Self-supervised learning of predictive segmentation models from video

Pauline Luc

Communauté
UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTÉ UNIVERSITÉ GRENOBLE ALPES

Spécialité : Mathématiques et Informatique

Arrêté ministériel : 25 mai 2016

Présentée par

Pauline LUC

Thèse dirigée par **Jakob VERBEEK**, INRIA
et codirigée par **Camille COUPRIE**, Facebook

préparée au sein du **Laboratoire Jean Kuntzmann,**
dans **l'École Doctorale Mathématiques, Sciences et technologies de l'information, Informatique**

Apprentissage autosupervisé de modèles prédictifs de segmentation à partir de vidéos

Self-supervised learning of predictive segmentation models from video

Thèse soutenue publiquement le **25 juin 2019**,
devant le jury composé de :

Monsieur JAKOB VERBEEK
DIRECTEUR DE RECHERCHE, INRIA CENTRE DE GRENOBLE RHÔNE-ALPES, Directeur de thèse

Monsieur CHRISTIAN WOLF
MAITRE DE CONFERENCES, INSA LYON, Rapporteur

Monsieur PATRICK PEREZ
DIRECTEUR SCIENTIFIQUE, VALEO.AI - PARIS, Rapporteur

Monsieur FLORENT PERRONNIN
DIRECTEUR SCIENTIFIQUE, NAVER LABS EUROPE - MEYLAN, Examinateur

Madame TINNE TUYTELAARS
PROFESSEUR, UNIV. CATHOLIQUE DE LOUVAIN - BELGIQUE, Examinateur

Madame ELISA FROMONT
PROFESSEUR, UNIVERSITE RENNES 1, Président

Contents

Abstract

The ability to anticipate future events is a key component of intelligence. Video prediction has been studied in recent years as a means to provide machines with this ability. Predictive models of the environment hold promise for allowing the transfer of recent reinforcement learning successes to many real-world contexts, by decreasing the number of interactions needed with the real world. Video prediction has been studied in recent years as a particular case of such predictive models, with broad applications in robotics and navigation systems. While RGB frames are easy to acquire and hold a lot of information, they are extremely challenging to predict, and cannot be directly interpreted by downstream applications. Here we introduce the novel tasks of predicting semantic and instance segmentation of future frames. The abstract feature spaces we consider are better suited for recursive prediction and allow us to develop models which convincingly predict segmentations up to half a second into the future. Predictions are more easily interpretable by downstream algorithms and remain rich, spatially detailed and easy to obtain, relying on state-of-the-art segmentation methods.

We first focus on the task of semantic segmentation, for which we propose a discriminative approach based on adversarial training. Then, we introduce the novel task of predicting future semantic segmentation, and develop an autoregressive convolutional neural network to address it. Finally, we extend our method to the more challenging problem of predicting future instance segmentation, which additionally segments out individual objects. To deal with a varying number of output labels per image, we develop a predictive model in the space of high-level convolutional image features of the Mask R-CNN instance segmentation model. We are able to produce visually pleasing segmentations at a high resolution for complex scenes involving a large number of instances, and with convincing accuracy up to half a second ahead.

Acknowledgements

First, I'd like to thank my PhD advisors Camille Couprie and Jakob Verbeek, for the huge amount of time, dedication and effort they have invested, throughout my PhD, consistently going out of their way for the outcome of our work to be of the highest quality. I have learned a great deal from you, and benefited from your rich and complementary perspectives, both in scientific terms and in terms of the research process. I have also deeply appreciated your day-to-day kindness and care.

I also want to thank some of the people I have been lucky enough to interact with, and who have been a huge inspiration throughout the years of my young career. Nikos Paragios, who is responsible for "saving my scientific career", by advising me throughout my final year of studies, to make choices I'm delighted I made, because they have led me to a most fulfilling start for my career. Fred Potter and Mehdi Felhi, along whose sides I had the chance to do an exciting research internship and who shook off the last doubts I had about starting a PhD. Florent Perronnin, who trusted me to start a PhD between two absolutely amazing labs, and has been an immense support ever since. Hervé Jégou and Iasonas Kokkinos, who have in turn, as my managers, always been pushing me to aim high, scientifically and career-wise, while showing a great deal of care and support. Olivier Teytaud, who, during the last year of my PhD, has been an amazing collaborator, a friend and a mentor, and whose dedication, team spirit, enthusiasm and trust have been a great source of motivation in the highs and resilience in the lows. And finally, Yann LeCun, who generously dedicated a great deal of his time to our scientific discussions, sharing with me some of his immense scientific culture and providing his invaluable insights to our projects. Along with Jakob and Camille, I want to thank each of them for their advice, mentoring and care, as well as the inestimable scientific insights they have shared with me, and the time they took to deliver all of these. It has been a great honour and pleasure to work closely with such brilliant, inspiring minds and generous, benevolent individuals.

I'm very fortunate and grateful also for our collaborations and interactions, with Natalia Neverova, Piotr Bojanowski and Soumith Chintala. Both labs of THOTH and FAIR Paris are amazing places to work and learn and I'm very grateful for the interactions, fruitful discussions, support and fun times there. For these, I want to thank Alexis Conneau, Matthijs Douze, Louis Martin, Pierre Stock, Guillaume Lample, Alexandre Defossez, Timothée Lacroix,

Thomas Lucas, Nikita Dvornik, Mathilde Caron, Vicky Kalogeiton, Alexandre Sablayrolles, Angela Fan, Nicolas Usunier, Gabriel Synnaeve, Jonas Gehring, Konstantin Shmelkov, Alberto Bietti, Pavel Tokmakov, Daan Wynen, Vlad Sydorov, Shreyas Saxena, Xavier Martin, Adria Ruiz, Neil Zeghidour, Nicolas Carion, Maxime Oquab, Diane Bouchacourt, David Lopez Paz, Marco Baroni, Ilija Radosavovic, Francisco Massa, Pierre Emmanuel Mazaré, Vasil Khalidov, Samuel Humeau, Shubho Sengupta, Ludovic Denoyer, Yann Ollivier, Benjamin Graham, Alexandre Lebrun, Martin Raison, Moustapha Cisse, Armand Joulin, Edouard Grave, Serhii Havrylov, Maxim Berman, Karteek Alahiri, and of course, Antoine Bordes, Julien Mairal and Cordelia Schmid. I also want to thank Piotr Dollar, Arthur Szlam, Lior Wolf, Lorenzo Torresani, Nicolas Ballas and Pascal Vincent for inspiring discussions. Many thanks also to Patricia Le Carré and Nathalie Gillot for their help with administrative tasks.

Next, I want to thank Patrick Pérez and Christian Wolf, for reviewing my thesis, and Tinne Tuytelaars, Elisa Fromont, Florent Perronnin, Patrick and Christian for accepting to be on my jury and travelling long distances for my defense.

Warm thanks also to Lina Marsso, Marie and Florent Raguin, Alexandre Marcireau, Adeline Richard, Clément Ricateau, Damien Frikha, Jean-Baptiste Bayle and many others, for becoming or staying great friends throughout these years; and to my family for their unconditional love and support. A special mention for Alberto Romagnoni and Sophie Duteil, for their experienced advice, for my Parisian cousins Laure Aussedat and Valentine Follin-Arbelet, for all the great times, and for my parents, for being the best. Finally, I want to thank my husband Albert, who is also my coach and my holiday planner, and whose support, patience and care have made him the behind-the-scene co-author of this thesis.

Chapter 1

Introduction

1.1 The ability to anticipate future events

Recent years have seen tremendous progress in various fields of artificial intelligence. In computer vision, deep learning approaches have revolutionized the field of recognition, with impressive achievements *e.g.* in classification (Krizhevsky et al., 2012; Simonyan and Zisserman, 2015; Szegedy et al., 2015; He et al., 2016), detection (Girshick et al., 2016) and semantic segmentation (Farabet et al., 2013; Long et al., 2015). While important challenges remain in providing a machine with the ability to perceive its environment, these advances have encouraged the research community to move forward and consider new and bold research directions towards developing other components necessary for intelligent behaviour.

Drawing inspiration from human intelligence is a rational route towards this goal. It is widely acknowledged that babies learn by observing and interacting with their environment. Rooted at the intersection between control theory, experimental psychology, computational neuroscience and statistics, the field of reinforcement learning (RL) is precisely concerned with learning behaviours and concepts through interactions with a given environment, and has also seen prodigious breakthroughs enabled by deep learning. In the context of games such as Backgammon (Tesauro, 1995), Atari (Mnih et al., 2013), Go (Silver et al., 2016), and recently StarCraft II (Vinyals et al., 2019), deep RL has enabled models to learn behaviours that surpass human performance.

However, all of these breakthroughs have occurred in environments that can be simulated. The main reason for this is that current RL algorithms require an immense number of interactions with the environment during learning. While this is possible in simulated environments, where interactions are cheap and can be sped up, in the real world, they are often slow, expensive and even dangerous. Learning invariably includes phases of exploration, where potentially completely random actions are taken, which can lead to catastrophic situations. For exemple, learning a model for autonomous driving using generic reinforcement learning algorithms directly on roads would involve crashing a huge number of

vehicles and endangering other users of the road, before it could reach – let alone surpass – human performance. Instead, current methods for autonomous driving require an important amount of prior expert knowledge specific to this task. Such knowledge is difficult to acquire and to incorporate optimally into a model. Just like hand-engineered vision features have been outperformed by far by deep features, deep RL has the potential to bring massive improvements to this field; or to any other robotic tasks, without having to renew, task after task, the same level of investment and effort by communities of experts.

The ability to anticipate future events is a key component of intelligence. In fact, our brain are essentially predictive machines, according to a currently dominant hypothesis in cognitive science (Bubic et al., 2010; Hohwy, 2013; Clark, 2013). Model-based RL consists in methods that rely on predictive models of the environment for learning, decision-making and/or planning. Such methods hold promise of allowing the transfer of recent RL successes to many real-world contexts, by decreasing the number of interactions needed with the real world, and this in various ways. First, simply because such predictive models of the world, also called *forward models*, can be used to replace simulators: instead of being programmed, the simulator is learned, and allows the same cheap and fast interactions. This can be seen as a form of data augmentation of the training set. Second, a forward model can be directly used in planning and control algorithms, *e.g.* using dynamic programming algorithms or model predictive control. Third, forward models have been used to formulate intrinsic reward signals, such as curiosity, to improve the exploration process and its efficiency, inspired by the observation by psychologists that curiosity is an important and widespread mechanism in the development of human knowledge, that fosters learning and exploring even in the absence of obvious external rewards (Silvia, 2012).

Finally, forward models can leverage the high dimensional observations they receive from the environment, to develop representations for it. Analogous to the fact that the scientific process heavily relies on experiments, checking that results are consistent with expectations given a proposed model, and correcting the model in case of inconsistency, forward models can be used to obtain an auxiliary learning signal and improve the intermediate representation they rely on (Mathieu, 2017). The underlying postulate is that a model that has learned to perform video prediction well will necessarily have developed a strong representation of the environment. This representation can then be shared across tasks and environments of similar nature. This applies beyond the RL setting, for learning predictive models from raw video: in both cases, exerting the ability to anticipate future events can in turn benefit the ability to perceive a dynamically evolving environment, without requiring human annotations. Tasks that leverage unlabeled data by predicting one part of the data given the other are called *self-supervised* tasks.

1.2 Video prediction

The task of video prediction is simply defined: given a sequence of frames extracted from a video, the goal is to predict a plausible future sequence. Conditioned on actions, it is an instance of forward-modeling with broad applications. Indeed, a video camera is a cheap sensor that acquires a rich, high-dimensional signal, and it is therefore a technology of choice for enabling any robot to capture the state of the environment. In combination with the reasons developed in the previous section, this has led to vivid and growing interest in recent years from the research community for the task of video prediction.

However, video prediction is excruciatingly challenging. The first challenge this task holds is its inherent uncertainty. From a probabilistic perspective, the task of video prediction can be viewed as learning a model that allows us to approximately sample from the distribution over the future output sequences, conditioned on the input sequence. In general, these conditional distributions have several modes and hence require more sophisticated methods than the classical regression losses that have been used in supervised learning. Generative modeling has also seen the emergence of methods that are able to leverage powerful deep learning techniques (Goodfellow et al., 2014; Kingma and Welling, 2014; van den Oord et al., 2016). In particular, recent developments of generative adversarial networks have distinguished themselves by the unprecedented quality of the generations, astonishing in realism (Karras et al., 2018; Brock et al., 2019). These methods have been applied to the setting of video prediction, but in general must settle for a trade-off between the diversity and the realism of the generated sequences. In this respect, video prediction is also a great playground, of fundamental interest, to develop methods that can model high-dimensional, complex and multi-modal, conditional distributions, which arise in a large number of other domains.

A second important challenge of the task resides in the multiple factors of variation in videos, that combine together and vary in time in a complex and intricate manner. These factors include: the presence, position, appearance and pose of various objects; characteristics of the scene, such as illumination and layout; and finally view-point and ego-motion of the camera. To capture the full distribution of plausible future sequences, models must not only disentangle these factors, but also anticipate how they can evolve and combine them adequately.

While RGB frames are easy to acquire and hold a lot of information, they are extremely challenging to predict. Furthermore, once predicted, they cannot be directly interpreted, which is why any machine learning pipeline will first proceed to feature extraction. Considering RGB intensities as low level features, we can question whether these features are the best trade-off between rich, spatially detailed representations of the state of the environment and ease of predictability, as well as ease of interpretability by downstream algorithms.

Let us recall the motivations that we have developed for forward modeling, to show in more detail that they do not require modeling of such low-level features. In the case where a predictive model is used to generate simulated

sequences, or for dynamic programming, current methods rely on extracting high-level features prior to predicting how good a state is, the reward that is to be obtained, or which action to take. These high-level features will need to undo some of the work we have put in when modeling each factor, since they may need to be invariant to some, *e.g.* illumination changes or appearance. In general, they will need to be semantically meaningful and possibly spatially detailed, *e.g.* to allow predictions relating to trajectories or interactions with objects. Control algorithms can also be used advantageously in abstract feature spaces, see *e.g.* (Watter et al., 2015). Concerning the use of forward models for intrinsic motivation, the curiosity-driven exploration approach proposed by Pathak et al., 2017 is also formulated in a high level feature space, bypassing the difficulties of directly predicting pixels, and, critically, learned to ignore the aspects of the environment the agent cannot affect. Directly using observation space for computing curiosity is shown to be significantly worse than using the learned embedding.

Predicting the RGB intensities of future frames seems therefore both overly complicated and in many ways not optimal. This is why, in this work, we set out to study candidates, in term of feature spaces, that make prediction easier, are more easily interpretable by downstream algorithms and remain rich, spatially detailed and easy to obtain.

1.3 Prediction in high-level feature spaces

Motivated by this, our goal in this thesis is to study video prediction in high-level, semantically meaningful and spatially detailed feature spaces, and to develop predictive models in these settings. These high-level representations should be sufficient to model important concepts, relating to object dynamics, scene dynamics and interactions between objects, which are necessary and sufficient for decision-making and planning in a wide variety of scenarios. While ultimately, modeling RGB features may enable learning more complete representations of the environment – under the condition that models can learn to model them well enough – we believe that focusing on such high-level representations will lead to methods that will be more widely applicable.

Semantic segmentation is one of the most complete forms of visual scene understanding. In this task, the goal is to label each pixel with the corresponding semantic label, *e.g. grass, car, dog, street,* etc. Human annotations are expensive to acquire, and this is even worse if we need annotations for each video frame. However, like several other recognition tasks, this task has seen important progress thanks to deep learning approaches. Methods have become extremely accurate on challenging datasets, with a large number of object classes, instances per image and in high resolution images. We therefore rely on state-of-the art semantic segmentation models to label all frames in videos. These automatically generated annotations meet the criteria that we have formulated, and constitute the first learning space for our predictive models.

However, semantic segmentation does not account for individual objects,

but rather lumps them together by assigning them to the same category label. Instance segmentation overcomes this shortcoming, by additionally associating with each pixel an instance label. It seems reasonable to assume that such an explicit notion of instance is important to model the concepts mentioned above, and accurately anticipate object motion and deformations by keeping track of their properties. Therefore, we extend our previous approach to predicting future instance segmentation.

1.4 Outline

The rest of this thesis is organised as follows. In Chapter 2, we begin by introducing the tasks of semantic and instance segmentations, as well as influential methods that have been proposed to address them. Then, we review the dominant frameworks in the field of generative modeling, to prepare for the presentation of video prediction methods. Finally, we provide additional context on how model-based RL employ forward models of the environment, and present the main directions that have been explored to address the task of video prediction, as well as applications of video prediction models.

In Chapter 3, we focus on the task of semantic segmentation itself, and propose a discriminative approach for semantic segmentation based on adversarial training. Inspired by recent successes in the field of generative modeling, we propose an adversarial training approach to train semantic segmentation models.

In Chapter 4, we introduce the novel task of predicting future frames in the space of semantic segmentation. Compared with the original video prediction task, we show that this task is better suited for autoregressive modeling and for predicting further into the future. We propose an autoregressive model which convincingly predicts segmentations up to half a second into the future. We conduct our experiments on a dataset for urban scene understanding, that consists of videos of much higher complexity than had been previously attempted by video prediction approaches, *e.g.* in terms of number of instances present on each image.

In Chapter 5, we extend our method to the more challenging problem of predicting future instance segmentation by forecasting convolutional features. To deal with a varying number of output labels per image, we develop a predictive model in the space of fixed-sized convolutional features of the Mask R-CNN instance segmentation model. We apply the "detection head" of Mask R-CNN on the predicted features to produce the instance segmentation of future frames. Besides producing semantically richer predictions, we show that this improves the quality of segmentations of individual object instances. Compared with concurrent video prediction approaches, our task allows us to produce visually pleasing segmentations at a high resolution for complex scenes involving a large number of instances, and with reasonable segmentation accuracy up to half a second ahead. Finally, in Chapter 6, we summarize our main contributions and present perspectives for future work.

Chapter 2

Background

2.1 Image segmentation

Image segmentation consists in breaking an image into meaningful regions, to represent it in a simplified and high-level form. As such, this task is extremely ambiguous, as we illustrate in Figure 2.1. We can try to characterize what makes a region "meaningful". Different definitions lead to different types of segmentation problems. Semantic segmentation consists in producing regions of pixels belonging to objects of the same category. Instance segmentation is more informative than semantic segmentation, in that regions should correspond to object instances of predefined categories. Both of these tasks are much less ambiguous and have seen tremendous progress in the past decade, thanks to large scale discriminative deep learning approaches.

Figure 2.1: Segmentation annotations performed by different subjects, from the Berkeley Segmentation Dataset 500 (Arbelaez et al., 2011). Important differences are due to the ambiguity of the task. To avoid distraction caused by inconsistencies between labels across annotations, we show edges between annotated regions rather than label-based colouring.

In the work we present in Chapter 4, we predict the segmentation of future frames as a proxy task towards providing the ability to anticipate future events. To do so, we build on recent progress both in semantic segmentation and instance segmentation. In the following, we present some of the most influential methods in both fields.

2.1.1 Semantic segmentation

Semantic segmentation is a visual scene understanding task formulated as a dense classification problem, where the goal is to predict a category label at each pixel in the input image. Just like in classification, the set of categories of interest are predefined beforehand and fixed.

More formally, given an image x, we wish to assign a label y_i to each spatial position i, representing the class of the object the pixel belongs to. We call $\mathcal{Y}$ the set of possible classes.

A distinction can be made between "things" and "stuff" to differentiate between classes of which the instances have "a specific size and shape", from those that are "defined by a homogeneous or repetitive pattern of fine-scale properties" but have "no specific or distinctive spatial extent or shape", as described by Forsyth et al., 1996. In other words, *things* have a well-defined shape and can for example be counted, whereas *stuff* cannot. The two kinds are generally needed to densely classify an image; for example, an image of a garden could be split into regions corresponding to on one side, the *stuff* classes, *e.g.* *grass*, *sky*, and on the other, the *thing* classes, *e.g.* *tree*, *toy*, *dog*. This distinction is sometimes used as a prior of the model *e.g.* in the works of Zheng et al., 2014; Heitz and Koller, 2008; Sun et al., 2014; Ladicky et al., 2010.

In most datasets, one of these categories is kept as a *catch-all-remaining* for all pixels that do not belong to any of the other categories, *e.g.* "background" or "other". A "void" class is sometimes introduced to indicate that the corresponding areas should not be used for learning; possibly because they have not been annotated for lack of time, or as an alternative to the *catch-all-remaining* class. In this case, predictions made for the pixels belonging to this class are not taken into account.

While most approaches focus on RGB data, extensions naturally apply for grayscale, stereo, RGB-D, as well as for 3D and video data where the main concerns are efficiency and cTime-Contrastive Networks: Self-Supervised Learning from Videoerency of the predictions.

Semantic segmentation can be seen as performing recognition and segmentation jointly. Segmentation as a low level task, requires methods to leverage fine and local details, for instance relying on image gradients. Recognition tasks on the other hand, typically involve building features that are invariant to view point, illumination and intra-class variations of (possibly deformable) objects, as well as being robust to occlusions. A very common strategy to achieve this is to aggregate local features into more robust representations, precisely losing some of the local information that is required by the low level task of segmentation. This makes one of the very specific challenges of the task of semantic

segmentation. Methods for recognition and segmentation have been tailored to these antagonistic goals; hence marrying them to benefit from their combined strengths, leveraging both local and global information, requires careful design.

One advantage of this task is that it is less ambiguous than segmentation alone, where several segmentations are often admissible. Ambiguity can still arise in the cases of semantic segmentation, from conditions in which the image was captured (*e.g.* over- or under-exposure, lens flare, blur, vignetting and so on). It can also result from wrong or ambiguous annotations, *e.g.* in the case of a photo of an object on a bus (should the photo be segmented as well or is it part of the bus ?), or in presence of (semi-)transparent occlusions like the silhouette of a person inside a vehicule. Imprecise annotations can additionally be caused by extremely complex occlusion borders, due to an animal seeking camouflage in a bush or a mesh fence over the image for example. Several other ambiguous scenarios could be encountered; however, these cases are rare and therefore they are usually ignored, so the task is generally assumed to be well-defined. This makes it well suited for discriminative approaches, which have made up the bulk of successful methods in the past two decades. Specifically, in recent years, as most other recognition tasks, the field has been dominated by deep learning approaches, typically relying on convolutional neural networks (CNNs).

Most of these approaches can be roughly described as belonging to one of two families: approaches that first segment the image, then classify the segments (Verbeek and Triggs, 2007; Csurka et al., 2008; He and Zemel, 2009; Gould et al., 2009; Kumar and Koller, 2010; Munoz et al., 2010; Lempitsky et al., 2011; Tighe and Lazebnik, 2013); and approaches that go in the reverse direction by first classifying each spatial position, and possibly refining the predictions, to account for spatial relationships (Shotton et al., 2006; Winn and Shotton, 2006; Krähenbühl and Koltun, 2011; Long et al., 2015; Chen et al., 2015; Schwing and Urtasun, 2015; Zheng et al., 2015; Noh et al., 2015; Yu and Koltun, 2016; Saxena and Verbeek, 2016). Hybrid approaches have also been explored, such as the works of Ladicky et al., 2009; Larlus and Jurie, 2009 and Farabet et al., 2013 The first family of methods were initially dominating, in particular due to their greater efficiency since feature extraction and prediction was restricted to a few thousands of segments at most. The second family of methods have been gradually favoured in recent years, as they naturally lend themselves to end-to-end learning with CNNs, and because dense prediction can be afforded efficiently by such methods. In both cases, conditional random fields (CRFs), presented below, provide a very natural framework for this task and have been extensively used.

We will focus on giving a self-contained presentation of some of the influential work in this area, including the approaches we build on. For a more exhaustive presentation, we refer the reader to the complementary works of Zhu et al., 2016 and Garcia-Garcia et al., 2017. The first is concerned with giving a broad picture of methods that have been proposed for segmentation tasks and the second focuses on the developments of recent deep learning approaches for semantic and instance segmentation.

First, we provide context on CRFs and describe how they can be applied for semantic segmentation. We also point to several formulations to illustrate the various forms that these models can take and the compromises one must make in their design. Next, we describe some of the strategies that have been used to leverage both global and local information in deep learning approaches.

Conditional random fields

A conditional random field is a probabilistic undirected graphical model that was introduced in 2001 by Lafferty et al., 2001 to discriminatively model a latent set of variables Y given an observed set of variables X. The core idea is to combine the strength of discriminative classification with graphical modeling. Discriminative classification generally leads to good performance but does not provide a formalism for specifying the relationships between the output variables. Graphical modeling on the other hand, allows us to do just that, either by specifying a set of conditional independence assumptions or equivalently a factorization that has to be admitted by the family of distributions we are considering. Conditional random fields lie at the intersection of these approaches, and enable us to incorporate prior knowledge into the model, to obtain better predictions than would be obtained by classifiers independently predicting each component of Y. We refer the interested reader to a full definition of these models in Appendix A.

Kumar and Hebert, 2003 were the first to use conditional random fields in computer vision. They applied these models to binary semantic segmentation, to detect man-made structures in images. As is standard when using random fields, they use two types of pairwise potentials: the first one encourages a smooth labelling, while the second adapts to the presence of discontinuities in the intensity values of the considered spatial positions. The connectivity is a simple 4-neighbourhood model.

For certain classes of pairwise potentials introduced by Krähenbühl and Koltun, 2011, mean-field inference is tractable in fully-connected CRFs with millions of variables, relying on the use of recent filter-based techniques. Such fully-connected CRFs have been found extremely effective in practice to recover fine details in the output maps. Chen et al., 2015 jointly leverage the strengths of these models with the discriminative power of a re-purposed CNN pre-trained for classification. They use the convolutional network to make coarse but globally accurate predictions, that can be refined by the CRF. Moreover, using a differentiable formulation of the mean-field iterations, Schwing and Urtasun, 2015 and Zheng et al., 2015 concurrently show that it is possible to train the CNN underlying the unary potentials in an integrated manner that takes into account the CRF inference during training.

