed with a convolutional layer instead of a fully connected one. For the CIFAR-10 experiments, we use the CIFAR-10 ResNet architecture proposed by (192). For the ImageNet-50 benchmark the discriminator features ResNet-18 architecture. Note that all architectures used have been modified to function as an AC-GAN (160).

All datasets are used to train a classification network in an incremental way, and the performance of our method is evaluated quantitatively through comparison with benchmark methods. Note that we compare our method mainly to the approaches that rely on the idea of generative memory replay, e.g. replaying generator synthesized samples of previous classes to the task solver without storing real samples of old data. For the sake of fairness we only consider benchmarks evaluated in class incremental single head evaluation setup. Hereby, to best of our knowledge (179) represent the state-of-the art benchmark followed by (173) and (180). Next, we relax the strict incremental setup and allow partial storage of real samples of previous classes. Here we compare to the iCarl (169), which is a state-of-the art method for continual learning with storing real samples.

4.1.5.2 Results

A quantitative comparison of the both variants of the proposed DGM approach with other methods is listed in Table 4.1. We use joint training (JT) as upper performance bound, here the task solver D is trained in a non-incremental fashion on all real samples without adversarial training being involved (not acting as discriminator for real/fakes). The first set of methods evaluated by (166) do not adhere to the strictly incremental setup, and thus make use of stored samples. Storing real samples if often refereed to as "episodic memory". The second set of methods we compare with do not store any real data samples. Notably, the generator of our method is initialized to by roughly 50% of the size of the network used by the other methods in this group.

Our method outperforms the state of the art (180) and (173) on the MNIST and SVHN benchmarks through the integration of the memory learning mechanism directly into the gen-

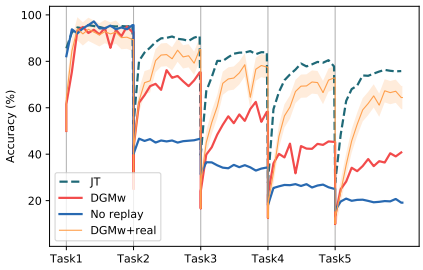


Figure 4.2: Top-5 performance of DGMw together with upper (JT) and lower (No replay) performance bounds measured for ImageNet-50 benchmark. DGM+real denotes variation with different ratios of real samples added to the replay loop (25%-75% of replayed samples being real)

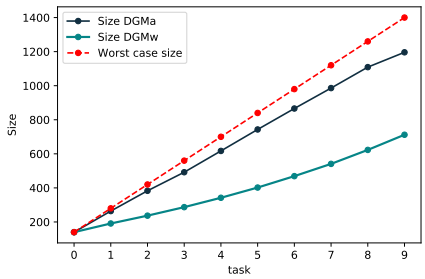


Figure 4.3: Network growth of DGMw and DGMa in the incremental MNIST setup. The worst case scenario refers adding the initial size after each task. Size is reported in terms of the number of neurons per layer.

erator network, and the expansion of said network as it saturates to accommodate new tasks. We yield an increase in performance over (179), a method that is based on a replay strategy for the generator network and does not provide dynamic expansion mechanism of the memory network, leading to increased training time and sensitivity to semantic drift. As it can be observed for both, our method and *MeRGAN*, the accuracy reported between 5-tasks and 10-tasks of the MNIST benchmark has changed a little, suggesting that for this dataset and evaluation methodology both approaches have largely curbed the effects of catastrophic forgetting.

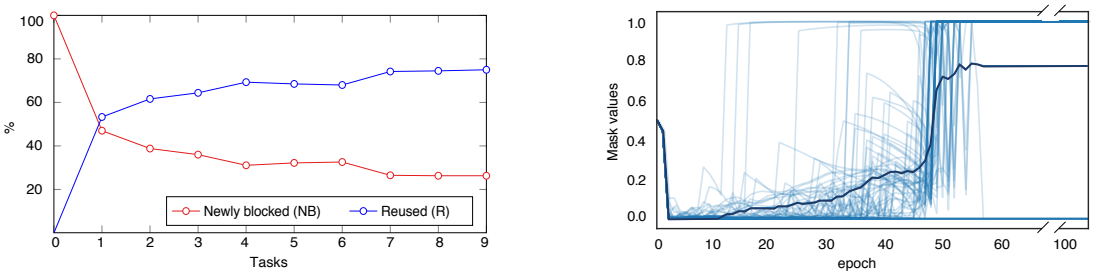
Interestingly, *DGM* outperforms joint training on the MNIST dataset using the same architecture. This suggests that the strictly incremental training methodology forced the network to learn better generalizations compared to what it would learn given all the data.

Given the high accuracies reached on the MNIST dataset largely give rise to questions concerning saturation, we opted to perform further evaluation on the more visually diverse SVHN dataset (189). In this context, increased data diversity translates to more difficult generation and susceptibility to catastrophic forgetting. In fact, as can be seen in Tab. 4.1, the difference between 5-task and 10-task accuracies is significantly larger in all methods than what can be observed in the MNIST experiments. DGMa strongly outperforms all other methods on the SVHN benchmark, whereas DGMw reaches the state of the art and slightly outperforms the best competitor only after the 10-th task. This can be attributed primarily to our efficient network expansion that allows for more redundancy in reserving representative neurons for each

task, and a less destructive joint use of neurons between tasks. DGM thus becomes more stable in the face of catastrophic forgetting.

The quality of the generated images after 10 stages of incremental training for MNIST, SVHN and CIFAR-10 can be observed in Fig. 4.5. The generative network is able to provide an informative and diverse set of samples for all previously seen classes without catastrophic forgetting.

Finally in the ImageNet-50 benchmark we incrementally add 50 classes with 10 classes per step and evaluate the classification performance of out system in a single-head evaluation. We evaluate the Frechet Inception Distance (FID) (193) metric of the generated images over 50 classes in order to asses the perceptual quality of the generation. Dynamics of the FID metric across the different tasks are provided in the appendix. In conjunction with the qualitative results shown in Fig. 4.6 it can be observed that little to no quality is lost for generating samples from previous tasks evidencing that no knowledge is forgotten. Nevertheless, for each newly learned task the discriminator network's classification layer is extended with 10 new classes, making the complexity of the classification problem to grow constantly (from 10-way classification to 50-way classification). With the more complex ImageNet samples also the generation task becomes much harder. These factors negatively impact the classification performance of the task solver presented in the Fig. 4.2, where DGMw performs significantly worse then the joint training (JT) upper bound.



(a) Parameter re-usability

(b) Mask learning

Figure 4.4: Mask behaviour analysis: parameter re-usability and mask learning dynamics.

Next, we relax the strict incremental setup and allow the DGM to partially store real samples of previous classes and compare the performance to the sate-of-the-art iCarl (169)[1]. Note that, iCarl relies only on storing real samples of previous classes introducing a smart sample

[1]We apply iCarl to the same classes of ImageNet as DGM with equal resolution using official implementation of iCarl: `https://github.com/srebuffi/iCaRL`

Method	Top-1(%)		Top-5(%)	
	A_{30}	A_{50}	A_{30}	A_{50}
JT	57.35	49.88	84.70	78.24
iCarl (K=1000)	29.38	28.98	69.98	59.49
iCarl (K=2000)	39.38	29.96	70.57	60.07
DGMw (K=1000)	47.27	30.24	76.73	56.76
DGMw (K=2000)	48.80	33.68	76.67	63.84
DGMw	29.67	17.32	62.53	40.76
DGMw (r=0.75)	50.80	38.22	78.27	64.64
DGMw (r=0.5)	49.33	40.40	78.20	67.12
DGMw (r=0.25)	42.33	30.20	74.60	57.32

Table 4.2: Performance comparison of DGM and iCarl. We run DGMw for different values of r, e.g. different ratios of real/synthesized samples, and different K - maximal memory size.

selection strategy (for details refer to Sec. 4.1.2). We define a ratio of stored real and total replayed samples $r = n_r/N$, where N is the total number of samples replayed per class and n_r is the number of randomly selected real samples stored per each previously seen class. We always keep the number of replayed samples balanced with the number of real samples of the currently observed classes, thus N is set to be equal to the average number of samples per class in the currently observed data chunk S_t. Furthermore, similarly to iCarl (169) we define K to be the total number of real samples that can be stored by the algorithm in at any point of time. We compare DGMw with iCarl for different values of K (e.g. $K = 1000$ and $K = 2000$) allowing the storage of $K/|\mathcal{Y}^t|$ samples per class.

From Tab. 4.2 we observe that DGM is outperformed by iCarl when no real samples are used (e.g. $r = 0$) after 50 classes in top-1 and after 30 and 50 classes in top-5 accuracy. DGMw with only synthesized samples being replayed reaches iCarl's performance in top-1 accuracy after 30 classes. Furthermore, we observe that adding real samples to the replay loop boosts DGM's classification accuracy beyond the iCarl's one. Thus, already for $r = 0.25$ the performance of our system can be improved significantly, encouraging that our method can be successfully applied within few-shot regime. We now consider DGM and iCarl with the same

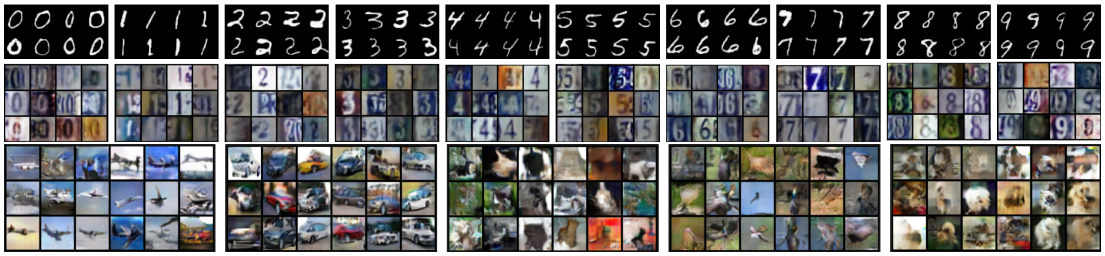


Figure 4.5: 4/images_cvpr generated by DGM from MNIST(top), SVHN(middle), and CIFAR(bottom) after learning 10 tasks incrementally.

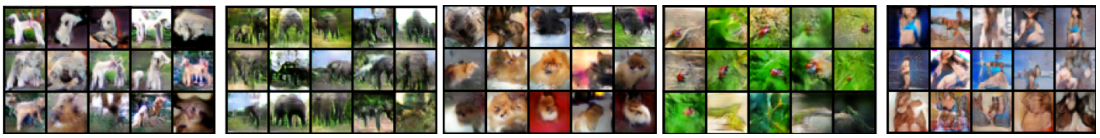


Figure 4.6: Examples of ImageNet samples generated by DGM after learning 5 tasks (50 classes)

memory size K (we test for $K = 1000$ and $K = 2000$). Here DGM outperforms iCarl in top-1 accuracy after 30 and 50 classes, in top-5 accuracy after 30 classes. This is largely attributed to the advantage of DGM using generated samples additionally to the stored real once.