Higher-order potentials have also been shown to be effective. He et al., 2004 introduce higher-order potentials learned from the data to eliminate impossible combinations *e.g. water-above-sky*. This incorporates the prior knowledge that physical laws restrict the set of possible layouts and spatial relationships between *objects* and *stuff*, depending on their nature, while allowing the actual combina-

tions to be learned from the data. Kohli et al., 2009 propose robust higher-order potentials based on label consistency across superpixels. Recent work by Arnab et al., 2016 has shown how specific classes of higher order potentials can be integrated in CNN-based segmentation models. While the parameters of these higher-order potentials can be learned, they are limited in number.

Deep learning methods

While early CNN-based semantic segmentation approaches were explicitly passing image patches through the CNN, see *e.g.* (Farabet et al., 2013), current state-of-the-art method indifferently use a fully convolutional approach (Long et al., 2015). This is more efficient, since it avoids redundant computation of low-level filters many times on pixels in overlapping patches. The extremely promising results demonstrated by the sole use of convolutional neural networks in the work of Long et al., 2015, lead a great number of works to focus on further improvement in the designs of architectures, so that the compromise between local and global information may be entirely learned from the data.

The field of view (FoV) (or receptive field) of a deep learning architecture is defined as the set of elements of the input image that can affect a prediction at any given spatial position, ignoring border effects. For example, an architecture that consists of a single convolutional layer has a FoV of size equal to the kernel size. The size of the FoV is an important characteristic, that helps us analyse how much context can be taken into account by an architecture. The size of the FoV increases linearly when stacking convolutional layers, and exponentially for pooling layers.

Since the work of Long et al., 2015, and just like in most other recognition tasks, it is widely acknowledged that competitive performance requires first pretrainining architectures for classification on a large dataset, such as ImageNet. This has favoured contributions that judiciously repurpose architectures initially designed for classification, to ease the fusion of global and local information, while avoiding dramatical changes to the architecture, so as to take the most advantage of the pretraining stage.

We present in the following several methods in this direction. We note that the presented methods can – and have been – used jointly.

Learned up-sampling Typical architectures involve a number of pooling steps, which can increase the receptive field size rapidly after several steps. As a result, however, the resolution of the output maps reduces, which means that a low-resolution label map is obtained. To obtain a label map of matching resolution, up-sampling is therefore necessary. This can be done using bi-linear interpolation or learned up-sampling filters (Long et al., 2015; Noh et al., 2015; Ronneberger et al., 2015).

Multi-scale architectures Multi-scale approaches are often used in computer vision, *e.g.* in coarse-to-fine approaches or to work across object scales. A generic approach is to apply a shared model to different scales of a Laplacian

pyramid of the input image and aggregate the predictions into a final prediction (Farabet et al., 2013). In CNNs, the very hierarchical nature of the feature representation leads to a more efficient set of multi-scale approaches. Intuitively, the high level features should provide coarse and semantically correct predictions, while the lower levels should provide less robust but better spatially localized cues. Aggregating intermediate predictions relying on these different levels can be done by relying on the use of *skip connections*, *i.e.* shallow convolutional maps from the features to intermediate predictions, [so called] because they take a shorter computational path from the current features to the output of the network, skipping the deeper path. Long et al., 2015 use skip connections to make parallel predictions corresponding to different levels of features and finally aggregate them. Ronneberger et al., 2015 iteratively refine the initial coarse predictions by leveraging the intermediate predictions coming from the lower level, giving rise to a "U-Net" where the name refers to the shape of the resulting architecture. A related family of methods use multi-resolution networks, *e.g.* (Saxena and Verbeek, 2016; Zhou et al., 2015).

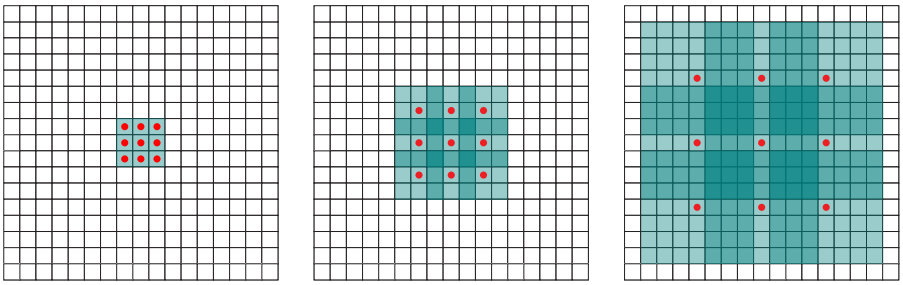
Shift-and-stitch and dilated convolutions The shift-and-stitch algorithm is a simple algorithm which involves shifting the input by one pixel S times in each spatial direction, where S denotes the network's sub-sampling ratio, to obtain slightly different predictions and interlace them together into a final prediction of the same size as the original input. This method has a high cost, but can be sped up using the "atrous" algorithm, avoiding redundant computation by adapting the kernels of the convolutional layers, as detailed by Long et al., 2015. This algorithm can be equivalently formulated in terms of dilated convolutions (Chen et al., 2015). In both works, it allows the authors to repurpose the classification architecture with large field of view into an architecture that performs dense prediction, with the same field of view.

We provide here the description given by Yu and Koltun, 2016 for dilated convolutions. To describe the computation that is done in a dilated convolution, we avoid considerations at the borders, and consider an infinite single channel feature map F. We can see it instead as a discrete function $F : \mathbb{Z}^2 \to \mathbb{R}$. Similarly, calling $\Omega_r = \mathbb{Z}^2 \cap [\![-r, r]\!]$, a filter k of size $(2r+1)^2$ can be seen as a function $k : \Omega_r \to \mathbb{R}$. The discrete convolution operator $*$ of F with k gives a new discrete function (or infinite feature map) $F * k$, defined in each point $\boldsymbol{p}$ by:

$$(F * k)(\boldsymbol{p}) = \sum_{S_r} F(\boldsymbol{s})k(\boldsymbol{t}), \tag{2.1}$$

where $S_r = \{(\boldsymbol{s}, \boldsymbol{t}) \in \mathbb{Z}^2 \times \Omega_r | \boldsymbol{s} + \boldsymbol{t} = \boldsymbol{p}\}$.

Dilated convolutions are a generalization of this operator, where for dilation parameter l, $S_r = \{(\boldsymbol{s}, \boldsymbol{t}) \in \mathbb{Z}^2 \times \Omega_r | \boldsymbol{s} + l\boldsymbol{t} = \boldsymbol{p}\}$. With $l = 1$, we recover the original definition. The definition of this operator for multi-channel feature maps follows directly from this. An intuitive illustration is provided in Figure 2.2



(a) (b) (c)

Figure 2.2: Dilated convolutions support exponential expansion of the receptive field without loss of resolution. Calling F_0 and F_1 respectively the input and output feature maps, F_1 is produced from F_0 (a) by a 1-dilated convolution: each element in F_1 has a receptive field of 3×3. (b) by a 2-dilated convolution; each element in F_2 has a receptive field of 7×7. (c) by a 4-dilated convolution; each element in F_3 has a receptive field of 15×15. The number of parameters associated with each layer is identical. When stacking layers of dilated convolutions, the receptive field grows exponentially while the number of parameters grows linearly. Figure from (Yu and Koltun, 2016).

Provided the adequate image-to-row algorithm (which now depends on the dilation parameter), dilated convolutions can leverage the same algorithms for fast matrix computation as normal convolutions ; there is simply an increase in computational complexity due to the fact that the resolution of feature maps is now maintained throughout the network.

Although the resolution of the input is recovered, this does not solve the problem that precise localization of borders is bound to be lost due to the pooling layers. Therefore, Yu and Koltun, 2016 propose a simpler and more accurate alternative. They note that dilated convolutions can be used to entirely replace pooling layers. Specifically they replace each of the two last pooling layer with pooling kernel of size p and its subsequent convolution layer by a dilated convolution layer with dilation parameter p. This allows the field of view to expand exponentially, but without aggregating values at different spatial positions.

To push the performance of their model, Yu and Koltun, 2016 additionally propose a context aggregation module, consisting of a stack of dilated convolutional layers, with increasing dilation parameter. This module is appended to the base network to increase its FoV significantly, leading to substancial performance gains on Pascal VOC 2012. Their final architecture for this dataset is called Dilation8, in reference to the eight layers of the context module. In the case of the Cityscapes dataset, due to the high image resolution of the Cityscapes dataset, they use a ten-layer context module, so the complete model is called Dilation10. We use this architecture in the work we present in Chapter 4.

Recurrent Networks Another line of work investigates the use of recurrent networks to predict globally coherent segmentations. Pinheiro and Collobert, 2014 propose to learn a convolutional network that is applied autoregressively to its own predictions, to iteratively refine initial predictions, by learning to correct its own errors. This architecture enables the network to have a large field of view, whil keeping the capacity of the model small. Byeon et al., 2015 propose 2D-Long Short-Term Memory (LSTM) layers, which they stack in their architecture, interleaved with convolutional layers. Here the recurrence is on the spatial dimension and is motivated by the modeling of complex spatial dependencies, through iterative propagation of the information of nearby predictions across four directions, from top-left, top-right, bottom-left and bottom-right. The outputs of the four LSTMs are aggregated in the convolutional layers. Visin et al., 2016 proposes a similar and simpler approach, where bidirectional LSTMs are used to model either horizontal or vertical relationships, in an alternating fashion. Because each LSTM processes each column (or row) independently, computation can be parallelized across columns (or rows). The obtained ReSeg network is used on top of a network pretrained for classification (in their case VGG) and learned upsampling layers map the predictions back to full resolution outputs.

2.1.2 Instance segmentation

Semantic segmentation provides a detailed semantic description of the scene. However, it does not distinguish between different object instances of the same class. The task of instance segmentation has been proposed to address this short-coming, by requiring for each pixel, the prediction of an additional label, corresponding to the instance the pixel belongs to.

An alternative view on this task is that it is an extension of object detection, where instead of predicting a bounding box for each detection, we require the method to predict a segmentation of the detected object. An important difference between the two views, is that the second one allows competing labels for a given spatial position. Depending on the down-stream application, post-processing may be required to determine unique predictions.

A note that instance segmentation usually focuses on *object* classes and neglects *stuff* classes, as previously defined in the context of semantic segmentation (see Section 2.1.1), leading to a partially segmented image. Joint methods for instance segmentation of *objects* and semantic segmentation of *stuff* have been explored, *e.g.* (Dai et al., 2015), and recently under the name of panoptic segmentation (Kirillov et al., 2017; Li et al., 2018c;b;d; de Geus et al., 2018; Kirillov et al., 2019; Xiong et al., 2019; Yang et al., 2019).

Detection-based versus segmentation-based

These two views have led approaches to build on methods developed either in the context of detection or of semantic segmentation. To remain coherent with terms used in the literature, we will call them respectively detection-based and

segmentation-based. We note however that this terminology can be slightly confusing, since the order between modules that aim to classify, segment or localise varies among methods of the same class, as we will see for example for object-centric methods. To better understand the difference, it can be helpful to notice that fundamentally, these classes of approaches differ in that detection-based approaches define object-centric losses, while segmentation-based approaches define pixel-centric losses. As a result, prior to post-processing, the outputs produced by the former are a (large) number of object-level feature volumes, while the latter produce a unified image-level feature volume.

Detection-based approaches In recent years, the field has been dominated by approaches that extend a detection-based framework, usually relying on object proposals. Hariharan et al., 2014 build on R-CNN (Girshick et al., 2016), but replace object proposal with region proposal, using a bottom-up, class agnostic process. For each proposal, features are extracted by a CNN from both the foreground region and its bounding box and classified using a SVM. Next, like most object detection pipelines, the method employs non maximum suppression (NMS), a post-processing step that consists in removing duplicate predictions for the same object, based on their intersection over union (IoU) with predictions of higher confidence. Following this, remaining region proposals are further refined in an iterative procedure between top-down classification and bottom-up segmentation methods. In follow up work, Hariharan et al., 2015 introduce *hypercolumns* to improve the segmentation refinement step: they use skip connections from each layer to the output, so that each pixel is described by features coming from low to high level layers, prior to classification. They also propose a more efficient framework, that proceeds first with an entire detection procedure, expands the set of retained boxes, segments them and scores features computed on the segmentation again. The gain in efficiency allows them to use a more recent deep architecture as feature extractor. The combination of their proposed hypercolumns, the deeper feature extractor, and the rescoring procedure, leads to a very large boost in performance. Dai et al., 2015 instead reverse feature extraction with masking: feature extraction is performed on the entire image, and each segment proposal is used to mask the features. Masked features are each passed to a multilayer perceptron (MLP) for classification. Interestingly, their framework is able to handle objects and *stuff* jointly. Li et al., 2016 also focus on improving the segmentation refinement step in a detection-followed-by-segmentation approach. They do so by learning a segmentation network that can be iteratively applied to refine predictions, serving as a learned, unconstrained inference procedure, similar in spirit to the approach of Pinheiro and Collobert, 2014 for semantic segmentation. Dai et al., 2016 are the first to propose an end-to-end learnable deep architecture for this task. They propose a cascaded structure, composed of three networks, that share their convolutional features. The first network is tasked with proposing class agnostic bounding boxes. Given these, the second extracts corresponding fixed-size representations and predicts a pixel-level mask. Finally, the fixed-size

representations, as well as masked copies, are given to a MLP for classification of each instance. To achieve end-to-end differentiability, in particular of the two latter networks with respect to the bounding box predictions, they introduce a differentiable warping layer to extract the fixed size representation. Overall their system is two orders of magnitude faster than previous approaches, and achieves a substantial boost in performance. Finally, He et al., 2017 build on Faster R-CNN (Ren et al., 2015) by adding a branch for predicting an object mask. To faithfully preserve spatial locations, they replace the RoI Pooling layer employed in the R-CNN family with a module called RoI Align, designed to allow pixel-to-pixel alignment between input and output. This method has enabled an outstanding boost in performance, and we build on it in the work we present in Chapter 5. We review this approach in detail in Section 2.1.2.

Object proposal methods, used in such proposal-based approaches, aim to maximize recall. They lead to a large number of redundant predictions, requiring post-processing steps to be filtered out. To avoid such post-processing steps, some works explore recurrent approaches, where at each time step, a new object is detected and segmented, and a score is predicted for use in a stopping condition. Romera-Paredes and Torr, 2016 are the first to propose this set up, and formulate a simple loss, that combines two terms. The first aims to maximize the maximum over matchings of the sum over pairs of matched predicted and ground truth masks of a soft version of the IoU. The second is trained to predict the termination of the sequence when the number of predictions matches the number of ground truth masks. Ren and Zemel, 2017 build on this framework, and propose a recurrent architecture that relies on an external memory which keeps track of the already segmented objects. Recurrence is both on the number of objects, and on the number of steps that the method employs, to refine its attention and look at several locations, before predicting a box location. In both of these works, the predicted bounding box coordinates are used to extract a patch, that is segmented and used to predict the termination score.

Segmentation-based approaches As we have mentioned earlier, a variety of works take the reverse direction, by starting from a semantic segmentation approach, and proposing methods to split the segmentations into instances. Liu et al., 2017a propose a sequential grouping of the foreground pixels predicted by semantic segmentation, by learning three successive networks. The first is tasked with predicting horizontal and vertical object breakpoints. Predictions are grouped into line segments, filling instances, that are grouped by a recurrent network into connected components. To deal with fragmented instances, *e.g.* due to occlusions, another recurrent network is used to merge the components into instances. Each instance is labeled by a simple majority vote over the spatial positions inside its mask. Arnab and Torr, 2017 propose a hybrid approach, where a semantic segmentation approach is augmented with a CRF, that produces a map of instance labels. Since the number of instances varies across images, this CRF is dynamically instanciated, using a fixed detection module to predict the number of instances, and whose predictions are used as

input to several unary potentials of the model. A fixed number of interations leads to the instance predictions, each labeled with the class originally predicted by the detector. Bai and Urtasun, 2017 propose to predict a modified watershed energy landscape such that each basin corresponds to a single instance, while all ridges are at the same height, using a convolutional network. This network is trained end-to-end to map predicted semantic segmentation and the input image to the energy map. A cut at a single energy level yields connected components, forming the instance predictions. Finally, Kong and Fowlkes, 2018 propose an end-to-end learnable framework for pixel-level grouping problems. A first stage embeds pixels, such that pixels belonging to the same group have high similarity, and pixels belonging to different groups have sufficiently low similarity. A second stage performs a fixed number of Mean Shift Clustering (Comaniciu and Meer, 2002) iterations on the embedding. These are differentiable, so the whole system can be trained end-to-end. Used as an object proposal method, far fewer proposals yield much higher recall than previous object proposal methods, including learned methods. This could therefore be used in a detection-based framework for instance segmentation. Instead, they propose to train the model to additionally predict semantic segmentation maps, leveraged in the majority voting strategy to transfer label predictions to the instances.

Discussion Detection-based approaches are naturally favoured by detection-based metrics, which are used for evaluation for some of the most common and challenging benchmarks, such as Microsoft COCO or Cityscapes.In particular, this metric does not require, nor evaluate the ability of the algorithm to resolve competing predictions for a given spatial position. Instead, it relies on the requirement that predicted masks should come with a confidence score. Arguably, such metrics are therefore unfair to segmentation-based metrics, and segmentation-based metrics have been proposed to address this. Instance segmentation methods should be compared based on a metric chosen according to whether competing predictions are a concern or not. If competing predictions are allowed, it should be noted that one of the strength of detection-based approaches, is that the tasks of classifying the object and segmenting it are usually decoupled. As a consequence, mask quality is not affected by competing classes, *e.g.* at the intersection between objects of different classes. He et al., 2017 show experimentally that predicting multinomial masks rather than binary masks for each class results in severe loss of performance.

Additionally, one can consider an amodal variant of the task of instance segmentation, as proposed by Zhu et al., 2017, where the full extent of the object is to be segmented – including potentially occluded parts. While this task is much more ambiguous than the original modal formulation, it may be a better high level representation, since it decouples factors of instance shapes and occlusions. In this case, detection-based approaches have a clear, natural advantage over segmentation-based approaches, since the latter are designed to produce a single prediction at each spatial position.

Nevertheless, pixel-centric losses usually enjoy simplicity, the tuning of fewer

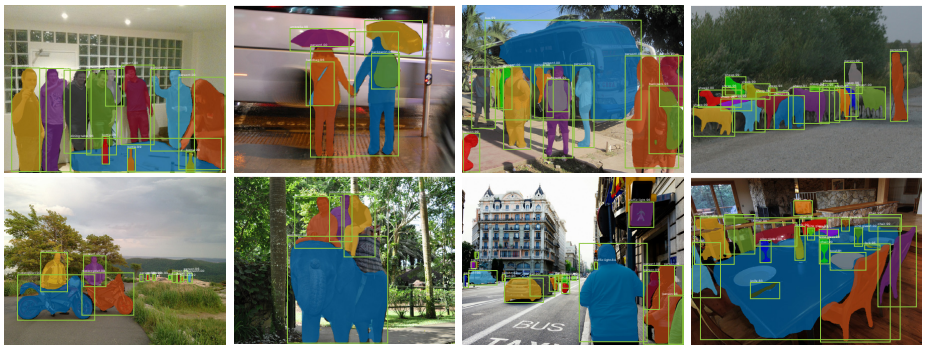


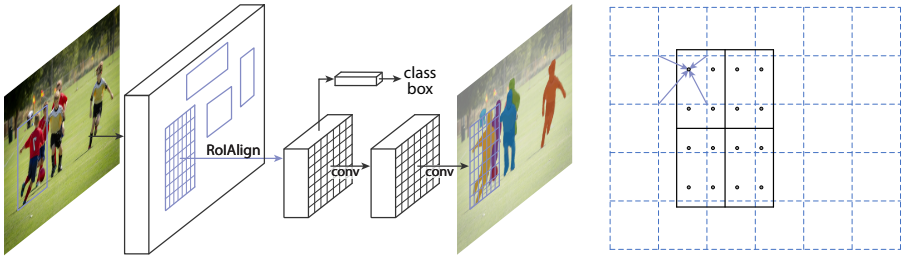
Figure 2.3: Instance segmentation predictions produced by Mask R-CNN on the COCO test set. Masks are shown in colour, and bounding box, category, and confidences are also shown. Figure from (He et al., 2017).

hyperparameters and fewer post-processing steps. Methods combining these strengths would therefore be desirable.

Mask R-CNN

Here we give a self-contained presentation of the Mask R-CNN framework, the current dominant detection-based approach. This framework was extensively demonstrated to be amenable to extremely impressive performance in the task of instance segmentation. In Figure 2.3, we show some predictions produced by this method on the COCO dataset (Lin et al., 2014). To give a complementary perspective to the one brought by the original papers, we describe the full approach on its own, without exhaustively mentioning earlier designs. For a good overview of the detection methods of the R-CNN family, we refer the reader to (Kalogeiton, 2017).

Framework Methods belonging to the R-CNN family for detection or instance segmentation implement a two-stage approach. The first stage performs *region proposal*, to predict bounding box coordinates for a large number of regions that are likely to contain an object, in a class-agnostic manner. Such predictions are called regions of interest (RoIs). They serve as a way to focus the capacity of the recognition pipeline of the second stage on relatively few promising areas of the image, in comparison with a sliding window technique. For each RoI, the second stage classifies the potential object, refines the bounding box coordinates to localize it, and in the case of Mask R-CNN, predicts a binary mask segmenting it. The two stages are complementary: while the first stage aims for rough localization, with high recall for the smallest number of proposals possible, the second devotes computation to fine localization and recognition on the selected regions. A note that both stages can share a deep convolutional network for feature extraction, with preliminary results indicating that this does not reduce performance.



(a) (b)

Figure 2.4: (a) The Mask R-CNN framework for instance segmentation. (b) RoIAlign: The dashed grid represents a feature map, the solid lines an RoI (with 2×2 bins in this example), and the dots the 4 sampling points in each bin. The value of each sampling point is computed by bilinear interpolation from the nearby grid points on the feature map. Both figures from (He et al., 2017).

Earlier approaches relied on low-level computer vision techniques for region proposal. Since its introduction as part of the Faster R-CNN framework by Ren et al., 2015, a Region Proposal Network (RPN) has been widely used instead to learn to perform this task, and as in the case of Mask R-CNN. Specifically, given a set of fixed-size bounding boxes, called *anchors*, of various ratios and scales, the network is trained to predict at each spatial position and anchor (i) a score, representing the probability that the anchor at this location contains an object, and (ii) coordinates, predicted relative to the anchor, as a coarse initial localization.

In the second stage, like its predecessors, Mask R-CNN extracts a fixed-size representation corresponding to each RoI. This representation is used as input to three *detection branches*, consisting of independent subnetworks and predicting respectively a class, bounding box coordinates (also relative to the RoI) and a segmentation mask. See Figure 2.4(a) for an illustration. At this stage, two elements contribute significantly to the performance of the system. The first is the introduction of their proposed RoIAlign layer, for the extraction of the fixed-size representation, given the image features and the coordinates of the RoI. This representation is obtained by bilinearly interpolating the nearby image features at regularly sampled locations in the RoI, as illustrated in Figure 2.4(b). Once the fixed-size mask has been predicted, it is resized back to RoI size. Hence, this operation is able to finely preserve the alignment between the content of the RoI and the predicted mask, in contrast with its predecessor RoI pooling (Ren et al., 2015), which first quantised the floating-number RoI coordinates to the granularity of the feature map, divided it into sub-windows that were themselves quantised, and finally pooled the feature values over each sub-window to obtain the desired representation. As a consequence, in comparison, RoIAlign brings huge improvements in mask accuracy; as well as a noticeable boost in box accuracy. The second element consists in decoupling between classification and segmentation. Rather than performing semantic segmentation of the RoI, a

binary mask is predicted, either in a class agnostic fashion, or using class-specific weights, determined by the predicted class, with minor performance gain for the second option. Compared with semantic segmentation, this prevents masks from competing across classes, and is shown to lead to a very large boost in performance.

Training As is now standard in computer vision, the two stages rely on features extracted by a deep convolutional network, whose architecture follows that of the current best performing architecture for image classification, in this case, ResNet or ResNeXt, and that has been pretrained on ImageNet. We note that the recent work of He et al., 2018b has recently challenged the conventional wisdom that pre-training on ImageNet helps achieve better performance than random initialization, and was shown to mainly improve training speed. In the case of Mask R-CNN, anchor boxes are labeled as positive if they intersect with any ground truth box with at least an IoU of 0.5, and negative otherwise. Similar heuristics have been used by its predecessors.

Given these labels, the region proposal network is trained using for each RoI i the combination of a classification loss and a regression loss:

$$L_i^{RPN}(\{p_i\},\{t_i\}) = L_{bce}(p_i, p_i^*) + p_i^* L_{reg}(t_i, t_i^*), \tag{2.2}$$

where L_{bce} is the binary cross entropy, $\{p_i\}$ and $\{t_i\}$ are respectively the predicted labels and relative coordinates for i, and $\{p_i^*\}$ and $\{t_i^*\}$ the corresponding ground truths and L_{reg} is the smooth ℓ_1 loss.

Note that the regression loss is only computed on RoIs labeled positively. Normalisation factors were originally used in (Ren et al., 2015) but have been removed since for simplification.

Minibatches are created in an image-centric way, by sampling an equal number N_r of RoIs per image, across a fixed number N_i of images, and respecting a given ratio between positives and negatives, *e.g.* 1:3, to avoid high imbalance towards negatives.

Coordinates are parameterized relatively to the anchor box, as follows:

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a, \tag{2.3}$$
$$t_w = \log(w/w_a), \quad t_h = \log(h/h_a), \tag{2.4}$$
$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a, \tag{2.5}$$
$$t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a), \tag{2.6}$$

where x, y, w and h denote the box's center coordinates and its witdh and height, and x, x_a, x^* denote respectively predicted, anchor and ground truth box, and likewise for the other notations.