4.1.5.3 Plasticity evolution

We analyze how learning is accomplished within a given task t, and how this further affects the wider algorithm. For a given task t, its corresponding binary mask M_t is initialized with the scaling parameter $s = 0$. Fig. 4.4(b) shows the learning trajectories of the mask values over the learning time of task t. As observed in Fig. 4.4(b), at task initialization of DGMa the mask is completely non-binary (all mask values are 0.5). As training progresses, the scaling parameter s is annealed, the network is encouraged to search for the most efficient parameter constellation (epoch 2-10 Fig. 4.4(b)). But with most mask values near 0 (most of units are not used, high efficiency is reached), the network's capacity to learn is greatly curtailed. The optimization process pushes the mask to become less sparse, the number of non-zero mask values is steadily increasing until the optimal mask constellation is found, a trend observed in the segment between the epoch 10 and 55 Fig. 4.4(b). This behaviour can be seen as a *short-term memory* formation - if learning was stopped at e.g. epoch 40 only a relatively small fraction of learnable units is binary masked, the units with non-binary mask values would be still partially overwritten during the subsequent tasks, thus, partial forgetting would still occur. A transition from short to the *long-term memory* occurs largely within the epochs 45-65. Here

the most representative units are selected and reserved by the network, parameters that have not made this transition are essentially left as unused for the learning task t. Finally, the optimal neuron constellation is optimized for the given task from epoch 60 onwards. The plasticity learning behaviors of DGMw is analyzed in the appendix.

For a given task t, masked units (neurons in DGMa, network weights in DGMw) can be broadly divided into three types: (i) units that are not used at all (U) [masked with 0] , (ii) units that are newly blocked for the task (NB_t), (iii) units that have been reused from previous tasks (NB_t).

Figure 4.4(a) presents the evolution of the ratio of the (NB_t) and (NB_t) types over the total number units blocked for the task t. Of particular importance is that the ratio of reused units is increasing between tasks, while the ratio of newly blocked units is decreasing. These trends can be justified by the network learning to generalize better, leading to a more efficient capacity allocation for new tasks.

4.1.5.4 Size vs. accuracy trade-off

One of the primary strengths of DGM is an efficient generator network expansion component, removing of which would lead to inability of the generator network to accommodate for memorizing new new task (analysis provided in the supplementary material). Its performance is directly related to how the network parameters are reserved during the incremental learning,which ultimately depends on the generator's ability to generalize from previously learned tasks. Fig. 4.3(b) reports network growth against the number of tasks learned. We find that learning masks directly for the layer weights (DGMw) significantly boosts the parameter re-usability, slowing down network growth as new tasks are introduced. Furthermore, one can observe the efficiency of our network growth method compared to a worst case scenario, where for every task we simply add the the initial number of network parameters.

Obviously, there is a trade-off between the network size and the quality of the synthesized images, which is further reflected in the classification performance of the system. As described in Sec. 4.1.4, hyperparameter that represents the re-usability factor during the training is λ_{RU}. We analyze the sensitivity of this parameter in the Tab. 4.3. As expected, we observe that λ_{RU} is negatively correlated with the network size and therefore the classification accuracy.

λ	$2e^{-6}$	$2e^{-1}$	0.75	1	2	5
$A_5(\%)$	98.35	98.10	97.22	96.91	96.67	88.70
$Size$	364	352	311	286	261	193

Table 4.3: Sensitivity of parameter λ_{RU} (MNIST benchmark).

4.1.6 Conclusions

In this section we studied the continual learning problem in a context in which evolving legislation can restrict their operation, specifically a single-head, strictly incremental context. We proposed a Dynamic Generative Memory approach for class incremental continual learning. Our results suggest that our approach successfully overcomes catastrophic forgetting by making use of a conditional generative adversarial model where the generator is used as a memory module through neural masking. We also show that the proposed dynamic memory expansion mechanism facilitates a resource efficient generator adaptation to successfully accommodate learning new tasks.

5

Concluding Remarks

5.1 Summary and Remarks

In this thesis we have investigated methods for overcoming commonly seen cases of data scarcity, through unsupervised annotation and feature estimation methods, few-shot learning techniques and finally through overcoming catastrophic forgetting in strictly class-incremental lifelong learning.

In Chapter 2 we have investigated a method of unsupervised, class agnostic spatio-temporal tube production system (Section 2.2), based on expanding the image-level "objectness" category to the temporal domain. Due to limitations in this method's temporal localization capabilities, we next proposed a method of providing more precise spatio-temporal annotations through an unsupervised, pixel-level segmentation system (Section 2.3) that makes use of multiple image level and temporal cues, organized into different topologies. For an optimal segmentation of the resulting graph, the similarity and graph cutting are jointly optimized.

While these methods tackle the lack of quality in gathered data by easing the annotation task through proposal generation and video segmentation, there are tasks for which low-level features are required but difficult to learn. A particular case is depth estimation, where supervised learning requires expensive LIDAR depth maps. As such, in Section 2.4 we proposed an unsupervised, depth estimation system that makes use of cycled adversarial learning. This system proved the usability of generative adversarial learning for this task, the more rigorous restrictions between the image views, provided by the synthesis cycle leading to a better optimization process. Overall this chapter provides a suite of unsupervised methods suitable for spatio-temporal video segmentation and depth estimation.

The next stage of our work detailed in Chapter 3 is focused on learning directly on data whose distribution is skewed, a data quantity restriction. We developed a low-shot learning system that makes use of a canonical, category-wise 3D model and predicted instance-specific texture and mesh deformations to it (Section 3.2) to generate samples for classes where samples are scarce. While learning a class specific 3D model does not provide the object kinematics, it can be sampled for novel viewing angles and positions, increasing generated data diversity. As a next stage we proposed a multi-modal generative few-shot learning classification system (Section 3.3), under the assumption that textual data is often more abundant than visual data for the scarce classes. Using this extra information we directly generate feature vectors, and provide a strategy to combine generated and real samples such that an extremely simple nearest neighbour approach is sufficient for state of the art classification performance. Within this chapter we observe that the data scarcity problem can be overcome using either data of superior quality, or data from an abundant modality to greatly outperform methods that do not use any supplementary data. We posit that this is a reasonable stance to take, as data quality can be improved using unsupervised systems, and textual data is particularly rich and comprehensive.

Finally, within Chapter 4 we tackle data storage restrictions caused by tightening legislation, and their effects on lifelong learning systems. Specifically, to overcome the exacerbated catastrophic forgetting problem in life-long learning systems operating in a strictly incremental fashion we have developed a generative, dynamically expanding network architecture. This network overcomes the lack of stored data by generating samples from old knowledge at each new incoming task, and retraining the classification network. An efficient binary mask module preserves the old information within the generative subnetwork, which is expanded as new tasks arrive to maintain enough capacity to learn and store incoming information.

5.2 Future Perspectives

As deep learning methods become more widely applied, methods of combating data scarcity will become ever more useful. As such, in the case of video annotation systems, computational efficiency and wider applicability are two of the paths in which they could be improved.

At the same time, the works detailed in this thesis on few-shot learning were only concentrated on the image domain, and more specifically on image-level fine-grained classification. More powerful generative strategies can help expand these methods to a broader classification scenario, and with the progress seen in video generation they can be expanded to the video

domain. In the case of the work detailed in Section 3.2, and expansion to the temporal domain would be extremely beneficial - learning an object's kinematics would allow us to generate extremely diverse and informative samples.

Finally, the catastrophic forgetting system presented has a series of generative and resource limitations - the generative system can only be improved to better "remember" previously seen data distributions, while the system itself grows its own memory footprint to handle new data. Note that the network expansion rate decreases for each new task as more knowledge is stored. This growth process can be further optimized to reach a saturation point earlier in the training, for wider applicability.

Bibliography

[1] M. Marian Puscas, E. Sangineto, D. Culibrk, and N. Sebe, "Unsupervised tube extraction using transductive learning and dense trajectories," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1653–1661. v, 2, 3, 5, 6, 12

[2] J. Song, L. Gao, M. M. Puscas, F. Nie, F. Shen, and N. Sebe, "Joint graph learning and video segmentation via multiple cues and topology calibration," in *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 2016, pp. 831–840. vi, 2, 3, 5, 6, 27

[3] A. Pilzer, D. Xu, M. Puscas, E. Ricci, and N. Sebe, "Unsupervised adversarial depth estimation using cycled generative networks," in *3DV*, 2018. vi, 2, 6, 7, 49

[4] F. Pahde, O. Ostapenko, P. Jähnichen, T. Klein, and M. Nabi, "Self-paced adversarial training for multimodal few-shot learning," *WACV*, 2019. vii, 2, 4, 6, 7, 70, 81, 84, 85, 89

[5] O. Ostapenko, M. Puscas, T. Klein, P. Jähnichen, and M. Nabi, "Learning to remember: A synaptic plasticity driven framework for continual learning," *arXiv preprint arXiv:1904.03137*, 2019. viii, 2, 7, 95

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012. 1

[7] K. Soomro, A. Roshan Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," in *CRCV-TR-12-01*, 2012. 1, 9

[8] P. Voigt and A. Von dem Bussche, *The EU General Data Protection Regulation (GDPR)*. Springer, vol. 18. 1, 5

[9] O. Ostapenko, M. Puscas, T. Klein, P. Jähnichen, and M. Nabi, "Learning to remember what to remember: A synaptic plasticity driven framework." 2, 7

[10] M. Jain, J. van Gemert, H. Jégou, P. Bouthemy, and C. G. M. Snoek, "Action localization with tubelets from motion," in *CVPR*, 2014, pp. 740–747. 3, 12, 14

[11] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph based video segmentation," in *CVPR*, 2010. 3, 12, 14, 23

[12] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata, "Feature generating networks for zero-shot learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 4, 70

[13] B. Hariharan and R. B. Girshick, "Low-shot visual object recognition," *CoRR*, vol. abs/1606.02819, 2016. [Online]. Available: http://arxiv.org/abs/1606.02819 4, 74

[14] L. Bertinetto, J. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners," in *Advances in Neural Information Processing Systems*, 2016, pp. 523–531. 4, 68

[15] D. Yoo, H. Fan, V. N. Boddeti, and K. M. Kitani, "Efficient K-Shot Learning with Regularized Deep Networks," in *AAAI*, 2018. 4, 67

[16] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1933–1941. 9

[17] G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 6, pp. 1510–1517, 2018. 9

[18] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6450–6459. 9

[19] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013. 10, 14

[20] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the objectness of image windows," *IEEE Trans. on PAMI*, vol. 34, no. 11, pp. 2189–2202, 2012. 10

[21] J. Carreira and C. Sminchisescu, "Constrained parametric min-cuts for automatic object segmentation," in *CVPR*, 2010, pp. 3241–3248. 10

[22] I. Endres and D. Hoiem, "Category independent object proposals," in *ECCV*, 2010, pp. 575–588. 10

[23] S. Yi and V. Pavlovic, "Multi-cue structure preserving MRF for unconstrained video segmentation," in *ICCV*, 2015. 10, 27

[24] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *CVPR*, 2010, pp. 2141–2148. 10, 11, 27, 29, 32, 42, 43

[25] S. Paris, "Edge-preserving smoothing and mean-shift segmentation of video streams," in *ECCV*, 2008, pp. 460–473. 11, 27

[26] F. Galasso, R. Cipolla, and B. Schiele, "Video segmentation with superpixels," in *ACCV*, 2012. 11, 27, 29, 32, 42, 43, 44

[27] A. Khoreva, F. Galasso, M. Hein, and B. Schiele, "Classifier based graph construction for video segmentation," in *CVPR*, 2015. 11, 27, 30, 32