The detection branches are also trained using a multi-task loss, to supervise each output, over each sampled RoI i. To simplify the expression, we drop the index i, yielding the following expression:

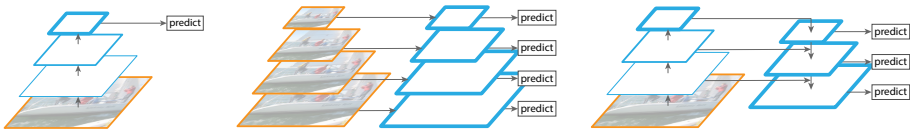


Figure 2.5: (a) Detection at single scale. (b) Detection is performed at each level of an image pyramid, more accurate but leading to increased computation. (c) Feature Pyramid Networks, both fast and accurate. Feature maps are indicate by blue outlines and thicker outlines denote semantically stronger features. Figure adapted from (Lin et al., 2017).

$$L^{DH} = L_{cls}(l, u) + [u \geq 1]L_{loc}(r^u, r^*) + [u \geq 1]L_{mask}(m^u, m^*), \tag{2.7}$$

where u is the ground truth class, equal to 0 for the catch-all background class; L_{cls} is the multi-class cross-entropy (MCE), L_{loc} is also a smooth ℓ_1 loss and L_{mask} is the average binary cross entropy over the RoI. Note that again, the two latter losses are only used if the RoI is labeled positively. l, r^u and m^u denote the predicted label, coordinates and masks, using the ground truth class u to choose the adequate predictions during training, when the branches are learned in class-specific fashion; r^* and m^* denote the corresponding ground truths. Coordinates are parametrized similarly to Equations (2.3) to (2.6), except that they are relative to the RoI's coordinates.

The authors preconize to perform training in an approximate end-to-end manner: the gradients of the detection branches with respect to the coordinates of the RoIs are not backpropagated in the RPN. This slightly improves the performance with respect to stage-wise training.

Inference During inference, the proposals are filtered with NMS, and the N_p (*e.g.* 1000) most confident RoIs are fed to the classification and localization branches. The predicted class is used to select the class-dependent box prediction. NMS is performed again on the refined bounding boxes, this time in terms of the classification score, and only the N_b (*e.g.* 100) most confident are kept, for segmentation, to keep the overhead of the mask prediction small. The refined bounding box predictions are used to extract improved fixed size representations, as input to the mask branch, and the final mask is again chosen according to the predicted class. Finally, the predicted masks are resized and binarised with threshold $\theta_m = 0.5$.

Multi-scale feature extraction Multi-scale approaches are extremely common in object detection, since objects can appear at a large range of scales. A widespread, simple procedure is to apply the detection approach on *image pyramids*, at the expense of a linear increase in computation with the chosen

number of scales. Lin et al., 2017 propose a generic approach to adapt a feature extractor to compute feature pyramids efficiently, forming a Feature Pyramid Network (FPN) A classification network is augmented with a top-down path to iteratively predict features at increasing resolutions. These features are not trained to match the features that would be obtained by applying the feature extractor on images of each resolution, but directly trained for the down-stream task (object proposal and/or instance segmentation). We illustrate this in Figure 2.5, comparing single scale detection, the multi-scale approach using a pyramid of images and FPNs. Both the RPN architecture and the detection branches are attached to each level of the FPN, sharing their weights. Training different weights was shown to lead to similar performance. As a consequence, features at the different levels effectively contain similar semantics. With the use of a FPN, the RPN needs anchors of different ratios only, because with its duplication across the different levels, its outputs effectively correspond to different image scales as well. Finally, to choose the feature level to extract from, feature extraction proceeds as if the feature pyramid had really been produced from an image pyramid, relying on the strategy introduced by He et al., 2014: given a RoI, the level is chosen such that the corresponding scaled candidate window at that (virtual) image resolution is most similar to the size of images used for pretraining the feature extractor on ImageNet (*i.e.* usually 224.)

FPNs have become a popular choice of feature extractor for object detection, and this architecture is used in the best performing instance of Mask R-CNN that was presented, both in speed and accuracy.

2.2 Deep generative modeling

A significant challenge of video prediction is to handle the inherent uncertainty of the task. As a consequence, an important line of work in video prediction focuses on applying and extending ideas developed in the context of image generative modeling to this setting.

Recent years have seen the emergence of methods that are able to leverage powerful deep learning techniques to learn generative models of images that have distinguished themselves by the unprecedented quality of the generations, astonishing in realism. Generations sampled from BigGAN, a state-of-the-art approach proposed by Brock et al., 2019 and based on generative adversarial networks (see Section 2.2.1), are show in Figure 2.6. In the present section, we review the dominant frameworks in this field, to prepare for the presentation of video prediction approaches in Section 2.3.3.

2.2.1 Generative adversarial networks

Generative adversarial networks (GANs) are an adversarial approach to learn deep generative models, proposed by Goodfellow et al., 2014. In this approach, two networks are trained jointly to compete against each other in a two-player minimax game. A generator G maps latent variables z drawn from a simple

Figure 2.6: Class-conditional samples generated by *BigGAN*. Figure from (Brock et al., 2019).

distribution p_z to the image space, thereby defining an implicit distribution p_g on image x. A discriminator D maps images sampled either from the data distribution p_{data} or from the model's distribution p_g to $[0, 1]$.

The goal is to find the two functions G and D that optimize the following objective:

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]. \tag{2.8}$$

For a fixed generator G, this objective can be interpreted as maximum likelihood training of the discriminator D, to classify between images coming from the real distribution as positives, and images coming from the generator as negatives. In this adversarial game, D tries to distinguish between the real data and the generated data, whereas the generator G's goal is to create samples that make the task as hard as possible for the discriminator. The global optimum of this objective is reached when $p_g = p_{data}$, where the generator has learned to synthesize images that are perfectly indistinguishable from the real images, so that the best discriminator can only predict 0.5 for all samples.

Usually, deep generative models relying on latent variables must resort either to approximate inference to estimate the log likelihood, as for variational autoencoders (see Section 2.2.2), or to approximation of its gradient, as for an earlier class of models called deep Boltzmann machines (Salakhutdinov and Hinton, 2009). The approach is motivated by the fact that the discriminator can be seen as defining a "variational" loss function, in the sense that the loss function of the generative model is defined by auxiliary parameters that are not part of the generative model, thereby entirely sidestepping these approximations.

The authors propose to parametrize D and G using MLPs and to optimize this objective using a algorithm that iterates between ascending the discriminator's stochastic gradient and descending the generator's stochastic gradient. In practice, an alternative objective is used for the generator, because it is observed to provide stronger gradients. Rather than minimizing $\log(1 - D(G(z)))$, G is trained to maximize $\log G(D(z))$. In other words, it is explicitly trained to maximize the probability that the discriminator will label the generated images as real ones.

Wasserstein GAN

Goodfellow et al., 2014 show that given an optimal discriminator, the criterion of the generator is equivalent to the Jensen-Shannon divergence between the true data distribution and the model distribution. Arjovsky et al., 2017 advocate for optimizing the Wasserstein distance rather than the Jensen-Shannon divergence for density modeling. Their approach, termed Wasserstein GAN (WGAN), brings small modifications of the original GAN training algorithm, to approximately optimize the formulation given by the Kantorovich-Rubinstein duality:

$$W(p_{data}, p_g) = \sup_{||f||_L \leq 1} \mathbb{E}_{x \sim p_{data}(x)}[f(x)] - \mathbb{E}_{x \sim p_g}[f(x)], \tag{2.9}$$

where the supremum is over all the 1-Lipschitz functions $f : X \to \mathbb{R}$. Instead of classifying the samples, the discriminator is hence trained to embed them far appart in $\mathbb{R}$, enforcing the values on the real samples to be large, and the values on the fake samples to be small, while the generator aims to reduce this distance, by maximizing the values taken on the fake samples. In practice, WGAN does not search over all the 1-Lipschitz functions $f : X \to \mathbb{R}$, but over the family given by the discriminator's architecture, and uses weight clipping for the discriminator, as a crude, but simple, way to enforce the Lipschitz constraint. They show that this leads to improved stability of training and correlation between the loss value and the generator's sample quality.

2.2.2 Variational autoencoders

Variational autoencoders (VAEs) were introduced concurrently by Kingma and Welling, 2014 and Rezende et al., 2014, to marry variational inference with deep learning. They owe their name to the training procedure, which we present after the generative process and summarize in Figure 2.7.

Generative process The generative model first draws a latent representation z from a chosen, simple prior distribution p_z. The likelihood $p_\theta(\cdot|z)$ is chosen to be a normal distribution, usually isotropic, with learned or fixed parameter σ, and whose mean $\mu_\theta(z)$ is output by a neural network that takes z as input. This neural network is called a probabilistic decoder, as it maps the latent representation to a distribution over images x, and is parametrized by θ.

Learning The objective function is derived from a variational inference approach to maximum likelihood estimation, chosen such that stochastic gradient descent can be used to learn the generative model over large datasets. First, the incomplete likelihood p_θ for a given sample $x^{(i)}$, under the described model is intractable, as computing it would require marginalizing the complete likelihood over the latent variable, as follows:

$$p_\theta(x^{(i)}) = \int_z p_\theta(x^{(i)}, z)dz = \int_z p_\theta(x^{(i)}|z)p_z(z)dz. \tag{2.10}$$

Additionally, the posterior distribution $p_\theta(\cdot|x^{(i)})$ is also intractable, so we cannot use the EM algorithm. Instead, provided any distribution q over z, a tractable lower bound can be formulated, relying on the following derivation:

$$D_{KL}(q(z)||p_\theta(z|x^{(i)})) = \int_z q(z) \log \frac{q(z)}{p_\theta(z|x^{(i)})} dz \tag{2.11}$$

$$= \int_z q(z) \log \frac{q(z)p_\theta(x^{(i)})}{p_\theta(x^{(i)}|z)p_z(z)} dz \tag{2.12}$$

$$= -\mathbb{E}_q[\log p_\theta(x^{(i)}|z)] + \log p_\theta(x^{(i)}) + D_{KL}(q(z)||p_z(z)), \tag{2.13}$$

where D_{KL} denotes the Kullback-Leibler (KL) divergence. Equation 2.12 uses Bayes' rule, and 2.13 relies on the fact that:

$$\int_z \log p_\theta(x^{(i)})q(z)dz = \log p_\theta(x^{(i)}) \int q(z)dz = \log p_\theta(x^{(i)}). \tag{2.14}$$

Rearranging the terms in Equation 2.13 yields:

$$\log p_\theta(x^{(i)}) - D_{KL}(q(z)||p_\theta(z|x^{(i)})) = \mathbb{E}_q[\log p_\theta(x^{(i)}|z)] - D_{KL}(q(z)||p_z(z)). \tag{2.15}$$

Given that the KL divergence is always non negative, we know that the right hand size is a lower bound on the incomplete data log likelihood $\log p_\theta(x^i)$. This bound is called the variational lower bound (VLB). We also see that the bound would be tight if $p_\theta(\cdot|x^{(i)}) = q$. Variational inference approximates the posterior in a predefined parametric family of simpler distributions q_ϕ, parametrized by ϕ. Whereas classic variational inference optimizes the parameters ϕ for each data point $x^{(i)}$, VAEs use a form of *amortized inference* (Gershman and Goodman, 2014), and learn an *inference network* that computes ϕ as a function of x. This improves the efficiency of inference, since it obviates the optimization step, offloading this to the inference network, that is trained at a cost that is amortized over the training samples. In the case of a VAE, the approximate posterior is chosen to be a Gaussian distribution with diagonal covariance, and given a sample $x^{(i)}$, the network learns to predict the mean and covariance of the distribution $q_\phi(\cdot|x^{(i)})$ that approximates the posterior.

The objective, denoted by $\mathcal{L}(\theta, \phi; x^{(i)})$ is now:

$$\mathcal{L}(\theta, \phi; x^{(i)}) = \mathbb{E}_{q_\phi}[\log p_\theta(x^{(i)}|z)] - D_{KL}(q_\phi(z|x^{(i)})||p_z(z)), \tag{2.16}$$

To learn the two models, jointly, we need to estimate the gradient of this objective over a minibatch, with respect to the parameters (θ, ϕ). The gradient

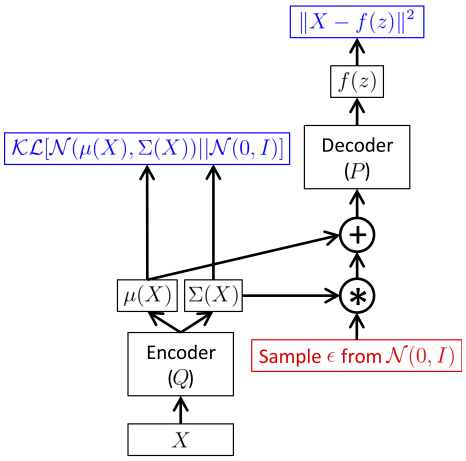


Figure 2.7: Training a VAE involves jointly training an encoder and a decoder. Red blocks are non-differentiable sampling operations. Blue blocks represent loss terms. Figure adapted from (Doersch, 2016).

of the second term can be computed analytically, but the first is more troublesome, due to the fact that the expectation is taken over a distribution that is parametrized by ϕ. To address this, VAEs introduce the *reparametrization trick*, relying on the fact that for a random variable z following a normal distribution $\mathcal{N}(\mu, \sigma^2 I)$, and provided that we sample ϵ from a standard normal distribution, we have, for any function f:

$$\mathbb{E}_z[f(z)] = \mathbb{E}_\epsilon[f(\mu + \sigma\epsilon)]. \tag{2.17}$$

This implies that we can derive a simple differentiable Monte Carlo estimator $\frac{1}{L}\sum_{l=1}^{L} f(\mu + \sigma\epsilon^{(l)})$ for the first term. In practice, a single sample is usually used in the estimation. Note that this trick can be extended to several families of distributions, *e.g.* families with tractable inverse CDF or "location-scale" families.

The resulting training procedure is summarized in Figure 2.7. From this, it is clear why the model is called a variational autoencoder: during training, each sample in a minibatch is encoded, its outputs serving to parametrize an approximate posterior, from which a latent representation is sampled, later decoded into the mean image. Because of the choice of a Gaussian distribution for the likelihood, the first term acts as a reconstruction loss, while the KL term acts as regularization on the encoder's outputs.

Attend, Infer and Repeat

Eslami et al., 2016 propose a structured image generative model, termed Attend, Infer and Repeat (AIR), based on VAEs. This model uses one latent

representation for each object present in the image, rather than a single one for the image as a whole. It is therefore a particularly interesting candidate for extension to video prediction, as such extensions could possess a built-in ability to track object properties. In this model, the inference network is a recurrent network that learns to predict parameters for the distributions of the presence of an additional object, of its position, and of its latent representation. The generator mirrors the inference network and reconstructs the image by summing the decoded patches adequately using the presence and position variables. The two networks are trained using the VLB. Gradients with respect to the parameters for the continuous latent variables are obtained with the *reparametrization trick*, and for the discrete with a *likelihood ratio estimator*, similar to (Mnih and Gregor, 2014). For binary images, or coloured images consisting of transparent objects against a uniform black background, this model learns to count the number of objects in the scene and localize them in an unsupervised manner.

2.2.3 Autoregressive models

In contrast with the previous classes of models, autoregressive models do not assume the existence of a latent representation, that would parametrize all factors of variation of the data. Instead, these models rely on the general product rule of probability:

$$p(x) = \prod_{d=1}^{D} p(x_d | x_{1:d-1}) \tag{2.18}$$

where D denotes the dimension of the data, x_d the $d-th$ element of x and $x_{1:-d-1}$ all of the preceding elements, according to any arbitrary order of the dimensions.

Autoregressive models parametrize each conditional distribution with a neural network. This approach was first proposed by Neal, 1992, relying on a fully visible network for each spatial position. Later, it was revisited by Bengio and Bengio, 1999, relying on single hidden-layer neural networks. In their case, this led to computational and memory costs squared in the input dimension. The architecture proposed by Larochelle and Murray, 2011 shares weights in the first layer of the neural network, bringing both costs down to linear. Finally, van den Oord et al., 2016 recently revived these ideas, proposing two deep architectures, PixelRNN and PixelCNN, whose parameters are shared across spatial positions. They are hence more expressive than previous approaches. Log-likelihood can be computed exactly for autoregressive models, and their architecture obtains state-of-the-art log-likelihood scores on held-out data.

While they introduce no independence assumption, the fact that autoregressive models lack a latent representation means that they do not lend themselves well to learning high-level representations in a self-supervised manner. These models also entail a slow sequential sampling process at test time. Perhaps as a consequence, these models have seen fewer applications in the context of video

prediction. Additionally, they are known to struggle to capture large structures. Hybrid approaches, combining autoregressive models with VAEs, have been explored in the works of Chen et al., 2017; Gulrajani et al., 2017; Lucas and Verbeek, 2018, to encode global image structure into latent variables while autoregressively modeling low level detail.

2.3 Anticipating future events

The ability to anticipate future events is an important prerequisite towards intelligent behaviour. In recent years, video prediction has been increasingly studied as a proxy task towards the goal of providing machines with this ability. In the field of reinforcement learning, forward modeling hold promise for more data-efficient algorithms, by leveraging a common representation of the world that can be learned using the rich observations provided by the environment and shared across tasks. In Section 2.3.1, we briefly review the basics of reinforcement learning, and present several model-based approaches. In Section 2.3.2, we present various ways in which forward modelling, and more generally self-supervision, can leverage observations and improve an agent's knowledge of the environment. In Section 2.3.3, we focus specifically on the task of video prediction and present the main directions that have been explored to address this task. Finally, in Section 2.3.4, we show that beyond these long term motivations, video prediction has allowed interesting applications.

2.3.1 A model of the world for planning and decision-making

Reinforcement learning basics

Reinforcement learning is a learning paradigm where an agent learns concepts and behaviours by interacting with an environment in which it evolves. At any time t, the environment provides the agent with observations corresponding to the current state x_t from state space $\mathcal{X}$, as well as a scalar *reward* signal r_t. In response, the agent chooses its actions $\{a_t\}_t$ in a pre-defined action space $\mathcal{A}$. An action leads to a new state of the environment and triggers new observations (x_{t+1}, r_{t+1}), possibly in a non-deterministic fashion. The goal of a reinforcement learning (RL) algorithm is for the agent to learn to take the actions that maximize its expected cumulative reward. RL algorithms specify a method for determining the *policy* to follow, in the form of a mapping π from states to actions; or to a probability distribution over actions, in the case of a non-deterministic policy. The algorithm aims to provide the policy yielding the greatest expected cumulative reward. Such a policy is called an *optimal policy*.

Classical RL algorithms rely deeply on the Markov assumption, that the future depends only on the current state and action (Sutton and Barto, 1998). In the simplest situation, we have a reasonable model of the way the environment evolves, conditionally to the action that was taken in a particular state, and of

the resulting observations. We call these the *dynamics of the system*. Together, the Markov assumption and the specification of the state space $\mathcal{X}$, the action space $\mathcal{A}$, the *transition probabilities* $p(x_{t+1}|x_t, a_t)$ and the reward function (or probability distribution) $r(x_{t+1}, x_t, a_t)$, form a Markov decision process (MDP) (Bellman, 1957), a formalism that is widely used. Usually, we consider that the transition probabilities and the reward do not depend on time; the MDP is then called *stationary*.

In this context, methods based on Dynamic Programming can be used to obtain an optimal policy. These methods define either or both of the following *value* functions. The *state value function* V^π maps each possible value x of the state x_t to the expected cumulative reward, given that we are in state x at time t and that we follow policy π. Likewise, the *state-action value function* Q^π maps each possible value (x, a) of the state-action pair (x_t, a_t) to the expected cumulative reward, given that we take action a in state x at time t and that we then follow policy π. The optimal policy π^* is a policy that maximizes the considered value function. The optimal value function is the value function for the optimal policy. The Bellman equations (Bellman, 1957) leverage the definitions of the value functions to express them in a recurrent relationship, involving the immediate reward and the expected value function for the next state, taking the form of a weighted average of the possible values, where the weights are given by the transition probabilities. This yields several algorithms for evaluating a given policy or for finding the optimal one.

In reinforcement learning settings however, usually we do not have such a model of the dynamics. We then have two possibilities: 1) we can attempt to learn one – this is called *forward modeling*; or 2) we can try to side-step this difficulty entirely, and directly learn a) a good policy or b) a value function from which to derive a policy. The first kind of methods are called *model-based*, while the second are called *model-free*.

Model-free approaches

To side-step the difficulty of learning a forward model, model-free approaches can instead attempt to learn directly a good policy. For example, *policy gradient* relies on a differentiable parametrization of the policy and estimates the gradient of a suitable objective function (*e.g.* the value function of the first state) with respect to the policy's parameters using the *likelihood ratio estimator* (Glynn, 1990). This gradient is used to perform gradient-based learning of the policy. Alternatively, model-free approaches can attempt to learn a good approximation to the optimal Q-function Q^*, from which a policy can be obtained by taking the action maximizing Q^*. This is called Q-learning and was originally proposed by Watkins and Dayan, 1992.

In fact, the *likelihood ratio estimator* also requires approximating Q^*. A simple approach is to use a Monte Carlo estimate, leading to the REINFORCE algorithm (Williams, 1992). Instead, an approximation to the optimal Q-function can be learned jointly with the policy, to reduce variance (but inducing bias), leading to a family of methods called *Actor-Critic methods*; see (Sutton and

Barto, 1998) for an extensive presentation.

Model-based approaches

It is generally acknowledged that when a large amount of samples can be used for training, model-based methods tend to perform worse than model-free methods, due to the intermediate modeling of the dynamics which can induce additional errors. Conversely, if it is possible to learn a good model efficiently, model-based methods can make up for the lack of data. For example, Deisenroth et al., 2009 show that compared to a state-of-the-art model-free approach, higher data-efficiency is achieved by learning the unknown transition dynamics for the pendulum swing up, a classical non linear control problem. Wahlström et al., 2015 show on the same task that model-based approaches can be scaled to high-dimensional inputs. Rather than using angle measurements, they propose to learn a forward model in the embedding learned by an autoencoder on raw images of the pendulum. They use this forward model in an online closed-loop control algorithm, that is able to reach the desired position of the pendulum, also provided in the form of an image. While this task may seem relatively restricted, it is an instance of a more general setting where an agent only has access to high-dimensional observations such as images, rather than direct access to a low-dimensional state, and must learn from a few trials. Such a setting is highly relevant to robotics, since it simply requires monitoring the robot with a video camera, from which the robot has to learn to solve tasks autonomously. In a similar flavour, Watter et al., 2015 propose to learn a forward model in the latent space of a VAE. They additionally constrain the latent space to be such that the dynamics are locally linear, thereby allowing more accurate use, on several complex problems, of effective control algorithms that rely on local linearization. More recently, Henaff et al., 2019 use an action-conditioned video prediction model to train a policy network, by unrolling the forward model and the policy network in time, and optimizing a differentiable objective at each time step, whose gradients are backpropated through time. The challenge is that the predictive model can produce arbitrary predictions outside the domain it was trained on, and hence cause actions leading to wrongly optimistic states. To avoid this, the authors add a reward concerned with encouraging actions that lead to low uncertainty predictions for the forward model.

Another advantage of model-based methods is that the forward model can be shared across different tasks. This might be similar to how humans learn to perform new tasks, by relying on their past experience and the representation they have built of the world and of its dynamics, to learn efficiently. Finn and Levine, 2017 demonstrate that an action-conditioned video prediction approach proposed in earlier work (Finn et al., 2016) can be used by a robot to plan the sequence of actions necessary to move objects into a desired position. The goal is specified by the user as a list of pixel initial locations (assumed to be part of the object) and a list of corresponding desired locations. The method leverages the implicit pixel flow predicted by the video prediction approach and supervised only by the video prediction task loss, by optimizing the maximum likelihood

of the desired location over the action sequences. Without any additional supervision, it is thus able to produce non-trivial action sequences. Furthermore it is shown to handle objects not seen during training.

Combinations of model-free and model-based approaches have been explored, where the forward model is used to improve upon a traditionally model-free methods, instead of being used inside a dynamic programming or control algorithm. For example, relaxing the requirement of full knowledge of the transition distributions, suppose that we have a generative model that can be used to generate simulated trajectories of actions and states. During training, such trajectories can be used in addition to real trajectories, in traditionally model-free approaches, as proposed in the famous Dyna algorithm (Sutton, 1991). Weber et al., 2017 augment an actor-critic method with a model-based path, which encodes simulated trajectories for each possible action, and provides these encodings as additional input to the policy network, leading to improved data efficiency and performance compared with the initial model-free method. In contrast to Dyna, the model is not used for data-augmentation, but to provide additional inputs, and therefore also used at test time. Simulation-based search also relies on simulated trajectories, but use the current state as the initial state for each simulation. This allows sampling-based estimation of the value function for a number of state-action pairs $(s_t, a_k)_{k=1}^K$, and the action maximizing the value function in this set is chosen and carried out. A similar idea is used in Monte Carlo Tree Search (Coulom, 2006), one of the core ingredients of the celebrated AlphaGo algorithm proposed by Silver et al., 2016.

2.3.2 Learning about the world through self-supervision

Model-based methods naturally provide a way to leverage the observations they receive from the environment. Besides applications in planning and control, the ability to anticipate future events can improve an agent's knowledge and representation of the environment.

First, a forward model can be used in several ways to improve exploration in a RL algorithm. For example, Oh et al., 2015 develop an action-conditional predictive model of video frames in Atari Games and use it in an *informed exploration* strategy, choosing the action leading to the predicted frame most dissimilar to the frames previously seen in a past moving temporal window. They show that this leads to improved performance for Deep Q-Network (Mnih et al., 2013) in several Atari games compared with random exploration.

In fact, it is possible to learn interesting behaviours *even in the absence of an external reward signal*. In this case, one or several reward signals are formulated and computed from the observations $\{x_t\}_t$ that the system receives. Such rewards are called *intrinsic*, as opposed to the *extrinsic* rewards considered in the classical case. The setting that relies only on intrinsic rewards, also called *intrinsic motivations*, is the analogue of unsupervised learning for reinforcement learning. We note however that while the formulation of such *intrinsic* rewards often relies on a forward model of the dynamics, this is not a requirement: this setting can also employ purely model-free approaches, as in the work of

Mohamed and Rezende, 2015.