[28] F. Galasso, M. Keuper, T. Brox, and B. Schiele, "Spectral graph reduction for efficient image and streaming video segmentation," in *CVPR*, 2014. 11, 27, 28, 29

[29] C.-P. Yu, H. Le, G. Zelinsky, and D. Samaras, "Efficient video segmentation using parametric graph partitioning," in *ICCV*, 2015. 11, 27, 28, 29

[30] W. Brendel and S. Todorovic, "Video object segmentation by tracking regions," in *ICCV*, 2009, pp. 833–840. 11, 29

[31] A. V. Reina, S. Avidan, H. Pfister, and E. L. Miller, "Multiple hypothesis video segmentation from superpixel flows," in *ECCV*, 2010, pp. 268–281. 11, 29

[32] J. Son, I. Jung, K. Park, and B. Han, "Tracking-by-segmentation with online gradient boosting decision tree," in *ICCV*, 2015. 11

[33] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *NIPS*, 2014. 11, 49, 50, 51, 57, 59, 62

[34] L. Ladicky, J. Shi, and M. Pollefeys, "Pulling things out of perspective," in *CVPR*, 2014. 11, 49

[35] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *TPAMI*, vol. 38, no. 10, pp. 2024–2039, 2016. 11, 49, 50, 51, 59, 62

[36] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, "Monocular depth estimation using multi-scale continuous crfs as sequential deep networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 11, 49, 50, 51, 59, 62

[37] H. Wang, A. Kläser, C. Schmid, and C. Liu, "Action recognition by dense trajectories," in *CVPR*, 2011, pp. 3169–3176. 12, 15

[38] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *ICCV*, 2013, pp. 3551–3558. 12, 15

[39] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari, "Learning object class detectors from weakly annotated video," in *CVPR*, 2012. 13, 22, 24

[40] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *ECCV*, 2010, pp. 282–295. 14, 24

[41] A. Papazoglou and V. Ferrari, "Fast object segmentation in unconstrained video," in *ICCV*, 2013, pp. 1777–1784. 14, 24

[42] Y. J. Lee, J. Kim, and K. Grauman, "Key-segments for video object segmentation," in *ICCV*, 2011, pp. 1995–2002. 14

[43] D. Banica, A. Agape, A. Ion, and C. Sminchisescu, "Video object segmentation by salient segment chain composition," in *Computer Vision Workshops (ICCVW), IEEE International Conference on*, 2013, pp. 283–290. 14

[44] D. Oneata, J. Revaud, J. J. Verbeek, and C. Schmid, "Spatio-temporal object detection proposals," in *ECCV*, 2014. 14, 22, 23, 24

[45] K. Fragkiadaki, P. A. Arbeláez, P. Felsen, and J. Malik, "Spatio-temporal moving object proposals," *CoRR*, vol. abs/1412.6504, 2014. [Online]. Available: http://arxiv.org/abs/1412.6504 14

[46] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *CVPR*, 2014, pp. 2155–2162. 14

[47] C. Szegedy, S. Reed, D. Erhan, and D. Anguelov, "Scalable, high-quality object detection," *CoRR*, vol. abs/1412.1441, 2014. [Online]. Available: http://arxiv.org/abs/1412.1441 14

[48] A. Khosla, T. Zhou, T. Malisiewicz, A. Efros, and A. Torralba, "Undoing the damage of dataset bias," in *ECCV*, 2012. 14

[49] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," *BMVC*, 2009. 15, 18, 26

[50] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. on PAMI*, vol. 32, no. 9, pp. 1627–1645, 2010. 15, 20, 26

[51] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014, pp. 580–587. 15, 19, 20, 26

[52] L. Bourdev, S. Maji, T. Brox, and J. Malik, "Detecting people using mutually consistent poselet activations," in *ECCV*, 2010. 15, 18, 26

[53] J. S. Supancic III and D. Ramanan, "Self-paced learning for long-term tracking," in *CVPR*, 2013, pp. 2379–2386. 19, 69

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1106–1114. 20

[55] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *CVPR Workshops*, 2014. 20

[56] M. D. Rodriguez, J. Ahmed, and M. Shah, "Action MACH a spatio-temporal maximum average correlation height filter for action recognition," in *CVPR*, 2008. 22

[57] C. Li, L. Lin, W. Zuo, S. Yan, and J. Tang, "Sold: Sub-optimal low-rank decomposition for efficient video segmentation," in *CVPR*, 2015. 27, 29, 30, 41, 42, 43, 44

[58] T. Ma and L. J. Latecki, "Maximum weight cliques with mutex constraints for video object segmentation," in *CVPR*, 2012, pp. 670–677. 28

[59] L. Gao, J. Song, F. Nie, F. Zou, N. Sebe, and H. T. Shen, "Graph-without-cut: An ideal graph learning for image segmentation," in *AAAI*, 2016, pp. 1188–1194. 28

[60] F. Shen, C. Shen, Q. Shi, A. van den Hengel, Z. Tang, and H. T. Shen, "Hashing on nonlinear manifolds," *IEEE Trans. Image Processing*, vol. 24, no. 6, pp. 1839–1851, 2015. 28

[61] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Trans. Multimedia*, vol. 15, no. 8, pp. 1997–2008, 2013. 28

[62] L. Gao, J. Song, F. Nie, Y. Yan, N. Sebe, and H. T. Shen, "Optimal graph learning with partial tags and multiple features for image and video annotation," in *CVPR*, 2015, pp. 4371–4379. 28

[63] F. Nie, X. Wang, and H. Huang, "Clustering and projected clustering with adaptive neighbors," in *SIGKDD*, 2014, pp. 977–986. 28