Curiosity driven exploration is an extremely promising avenue for learning useful skills, in addition to or even in the absence of extrinsic rewards. Curiosity is an intrinsic reward defined by the error in a forward model's predictions. By considering this reward, the agent is encouraged to take actions from which it can learn the most, and indirectly encouraged to avoid actions that lead to terminal states (*e.g.* fatal ones) as well as to learn navigation skills that enable it to explore its environment. For instance, Pathak et al., 2017 show that compared with random exploration, the curiosity intrinsic reward they consider leads to better performance and fewer necessary interactions with the environment in sparse extrinsic reward settings like the Vizdoom environment. In the no reward setting, curiosity allows the agent to navigate efficiently (eg. without bumping into walls in the Vizdoom environment). In the Super Mario World, it learns to avoid fatal events, which would otherwise bound the cumulative intrinsic reward of the agent: getting killed due to a bad move is easily predictable and additionally shortens the time during which the agent can be "surprised". Additionally, this exploration policy is shown to generalize well in the next level. Their method thus enables the agent to learn basic and generalizable skills, including in the absence of an explicit goal. When such a goal is given, these skills can be efficiently specialized towards it.

Intrinsic motivation is also motivated in general by similarities with the way humans and animals learn. A good overview of several approaches, and how they relate to concepts in developmental psychology, can be found in (Oudeyer and Kaplan, 2007).

Beyond the formulation of intrinsic rewards, self-supervision can serve to define more general loss terms, to leverage the observations coming from the environment as additional free supervision, of potentially much higher dimensionality than the reward signal. The postulate that is made is that such loss terms will lead the model to develop a representation of the environment that will serve the variety of tasks for which it obtains only sparse and low dimensional reward signal. Mirowski et al., 2017 investigate two auxiliary loss terms by augmenting a state-of-the-art model-free policy-gradient method with two streams, that respectively output a depth map and prediction of loop closures of the agent's ego-motion. Both are learned in a supervised fashion. These streams share a representation with the policy network and their role is to improve it, so as to increase data-efficiency and task performance. Dosovitskiy and Koltun, 2017 do away with a reward-based formulation and cast the problem of learning to act into a self-supervised problem, where the agent is simply trained to predict the consequences of its actions, and actions are explicitly chosen according to which consequences are sought. They differentiate between high-dimensional *sensory* signal, and low-dimensional *measurements*; their key assumption being that any goal can be expressed as a function of the measurements. They train a model to predict the future measurements, at several fixed time intervals, given the current sensory signal, measurement and goal at hand. During training, exploration is achieved by following an ϵ-greedy policy: with probability ϵ, the next action is chosen randomly, and otherwise it is chosen greedily with respect

to the goal at hand. The ϵ value is annealed from 1 according to a fixed schedule: in the beginning, the policy is entirely random, and gradually, the trade-off is shifted towards exploitation. Remarkably, this simple formulation leads to state-of-the-art performance, particularly on complex tasks. They show good generalization properties of the trained models across goals and environments. Their approach supports training without a fixed goal and dynamically changing goals at test time. These results form a direct motivation to our goal of developing models that can better anticipate future events.

Applications of the learned representation in video recognition tasks

Self-supervised tasks have also been proposed to develop useful representations of video beyond the reinforcement learning setting. Srivastava et al., 2015 show that the representation learned through video prediction leads to improved performance for human action classification. Agrawal et al., 2015 propose to learn feature representation through the task of predicting the camera motion from pairs of images, and evaluate the obtained representations for various computer vision tasks. They show that in medium data regimes, these representations compare favourably with class-based supervision. Walker et al., 2016 learn to predict dense trajectories from still images, and show that the learned representation obtains competitive results on the Pascal VOC 2012 test dataset, without supervised image pre-training. In particular, they outperform the supervised method on the human class, which is well represented in the video dataset used to learn the representation, UCF 101. Vondrick et al., 2016b learn a GAN on videos, using a 3D convolutional discriminator on the generated videos. They show that the discriminator can be fine-tuned for action classification with $1/8^{th}$ of the data required by the same randomly initialized architecture to reach the same performance. Sermanet et al., 2018 learn a representation that is invariant to viewpoint and agent appearance, but varies with pose, by leveraging unlabelled videos recorded from multiple viewpoints. They show that this representation is useful for imitation learning. Vondrick et al., 2018 learn models for visual tracking by leveraging the temporal coherency of colour in video. Specifically, a predictive model is trained to copy the colour intensities from a reference frame to future gray-scale frames, using the original coloured frames as supervision. They show that the model learns to effectively track visual regions, foreground and human pose. Their method outperforms unsupervised baselines relying on optical flow, suggesting that the model is learning useful motion and instance features.

2.3.3 Video prediction methods

In the previous sections, we have shown that action-conditioned forward models can be used in a variety of ways: (i) to improve knowledge of the environment, through more efficient exploration or through better representations; (ii) to learn important skills; (iii) for use in planning and control. Arguably, a predictive model can always be conditioned on an action, by slight modification

to the architecture. Video prediction methods can therefore – and have been – proposed independently, without requiring a RL environment for training and evaluation. In the present section, we review several research directions that have been explored to address this problem.

Generative modeling of videos

A significant challenge of video prediction is to handle the inherent uncertainty of the task: For a given input sequence, there is generally a continuous spectrum of plausible output sequences. Worst, two perfectly plausible output sequences could be very far apart in pixel space, while a combination of the two, *e.g.* by weighted mean, would usually not correspond to a plausible sequence.

Following the terminology of Sutton and Barto, 1998, we could try to learn *a distribution model* mapping any value of the input sequence to the corresponding conditional probabilitiy distribution over the output sequences. From a probabilistic perspective, the task of video prediction can be viewed as learning a *sample model* that allows us to directly obtain samples from these distributions. This is easier than learning a distribution model – and is already by itself extremely challenging, as we will illustrate in the rest of this section. One of the core challenges we are underlining here is that in general, these conditional distributions will have several modes and their respective expectation will likely not correspond to one of of them. This forms a fundamental reason why we cannot expect to apply traditional supervised regression methods by themselves to solve this task.

As a consequence, an important line of work in video prediction focuses on applying ideas developed in the context of image generative modeling, either to learn such stochastic sample models or at least to formulate better training objectives for deterministic models.

In practice and in line with this analysis, Mathieu et al., 2016 observe that training with a squared ℓ_2 loss leads to blurry predictions. To address this, they propose a combination of losses: a ℓ_1 loss, whose optimal solution is the pixelwise conditional median, leading to sharper predictions than the ℓ_2 loss, a gradient difference loss (GDL), defined as the ℓ_1 loss over the gradients of the image, and a loss term based on adversarial training. Zhou and Berg, 2016 employed a similar training strategy for future frame predictions in time-lapse videos. Following this work, it has become common to use an adversarial loss term for this task, see for example the works of Bhattacharjee and Das, 2017; Vondrick and Torralba, 2017; Liu et al., 2018.

Vondrick et al., 2016b apply the GAN framework to generative modeling of videos. They use this model for single frame conditioned video prediction, by jointly learning an encoder with the GAN, that maps the input image to the latent space. This encoder is supervised by a squared ℓ_2 loss between the input and the first generated image. Jang et al., 2018 propose a GAN framework conditioned on an appearance variable on one hand, and a motion variable, including action labels, on the other. Tulyakov et al., 2018 learn a disentangled representation for appearance and motion – without requiring labels – thanks

to the generator architecture they propose. We further detail these approaches in a dedicated section.

Xue et al., 2016 rely on a VAE framework to learn a probabilistic model of the frame difference between the next and current frames, given the current. Babaeizadeh et al., 2018 propose a VAE-based model to model the future frame, given several past frames. An inference network predicts the parameters of an approximate posterior distribution over the latent representation, given the whole video. They rely on an architecture proposed by Finn et al., 2016, and condition it on a latent sample that comes from the posterior at train time, and from the prior at test time. They also propose a time-varying formulation where a new latent representation is sampled at each time step, both at train and test time. They show that their architecture is able to perform diverse predictions. They evaluate the best out of N predictions, and show that the metric increases significantly with N. Additionally, they show that on average, only 3 samples are enough to predict outcomes with higher quality than the deterministic baseline. Denton and Fergus, 2018 propose a closely related approach, but in contrast, use a more flexible inference mechanism, by inferring a different posterior distribution for every time step, conditioned on the past, instead of a single approximate distribution conditioned on the whole video. Additionally, they propose a learnable prior distribution, that is conditioned on the past frames, to replace the fixed prior. Indeed, the level of uncertainty should depend on the context: For instance, prediction aboard a vehicule driving in a city like Paris is always highly uncertain, but less so when one is driving on the beltway than in charming historical neighbourhoods with narrow streets and numerous intersections like Le Marais. Many sources of uncertainty should be expected: upcoming intersections are usually visible, just as are large potential occluders like trucks, or pedestrians waiting on the side of the road. Overall, Denton and Fergus, 2018 show that their method leads to sharper images, and can be trained using a simple training procedure, while Babaeizadeh et al., 2018 reported a three-stage procedure to reach the desired diversity.

In comparison with VAE-based approaches, GAN-based approaches produce more realistic results but struggle with diversity. We are not aware of any GAN based method that can produce diverse predictions from more than a single frame of history. The works of Jang et al., 2018; Xiong et al., 2018; Tulyakov et al., 2018 all condition on a single frame, and in this case, motion is largely underdetermined. All works relying on more than a single frame of context produce deterministic predictions (Xiong et al., 2018; Vondrick and Torralba, 2017). Avoiding mode collapse and mode dropping in GANs is an active research topic, see for example the works of Salimans et al., 2016; Che et al., 2017; Metz et al., 2017; Zhang et al., 2017; Nguyen et al., 2017; Srivastava et al., 2017; Tolstikhin et al., 2017; Ghosh et al., 2018; Lucas et al., 2018; Hoang et al., 2018; Lin et al., 2018; Xiao et al., 2018; Elfeki et al., 2018; Shmelkov et al., 2019. Applications of such methods could potentially lead to improvements in the diversity of video predictions made by GANs.

Inspired by the work of Larsen et al., 2016 in the space of images, Lee et al., 2018 therefore propose to combine the strength of VAE- and GAN- based

approaches. They formulate a conditional VAE framework, similar to (Denton and Fergus, 2018), and add adversarial loss terms on both the reconstructions and the generations. This yields small improvements both in terms of diversity and realism compared to the VAE baseline.

Ranzato et al., 2014 introduce the task of next frame prediction and address it using an autoregressive model on image patches. Inspired by language-modeling approaches relying on the rule chain of probability, given a history of patches, they propose to learn to predict a multinomial probability distribution over a vocabulary of patches, learned beforehand by clustering. Specifically, for each patch to predict, they first quantizing a larger surrounding patch in the past frames, embed it and feed it to a recurrent network, trained using the MCE to predict the index of the nearest neighbour of the future patch in the learned vocabulary. More closely inspired by recent autoregressive image models, Kalchbrenner et al., 2017 propose to predict the discrete multinomial distribution over the raw pixel intensity of each pixel, given all pixels in preceding frames, or above and to left in the current frame. Their architecture first models temporal dependencies with framewise convolutional encodings fed to a convolutional LSTM (ConvLSTM) (Shi et al., 2015). Next, the image autoregressive model of van den Oord et al., 2016 , conditioned on the previous outputs, is employed to capture dependencies across the spatial and colour dimensions.

A variety of learned perceptual losses

Inspired by approaches proposed in the context of style transfer (Gatys et al., 2016; Johnson et al., 2016) and image generation (Lamb et al., 2016; Dosovitskiy and Brox, 2016), several works propose various types of *perceptual losses*, *i.e.* reconstruction losses that are applied, not in the space of RGB intensities but instead in that of features extracted by a network that has been pretrained for an adequate auxiliary task. Liu et al., 2018 propose a loss that enforces consistency between the predictions of Flownet (Dosovitskiy et al., 2015a), a network pretrained for optical flow estimation of pairs of images. They take as input in both cases the last input frame and, on one hand, the predicted frame and on the other, the ground truth frame. Li et al., 2018e predict multiple time-step optical flow first, followed by a flow-to-frame synthesis phase. This second phase is trained using a perceptual loss, relying on VGG (Simonyan and Zisserman, 2015) for the feature extraction. Jang et al., 2018 propose a perceptual ranking module for conditional generation, that combines the idea of perceptual and deep ranking losses (Wang et al., 2014; Schroff et al., 2015), in a self-supervised setting. Specifically, they train two conditional discriminators, responsible respectively for motion and appearance. Each discriminator is used as a feature extractor. Following the ideas developed by Gatys et al., 2015, the Gram matrix of these features is used to represent the input. It is not directly the distance between the two representations that is optimized, as would be done in a standard perceptual loss, but a ranking loss relying on this distance, that aims to enforce similarity between real and fake videos conditioned on the

same variable and dissimilarity when conditioned on different variables. Xiong et al., 2018 propose a related adversarial ranking loss, in the context of a two-stage GAN approach. While the first stage is learned using the combination of a reconstruction and adversarial loss terms, the second uses in addition their proposed learned perceptual ranking loss. Specifically, the refinement network of the second stage is trained to minimize the distance on such Gram matrices between the fake videos from the second stage and the real videos, while increasing the distance between the fake videos from the first and second stages. A conceptual difference with the previous work is that they in turn train the discriminator to maximize the ranking loss, in addition to its usual discriminative objective. The refinement network is shown to largely improve the motion dynamics of the generations in comparison with the first stage only and they show visually pleasing long term generations of time-lapse videos of 32 frames.

Disentangling the factors of variation for a structured representation

Another challenge of video prediction lies in the multiple factors of variation in videos: position of different instances present in the scene, as well as their pose and appearance; position and viewpoint of the camera; illumination and content of the scene; and so on – all of which combine together and vary in time in a complex and intricate manner.

The postulate that the task of video prediction will necessarily lead to useful and meaningful representations implies that some kind of internal semantic representation of the scene will be developed. To put it more concretely, to anticipate the motion of an object, a perfect method will probably need to recognize some high level characteristics of the object and of the environment. Such characteristics could be the object's class and scene's layout. Like in recognition tasks, this abstract representation will need to be invariant to many of the aforementioned factors of variation; but additionally, a plausible evolution of each of these factors will need to be anticipated and their combination will need to allow the fine and dense prediction of a plausible set of future frames.

To break it down into a set of easier problems, a number of works aim at disentangling some of these factors of variation so as to learn a better representation, that can lead to improved prediction. Besides, when the motivation is to learn a good representation, it is also a way to design explicitly the representation so that each part is invariant to the other factors – which is often desirable.

Several works rely on the assumption that the semantic content of the scene stays roughly identical throughout the sequence, and decompose the representation into a content and a time varying representing, corresponding to motion or pose. Villegas et al., 2017a propose a two-stream architecture, where motion and content are separately encoded: the first stream takes as input the sequence of frame differences, and processes them with a ConvLSTM to encode the dynamics, while the second stream requires only the last observed frame, to encode the general content of the scene using a CNN. Denton and Birodkar, 2017 propose to separately encode content and pose information extracted from

each frame. To this end, they introduce an adversarial loss to prevent the pose features from being discriminable from one video to another, and constrain the content representation to vary slowly in time. This is another example of the strategy mentioned above. Tulyakov et al., 2018 also learn disentangled representations for content and motion, using a GAN framework. Different from the previous works, they sample a single content representation for each sequence, and map a random vector for each frame using recurrent network, to give a path in the motion representation space. Representations are concatenated and decoded per image. Two discriminators, one acting on each frame and another on each video, complement each other to train the generator. They show that their framework allows to keep one representation fixed while sampling the other, yielding convincing videos and allowing more controlled generation than previous GAN-based frameworks (Vondrick et al., 2016b).

Disentangling foreground and background has also been explored: Vondrick et al., 2016b rely on the assumption that the background is static, and use one stream to generate a fixed background image, and another to generate dynamic foreground and mask sequences, used to blend the foreground onto the background to obtain the final video.

A third direction is structured object-oriented representation learning. In principle this can be viewed as an extension of disentanglement between foreground and background. Kosiorek et al., 2018 propose a temporal extension of the object-centric VAE-based generative model of Eslami et al., 2016. This approach allows them to detect and track objects in an unsupervised fashion, to perform video prediction and to use the latent variables for solving high level simple tasks. They are still restricted to videos that consist of binary images. Zhu et al., 2018a propose an object-oriented action-conditioned video prediction method that decomposes the environment into objects and models inter-object relationships to model the dynamics. This method first performs semantic segmentation of the current image into a fixed number of dynamic and static classes of objects. These classes are not specifed explicitly beforehand; rather it is up to the model to learn to distinguish between different types of objects, based on the effect they have on other objects and on their own dynamics, thanks to specific weights dedicated to each channel for modeling the dynamics. Relying on these predictions, another network is charged with predicting a motion vector for each dynamic object, used to warp the image and compose it with a prediction of the background. A variety of auxiliary losses are used to enforce certain priors and help the training procedure.

Predicting intermediate image transformations

Several authors reparametrize the problem to predict transformations instead of raw pixel values. Following Reda et al., 2018, we distinguish between two kinds of transformations: *kernel-based* and *vector-based*. Kernel-based methods predict the kernel weights of convolutional layers – or of the analogous operation with untied weights across spatial position. These operations can be used directly on the image or on features extracted from it. Vector-based methods

are based on spatial transformer networks (STNs) (Jaderberg et al., 2015): a sampling grid is created based on the predicted parametric transformation (*e.g.* affine or pixelwise displacement), and then used to sample the input values to form the output.

Ranzato et al., 2014 and Mathieu et al., 2016 implement an optical flow baseline, that consists in warping the first two frames using the optical flow computed between them. Conceptually, it is a vector-based approach where the transformation is given for each pixel by the flow field. This method is shown to work well for the next frame, but subsequent frames contain significant artifacts. In Section 4.4.2, we will present an improvement of this baseline, first by introducing a conceptually different and much more accurate method for transforming frames; second by also transforming the optical flow to account for the displacement of physical points when predicting multiple steps.

On one hand, most patches in video are near-copies of patches in nearby existing frames, it should be easier to transform given frames rather than predict future frames directly. On the other, baselines relying on optical flow estimation are prone to errors, due to several reasons. First, they can arise due to challenges in optical flow estimation. Second, when relying on a vector-based approach, it is the future optical flow that must be estimated, which is harder than optical flow estimation. Third, some pixe predictions cannot be handled alone by optical flow, *e.g.* in the case of disocclusions (occurences where previously occluded regions become visible) or of content entering the scene. To avoid pitfalls of both direct RGB prediction and optical-flow-based methods, a natural approach is hence to predict intermediate pseudo-flow, used to tranform the inputs and only supervised by the video prediction task loss. With this motivation, Liu et al., 2017b propose to use a deep learning approach to learn to predict *deep voxel flow*, a 3D pseudo-optical flow field across space and time, that is used for trilinear extrapolation across the input video volume to produce the next frame. An interesting aspect of their method is that thanks to the 3D nature of their predictions, it can leverage more than the past two frames, to which the 2D optical flow fields are limited. They show that their multi-scale variant leads to sharper predictions than a direct RGB prediction method like that of Mathieu et al., 2016.

Inspired by layered motion representations (Wang and Adelson, 1993), Xue et al., 2016 propose a *cross-convolutional network* that encodes images and motion information as feature maps and convolutional kernels respectively. Specifically, the network encodes both the current frame and the difference between current and future frames to predict the weights of convolutional layers, each used on a different scale of a pyramid of feature maps extracted on various scales of the current frame. It is assumed that during learning, feature maps will learn to characterize image layers while the corresponding kernels will correspond to their motion.

Concurrently, Finn et al., 2016 introduce three transformation-based architectures for action-conditioned video prediction. The motivation is that this reformulation should lead models to focus on learning physics, rather than additionally modeling object appearance. The Dynamic Neural Advection (DNA)

network predicts a kernel, with weights summing to one, for each spatial position, so that the predicted pixel value is a convex combination of nearby pixel values in the current frame. A convolutional variant (CDNA) is proposed, similar to the work of Xue et al., 2016. The network predicts N kernels, but uses them directly on the input image; $N + 1$ masks summing to 1 at each position are additionally predicted, to compose the obtained images and the input, to produce the prediction. In case of static background, this allows the architecture to copy-paste several pixels easily. Finally, the Spatial Transformer Predictor outputs the parameters of multiple affine transformations, each applied to the input image, this time in a vector-based fashion. Similarly, masks are predicted for composition with the original image. All architecture are shown to have similar performance. Transformation are performed in a pixelwise manner for DNA, whereas CDNA and STP are argued to be more object-centric, as the transformations could correspond to motion of different objects. It is shown that these models are better able to generalize to unseen objects, compared to models that reconstruct the pixels directly.

Vondrick and Torralba, 2017 propose a related kernel-based approach. Different from CDNA, weights are not shared accross location, and they are applied to the input frame directly, to generate future frames by transforming the input pixels.

The work of Zhu et al., 2018a for object-centric representations is most similar to the spatial transformer predictors proposed by Finn et al., 2016, in that the input image is directly transformed in a vector-based fashion and composed with the background using a predicted mask. However, the architecture is more explicitly designed to model object motion and relationships between objects. Additionally, the model is expected to learn to detect static objects thanks to the effect they have on the dynamic objects.

Xu et al., 2018 explore similar ideas for a ConvLSTM, to allow the network to preserve fine details of objects like poles or traffic lights over multiple time step predictions. Kernel weights are predicted dynamically, based on past input features, and used to transform respectively input features, hidden state dynamics and memory cell dynamics. These dynamics are accumulated in the form of sequences of difference of past values. Finally the module combines the obtained representation to produce the usual input, forget, new memory cell state, output and new hidden state values.

Reda et al., 2018 propose a *Spatially-displaced convolutional* module, that combines the strength of *kernel-based* and *vector-based* transformers. Both types of transformation-based approaches are shown to improve upon a convolution-based encoder-decoder architecture like that of Villegas et al., 2017a, which tends to produce blurry results and checkerboard effects due to the deconvolutions. They observe that on one hand, *kernel-based* approaches produce visually pleasing results but do not scale for large motion, since they require increasing the kernel size. On the other hand, *vector-based* approaches do not have this issue, but lead to artifacts in the presence of disocclusions, in the form of speckled noise. The combination leads to crisp frames and consistent motion, in high resolution images. As a result, error propagation is much slower.

Processing a sequence of input frames

To process a sequence of input frames, one of the simplest approaches is to concatenate input frames along along the channel dimension to form the input. This approach has been widely used, *e.g.* in Mathieu et al., 2016; Vondrick and Torralba, 2017; Liu et al., 2017b; Bhattacharjee and Das, 2017. A concern with this is that important long-range information will be missing, when it is not comprised in the temporal context given as input to the model. Therefore, a number of works consider classical recurrent architectures that have been successful in previous sequence-to-sequence tasks, such as LSTMs (Hochreiter and Schmidhuber, 1997) or Gated Recurrent Units (GRUs) (Cho et al., 2014) and their convolutional extensions, respectively proposed by Shi et al., 2015 and N. Ballas, 2016. Another advantage of recurrent architectures is that they can be used flexibly on input sequences on varying length.

Srivastava et al., 2015 propose to stack LSTMs to form both an encoder and a decoder. Similarly, Finn et al., 2016 introduce a predictive autoencoder, composed of several layers of ConvLSTMs and deconvolutional LSTMs, with skip connections from the encoder to the decoder. This is also the base architecture of Babaeizadeh et al., 2018; Lee et al., 2018. Some works propose instead to use a LSTM in the representation space learned by an autoencoder (Zhou and Berg, 2016; Denton and Birodkar, 2017); or alternatively, a ConvLSTM if this space retains a spatial extent (Jang et al., 2018; Denton and Fergus, 2018; Xu et al., 2018). Finally, Tulyakov et al., 2018 employ a GRU to map random samples to a coherent random path in the motion space, which is then decoded with a fixed content representation in a GAN framework.

Multi-step prediction and compounding error

First, given a next frame prediction approach, a simple approach for multi time-step prediction is to use the model autoregressively: we drop the first input frame and append the newly predicted frame to the inputs, and use this as input for the subsequent prediction. This method is flexible in the number of time-steps to predict and has been used by various works (Mathieu et al., 2016; Villegas et al., 2017a; Reda et al., 2018; Jayaraman et al., 2019), but it is prone to error-accumulation: errors made in predictions will cause additional errors in the next predicted frames, leading to poor predictions after a number of time-steps that depends on the difficulty of the dataset. Back-propagation through time can be used to mitigate this problem.

Just like in the case of multiple inputs, one can also simply modify the model so that is learns to predict a batch of future frames instead of a single frame; simply amounting to multiply the number of output channels by the number of frames desired. This model does not lead to compounding errors, precisely due to its main weakness: it is not built to leverage the temporal structure of the data and share weights across time-steps. It is therefore an expensive way (in terms of capacity) of addressing the issue. Additionally, prediction is restricted to the chosen number of time steps. Examples of methods relying on

this include the works of Mathieu et al., 2016; Walker et al., 2016; Liu et al., 2017b; Bhattacharjee and Das, 2017.

Both in the cases of multiple inputs and outputs, 3D convolutions (Tran et al., 2015) provide a more flexible trade-off than the previous batch approach between model capacity and modeling of temporal long-range dependencies. Indeed, in the temporal domain, they are to the batch approach, what convolutions are to fully connected layers in the spatial domain. Hence, for a fixed dimension of output representation, their local temporal connectivity allows them to use a number of parameters that only depends on their kernel size, while the previous approach directly depends on the number of input or output frames. Just like convolutions however, they must be stacked to obtain a sufficient temporal field of view. Xiong et al., 2018; Li et al., 2018e rely on 3D convolutional autoencoders, while Vondrick et al., 2016a proposes a 3D GAN architecture. Reda et al., 2018 also uses a 3D convolutional autoencoder, but in single time-step setting, and chooses recursive prediction for multi-time step.

A note that contrary to convolutional networks that can be used on images of arbitrary size, to make predictions of proportional spatial dimensions, in this context, 3D convolutional networks are trained and used for a fixed number of input and output frames. This is due to the fact that the predictions are not equivariant in time, as they are in space: important differences in the last input frames will affect the predictions a lot more than if they occur in the first input frames. This is a specificity of the problem of video prediction, compared with video recognition tasks for example.