[64] F. Nie, X. Wang, M. I. Jordan, and H. Huang, "The constrained laplacian rank algorithm for graph-based clustering," in *AAAI*, 2016, pp. 1969–1976. 28

[65] N. S. Nagaraja, F. R. Schmidt, and T. Brox, "Video segmentation with just a few strokes," in *ICCV*, 2015, pp. 3235–3243. 28, 29

[66] M. Keuper, B. Andres, and T. Brox, "Motion trajectory segmentation via minimum cost multicuts," in *ICCV*, 2015, pp. 3271–3279. 28, 29

[67] P. Ochs, J. Malik, and T. Brox, "Segmentation of moving objects by long term video analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1187–1200, 2014. 28

[68] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *ECCV*, 2010, pp. 282–295. 28, 29

[69] P. Ochs and T. Brox, "Higher order motion models and spectral clustering," in *CVPR*, 2012, pp. 614–621. 28

[70] V. Zografos, R. Lenz, E. Ringaby, M. Felsberg, and K. Nordberg, "Fast segmentation of sparse 3d point trajectories using group theoretical invariants," in *ACCV*, 2014, pp. 675–691. 28

[71] L. Chen, J. Shen, W. Wang, and B. Ni, "Video object segmentation via dense trajectories," *IEEE Trans. Multimedia*, vol. 17, no. 12, pp. 2225–2234, 2015. 29

[72] K. Fragkiadaki and J. Shi, "Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement," in *CVPR*, 2011, pp. 2073–2080. 29

[73] B. Luo, H. Li, T. Song, and C. Huang, "Object segmentation from long video sequences," in *ACM Multimedia*, 2015, pp. 1187–1190. 29

[74] C. Xu, C. Xiong, and J. J. Corso, "Streaming hierarchical video segmentation," in *ECCV*, 2012, pp. 626–639. 29, 30, 42, 43

[75] H. Jiang, G. Zhang, H. Wang, and H. Bao, "Spatio-temporal video segmentation of static scenes and its applications," *IEEE Trans. Multimedia*, vol. 17, no. 1, pp. 3–15, 2015. 29

[76] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011. 29, 31, 32, 42, 45

[77] P. Ochs and T. Brox, "Object segmentation in video: A hierarchical variational approach for turning point trajectories into dense regions," in *ICCV*, 2011, pp. 1583–1590. 29

[78] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in *CVPR*, 2009, pp. 2294–2301. 29

[79] F. Galasso, N. S. Nagaraja, T. J. Cardenas, T. Brox, and B. Schiele, "A unified video segmentation benchmark: Annotation, metrics and analysis," in *ICCV*, 2013. 30, 40, 41, 43

[80] B. Liu and X. He, "Multiclass semantic video segmentation with object-level active inference," in *CVPR*, 2015, pp. 4286–4294. 30

[81] Y. Wang, J. Liu, Y. Li, and H. Lu, "Semi- and weakly- supervised semantic segmentation with deep convolutional neural networks," in *ACM Multimedia*, 2015, pp. 1223–1226. 30

[82] X. Yao, J. Han, G. Cheng, and L. Guo, "Semantic segmentation based on stacked discriminative autoencoders and context-constrained weakly supervised learning," in *ACM Multimedia*, 2015, pp. 1211–1214. 30

[83] K. Fan, "On a Theorem of Weyl Concerning Eigenvalues of Linear Transformations. I," *Proceedings of the National Academy of Science*, vol. 35, pp. 652–655, Nov. 1949. 36

[84] J. Corso, E. Sharon, S. Dube, S. El-Saden, U. Sinha, and A. Yuille, "Efficient multilevel brain tumor segmentation with integrated bayesian model classification," *Medical Imaging, IEEE Transactions on*, vol. 27, no. 5, pp. 629–640, 2008. 42, 43

[85] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *ICCV*, 2013. 43

[86] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *NIPS*, 2006. 50, 59, 62

[87] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *CVPR*, 2016. 50, 52, 53

[88] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *CVPR*, 2016. 50, 52

[89] R. Garg, V. K. BG, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *ECCV*. Springer, 2016. 50, 52, 57, 59, 62

[90] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *CVPR*, vol. 2, no. 6, 2017, p. 7. 50, 52, 57, 58, 59, 62

[91] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey, "Learning depth from monocular videos using direct methods," in *CVPR*, 2018. 50, 51

[92] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *CVPR*, vol. 2, no. 6, 2017, p. 7. 50, 52, 57, 59, 62

[93] R. Mahjourian, M. Wicke, and A. Angelova, "Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints," in *CVPR*, 2018. 50, 51

[94] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012. 51

[95] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *TPAMI*, vol. 31, no. 5, pp. 824–840, 2009. 51

[96] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, 2012. 51, 57, 59

[97] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *CVPR*, 2016. 51, 57

[98] W. Zhuo, M. Salzmann, X. He, and M. Liu, "Indoor scene structure analysis for single image depth estimation," in *CVPR*, 2015. 51

[99] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *3DV*, 2016. 51

[100] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, "Multi-scale continuous crfs as sequential deep networks for monocular depth estimation," in *CVPR*, 2017. 51

[101] D. Xu, W. Wang, H. Tang, H. Liu, N. Sebe, and E. Ricci, "Structured attention guided convolutional neural fields for monocular depth estimation," in *CVPR*, 2018. 51

[102] Y. Kuznietsov, J. Stückler, and B. Leibe, "Semi-supervised deep learning for monocular depth map prediction," *CVPR*, 2017. 51, 59, 62

[103] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, "Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction," *arXiv preprint arXiv:1803.03893*, 2018. 51

[104] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey, "Learning depth from monocular videos using direct methods," in *CVPR*, 2018. 52

[105] Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia, "Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding," *arXiv preprint arXiv:1806.10556*, 2018. 52

[106] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014. 52, 54

[107] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017. 52

[108] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," *arXiv preprint*, 2017. 52

[109] J. N. Kundu, P. K. Uppala, A. Pahuja, and R. V. Babu, "Adadepth: Unsupervised content congruent adaptation for depth estimation," in *CVPR*, 2018. 52, 59, 62

[110] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015. 57

[111] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/ 58

[112] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. Yuille, "Towards unified depth and semantic prediction from a single image," in *CVPR*, 2015. 59

[113] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1717–1724. 66

[114] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *InternationalConference on Learning Representations*, 2017. 66, 67, 86, 89

[115] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *NIPS*, 2017, pp. 4080–4090. 66, 68, 83, 86, 89

[116] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," in *NIPS*, 2016, pp. 3630–3638. 66, 68, 86, 89

[117] F.-F. Li, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594–611, 2006. 66

[118] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Deep Learning Workshop*, vol. 2, 2015. 66, 67

[119] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in *Proceedings of the Cognitive Science Society*, vol. 33, 2011. 66

[120] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *International conference on machine learning*, 2016, pp. 1842–1850. 66

[121] H. Edwards and A. Storkey, "Towards a neural statistician," *ICLR*, 2017. 66

[122] B. Hariharan and R. Girshick, "Low-shot Visual Recognition by Shrinking and Hallucinating Features," in *ICCV*, 2017. 66, 67, 75, 78, 80, 83, 87

[123] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *CVPR*, 2014, pp. 1701–1708. 67

[124] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in Neural Information Processing Systems*, 1994, pp. 737–744. 67

[125] M. Douze, A. Szlam, B. Hariharan, and H. Jégou, "Low-shot learning with large-scale diffusion," *CoRR*, 2017. 67

[126] Y.-X. Wang, R. Girshick, M. Hebert, and B. Hariharan, "Low-Shot Learning from Imaginary Data," in *CVPR*, 2018. 67

[127] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *ICML*, 2017. 68, 86, 89

[128] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, p. 248, 2015. 68

[129] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it smpl: Automatic estimation of 3d human pose and shape from a single image," in *European Conference on Computer Vision*. Springer, 2016, pp. 561–578. 68

[130] S. Zuffi, A. Kanazawa, D. W. Jacobs, and M. J. Black, "3d menagerie: Modeling the 3d shape and pose of animals." 68

[131] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik, "Learning category-specific mesh reconstruction from image collections," in *ECCV*, 2018. 68, 70, 71, 73, 75

[132] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *ICML*, 2009, pp. 41–48. 68, 71

[133] M. P. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *NIPS*, 2010, pp. 1189–1197. 69, 71

[134] L. Jiang, D. Meng, S.-I. Yu, Z. Lan, S. Shan, and A. Hauptmann, "Self-paced learning with diversity," in *Advances in Neural Information Processing Systems*, 2014, pp. 2078–2086. 69

[135] A. Pentina, V. Sharmanska, and C. H. Lampert, "Curriculum learning of multiple tasks," in *CVPR*, 2015, pp. 5492–5500. 69

[136] D. Zhang, D. Meng, L. Zhao, and J. Han, "Bridging saliency detection to weakly supervised object detection based on self-paced curriculum learning," *arXiv preprint arXiv:1703.01290*, 2017. 69

[137] X. Liang, S. Liu, Y. Wei, L. Liu, L. Lin, and S. Yan, "Towards computational baby learning: A weakly-supervised approach for object detection," in *ICCV*, 2015, pp. 999–1007. 69

[138] E. Sangineto, M. Nabi, D. Culibrk, and N. Sebe, "Self paced deep learning for weakly supervised object detection," *arXiv preprint arXiv:1605.07651*, 2016. 69, 71

[139] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models," *arXiv:1411.2539 [cs]*, Nov. 2014, arXiv: 1411.2539. 69

[140] F. Faghri, D. J. Fleet, J. R. Kiros, and S. Fidler, "VSE++: Improving Visual-Semantic Embeddings with Hard Negatives," *arXiv:1707.05612 [cs]*, Jul. 2017, arXiv: 1707.05612. 69

[141] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *CVPR*, 2015, pp. 3128–3137. 69

[142] S. Reed, Z. Akata, H. Lee, and B. Schiele, "Learning deep representations of fine-grained visual descriptions," in *CVPR*, 2016, pp. 49–58. 69, 87

[143] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 48. PMLR, 2016, pp. 1060–1069. 69, 84

[144] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *ICCV*, 2017. 69, 74, 75, 84, 87, 88

[145] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks," *arXiv:1711.10485 [cs]*, Nov. 2017, arXiv: 1711.10485. 69, 84

[146] S. Sharma, D. Suhubdy, V. Michalski, S. E. Kahou, and Y. Bengio, "Chatpainter: Improving text to image generation using dialogue," *arXiv preprint arXiv:1802.08216*, 2018. 69

[147] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013. 69

[148] A. Mishra, M. Reddy, A. Mittal, and H. A. Murthy, "A generative model for zero shot learning using conditional variational autoencoders," *arXiv preprint arXiv:1709.00663*, 2017. 69

[149] M. Elhoseiny, Y. Zhu, H. Zhang, and A. Elgammal, "Link the head to the "beak": Zero shot learning from noisy text description at part precision," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 70

[150] Y. Zhu, M. Elhoseiny, B. Liu, X. Peng, and A. Elgammal, "A generative adversarial approach for zero-shot learning from noisy texts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 70, 85

[151] S. Vicente, J. Carreira, L. Agapito, and J. Batista, "Reconstructing pascal voc," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 41–48. 71

[152] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, "Category-specific object reconstruction from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1966–1974. 71

[153] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. 75

[154] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," Tech. Rep., 2011. 75, 87