At the intersection between batch and 3D approaches, Vondrick and Torralba, 2017 decouple the spatial and temporal processing: first they employ a resolution-preserving convolutional encoder for the input clip, then they upsample temporally the representation using a temporal up-convolutional network.

Error propagation can be argued to be less severe in the case of recurrent networks, depending on the architecture. In some architectures, the recurrent layers require the encoding of the previous predictions as input, as in (Finn et al., 2016; Xu et al., 2018; Denton and Fergus, 2018; Babaeizadeh et al., 2018; Lee et al., 2018). In others, they do not, *e.g.* it is the case of Zhou and Berg, 2016; Denton and Birodkar, 2017; Jang et al., 2018; Tulyakov et al., 2018. In both cases, error-propagation should eventually happen in time, but the learned representation can be such that prediction is easier in this space, so that this phenomenon happens more slowly. In the first case, error-propagation can additionally happen in depth, *i.e.* throughout the encoding process, similarly to the original problem met in autoregressive prediction.

Oliu et al., 2018 propose "bijective Gated Recurrent Units", a variation on GRUs that defines a bidirectional mapping between input and output by duplicating the logic gates of the standard GRU. This "backward" function can be seen as a learned inverse on the standard "forward" function. Such modules can be stacked to obtain a recurrent autoencoder, mitigating error propagation for multiple time step prediction by avoiding the re-encoding of predictions. Additionally this leads computational and time savings. The authors also demonstrate interesting interpretability properties.

Villegas et al., 2017b address the problem of compounding errors in long term recursive pixel-level prediction, by proposing a hierarchical, two-stage approach, quite different in spirit from the previous approaches. First, prediction is performed at a high level, in a semantic space. Next, each future frame is predicted using an analogy: the future frame should be to the high level representation what the last input frame is to the last input representation. They apply this generic two-stage framework in the context of videos of human motion with static background. A pretrained CNN extracts the high level representations in the form of a sequence of vectors corresponding to estimated human poses (requiring image-level annotations). For the second step, they use the analogy-based CNN of Reed et al., 2015: two encoders embed in a shared space on one hand the given image, and on the other the corresponding pose and the desired pose; while a decoder maps the result of the desired pose, minus the given pose, plus the given image (or rather their representations) to the desired image.

Following the same philosophy, Vondrick and Torralba, 2017 propose to predict, for each time step, the kernel weights for transformation of the last input frame. This is in contrast with several transformation based architectures, which directly process the preceding frame. This requires larger capacity architectures, to account for larger potential motion, they argue that this is less a concern in the context of video prediction, since supervision can be obtained easily.

Jayaraman et al., 2019 propose an original reformulation of the task: they propose to predict of a future frame *without committing to a time offset at which the frame would be to occur*. To this end, they propose a minimum-over-time loss, which chooses as groundtruth frame, the frame belonging to the ground truth sequence, which minimizes the reconstruction loss for a given prediction. They extend this loss for autoregressive prediction. By predicting such "low uncertainty frames", the predictions are shown to be less blurry, leading to less compounding errors. When used in the context of intermediate frame prediction, they are likely to correspond to intermediate state that must be traversed in any case. The authors show that these predictions can be used in the context of hierarchical planning, as visual subgoals.

Leveraging context while preserving fine spatial structure

In general, concerning dense prediction tasks, the greater the context an approach can leverage, the more accurate its predictions will be; however, this must not come at the price of sacrificing fine local details and structure in the input, which would lead to unrealistic predictions. As we have seen in the context of semantic segmentation, methods must therefore fuse global and local information adequately. Hence many of the techniques presented in Section 2.1.1 to address these antagonistic goals remain relevant here.

The most common approach is to use an autoencoder with skip connections, similar to U-Net (Ronneberger et al., 2015). See for instance the works of Finn et al., 2016; Villegas et al., 2017a; Liu et al., 2017b; Denton and Birodkar, 2017; Lee et al., 2018; Babaeizadeh et al., 2018; Denton and Fergus, 2018; Jang

et al., 2018; Xiong et al., 2018; Liu et al., 2018; Reda et al., 2018. Alternatively, some works propose a multi-scale approach. Mathieu et al., 2016 run a resolution-preserving network on a down-sampled version of the inputs. The coarse prediction is upsampled and concatenated to the inputs at the original resolution if only two scales are used, or down-sampled to the second smallest chosen scale otherwise. A second network is used to predict refinements to the coarse prediction, forming the final prediction in the case of two scales, or yet another intermediate prediction to refine otherwise. This can be extended to any number of scales and refinement networks can be shared across scales. Loss terms are defined for each scale and optimized jointly. Xue et al., 2016 also use a multi-scale approach for their cross-convolutional network: features are extracted and kernels are predicted at each scale of a pyramid of images. After cross-convolution, the transformed feature maps are upsampled accordingly and combined.

Resolution-preserving architectures, leveraging dilated convolutions, have also been explored. Vondrick and Torralba, 2017 employ a dilated convolutional network on the concatenated frames to capture long-range spatial, followed by a temporal up-convolutional network that transforms the obtained feature maps into transformation parameters. These are used to transform the last input image into each of the multi time-step predictions. Kalchbrenner et al., 2017 also use a resolution-preserving convolutional encoder, followed by a ConvLSTM module and an image conditional autoregressive model.

Wang et al., 2017 propose a variant on stacked ConvLSTMs, where the memory is not constrained inside each LSTM unit, but instead shared across the stacked recurrent layers, as well as across the states. The rational is that more low and mid- level information related to visual appearance can be stored and are directly accessible by the higher layers for prediction, forming an alternative to skip connections. Although this allows preserving fine spatial details, the proposed architecture has a relatively small field of view, since the field of view expands only linearly with the number of layers, so it could still benefit from some of the previous approaches.

In fact, Byeon et al., 2018 observe that an unfortunate property of ConvLSTMs is that the field of view diminishes from the first input frame to the last input frame. This is undesirable, as it means that closer time frames contribute less information to the prediction. To address this issue, they introduce a video prediction model that takes into account the full context for the prediction at each spatial position. They do so by blending together the predictions of LSTM modules with pyramidal connection topology (Stollenga et al., 2015), each spanning one of the five possible directions: left to right, right to left, top to bottom, bottom to top and past to current. In comparison with convolutional (non recurrent) architectures, depth is used only to increase power of the representation, rather than to increase the context that the network has access to. Computation between the different LSTMs can be parallelised. Their architecture alone, with standard reconstruction loss and no disentangling of the representation, yields state-of-the-art result on next frame prediction across several challenging datasets.

Video prediction in other feature spaces

A number of works depart from prediction in the space of RGB intensities to forecast different features of the video. Often these reformulations retain motivations of the original task, but address a less challenging problem; so that the methods are more directly applicable. If necessary, in many cases, mapping back to the RGB space could be done, *e.g.* with the analogy-based approach proposed by Villegas et al., 2017b or with a direct mapping like in Li et al., 2018e.

Several methods propose to predict motion features. Walker et al., 2015 learn to predict the future motion of every pixel in the form of dense optical flow from a single frame, and show preliminary results when the network is used for long-range prediction of future motion. In subsequent work, Walker et al., 2016 perform forecasting with a VAE, predicting diverse future pixel trajectories up to one second, from static images. Similarly, Luo et al., 2017 employ a ConvLSTM architecture to predict sequences of up to eight frames of optical flow in RGB-D videos.

Future prediction of more abstract representations has also been considered in a variety of contexts, and very early on. Srivastava et al., 2015 already propose to perform prediction, either in the space of RGB patches, or in that of the activations of the VGG network (Simonyan and Zisserman, 2015), or in that of the temporal stream of the two-stream architecture for activity recognition (Simonyan and Zisserman, 2014). Kitani et al., 2012 predict future trajectories of people from semantic segmentation of an observed video frame, modeling potential destinations and transitory areas that are preferred or avoided. Lan et al., 2014 predict future human actions from automatically detected atomic actions. Lee et al., 2017 predict future object trajectories from past object tracks and object interactions. Vondrick et al., 2016a predict the activations of the last hidden layer of AlexNet (Krizhevsky et al., 2012) in future frames, and use these to anticipate objects and actions. Gui et al., 2018 use a recurrent encoder framework to forecast human motion up to a second in the future. Bhattacharyya et al., 2018 jointly predict ego-motion and people trajectories over the next second, in the form of bounding boxes. They additionally predict the uncertainty of their predictions. Vu et al., 2018 learn video representations relying on a novel memory module, and use them both to detect objects in video and to anticipate their positions in future frames. Abu Farha et al., 2018 anticipate future actions over long term horizons of up to five minutes. Pavllo et al., 2018 predict human pose sequences with improved short-term accuracy by predicting a quaternion representation for the joint rotations. An extension of their framework is proposed in Pavllo et al., 2019.

2.3.4 Applications of video prediction

Besides emerging applications in RL of forward modeling, and applications of self-supervised learning in video recognition tasks (see Sections 2.3.1 and 2.3.2), video prediction has seen other interesting applications.

Liu et al., 2018 show an application of video prediction in the context of anomaly detection. Their method learns to predict the next frame and thresholds a score based on the similarity between the prediction and the ground-truth frames to predict whether a frame is normal or not. This is because unusual events, such as the presence of bicycles or the occurence of a fight in a walking zone, will be harder to predict, so the corresponding frames' scores should be low. They show that this method obtains state-of-the-art performance on three benchmarks.

In the partial observation strategy game of StarCraft, Synnaeve et al., 2018 learn a predictive model of the full game state, both present and future, given past and current partial observations. The predictions are given to their rule based bot, minimally adapted to take them into account. While they seem to hurt a module concerned with choosing tactics, predictions five seconds into the future improve the performance of build actions, leading to a significant improvement in win rate.

Several works investigate the use of predictive models to forecast complex spatio-temporal phenomena like those occurring in natural physical processes. ConvLSTMs were initially proposed by Shi et al., 2015 in the context of precipitation nowcasting, that aims to predict the future rainfall intensity in a local region over a relatively short period of time. They use this module in a predictive autoencoder architecture, consisting of two stacks of ConvLSTM layers, and obtain state-of-the-art performance. Bezenac et al., 2018 propose a transformation-based predictive model incorporating prior scientific knowledge to forecast sea surface temperature, which itself has applications in weather forecasting. Ziat et al., 2017 propose a recurrent model and evaluate it on various spatio-temporal forecasting tasks: disease spread, wind speed and orientation, sea surface temperature and car traffic forecasting. These applications can be seen as video prediction in particular domains, and directly benefit from advances on this task.

Chapter 3

Semantic Segmentation using Adversarial Training

We are interested in semantic segmentation as a high level, yet spatially detailed representation for the content of images. In this chapter, we present an adversarial training approach to train semantic segmentation models, published at the NIPS Workshop on Adversarial Training, 2016 (Luc et al., 2016). Specifically, we train a convolutional semantic segmentation network along with a discriminator, trained to classify whether segmentation maps come from the ground truth or from the segmentation network. The motivation for our approach is that it can detect and correct higher-order inconsistencies between ground truth segmentation maps and the ones produced by the segmentation net. Our experiments show that our adversarial training approach leads to improved accuracy on the Stanford Background and PASCAL VOC 2012 datasets.

In Section 3.1, we give the motivations for our method. We discuss related work on adversarial training approaches, as well as recent segmentation models, in Section 3.2. We present our adversarial training approach and network architectures in Section 3.3 and our experimental results in Section 3.4. We conclude with a discussion in Section 3.5.

3.1 Introduction

As presented in Section 2.1.1, current state-of-the-art methods rely on CNN approaches, following early work using CNNs for this task by Grangier et al., 2009 and Farabet et al., 2013. Despite many differences in the CNN architectures, a common property across all these approaches is that all label variables are predicted independently from each other. This is the case at least during training; various post-processing approaches have been explored to reinforce spatial contiguity in the output label maps since the independent prediction model does not capture this explicitly.

We also discussed that conditional random fields have been extensively used

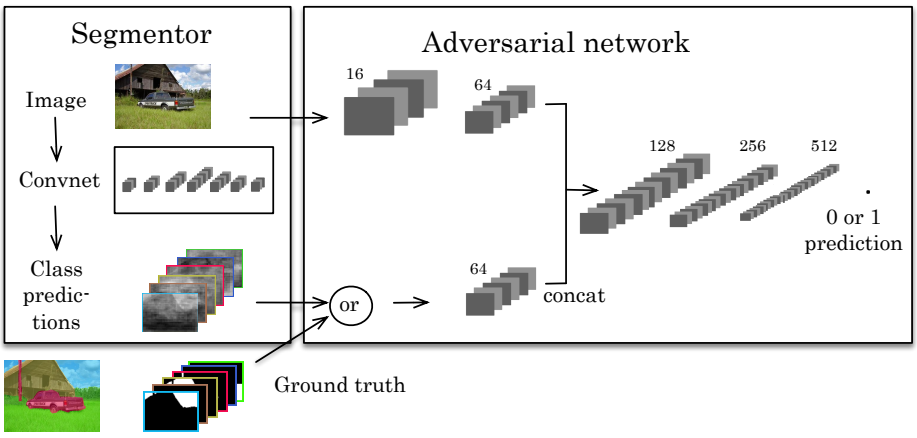


Figure 3.1: Overview of the proposed approach. Left: The segmentation net takes RGB image as input, and produces per-pixel class predictions. Right: The discriminator takes label map as input and produces class label (1=ground truth, or 0=synthetic). It optionally also takes RGB image as input.

to reinforce spatial contiguity in the output label maps since the independent prediction model does not capture this explicitly. Various forms for the graph connectivity and the pairwise or higher order potentials have been explored, where the former largely determines the expressivity of the latter, to lead to a tractable inference procedure.

We are interested in enforcing higher-order consistency without being limited to a very specific class of high-order potentials. Instead of seeking to directly integrate higher-order potentials in a CRF model, we explore an approach based on adversarial training inspired by GANs We optimize an objective function that combines a conventional multi-class cross-entropy loss with an adversarial term. The adversarial term encourages the segmentation model to produce label maps that cannot be distinguished from ground-truth ones by a discriminatively trained binary classification model. Since the discriminator can assess the joint configuration of many label variables, it can enforce forms of higher-order consistency that cannot be enforced using pair-wise terms, nor measured by a per-pixel cross-entropy loss. Our approach is illustrated in Figure 3.1.

The contributions of our work are the following:

1. We present, to the best of our knowledge, the first application of adversarial training to semantic segmentation.

2. The adversarial training approach enforces long-range spatial label contiguity, without adding complexity to the model used at test time.

3. Our experimental results on the Stanford Background and Pascal VOC 2012 dataset show that our approach leads to improved labeling accuracy.

3.2 Related work

Adversarial training As discussed in Section 2.2.1, generative adversarial networks (GANs) (Goodfellow et al., 2014) have led to state of the art results for generative image modeling. In follow-up work, Radford et al., 2016 present a number of architectural design choices that enable stable training of GANs that are able to synthesize realistic images. Similar to Goodfellow et al., 2014, they use deep "deconvolutional" networks that progressively construct the image by up-sampling, using essentially a reverse CNN architecture. Denton et al., 2015 use a Laplacian pyramid approach to learn a sequence of GAN models that successively generate images with finer details. Extensions of GANs for conditional modeling have been explored, *e.g.* for image tag prediction (Mirza and Osindero, 2014), face image generation conditioned on attributes (Gauthier, 2015), and for caption-based image synthesis (Reed et al., 2016).

Deep conditional generative models have also been defined in a non-stochastic manner, where for a given conditioning variable a single deterministic output is generated. For example, Dosovitskiy et al., 2015b developed deep generative image models where the conditioning variables encode the object class, viewpoint, and colour-related transformations. In this case a conventional regression loss can be used, since inference or integration on the conditioning variables is not needed. Dosovitskiy et al., 2015b train their models using an ℓ_2 regression loss on the target images.

In other examples, the conditioning variable takes the form of one or more input images. In Section 2.3.3, we presented several approaches that use a deterministic model for the problem of predicting the next frame in video given several preceding frames (Mathieu et al., 2016; Vondrick and Torralba, 2017; Xiong et al., 2018). Pathak et al., 2016 considered the problem of image inpainting, where the missing part of the images has to be predicted from the observed part. Such models are closely related to deep convolutional semantic segmentation models that deterministically produce a label probability map, conditioned on an input RGB image. In the latter two cases, a regression loss is combined with an adversarial loss term. The motivation in both cases is that per-pixel regression losses typically result in too blurry outputs, since they do not account for higher-order regularities in the output. Since the discriminator has access to large portions or the entire output image, it can be interpreted as a learned higher-order loss, which obviates the need to manually design higher-order loss terms. The work of Tarlow and Zemel, 2012 is related to this approach as they also suggested to learn with higher-order loss terms, while not including such higher-order terms in the predictive model to ensure efficient prediction.

Semantic Segmentation In our work, we build on recent advances in deep learning architectures for semantic segmentation, such as the ones presented in Section 2.1.1. Specifically, we investigate the robustness our approach with respect to segmentation architectures, using on one hand the multi-scale architecture of Farabet et al., 2013 and on the other the dilated convolutional architecture of Yu and Koltun, 2016.

As we have outlined in Section 2.1.1, most work that combines CNN unary label predictions with CRFs is based on models with pairwise or higher-order terms with few trainable parameters, *e.g.* (Arnab et al., 2016; Schwing and Urtasun, 2015; Zheng et al., 2015). An exception is the work of Lin et al., 2016 which uses a second CNN to learn data dependent pairwise terms. Another approach that exploits high-capacity trainable models to drive long-range label interactions is to use recurrent networks (Pinheiro and Collobert, 2014), where each iteration maps the input image and current label map to a new label map.

In comparison to these approaches our work has the following merits: (i) The discriminator has a high capacity, and is thus flexible enough to detect mismatches in a wide range of higher-order statistics between the model predictions and the ground-truth, without having to manually define these. (ii) Once trained, our model is efficient since it does not involve any higher-order terms or recurrence in the model itself.

3.3 Adversarial training for semantic segmentation networks

Inspired by GANs, our approach trains two networks, a segmentation model and a discriminator to compete against each other, as a way to enforce higher-order consistency in the predictions. We begin by describing our general framework for adversarial training of semantic segmentation models in Section 3.3.1. Next, we present the architectures used in our experiments in Section 3.3.2.

3.3.1 Adversarial training

We propose to use a hybrid loss function that is a weighted sum of two terms. The first is a multi-class cross-entropy term that encourages the segmentation model to predict the right class label at each pixel location independently. This loss is standard in state-of-the-art semantic segmentation models, see *e.g.* (Chen et al., 2015; Lin et al., 2016; Long et al., 2015; Noh et al., 2015). We use $s(\boldsymbol{x})$ to denote the class probability map over C classes of size $H \times W \times C$ that the segmentation model produces given an input RGB image $\boldsymbol{x}$ of size $H \times W \times 3$.

The second loss term is based on an auxiliary discriminative convolutional network. This loss term is large if the discriminator can differentiate the output of the segmentation network from ground-truth label maps. Since the discriminative CNN has a field-of-view that is either the entire image or a large portion of it, mismatches in the higher-order label statistics can be penalized by the adversarial loss term. Higher-order label statistics (such as *e.g.* the shape of a region of pixels labeled with a certain class, or whether the fraction of pixels in a region of a certain class exceeds a threshold) are not accessible by the standard per-pixel factorized loss function. We use $a(\boldsymbol{x}, \boldsymbol{y}) \in [0, 1]$ to denote the scalar probability with which the discriminator predicts that $\boldsymbol{y}$ is the ground truth label map of $\boldsymbol{x}$, as opposed to being a label map produced by the segmentation model $s(\cdot)$.

Given a data set of N training images $\boldsymbol{x}_n$ and a corresponding label maps $\boldsymbol{y}_n$, we define the loss as

$$\ell(\boldsymbol{\theta}_s, \boldsymbol{\theta}_a) = \sum_{n=1}^{N} \ell_{\text{mce}}(s(\boldsymbol{x}_n), \boldsymbol{y}_n) - \lambda \Big[\ell_{\text{bce}}(a(\boldsymbol{x}_n, \boldsymbol{y}_n), 1) + \ell_{\text{bce}}(a(\boldsymbol{x}_n, s(\boldsymbol{x}_n)), 0) \Big], \tag{3.1}$$

where $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_a$ denote the parameters of the segmentation model and discriminator respectively, and λ is a positive scalar used to balance the loss terms. In the above, $\ell_{\text{mce}}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_{i=1}^{H \times W} \sum_{c=1}^{C} y_{ic} \ln \hat{y}_{ic}$ denotes the multi-class cross-entropy loss for predictions $\hat{\boldsymbol{y}}$, which equals the negative log-likelihood of the target segmentation map $\boldsymbol{y}$ represented using a 1-hot encoding. Similarly, we use $\ell_{\text{bce}}(\hat{z}, z) = -\big[z \ln \hat{z} + (1 - z) \ln(1 - \hat{z})\big]$, the binary cross-entropy loss. We *minimize* the loss with respect to the parameters $\boldsymbol{\theta}_s$ of the segmentation model, while *maximizing* it w.r.t. the parameters $\boldsymbol{\theta}_a$ of the discriminator.

Training the discriminator. Since only the second term depends on the discriminator, training the discriminator is equivalent to *minimizing* the following binary classification loss

$$\sum_{n=1}^{N} \ell_{\text{bce}}(a(\boldsymbol{x}_n, \boldsymbol{y}_n), 1) + \ell_{\text{bce}}(a(\boldsymbol{x}_n, s(\boldsymbol{x}_n)), 0). \tag{3.2}$$

In our experiments we let $a(\cdot)$ take the form of a CNN. Below, in Section 3.3.2, we describe several variants for the discriminator's architecture, exploring different possibilities for the combination of the inputs and the field-of-view of the discriminator.

Training the segmentation model. Given the discriminator, the training of the segmentation model minimizes the multi-class cross-entropy loss, while at the same time degrading the performance of the discriminator. This encourages the segmentation model to produce segmentation maps that are hard to distinguish from ground-truth ones for the discriminator. The terms of the objective function Eq. (3.1) relevant to the segmentation model are

$$\sum_{n=1}^{N} \ell_{\text{mce}}(s(\boldsymbol{x}_n), \boldsymbol{y}_n) - \lambda \ell_{\text{bce}}(a(\boldsymbol{x}_n, s(\boldsymbol{x}_n)), 0). \tag{3.3}$$

We follow Goodfellow et al., 2014, and replace the term $-\lambda \ell_{\text{bce}}(a(\boldsymbol{x}_n, s(\boldsymbol{x}_n)), 0)$ with $+\lambda \ell_{\text{bce}}(a(\boldsymbol{x}_n, s(\boldsymbol{x}_n)), 1)$ when updating the segmentation model in practice. In other words: instead of minimizing the probability that the discriminator predicts $s(\boldsymbol{x}_n)$ to be synthetic label map for $\boldsymbol{x}_n$, we maximize the probability that the discriminator predicts it to be a ground truth map for $\boldsymbol{x}_n$. It is easy to show that $\ell_{\text{bce}}(a(\boldsymbol{x}_n, s(\boldsymbol{x}_n)), 0)$ and $-\ell_{\text{bce}}(a(\boldsymbol{x}_n, s(\boldsymbol{x}_n)), 1)$ share the same set of critical points. The rationale for this modified update is that it leads to a

stronger gradient signal when the discriminator makes accurate predictions on the synthetic/ground-truth nature of the label maps. Preliminary experiments confirmed that this is indeed important in practice to speedup training.

3.3.2 Network architectures

We now detail the architectures we used for our preliminary experiments on the Stanford Background dataset and large-scale experiments on the Pascal VOC 2012 segmentation benchmark.

Stanford Background dataset. For this dataset we used the multi-scale segmentation network of Farabet et al., 2013, and train it patch-wise from scratch. The discriminator takes as input a label map, and the corresponding RGB image. The label map is either the ground truth corresponding to the image, or produced by the segmentation net. The ground truth label maps are downsampled to match the output resolution of the segmentation net, and fed in a 1-hot encoding to the discriminator. We apply local contrast normalization to the RGB images before feeding them to either the discriminator or segmentation network. The architecture of the discriminator is shown in Figure 3.2. At first, two separate branches process the image and the label map, to allow different low level representations for the two different signals. We follow the observation of Pinheiro et al., 2016 that it is preferable to have roughly the same number of channels for each input signal, so as to avoid that one signal dominates the other when fed to subsequent layers. When fusing the two signal branches, we represent both inputs using 64 channels. The signals are then passed into another stack of convolutional and max-pooling layers, after which the binary class probability is produced by a sigmoid activation. The discriminator applies two max-pooling operators to the label maps. It is applied in a fully convolutional fashion to the inputs, resulting in a number synthetic/ground-truth predictions of the discriminator that is $4 \times 4 = 16$ times smaller than the number of predictions generated by the segmentation network.

Pascal VOC 2012 dataset. For this dataset we used the state-of-the-art Dilated-8 architecture of Yu and Koltun, 2016, and fine-tune the pre-trained model. This architecture is built upon the VGG-16 architecture of Simonyan and Zisserman, 2015, but does not include the two last max-pooling layers to maintain a higher resolution. The convolutions that follow the modified pooling operators are dilated with a factor of two for each preceding suppressed max-pooling layer. Following the last convolutional layer, a "context module" composed of eight convolutional layers with increasing dilation factors, is used to expand the network's field-of-view while maintaining the resolution of the feature maps. We explore three variants for the discriminator input, which we call respectively *Basic*, *Product* and *Scaling*.

In the first approach, *Basic*, we directly input the probability maps generated by the segmentation network. Preliminary experiments in this set-up show no

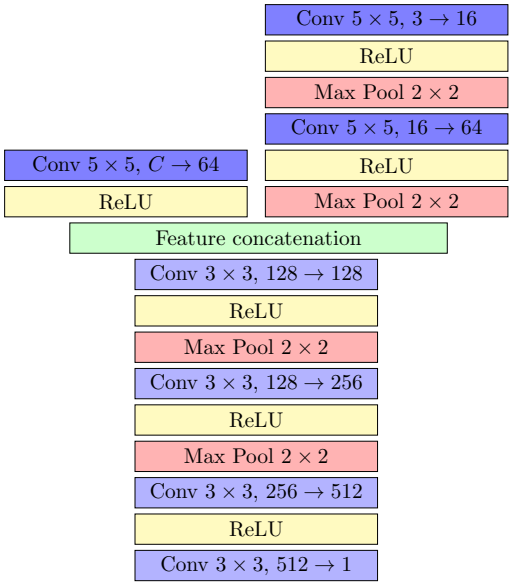


Figure 3.2: Discriminator architecture used for the Stanford Background dataset. The left and right branch processes respectively the class segmentations and the RGB image.

difference when adding the corresponding RGB image, we therefore do not use it for simplicity. One concern for this choice of the inputs is that the discriminator can potentially trivially distinguish the ground truth and generated label maps by detecting if the map consists of zeros and ones (one-hot coding of ground truth), or of values between zero and one (output of segmentation network).

In the second case, *Product*, we use the label maps to segment the input RGB image, and use it as input for the discriminator. In particular, we multiply the input image with each of the class probability maps (or ground truth), leading to a discriminator input with $3C$ channels. See Figure 3.3 for illustration. Prior to multiplication, the RGB image is mean-centered per pixel, in the same way as it is for input to the segmenting model.

In the third case, *Scaling*, we replace the 1-hot coding of the ground-truth label maps $\boldsymbol{y}$ with distributions over the labels $\overline{\boldsymbol{y}}$ that put at least mass τ at the correct label, but are otherwise as similar as possible (in terms of KL divergence) to the distributions produced by the segmenting network. For each spatial position i, given its ground-truth label l, we set the probability for that pixel and that label to be $\overline{\boldsymbol{y}}_{il} = \max(\tau, s(\boldsymbol{x})_{il})$, where $s(\boldsymbol{x})_{il}$ is the corresponding prediction of the segmentation net. For all other labels c we set $\overline{\boldsymbol{y}}_{ic} = s(\boldsymbol{x})_{ic}(1 - \overline{\boldsymbol{y}}_{il})/(1 - s(\boldsymbol{x})_{il})$, so that the label probabilities in $\overline{\boldsymbol{y}}$ sum to one for each pixel. In our experiments we have used $\tau = 0.9$.

We also need to handle the presence of unlabeled pixels in the ground-truth for the input to the discriminator. We adopt an approach similar to what is done

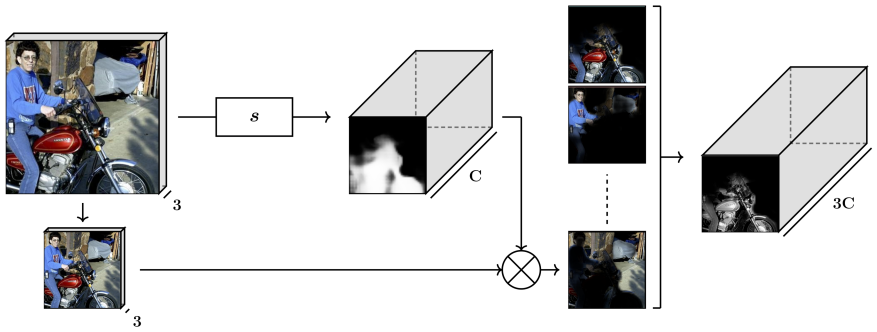


Figure 3.3: Illustration of using the product of the RGB input and the output of the segmentation network to generate input for the discriminator. The image is down-sampled by the stride of the segmentation network. The probability maps are then multiplied element-wise with each colour channel. These outputs are concatenated and form the input to the discriminator.

in image-wise training of the segmentation model with the cross entropy loss. We zero-out the values at the spatial positions of unlabeled pixels in both the ground-truth and the output of the segmentation network. We also zero-out the corresponding gradient values during the backward pass corresponding to the second term of Eq. 3.3. Indeed, those gradients do not correspond to predictions produced by the segmentation net, but to the presence of zeros introduced by this procedure, and should therefore be ignored.

We experiment with two architectures for the discriminator with different fields-of-view. The first architecture, we call *LargeFOV*, has a field-of-view of 34×34 pixels in the label map, whereas the second one, *SmallFOV*, has a field-of-view of 18×18. Note that this corresponds to a larger image region since the outputs of the segmentation net are eight times down-sampled with respect to the input image. We expect LargeFOV to be more effective to detect differences in patterns of relative position and co-occurrence of class labels over lager areas. Whereas we expect SmallFOV to focus on more fine local details, such as the sharpness and shape of class boundaries and spurious class labels.

Finally, we test a high capacity variant as well as a lighter one of each architecture. The smaller capacity variants have less channels per layer and respectively called LargeFOV-light and SmallFOV-light. Figure 3.4 summarizes the architectures used. The number of parameters of each model also depends on the input encoding.

3.4 Experimental evaluation results

Datasets. In our experiments we used two datasets. The Stanford Background dataset by Gould et al., 2009 contains 715 images of eight classes of

Conv 3×3	$C' \to 96$	$C' \to 96$
ReLU		
Conv 3×3	$96 \to 128$	$96 \to 128$
ReLU		
Conv 3×3	$128 \to 128$	$128 \to 128$
ReLU		
Max Pool 2×2		
Conv 3×3	$128 \to 128$	$128 \to 256$
ReLU		
Conv 3×3	$128 \to 128$	$256 \to 256$
ReLU		
Max Pool 2×2		
Conv 3×3	$128 \to 256$	$256 \to 512$
ReLU		
Conv 3×3	$256 \to 1$	$512 \to 1$
$C' = C$	$8,7.10^5$	$2,4.10^6$
$C' = 3C$	$9,1.10^5$	$2,4.10^6$

Conv 3×3	$C' \to 96$	$C' \to 96$
ReLU		
Conv 1×1	$96 \to 128$	$96 \to 128$
ReLU		
Max Pool 2×2		
Conv 3×3	$128 \to 128$	$128 \to 256$
ReLU		
Conv 1×1	$128 \to 128$	$256 \to 256$
ReLU		
Max Pool 2×2		
Conv 3×3	$128 \to 256$	$256 \to 512$
ReLU		
Conv 1×1	$256 \to 1$	$512 \to 1$
$C' = C$	$4,9.10^5$	$1,6.10^6$
$C' = 3C$	$5,3.10^5$	$1,6.10^6$

Figure 3.4: Summary of the architectures used for the discriminator, from left to right : LargeFOV-light, LargeFOV, SmallFOV-light, SmallFOV, with layers organized from top to bottom, along with the approximate number of parameters for each model. This number depends on the number of channels of the input (C channels for *Basic* and *Scaling* encodings, $3C$ channels for *Product* encoding.)

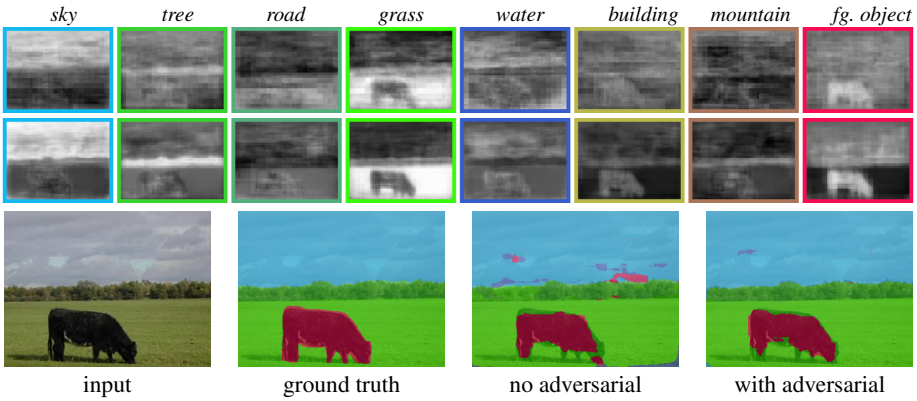


Figure 3.5: Segmentations on Stanford Background. Class probabilities without (first row) and with (second row) adversarial training. In the last row the class labels are superimposed on the image.

scene elements. We used the splits introduced by Gould et al., 2009: 573 images for training, 142 for testing. We train the multi-scale network of Farabet et al., 2013 using the same hyper-parameters. We have further split the training set into eight subsets, and we train on all subsets but one, which we use as our validation set to choose an appropriate weight λ, learning rate for the discriminator and to select the final model. The discriminator is trained using a weight $\lambda = 2$ and learning rate 10^{-3}. We compute the three standard performance measures: per class accuracy, per pixel accuracy, and the mean Intersection over Union (IoU) as defined by Everingham et al., 2015.

The second dataset is Pascal VOC 2012. As is common practice, we train our models on the dataset augmented with extra annotations from Hariharan et al., 2011, which gives a total of $10,582$ training images. For validation and test, we use the challenge's original $1,449$ validation images and $1,456$ test images.

In addition to the standard IoU metric, we also evaluate our models using the BF measure introduced by Csurka et al., 2013, to measure accuracy along object contours. This measure extends the Berkeley contour matching score of Martin et al., 2004, a commonly used metric in segmentation, to semantic segmentation. It is based on the closest match between boundary points in the prediction and the ground-truth segmentation. The tolerance in the distance error, used to decide whether a point has a match or not, is a factor θ times the length of the image diagonal. We choose θ such that this distance error tolerance is 5 pixels for the smallest image diagonal. In the original annotations of the dataset, however, the labels around the border of the objects are not given, since they are marked as 'void' and ignored in evaluation. Instead, to measure the mean BF, we use the 1,103 images out of 1,449 images of the validation set which were annotated on all pixels by Hariharan et al., 2011.

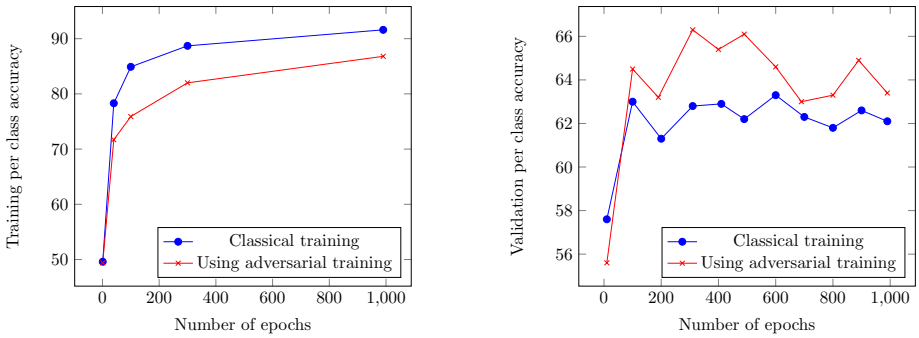


Figure 3.6: Per-class accuracy across training ephochs on the Stanford Background dataset on train data (left) and validation data (right), with and without adversarial training.

	Per-class acc.	Pixel acc.	Mean IoU
Standard	66.5	73.3	51.3
Adversarial	**68.7**	**75.2**	**54.3**

Table 3.1: Segmentation accuracy on the Stanford Background dataset.

Results on the Stanford Background dataset. In Figure 3.5 we give an illustration of the segmentations generated using this network with and without adversarial training. The adversarial training better enforces spatial consistency among the class labels. It smoothens and strengthens the class probabilities over large areas, see *e.g.* the probability maps for *sky* and *grass*, but also sharpens class boundaries, and removes spurious class labels across small areas.

In Figure 3.6 we display the evolution of the per-class prediction accuracy on the train and validation sets, using either standard or adversarial training. Note that the adversarial strategy results in less overfitting, *i.e.* generating a regularization effect, resulting in improved accuracy on validation data. This is also reflected in all three performance metrics on the test set, as reported in Table 3.1.

Results on Pascal VOC 2012. In order to set the learning rates of both the segmentation and discriminator, as well as the trade-off weight of the losses λ, we conduct a small grid search for each combination of discriminator architecture and input encoding.

To train the two networks, we first experimented with pre-training the discriminator before using the adversarial loss to fine-tune the segmentation network, so as to ensure that the adversarial loss is meaningful. This, however, led to the training to be rapidly instable after just a few epochs in many experiments. We found that training instead using an alternating scheme is more effective. We experimented with a *fast* alternating scheme, where we alternate

	Basic		*Product*		*Scaling*	
	mIOU	mBF	mIOU	mBF	mIOU	mBF
LargeFOV	**72.0**	47.2	**72.0**	**47.7**	**72.0**	**47.9**
SmallFOV	**72.0**	**47.6**	71.9	46.4	71.9	47.1
LargeFOV-light	**72.0**	47.0	**72.0**	**47.7**	**72.0**	47.4
SmallFOV-light	71.9	47.2	71.9	47.4	**72.0**	46.9

Table 3.2: Performance using different architectures and input encodings for the discriminator on the Pascal VOC 2012 validation set.

between updating the segmenting network's and the discriminator's weights at every iteration of SGD and a *slow* one, where we alternate only after 500 iterations of each. We found the second scheme to lead to the most stable training, and used it for the results on the validation set reported in Table 3.2. For details on the hyper-parameter search, and the final hyper-parameters used for each model, we refer the reader to the supplementary material of our publication (Luc et al., 2016).

We compare the results of adversarial training with a baseline consisting of fine-tuning of Dilated8 using the cross-entropy loss only. For the baseline we obtained a mean IoU of 71.8 and mean BF of 47.4. As shown in Table 3.2, we observe small but consistent gains for most adversarial training setups. In particular, the LargeFOV architecture is the most effective overall. Moreover, it is interesting to note that the different discriminator input encodings lead to comparable results. In fact, we found that for the basic input encoding, the discriminator does not succeed in perfectly separating ground-truth and predicted label maps, it rather has a discrimination accuracy that is comparable to that obtained with the other input encodings.

Using the evaluation server we also tested selected models on the Pascal VOC 2012 test set. For the baseline model we obtain (73.1), while for LargeFOV-Product and LargeFOV-Scaling we obtained 73.3 and 73.2 respectively. This confirms the small but consistent gains that we observed on the validation data.

3.5 Discussion

We have presented an adversarial approach to learn semantic segmentation models. In the original work of Goodfellow et al., 2014 the discriminator is used to define a proxy loss for a generative model in which the calculation of the cross-entropy loss is intractable. In contrast, the CNN-based segmentation models we use allow for tractable computation of the exact multi-class cross-entropy loss. In our work we use the discriminator as a "variational" loss, with adjustable parameters, to regularize the segmentation model by enforcing higher-order consistency in the factorized prediction model of the label variables. Methodologically, this approach is related to work by Roweis et al., 2002, where variational

inference was used in tractable linear-Gaussian mixture models to enforce consistency across multiple local dimension reduction models, and to work by Tarlow and Zemel, 2012 which learn models with higher-order loss terms, while not including such higher-order terms in the predictive model to ensure efficient inference.

To demonstrate the regularization property of adversarial training, we conducted experiments on the Standford Background dataset and the Pascal VOC 2012 dataset. Our results show that the adversarial training approach leads to improvements in semantic segmentation accuracy on both datasets. The gains in accuracy observed on the Stanford Background dataset are more pronounced. This is most likely due to higher risk of over fitting using this smaller data set, and also due to the more powerful segmentation architectures used for the Pascal VOC 2012 dataset.

Chapter 4

Predicting Future Semantic Segmentation

The ability to anticipate the future is an important attribute of intelligence. It is also of utmost importance in real-time systems, *e.g.* in robotics or autonomous driving, which depend on visual scene understanding for decision making. While prediction of the raw RGB pixel values in future video frames has been studied in previous work, in our work, we introduce the novel task of predicting semantic segmentations of future frames. Given a sequence of video frames, our goal is to predict segmentation maps of not yet observed video frames that lie up to a second or further in the future. We develop an autoregressive convolutional neural network that learns to iteratively generate multiple frames. Our results on the Cityscapes dataset show that directly predicting future segmentations is substantially better than predicting and then segmenting future RGB frames. Prediction results up to half a second in the future are visually convincing and are much more accurate than those of a baseline based on warping semantic segmentations using optical flow. The work presented in this chapter was published at ICCV 2017 (Luc et al., 2017).

In Section 4.1, we motivate our novel task and introduce our work in more detail. We discuss related work on video prediction in Section 4.2. We present our approach in Section 4.3 and our baselines and experimental results in Section 4.4. We conclude in Section 4.5.

4.1 Introduction

Anticipating future events is a key component of intelligent decision-making. In Chapter 2, we described how reinforcement learning (RL) approaches can leverage video prediction models for planning (Finn et al., 2016), control (Wahlström et al., 2015), simulation of sequences for data augmentation (Sutton, 1991) or additional inputs to a policy network (Weber et al., 2017). Decision making can also rely on intermediate prediction of the consequences of actions in a

given setting (Dosovitskiy and Koltun, 2017). Finally, such forward models can be used to formulate intrinsic reward signals, such as curiosity (Pathak et al., 2017); as well as to improve exploration (Oh et al., 2015). In a nutshell, video prediction has become an important research direction towards increasing the data-efficiency of RL algorithms. Doing so is crucial to transfer the impressive successes of RL methods, to this day demonstrated in simulated environments, to real-world applications, where interactions with the environment are typically slow, expensive or even dangerous, *e.g.* in the case of robotic systems and autonomous vehicles.

While humans can predict vehicle or pedestrian trajectories effortlessly and at the reflex level, it remains an open challenge for current computer vision systems. An application which directly benefits from our work is autonomous driving. In this domain, approaches are either based on a number of semantic decompositions such as road and obstacle detection, or directly learn a mapping from visual input to driving instructions end-to-end. Recent work from Mobileye (Shalev-Shwartz and Shashua, 2016) demonstrated an advantage of the semantic abstraction approach in lowering the required amount of training data and decreasing the probability of failure. Other work by Shalev-Shwartz et al., 2016 uses future prediction to facilitate long-term planning problems and forms a direct motivation for our work.

As discussed in Chapter 2, the task of predicting future RGB video frames given preceding ones has recently received significant attention. Modeling raw RGB intensities is, however, overly complicated as compared to predicting future high-level scene properties, while the latter is sufficient for many applications. Such high-level future prediction has been studied in various forms, *e.g.* by explicitly forecasting trajectories of people and other objects in future video frames (Alahi et al., 2014; Fouhey and Zitnick, 2014; Hoai and De la Torre, 2014; Kitani et al., 2012; Lan et al., 2014; Pei et al., 2011). In the work we present here, we do not explicitly model objects or other scene elements, but instead model the dynamics of semantic segmentation maps of object categories with convolutional neural networks. Semantic segmentation is one of the most complete forms of visual scene understanding, where the goal is to label each pixel with the corresponding semantic label (*e.g.*, *tree, pedestrian, car*, *etc.*). In our work, we build upon the recent progress in semantic segmentation, generative modeling and video prediction (Farabet et al., 2013; Long et al., 2015; Chen et al., 2015; Yu and Koltun, 2016; Goodfellow et al., 2014; Mathieu et al., 2016; Patraucean et al., 2016; Arjovsky et al., 2017), and develop models to predict the semantic segmentation of future video frames, given several preceding frames. See Figure 4.1 for an illustration.

The pixel-level annotations needed for semantic segmentation are expensive to acquire, and this is even worse if we need annotations for each video frame. To alleviate this issue we rely on the state-of-the-art semantic image segmentation model of (Yu and Koltun, 2016) to label all frames in videos, and then learn our future segmentation prediction models from these automatically generated annotations.

We systematically study the effect of using RGB frames and/or segmen-

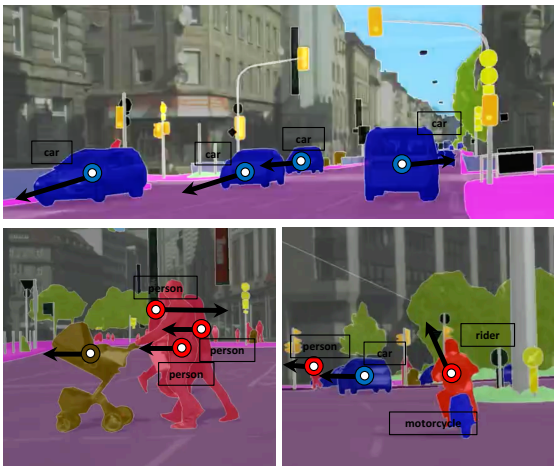


Figure 4.1: Our models learn semantic-level scene dynamics to predict semantic segmentations of unobserved future frames given several past frames.

tations as inputs and targets for our models and the impact of various loss functions. Our experiments on the Cityscapes dataset (Cordts et al., 2016) suggest that it is advantageous to directly predict future frames at the abstract semantic-level, rather than to predict the low-level RGB appearance of future frames and then to apply a semantic segmentation model on these. By moving away from raw RGB predictions and modeling pixel-level object labels instead, the network's modeling capacity seems better allocated to learn basic physics and object interaction dynamics.

In this work we make two contributions:

- We introduce the novel task of predicting future frames in the space of semantic segmentation. Compared with prediction of the RGB intensities, we show that we can predict further into the future, and hence model more interesting distributions.
- We propose an autoregressive model which convincingly predicts segmentations up to 0.5 seconds into the future. The mean IoU of our predictions reaches two thirds of the one obtained by the method used to automatically generate the dense video annotations.

Our approach does not require extremely costly temporally dense video annotation and its genericity allows different architectures for still-image segmentation and future segmentation prediction to be swapped in.

4.2 Related Work

Amongst the video prediction methods presented in Section 2.3.3, works that propose to shift the video prediction task to another output space are partic-

ularly relevant to the work we present here. Kitani et al., 2012 predict future trajectories of people from semantic segmentation of an observed video frame, modeling potential destinations and transitory areas that are preferred or avoided. Lan et al., 2014 predict future human actions from automatically detected atomic actions. Lee et al., 2017 predict future object trajectories from past object tracks and object interactions. Several methods produce motion features, such as dense optical flow Walker et al., 2015; Luo et al., 2017 or future pixel trajectory Walker et al., 2016. Future prediction of more abstract representations has also been considered in a variety of contexts: Srivastava et al., 2015 and Vondrick et al., 2016a propose to predict the high level activations of networks trained for image or video classification. Our work distinguishes itself by predicting in a space of features that are both semantically strong, like the high level features of a classification network, and detailed spatially, like the motion features.

Our work is also related to approaches that attempt to mitigate rapid error propagation in video prediction. In concurrent work, Villegas et al., 2017b propose a hierarchical two-stage approach, where prediction is first performed in a high-level semantic space, and mapped back to the pixel space using an analogy-based approach. Oliu et al., 2018 propose a new recurrent module that learns simultaneously an approximate inverse function to itself. Stacked together, these modules form a recurrent autoencoder that does not require re-encoding previous predictions for predicting multiple time-steps. Orthogonal to these directions, we study a novel video prediction task, and show experimentally that this task is better suited for autoregressive modeling and for predicting further ahead into the future.

Perceptual losses, introduced in Section 2.3.3, are also related to our approach. While the recent works of Li et al., 2018e; Liu et al., 2018 rely on pre-trained networks to formulate semantically stronger reconstruction losses, we learn a model in a semantically strong space, using standard reconstruction losses.

Finally, several authors developed methods related to our work to improve the temporal stability of semantic video segmentation. Jin et al., 2017a train a model to predict the semantic segmentation of the immediate next image from the preceding input frames, and fuse this prediction with the segmentation computed from the next input frame. Patraucean et al., 2016 employ a convolutional RNN to implicitly predict the optical flow, and use these to warp and aggregate per-frame segmentations. In contrast, our work is focused on predicting future segmentations without seeing the corresponding frames. Nilsson and Sminchisescu, 2018 use a convolutional RNN model with a spatial transformer component (Jaderberg et al., 2015) to accumulate the information from past and future frames in order to improve prediction of the current frame segmentation. In contrast, our work is focused on predicting future segmentations without seeing the corresponding frames. Most importantly, we target a longer time horizon than a single frame.

The approach we develop for predicting future semantic segmentation builds on the work of Mathieu et al., 2016, state-of-the-art at the time of publication,

and our best performing architecture uses dilated convolutions introduced by Chen et al., 2015 in the context of semantic segmentation.

4.3 Predicting future frames and segmentations

We start by presenting different scenarios to predict RGB pixel values and/or segmentations of the next video frame in Section 4.3.1. Then, in Section 4.3.2, we describe two extensions of the single-frame prediction model to predict further into the future.

4.3.1 Single-frame prediction models

Pixel-level supervision is laborious to acquire for semantic image segmentation, and even more so for its video counterpart. To circumvent the need for datasets with per-frame annotations, we use the state-of-the-art Dilation10 semantic image segmentation network of Yu and Koltun, 2016, presented in detail in Section 2.1.1, to provide input and target semantic segmentations for all frames in each video. We use the resulting temporally dense segmentation sequences to learn our models.

Let us denote with X_i the i-th frame of a video sequence and denote the sequence of frames from X_t to X_T as $X_{t:T}$. We denote by S_i the semantic segmentation of frame X_i given the Dilation10 network. We represent the segmentations S_i using the final softmax layer's pre-activations, rather than the probabilities it produces. This is motivated by recent observations in network distillation that the softmax pre-activations carry more information (Ba and Caruana, 2014; Hinton et al., 2014). For single-frame future prediction, we consider five different models that differ in whether they take RGB frames and/or segmentations as their inputs and targets: model X2X takes $X_{1:t}$ and predicts X_{t+1}, model S2S takes $S_{1:t}$ and predicts S_{t+1}, models XS2X and XS2S take $(X_{1:t}, S_{1:t})$ and predict respectively X_{t+1} and S_{t+1}, and finally model XS2XS takes $(X_{1:t}, S_{1:t})$ and predicts (X_{t+1}, S_{t+1}).

Architectures. The X2X model is a next frame prediction model, for which we use the multi-scale network of Mathieu et al., 2016, with two spatial scales. Denoting with C the number of output channels, each scale module is a four-layer convolutional network alternating convolutions and ReLU operations, outputting feature maps with 128, 256, 128 and C channels each, and filters of size 3 for the smaller scale, and 5, 3, 3, 5 for the larger scale. The last non-linear function is a hyperbolic tangent, to ensure that the predicted RGB values lie in the range $[-1, 1]$. The output at the coarser scale is upsampled, and used as input to the next scale module together with a copy of the input RGB frames at that scale. This module then outputs a refinement that is summed with the previous upsampled coarse prediction to form the final prediction.