[155] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 76, 87

[156] R. G. Krishnan, A. Khandelwal, R. Ranganath, and D. Sontag, "Max-margin learning with the bayes factor," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018. 79

[157] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 80, 87

[158] M. Bauer, M. Rojas-Carulla, J. B. Świątkowski, B. Schölkopf, and R. E. Turner, "Discriminative k-shot learning using probabilistic models," *arXiv preprint arXiv:1706.00326*, 2017. 81, 83, 89

[159] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014, pp. 2672–2680. 84, 85

[160] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," *arXiv preprint arXiv:1610.09585*, 2016. 85, 100, 101, 104

[161] N. Hilliard, L. Phillips, S. Howland, A. Yankov, C. D. Corley, and N. O. Hodas, "Few-shot learning with metric-agnostic conditional embeddings," *arXiv preprint arXiv:1802.04376*, 2018. 86, 89

[162] Z. Chen, Y. Fu, Y. Zhang, Y.-G. Jiang, X. Xue, and L. Sigal, "Semantic feature augmentation in few-shot learning," *arXiv preprint arXiv:1804.05298*, 2018. 86, 89

[163] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *ICVGIP*. IEEE, 2008, pp. 722–729. 87

[164] J. Kirkpatrick, R. Pascanu, N. C. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *CoRR*, vol. abs/1612.00796, 2016. [Online]. Available: http://arxiv.org/abs/1612.00796 95, 98, 102

[165] F. Zenke, B. Poole, and S. Ganguli, "Improved multitask learning through synaptic intelligence," *CoRR*, vol. abs/1703.04200, 2017. [Online]. Available: http://arxiv.org/abs/1703.04200 95, 98, 102

[166] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. S. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," *CoRR*, vol. abs/1801.10112, 2018. [Online]. Available: http://arxiv.org/abs/1801.10112 95, 96, 99, 102, 103, 104

[167] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," *CoRR*, vol. abs/1711.09601, 2017. [Online]. Available: http://arxiv.org/abs/1711.09601 95, 98

[168] J. Serrà, D. Surís, M. Miron, and A. Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," *CoRR*, vol. abs/1801.01423, 2018. [Online]. Available: http://arxiv.org/abs/1801.01423 95, 98, 99, 101

[169] S. Rebuffi, A. Kolesnikov, and C. H. Lampert, "icarl: Incremental classifier and representation learning," *CoRR*, vol. abs/1611.07725, 2016. [Online]. Available: http://arxiv.org/abs/1611.07725 96, 98, 102, 103, 104, 106, 107

[170] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning," *arXiv preprint arXiv:1710.10628*, 2017. 96, 98

[171] M. Mayford, S. A. Siegelbaum, and E. R. Kandel, "Synapses and memory storage," *Cold Spring Harbor perspectives in biology*, p. a005751, 2012. 96

[172] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, Z. Zhang, and Y. Fu, "Incremental classifier learning with generative adversarial networks," *CoRR*, vol. abs/1802.00853, 2018. [Online]. Available: http://arxiv.org/abs/1802.00853 96

[173] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Advances in Neural Information Processing Systems*, 2017, pp. 2990–2999. 96, 98, 102, 104

[174] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999. 98

[175] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," ser. Psychology of Learning and Motivation, G. H. Bower, Ed. Academic Press, 1989, vol. 24, pp. 109 – 165. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0079742108605368 98

[176] R. Ratcliff, "Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions," *Psychological Review*, pp. 285–308, 1990. 98

[177] Z. Li and D. Hoiem, "Learning without forgetting," *CoRR*, vol. abs/1606.09282, 2016. [Online]. Available: http://arxiv.org/abs/1606.09282 98

[178] R. Kemker and C. Kanan, "Fearnet: Brain-inspired model for incremental learning," *arXiv preprint arXiv:1711.10563*, 2017. 98, 99

[179] C. Wu, L. Herranz, X. Liu, Y. Wang, J. van de Weijer, and B. Raducanu, "Memory Replay GANs: learning to generate images from new categories without forgetting," in *Advances In Neural Information Processing Systems*, 2018. 98, 102, 103, 104, 105

[180] A. Seff, A. Beatson, D. Suo, and H. Liu, "Continual learning in generative adversarial nets," *arXiv preprint arXiv:1705.08395*, 2017. 98, 102, 104

[181] A. Mallya and S. Lazebnik, "Piggyback: Adding multiple tasks to a single, fixed network by learning to mask," *arXiv preprint arXiv:1801.06519*, 2018. 99

[182] M. Mancini, E. Ricci, B. Caputo, and S. R. Bulò, "Adding new tasks to a single network with weight trasformations using binary masks," *arXiv preprint arXiv:1805.11119*, 2018. 99, 101

[183] M. Courbariaux, Y. Bengio, and J.-P. B. David, "Training deep neural networks with binary weights during propagations. arxiv preprint," *arXiv preprint arXiv:1511.00363*, 2015. 99

[184] J. Yoon, E. Yang, J. Lee, and S. J. Hwang, "Lifelong learning with dynamically expandable networks," 2018. 99

[185] N. Kamra, U. Gupta, and Y. Liu, "Deep generative dual memory network for continual learning," *arXiv preprint arXiv:1710.10368*, 2017. 99

[186] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," *arXiv preprint arXiv:1312.6211*, 2013. 100

[187] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777. 103

[188] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998. 103

[189] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, vol. 2011, no. 2, 2011, p. 5. 103, 105

[190] A. Krizhevsky, V. Nair, and G. Hinton, "The cifar-10 dataset," *online: http://www. cs. toronto. edu/kriz/cifar. html*, 2014. 103

[191] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015. 103

[192] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015. 104

[193] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a nash equilibrium," *arXiv preprint arXiv:1706.08500*, 2017. 106