For models that predict segmentations S_{t+1}, we removed the last hyperbolic tangent non-linearities for the corresponding output channels, since the softmax pre-activations are not limited to a fixed range. Apart from this difference, the

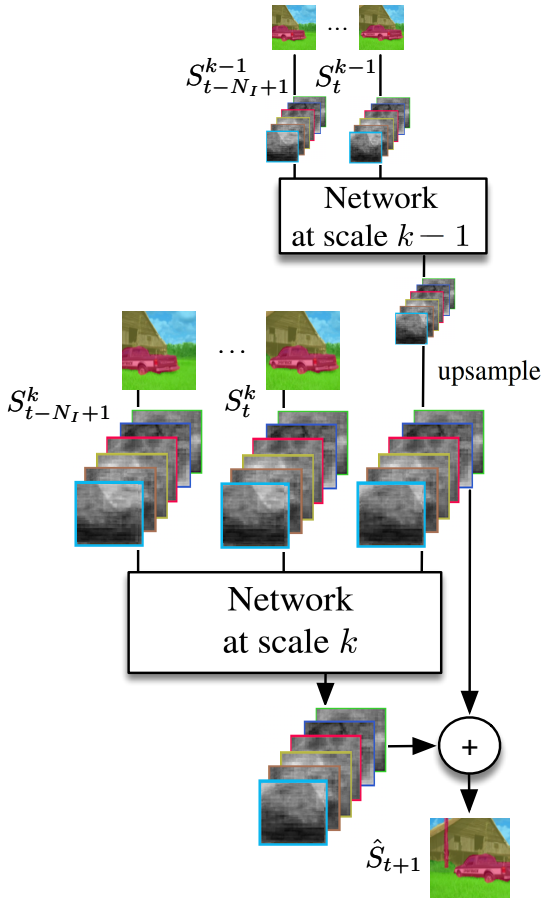


Figure 4.2: Multi-scale architecture of the S2S model that predicts the semantic segmentation of the next frame given the segmentation maps of the N_I previous frames.

S2S model, that predicts the next segmentation from past ones, has the same architecture as the X2X model.

The multi-scale architecture of the S2S model is illustrated in Figure 4.2. The other models (XS2X, XS2S, and XS2XS), which take both RGB frames and segmentation maps as input, also use the same internal architecture and differ only in the number of channels in their input and output.

Loss function. Following Mathieu et al., 2016, for all models, the loss function between the model output $\hat{Y}$ and the target output Y is the sum of an ℓ_1 loss and a gradient difference loss (GDL):

$$\mathcal{L}(\hat{Y}, Y) = \mathcal{L}_{\ell_1}(\hat{Y}, Y) + \mathcal{L}_{\text{gdl}}(\hat{Y}, Y). \tag{4.1}$$

Using Y_{ij} to denote the pixel elements in Y, and similarly for $\hat{Y}$, the losses are defined as:

$$\mathcal{L}_{\ell_1}(\hat{Y}, Y) = \sum_{i,j} \left|Y_{ij} - \hat{Y}_{ij}\right|, \tag{4.2}$$

$$\begin{aligned}\mathcal{L}_{\text{gdl}}(\hat{Y}, Y) = &\sum_{i,j} \Big|\left|Y_{i,j} - Y_{i-1,j}\right| - \left|\hat{Y}_{i,j} - \hat{Y}_{i-1,j}\right|\Big| \\ &+\Big|\left|Y_{i,j-1} - Y_{i,j}\right| - \left|\hat{Y}_{i,j-1} - \hat{Y}_{i,j}\right|\Big|,\end{aligned} \tag{4.3}$$

where $|\cdot|$ denotes the absolute value function. The ℓ_1 loss tries to match all pixel predictions independently to their corresponding target values. The GDL, instead, penalizes errors in the gradients of the prediction. This loss is relatively insensitive to low-frequency mismatches between prediction and target (*e.g.*, adding a constant to all pixels does not affect the loss), and is more sensitive to high-frequency mismatches that are perceptually more significant (*e.g.* errors along the contours of an object). We present a comparison of this loss with a multi-class cross entropy loss in Section 4.4.

4.3.2 Predicting deeper into the future

To predict further into the future than a single frame, we consider two extensions of the previous models. The first is to expand the output of the network to comprise a batch of m frames, *i.e.* to output $X_{t+1:t+m}$ and/or $S_{t+1:t+m}$. We refer to this as the "batch" approach. The drawback of this approach is that it ignores the recurrent structure of the problem. That is, it ignores the fact that S_{t+1} depends on $S_{1:t}$ in the same manner as S_{t+2} depends on $S_{2:t+1}$. As a result, the capacity of the model is split to predict the m output frames, and the number of parameters in the last layer scales linearly with the number of output frames.

In our second approach, we leverage the recurrence property, and iteratively apply a model that predicts a single step into the future, using its prediction for time $t+1$ as an input to predict at time $t+2$, and so on. This allows us to predict arbitrarily far into the future in an autoregressive manner, without resources scaling with the number of time-steps we want to predict. We refer to this approach as "autoregressive". See Figure 4.3 for a schematic illustration of the two extensions for multiple time-step predictions.

In contrast with the batch mode however, the autoregressive mode is prone to error accumulation: mistakes at each time-step affect all later time-steps. One can see this as a domain shift between training, where only inputs provided by the dataset are used, and inference, where its predictions are fed back in, and could be quite different from the training samples. To mitigate this issue, we fine-tune these models using backpropagation through time (BPTT) (Werbos, 1988). This way, the network can learn to take into account its own mistakes, and to make predictions that will not cause improper outputs in the following steps. To reduce the domain shift, an alternative would be to adapt scheduled

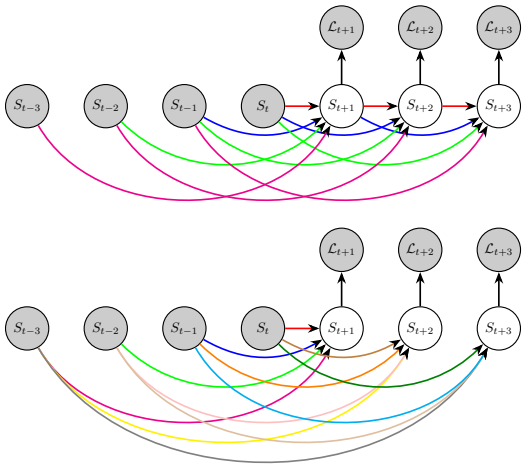


Figure 4.3: Illustration of the autoregressive (top) and batch (bottom) models. The autoregressive model shares parameters over time; dependency links are coloured accordingly.

sampling (Bengio et al., 2015), a form of curriculum learning that randomly chooses at training, whether to use as inputs the outputs of the network or the samples of the dataset, and gently grows the proportion of the network outputs during training. We evaluate both the models trained for single-frame prediction, as well as those fine-tuned using BPTT.

4.4 Experiments

We first describe the dataset and evaluation metrics in Section 4.4.1 and our baselines in Section 4.4.2. We then present results on short-term (i.e. single-frame) prediction, mid-term prediction (0.5 sec.), and long-term prediction (10 sec.).

4.4.1 Dataset and evaluation metrics

The Cityscapes dataset (Cordts et al., 2016) contains 2,975 training, 500 validation and 1,525 testing video sequences of 1.8 second. Each sequence consists of 30 frames, and a ground truth semantic segmentation is available for the 20-th frame. The segmentation outputs of the Dilation10 network (Yu and Koltun, 2016) are produced at a resolution of 128×256 and we perform all experiments at this resolution. For this purpose, we also downsample RGB frames and ground truth to this resolution. We report performance of our models on the Cityscapes validation set, and mid-term performance for our best performing models on the test set.

We assess performance using the standard mean intersection over union (IoU) measure, computed w.r.t. the ground truth segmentation of the 20-th frame in each sequence (IoU GT). We also compute the IoU measure w.r.t. the segmentation produced using the Dilation10 network for the 20-th frame (IoU SEG). The IoU SEG metric allows us to validate our models w.r.t. the target segmentations from which they are trained. Finally, we compute the mean IoU across categories that can move in the scene: *person, rider, car, truck, bus, train, motorcycle, and bicycle* (IoU-MO, for "moving objects").

To evaluate the quality of the frame RGB predictions, we compute the peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM) measures (Wang et al., 2004). The PSNR only depends on the inverse of the mean square error. The SSIM measures similarity between two images, ranging between -1 for very dissimilar inputs to $+1$ when the inputs are the same. It is based on comparing local patterns of pixel intensities normalized for luminance and contrast.

Unless specified otherwise, we train our models using a frame interval of 3, corresponding to 0.18 sec., and taking 4 frames and/or segmentations as input. That is, the input sequence consists of frames $\{X_{t-9}, X_{t-6}, X_{t-3}, X_t\}$, and similarly for segmentations. We performed patch-wise training with 64×64 patches for the largest scale resolution, enabling equal class frequency sampling as in Farabet et al., 2013, using mini-batches of four patches and a learning rate of 0.01.

4.4.2 Baselines

We include two baselines in our experiments. The first baseline copies the last input frame to the output. The second baseline is an optical flow baseline, which we present in detail next. Comparison with tracking-based approaches is difficult since (i) segmentation is performed densely and lacks the notion of object instances used by object trackers, and (ii) "stuff" categories (road, vegetation, *etc.*), useful for drivable area detection in the context of autonomous driving, are not suitable for modeling with tracking-based approaches.

The baseline relying on optical flow implemented in (Ranzato et al., 2014; Mathieu et al., 2016) leads to systematic failure cases, due to the fact that the past flow field is used in place of the inaccessible future flow field to warp the last input. In the following, we analyse these failures, and then we propose an improved optical flow baseline that avoids this by projecting each pixel forward, based on its past displacement.

Both baselines rely on the optical flow field $\mathbf{F}_{t \to t-1}$ computed from X_t, the RGB frame at time t, to X_{t-1}, the RGB frame at time $t-1$. The flow fields are computed using Full Flow (Chen and Koltun, 2016), a state of the art optical flow estimation method, using the default parameters given by the authors on the MPI Sintel Flow Dataset (Butler et al., 2012).

Figure 4.4: From left to right: two input frames and three predictions, framed in red, obtained from the warping baselines, "$t+1$ - centric" (top) and "t - centric" (bottom). Static pixels that are about to be occluded by the moving objects are systematically mis-predicted to hold their previous values in the original baseline - see the pixels in front of the left car on the top row - leading moving objects to shrink instead of moving. This is corrected in our improved baseline, shown in the bottom row.

Initial flow baseline

Here we describe the baseline used by Ranzato et al., 2014 and Mathieu et al., 2016 for next frame prediction in the space of RGB intensities relying on optical flow, to introduce our notations and differentiate this baseline explicitly from the one we develop. Let us consider a spatial position p of the frame we wish to predict X_{t+1}. The original baseline sets $\hat{X}_{t+1}(p)$ by bilinearly interpolating the values of X_t surrounding spatial position $p+d$, where $d = \mathbf{F}_{t\to t-1}(p)$. The issue is that one should be using the flow vector $d' = \mathbf{F}_{t+1\to t}(p)$ instead, which we cannot access since we are trying to predict X_{t+1}. The displacements d and d' are in general not equal, since the physical point corresponding to d may have been replaced by another, which potentially does not have the same displacement. For instance, this is the case for still points that are about to be occluded by moving objects. In this case, d is zero, whereas d' could correspond to a fast moving point. A qualitative example is shown is Figure 4.4, where the values for the pixels in front of the moving car are not replaced by those of the car as they should be. Note that this is a systematic failure case of the baseline, which concerns all pixels which are about to be occluded by a moving object. We call this baseline "$t+1$ - centric", as it can be viewed as looping over the spatial positions of the prediction $\hat{X}_{t+1}$.

Improved baseline for future semantic segmentation

We propose an improved baseline which does not have this shortcoming. Considering a spatial position p in the last input frame X_t, we project its value into the next frame by using the opposite of the flow vector at p as an estimation for the displacement of the corresponding physical point between time steps t and $t+1$, setting

$$\hat{X}_{t+1}([p+d]) = X_t(p), \tag{4.4}$$

where $d = -\mathbf{F}_{t\to t-1}(p)$ and where $[\cdot]$ denotes rounding.

Method	PSNR	SSIM	IoU GT (SEG)	IoU-MO GT (SEG)
Copy last input	20.6	0.65	49.4 (54.6)	43.4 (48.2)
Warp last input	23.7	0.76	59.0 (**67.3**)	54.4 (**63.3**)
Model X2X	24.0	**0.77**	23.0 (22.3)	12.8 (11.4)
Model S2S	—	—	58.3 (64.9)	53.8 (59.8)
Model XS2X	**24.2**	**0.77**	22.4 (22.5)	10.8 (10.0)
Model XS2S	—	—	58.2 (64.6)	53.7 (59.9)
Model XS2XS	24.0	0.76	55.5 (61.1)	50.7 (55.8)
Model S2S-adv.	—	—	58.3 (65.0)	53.9 (60.2)
Model S2S-dil	—	—	**59.4** (66.8)	**55.3** (63.0)

Table 4.1: Short-term prediction (0.18 sec.) accuracy of baselines and of our models taking either RGB frames (X) and/or segmentations (S) as input and output, on the Cityscapes validation set. For reference: the 59.4 IoU corresponds to 91.8% per-pixel accuracy.

In case of competing values for position $[p+d]$, we prioritize these corresponding to the largest flow to favor displacement of moving and close-by objects, as opposed to still and far objects or stuff. This baseline is called "t - centric"

We apply the same transformation procedure to the flow field $\mathbf{F}_{t\to t-1}$ to get $\hat{\mathbf{F}}_{t+1\to t}$, which we use to predict X_{t+2} and so on. We solve a Dirichlet boundary value problem to interpolate the missing values that were not determined by the warping in Equation 4.4. As in our other experiments, we employed a frame interval of 3. Decreasing the frame interval down to 1 leads to worse performance for mid-term prediction (43.5 instead of 44.3 IoU GT for a frame rate of 3) because of error propagation, as more predictions are needed to reach 0.54 sec.

4.4.3 Short-term prediction

In our first experiment, we compare the five different input-output representations. We also include the baseline results. Our models take in input frames 8, 11, 14 and 17 and predict outputs for frame 20, that is 0.18 sec. ahead. For models that do not directly predict future segmentations, we generate segmentations using the Dilation10 network based on the predicted RGB frames.

In Figure 4.5, we show qualitative results of the predictions for one of the validation sequences. From the quantitative result in Table 4.1 we make several observations. First, in terms of RGB frame prediction (PSNR and SSIM), the performance is comparable for the three models X2X, XS2X, and XS2XS, and improves over the two baselines. This shows that our models learn non-trivial scene dynamics in the RGB pixel space, and that adding semantic segmentations either at input and/or output does not have a substantial impact on this ability.

Second, in terms of the IoU segmentation metrics, the models that directly predict future segmentations (S2S, XS2S, XS2XS) perform much better than the models that only predict the RGB frames. This suggests that artifacts in the

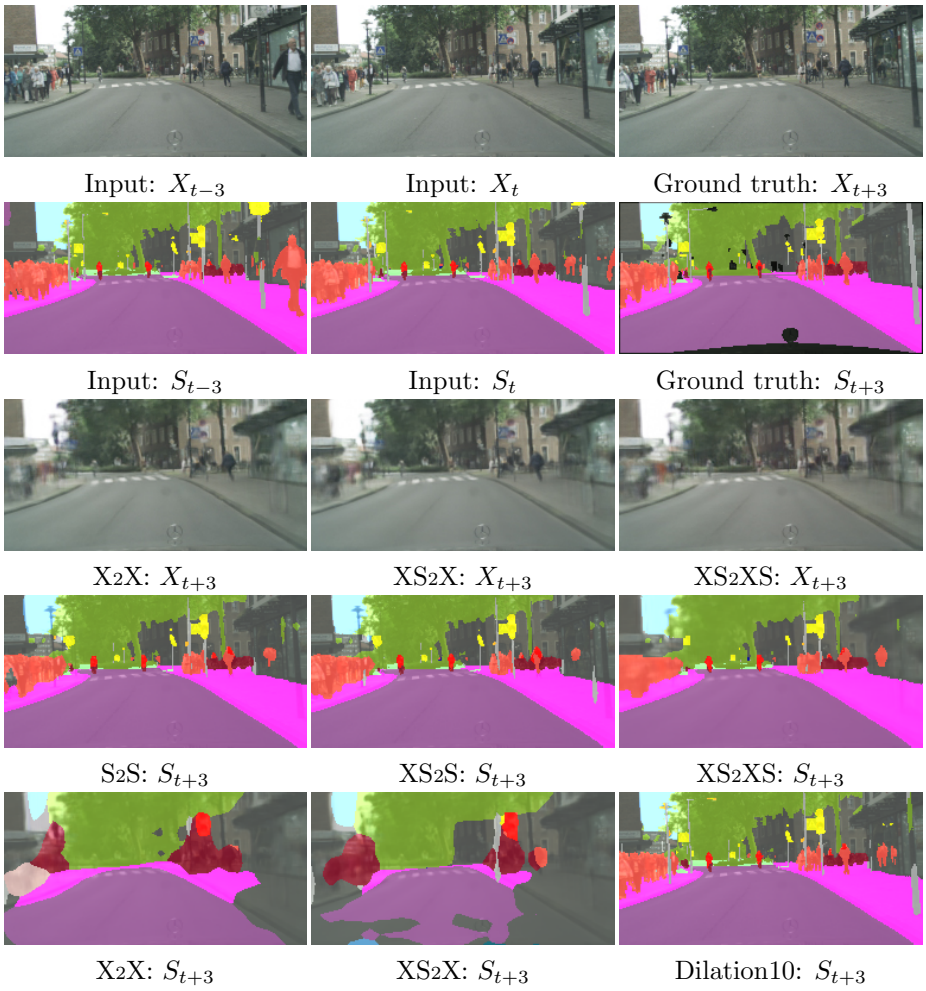


Figure 4.5: Short-term predictions of RGB frame X_{t+3} and segmentation S_{t+3} using our different models, compared to ground truth, and Dilation10 oracle that has seen X_{t+3}.

Model	IoU GT	IoU SEG	IoU-MO GT
Dilation10 oracle	68.8	100	64.7
S2S, 2 scales, ℓ_1+GDL	**58.3**	**64.9**	**53.8**
S2S, 1 scale, ℓ_1+GDL	57.7	63.9	52.6
S2S, 2 scales, ℓ_1	57.6	64.0	53.2
S2S, 2 scales, MCE	55.5	60.9	49.7

Table 4.2: Ablation study with the S2S model on the Cityscapes validation set and comparison to a Dilation10 oracle that predicts the future segmentation using the future RGB frame as input.

RGB frame predictions degrade the performance of the Dilation10 network. See also the corresponding RGB frame predictions in Figure 4.5.

Third, the XS2XS model, which predicts both segmentations and RGB frames performs somewhat worse than the models that only predict segmentations (S2S and XS2S), suggesting that some of the modeling capacity is compromised by jointly predicting the RGB frames.

Finally, we report the results of an experiment where we fine-tune the S2S model using adversarial training (S2S-adv). We found that this does not lead to a significant improvement over normal training. While we expect "blur" to occur in the predicted segmentation maps, as in the case of RGB predictions, we believe that it results only in small deformations of the segmentations in the final label map, (after the argmax operation), that are hard to quantify and observe qualitatively.

Table 4.2 presents results of an ablation study of the S2S model, assessing the impact of the different loss functions, as well as the impact of using one or two scales. We include the results obtained using the Dilation10 model as an "oracle", that predicts the future segmentation based on the future RGB frame, which is not accessible to our other models. This oracle result gives the maximum performance that could be expected, since this oracle was used to provide the training data - we can thus only expect our models to have at best comparable performance with this oracle. All variants of the S2S model were trained during about 960,000 iterations, taking about four days of training on a single GPU. The results show that using two scales improves the performance, as does the addition of the gradient difference loss. Training with the ℓ_1 and/or GDL loss on the softmax pre-activations gives better results as compared to training using the multi-class cross-entropy (MCE) loss on the segmentation labels. This is in line with observations made in network distillation (Ba and Caruana, 2014; Hinton et al., 2014).

Finally, we perform further architecture exploration for the S2S model, which performed best. Our multiscale S2S model based on standard convolutions has a field of view of 30 over input resolution 128×256. We propose a simpler, deeper, and more efficient architecture with dilated convolutions (Yu and Koltun, 2016), to expand the field of view while retaining accurate localization for the predic-

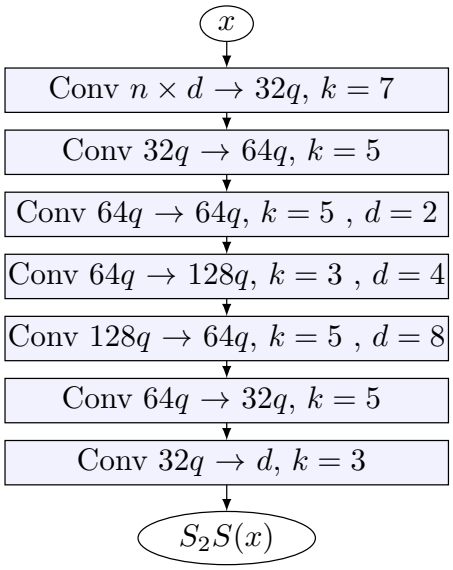


Figure 4.6: Architecture for S2S-dil model, taking the segmentations of $n = 4$ frames in input, each of channel dimension C. Each inner convolutional layer is followed by a ReLU. Each convolutional layer has a kernel of size $k \times k$ and dilation parameter d. q is a hyperparameter used to scale the number of feature maps linearly for simple control over the model capacity. Stride is always one and padding is chosen so as to maintain the size of the input.

tions. We summarize its architecture in Figure 4.6.

With $q = 4$, this architecture has 8.2M parameters and obtains overall best performance of 60.4. Finally, to retain the possibility of fine-tuning this architecture in an autoregressive fashion on a single GPU, we scale the parameters back down to 0.9M, corresponding to a choice of $q = 1.25$. We call this model S2S-dil and report its performance in Table 4.1.

4.4.4 Mid-term prediction

We now address the more challenging task of predicting the mid-term future, *i.e.* the next 0.5 second. In these experiments we take in input frames 2, 5, 8, and 11, and predict outputs for frames 14, 17 and 20. We compare different strategies: batch models, autoregressive models (AR), and models with autoregressive fine-tuning (AR fine-tune). We compare these strategies to our two baselines consisting in copying the last input, and the second one relying on optical flow. For the optical flow baseline, after the first prediction, we also warp the flow field so that the flow is applied to the correct locations at the next time-step, and so on. For models XS2X and XS2S, the autoregressive mode is not used because either the frame or the segmentation input are missing for

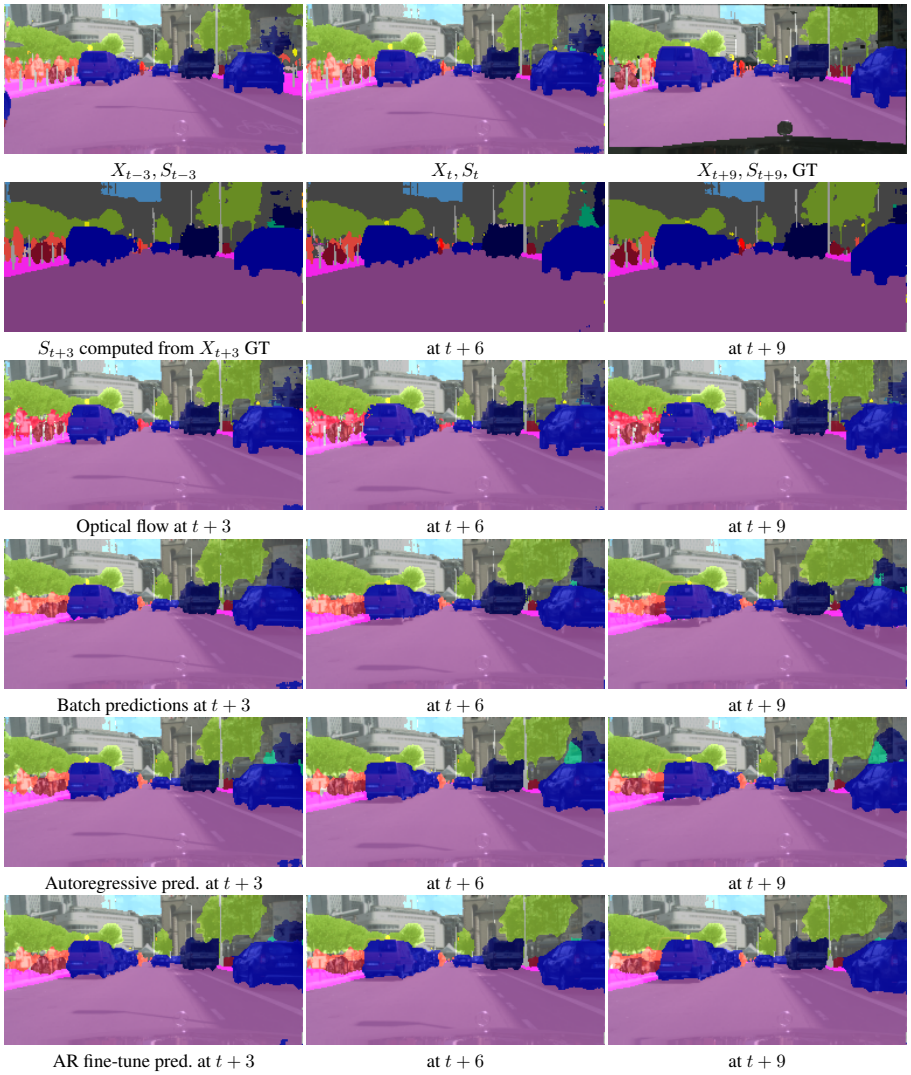


Figure 4.7: Comparison between optical flow baseline, batch, autoregressive, and autoregressive fine-tuned S2S model predictions. First row: last inputs and ground truth segmentation. Second row: target segmentations obtained using the Dilation10 network. Other rows show predictions overlaid with the true future frames.

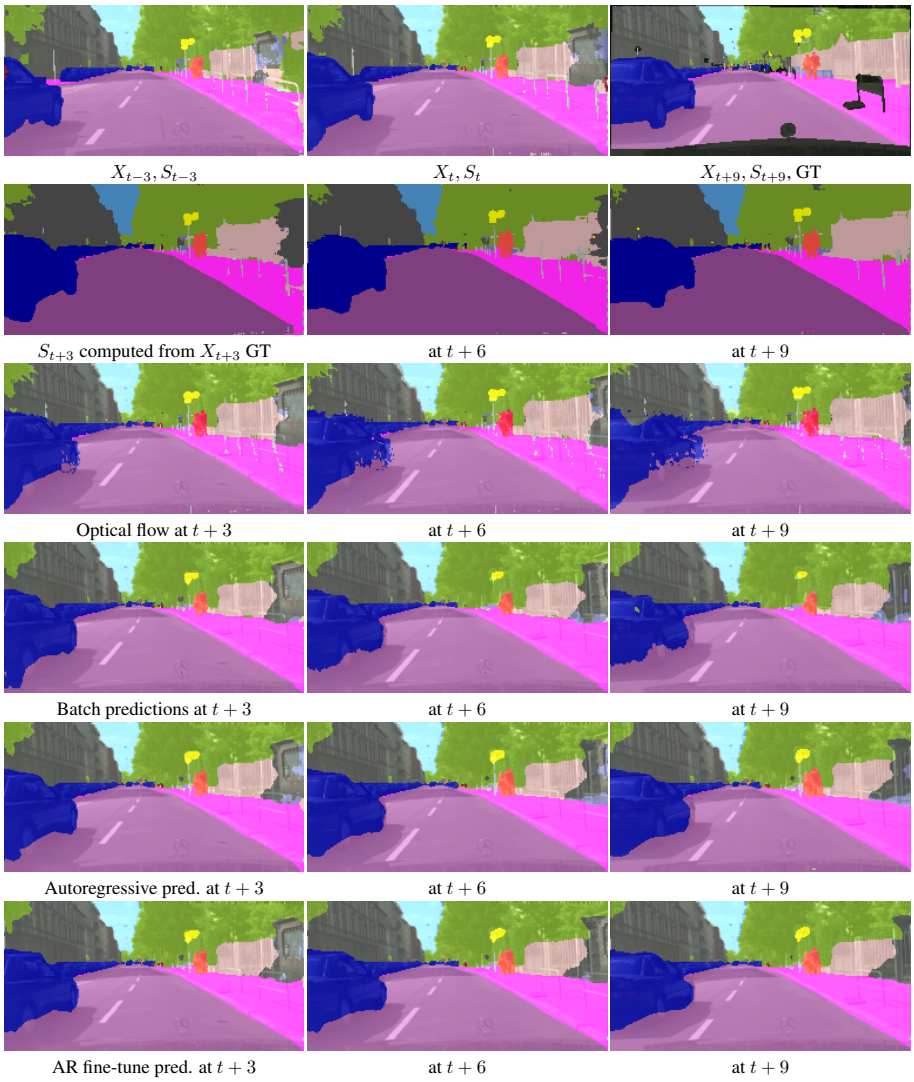


Figure 4.8: Comparison between optical flow baseline, batch, autoregressive, and autoregressive fine-tuned S2S model predictions. First row: last inputs and ground truth segmentation. Second row: target segmentations obtained using the Dilation10 network. Other rows show predictions overlaid with the true future frames.

Model	Frame 14 PSNR	Frame 14 SSIM	Frame 20 PSNR	Frame 20 SSIM
Copy last input	20.4	0.64	18.0	0.55
Warp last input	23.5	**0.76**	19.4	0.59
X2X, AR	**23.9**	**0.76**	19.2	0.61
XS2XS, AR	23.8	**0.76**	19.3	0.61
X2X, batch	23.8	**0.76**	20.6	**0.65**
XS2X, batch	**23.9**	**0.76**	**20.7**	**0.65**
XS2XS, batch	23.8	**0.76**	**20.7**	0.64

Table 4.3: Mid-term (0.54 sec.) RGB frame prediction results for frame 20 on the Cityscapes validation set using different models in batch and autoregressive mode.

predicting from the second output on.

The results for RGB frame prediction in Table 4.3 show that for frame 14, all models give comparable results, consistently improve over the copy baseline and perform somewhat better than the warping baseline. For frame 20, the batch models perform best. On the contrary, when predicting segmentations, we find that the autoregressive models perform better than the batch models, as reported in Table 4.4. This is probably due to the fact that the single-step predictions are more accurate for segmentation, which makes them more suitable for autoregressive modeling. For RGB frame prediction, errors accumulate quickly, leading to degraded autoregressive predictions. Among the batch models, using the images as input (XS2S model) helps slightly. Predicting both the images and segmentation (XS2XS model) performs worst, the image prediction task presumably taking up resources otherwise available for modeling the dynamics of the sequence.

Model S2S is the most effective, as it can be applied in autoregressive mode, and outperforms XS2XS in this setting. In Figures 4.7 and 4.8, we compare different versions of this model: batch, autoregressive, and autoregressive fine-tuned. Visually, the first sequence shows some improvements using the autoregressive fine-tuned model, by more accurately matching contours of the moving cars than the other strategies. Results are also compared with our optical flow baseline. The second sequence displays typical failures of the optical flow baseline, where certain values cannot be estimated because they correspond to points that were not present in the input, *e.g.* those at the back of the incoming car, and must be filled using a standard region filling algorithm. All segmentations are overlaid with the true video sequence to facilitate assessment of the predictions.

Difficult cases for our methods include dealing with occlusions and with fast ego-motion. Figures 4.9 and 4.10 show two failures cases of our S2S model (fine-tuned in autoregressive mode), where the model respectively underestimates the speed of the camera and fails to predict a future occlusion.

Finally, we measure performance on the Cityscapes test set for mid-term

Model	IoU GT	IoU SEG	IoU-MO GT
Copy last input	36.9	39.2	26.8
Warp last input	44.3	47.2	37.0
S2S, AR	45.3	47.2	36.4
S2S, AR, fine-tune	46.7	49.7	39.3
XS2XS, AR	39.3	40.8	27.4
S2S, batch	42.1	44.2	32.8
XS2S, batch	42.3	44.6	33.1
XS2XS, batch	41.2	43.5	31.4
S2S-adv, AR	45.1	47.2	37.3
S2S-dil, AR	46.5	48.6	38.8
S2S-dil, AR, fine-tune	**47.8**	**50.4**	**40.8**

Table 4.4: Mid-term (0.54 sec.) segmentation prediction results on the Cityscapes validation set. For reference: the 47.8 IoU corresponds to 87.9% per-pixel accuracy.

Model	IoU GT	IoU SEG	IoU-MO GT
Dilation10 oracle	67.1	100	61.5
Warp last input	45.9	49.5	39.1
S2S, AR, fine-tune	47.8	51.8	40.2
S2S-dil, AR, fine-tune	**48.0**	**52.0**	**40.4**

Table 4.5: Mid-term (0.54 sec.) segmentation prediction for frame 20 using our best S2S models on the Cityscapes test set.

prediction of our optical flow baseline and of our two models S2S, AR, fine-tune and S2S-dil, AR, fine-tune. We use the same setup as we used on the validation set: we take in input frames 2, 5, 8, and 11, and predict outputs for frames 14, 17 and 20 of each sequence. Results for frame 20 are shown in Table 4.5. For reference, we also show the performance reported by the authors of the Dilation10 architecture on the test set. Consistent with observations on the validation set, we observe significant improvements of the S2S models over the optical flow baseline. The gains on the IoU GT metrics are slightly less pronounced, but still very close to those obtained on the validation set, while the oracle performance suffers a large decrease (from 64.7 to 61.5 on moving objects). The IoU SEG metric on the other hand seems to be much less affected and even enjoys an important boost in comparison with the one measured on the validation set. These results hint at the importance of choosing an oracle that has good generalization properties.

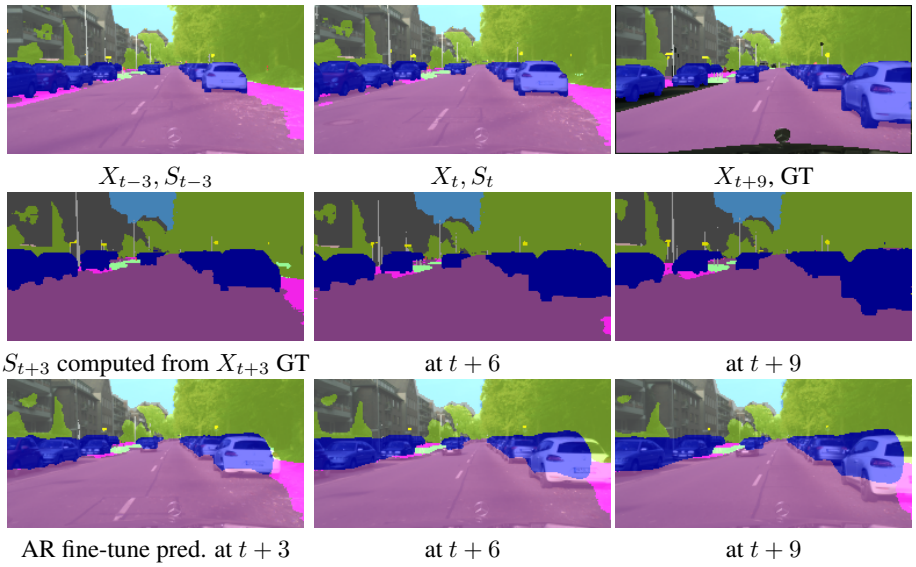


Figure 4.9: Failure case of the autoregressive fine-tuned S2S model. First row: last inputs and ground truth. Second row: future segmentations obtained using the Dilation10 network computed using the future RGB frames. Third row: S2S, AR, fine-tune predictions overlaid with the true future frames. In this example, the speed of the camera is underestimated by our model, resulting in large errors in the segmentation of the closest car.

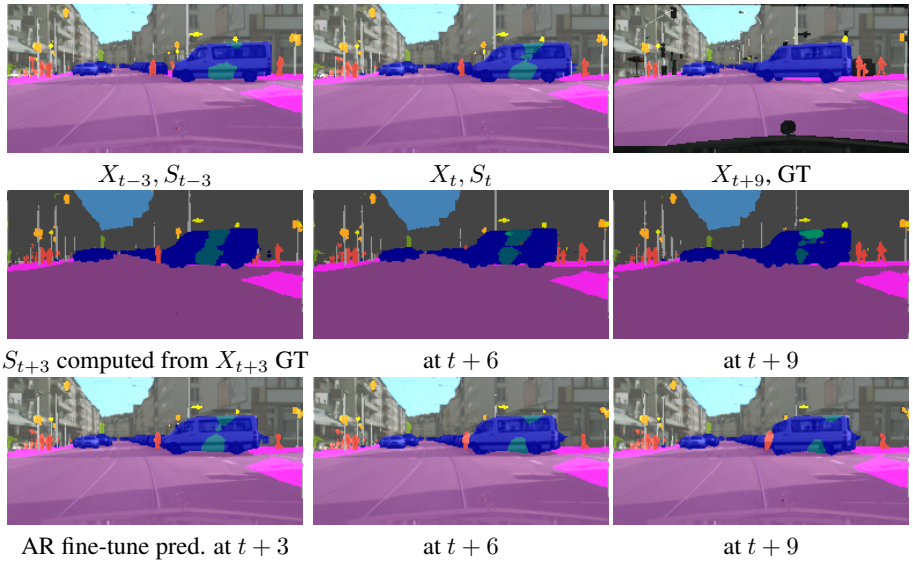


Figure 4.10: Failure case of the autoregressive fine-tuned S2S model. First row: last inputs and ground truth. Second row: future segmentations obtained using the Dilation10 network computed using the future RGB frames. Third row: S2S, AR, fine-tune predictions overlaid with the true future frames. In this example, the occlusion of the pedestrian by the vehicle coming from the right is not predicted by our system. The green blobs that appear in the vehicle correspond to the "bus" category. This second mistake is hard to avoid because it also appears in the Dilation10 input segmentations.

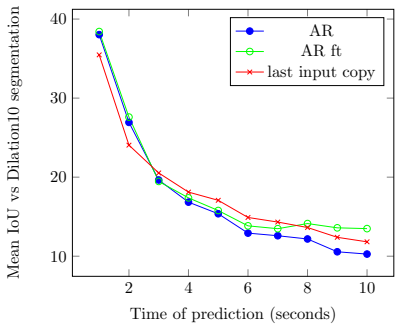


Figure 4.11: Mean IoU SEG of long-term segmentation prediction for the AR and AR fine-tune S2S models.

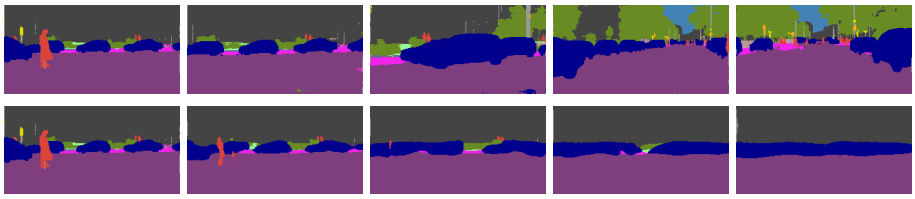


Figure 4.12: Last input segmentation, and ground truth segmentations at 1, 4, 7, and 10 seconds into the future (top row), and corresponding predictions of the autoregressive S2S model trained with fine-tuning (bottom row).

4.4.5 Long-term prediction

To evaluate the limits of our S2S autoregressive models on arbitrarily long sequences, we use them to make predictions of up to ten seconds into the future. To this end, we evaluate our models on ten sequences on 238 frames extracted from the long Frankfurt sequence of the Cityscapes validation set. Given four segmentation frames with a frame interval of 17 images, corresponding to exactly one second, and hence much increased in comparison with the one used at train time, we apply our models to predict the ten next ones. In Figure 4.11 we report the IoU SEG performance as a function of time. In this extremely challenging setting, the predictive performance quickly drops over time. Fine-tuning the model in autoregressive mode improves its performance, but only gives a clear advantage over the input-copy baseline for predictions at one and two seconds ahead. We also applied our model with a frame interval of 3 to predict up to 55 steps ahead, but found this to perform much worse. Figure 4.12 shows an example of predictions compared to the actual future segmentations. The visualization shows that our model averages the different classes into an average future, which is perhaps not entirely surprising, given the deterministic nature of our model. Sampling different possible futures using a GAN or VAE approach might be a way to resolve this issue.

4.4.6 Cross-dataset generalization

To evaluate the generalization capacity of our approach, we test our S2S model on the Camvid dataset (Brostow et al., 2008), specifically on the test set of 233 images with 11 classes grouping employed in (Badrinarayanan et al., 2017). Ground truth segmentations are provided for every second on 30 fps video sequences. We first generate the Dilation10 segmentations - without fine-tuning the oracle to the CamVid dataset - using a frame interval of 5, roughly corresponding to a frame interval of 3 on Cityscapes. We note that the class correspondence between Cityscapes and CamVid is not perfect; for instance we associate the class “tree” to “vegetation”. As reported in Table 4.6, our models have very good mid-term performance on this dataset, considering the oracle results. For reference, Yu and Koltun, 2016 reports an IoU of 65.3 using a fine-tuned Dilation8.

	Dilation10 oracle	Copy last input	Warp last input	S2S AR ft
Cityscapes	67.1	–	45.9	**47.8**
Camvid	55.4	40.8	43.7	**46.8**

Table 4.6: IoU GT of oracle and mid-term (0.54 sec.) predictions on test sets of Cityscapes and Camvid.

4.5 Conclusion

We introduced a new visual understanding task of predicting future semantic segmentation. This task led us to develop models that predict semantically rich information, for use in downstream applications, much more accurately than a two-stage procedure that first predicts future RGB frames and then segments these. We systematically studied the effect of using RGB frames and/or segmentations as inputs and targets for our models and the impact of various loss functions. Against our expectations, we found that for single time-step prediction, training a model to predict segmentations led to significantly better performance than training it to jointly predict future segmentations and frames. This suggests that rather than helping the model with extra supervision, the video prediction task is instead hurting performance by compromising some of the model capacity. For prediction beyond a single frame, we considered batch models that predict all future frames at once, and autoregressive models that sequentially predict the future frames. While batch models were more effective in the RGB intensities space because of otherwise large error propagation, the more flexible autoregressive mode was more accurate in the semantic segmentation space, thanks to slower error accumulation in this setting. This new task is hence better suited for autoregressive modeling and for predicting further ahead in the future, thanks to which we can aim to model more interesting distributions.

In this respect, there is still room for improvement. Where the Dilation10 network for semantic image segmentation gives around 69 IoU, this drops to about 59 when predicting 0.18 sec. ahead and to about 48 for 0.54 sec. Most predicted object trajectories are reasonable, but do not always correspond to the actual observed trajectories. GAN or VAE models may be useful to address the inherent uncertainty in the prediction of future segmentations.

Since our publication, several works have taken up the task of predicting future semantic segmentation. Jin et al., 2017b propose a model that jointly predicts future semantic segmentation and future optical flow. These two tasks are expected to complement each other: structure motion prediction is made possible by decomposing the optical flow into different groups according to semantic segmentation, and semantic segmentation can in turn leverage pixel-wise correspondence brought by optical flow estimation. Their architecture is designed to leverage this mutually beneficial relationship. Other approaches focus on future semantic segmentation alone, and make architectural contributions. To

preserve rich structure and complex motion, Xu et al., 2018 separately treat low and high frequency components of the frame and propose a convolutional LSTM module with temporal-adaptive kernels. Nabavi et al., 2018 propose an autoencoding architecture that relies on a ConvLSTM module for prediction in the code space. Vora et al., 2018 propose a geometry-based approach, and design a deep architecture that leverages intermediate predictions from sub-modules responsible for (single image) semantic segmentation, depth and ego-motion. The two latter modules are trained in an unsupervised fashion, requiring no additional supervision. Terwilliger et al., 2019 aggregate past optical flow features using a ConvLSTM to predict future optical flow. These predictions are used by a learnable warp layer to produce future semantic segmentation. This set up is both efficient and surprisingly accurate. Finally, Bhattacharyya et al., 2019 propose a Bayesian learning framework that encourages the learning of diverse models to accurately capture the multi-modal nature of the future scenes. Each model corresponds to a set of weights, so that sampling different weights for the model allows generating different predictions for a given input sequence. Their framework accurately captures model uncertainty and significantly improves the accuracy of mean predictions. Considering the top K predictions yields an additional boost in performance, suggesting that the predictions are diverse as well as being accurate.

This task has seen new applications since as well. Motivated in part by our approach, Zhang et al., 2018 investigate the use of convnets to predict the future states of tumor growth, by predicting future labels of tumor segmentation. For them, the process of gathering the equivalent of a video dataset, let alone an annotated one, is extremely challenging, since it requires gathering each sequence over multiple years, to capture the tumor evolution. Hence, first learning strong, semantically meaningful features on an image dataset is critical. Their approach obtains substantial improvements over a state-of-the-art mathematical model- based approach in both accuracy and efficiency. In follow-up work, Zhang et al., 2019 extend this approach to the 3D domain to predict future tumor volumes. Zhu et al., 2018b propose to use the semantic segmentation predicted by a video prediction model as pseudo ground truth to improve semantic segmentation models. They rely on a boundary label relaxation technique that makes training robust to annotation noise and propagation artifacts along object boundaries. This leads to a nice boost in accuracy across several datasets for autonomous driving.

We provide the code for our Torch-based implementation at `https://github.com/facebookresearch/SegmPred`, and invite the reader to watch videos of our predictions at `https://thoth.inrialpes.fr/people/pluc/iccv2017`.

Chapter 5

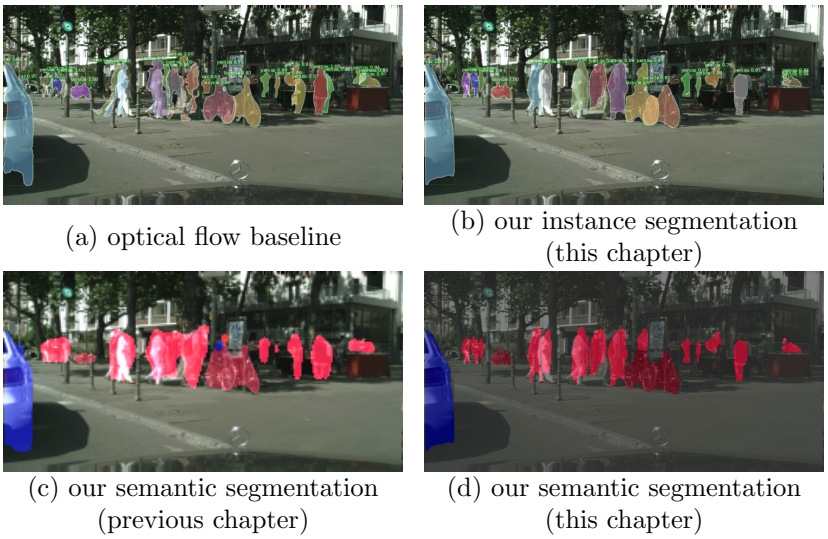
Predicting Future Instance Segmentation

The ability to anticipate future events is an important prerequisite towards intelligent behaviour. In Chapter 2, we motivated the task of video prediction, as a proxy task towards this goal. In the previous chapter, we proposed to shift this task to the space of semantic segmentation, as a high level, semantically rich and spatially detailed representation, which can be directly useful for downstream applications such as autonomous driving. We showed that this new task was better suited for predicting further ahead into the future, and that forecasting at the semantic level was more effective than forecasting RGB frames and then segmenting these. In this chapter, we consider the more challenging problem of future *instance* segmentation, which additionally segments out individual objects. To deal with a varying number of output labels per image, we develop a predictive model in the space of fixed-sized convolutional features of the Mask R-CNN instance segmentation model. We apply the "detection head" of Mask R-CNN on the predicted features to produce the instance segmentation of future frames. Experiments show that this approach significantly improves over strong baselines based on optical flow and repurposed instance segmentation architectures. The work presented in this chapter was previously published at ECCV 2018 (Luc et al., 2018).

In Section 5.1, we further motivate our approach. We discuss related work on video prediction and instance segmentation in Section 5.2. We present our approach in Section 5.3 and our baselines, experiemental setup and results in Section 5.4. We conclude in Section 5.5.

5.1 Introduction

Predictive models have important applications in decision-making contexts, such as autonomous driving, where rapid control decisions can be of vital importance (Shalev-Shwartz and Shashua, 2016; Shalev-Shwartz et al., 2016) and where



(a) optical flow baseline

(b) our instance segmentation (this chapter)

(c) our semantic segmentation (previous chapter)

(d) our semantic segmentation (this chapter)

Figure 5.1: Predicting half a second into the future. Instance modeling significantly improves the segmentation accuracy of the individual pedestrians.

interactions with the real-world are slow, expensive and dangerous, as discussed in Section 2.3. In such contexts, however, the goal is often not to predict the raw RGB values of future video frames, but to make predictions about future video frames at a semantically meaningful level, *e.g.* in terms of presence and location of object categories in a scene. In the previous chapter, we showed that for prediction of future semantic segmentation, modeling at the semantic level is much more effective than predicting raw RGB values of future frames, and then feeding these to a semantic segmentation model.

Although spatially detailed, semantic segmentation does not account for individual objects, but rather lumps them together by assigning them to the same category label, *e.g.* the pedestrians in Figure 5.1(c). As discussed in Section 2.1, instance segmentation overcomes this shortcoming by additionally associating with each pixel an instance label, as show in Figure 5.1(b). This additional level of detail is crucial for downstream tasks that rely on instance-level trajectories, such as encountered in control for autonomous driving. Moreover, ignoring the notion of object instances prohibits by construction any reasoning about object motion, deformation, *etc*. Including it in the model can therefore greatly improve its predictive performance, by keeping track of individual object properties, *c.f.* Figure 5.1 (c) and (d).

Since the instance labels vary in number across frames, and do not have a consistent interpretation across videos, the approach presented in Chapter 4 does not apply to this task. Instead, we build upon Mask R-CNN, the recent state-of-the-art instance segmentation model of He et al., 2017 presented in Section 2.1.2, that extends an object detection system by predicting with each object bounding box a binary segmentation mask of the object. In order to

forecast the instance-level labels in a coherent manner, we predict the fixed-sized high level convolutional features used by Mask R-CNN. We obtain the future object instance segmentation by applying the Mask R-CNN "detection head" to the predicted features.

Our approach offers several advantages: (i) we handle cases in which the model output has a variable size, as in object detection and instance segmentation, (ii) we do not require labeled video sequences for training, as the intermediate CNN feature maps can be computed directly from unlabeled data, and (iii) we support models that are able to produce multiple scene interpretations, such as surface normals, object bounding boxes, and human part labels, such as the model proposed by Kokkinos, 2017, without having to design appropriate encoders and loss functions for all these tasks to drive the future prediction. Our contributions are the following:

- we introduce of the new task of future instance segmentation, which is semantically richer than previously studied anticipated recognition tasks,
- we propose a self-supervised approach based on predicting high dimensional CNN features of future frames, which can support many anticipated recognition tasks,
- experimental results show that our feature learning approach improves over strong baselines, relying respectively on optical flow and repurposed instance segmentation architectures.

5.2 Related work

Video prediction. Like the previous chapter, the work presented in this chapter is highly related to video prediction methods introduced in Section 2.3.3 and especially approaches that perform prediction in other output spaces, such as the ones described in Sections 4.2 and 4.5. The work of Vondrick et al., 2016a, who predict high level CNN features of future video frames to anticipate actions and object appearances in video, is particularly relevant to ours. However, while they only predict image-level labels, we consider the more complex task of predicting future instance segmentation, requiring fine spatial detail. To this end, we forecast spatially dense convolutional features, where Vondrick et al., 2016a were predicting the activations of much more compact fully connected CNN layers. Our work demonstrates the scalability of CNN feature prediction, from 4K-dimensional to 32M-dimensional features, and yields results with a surprising level of accuracy and spatial detail.

In Chapter 4, we predicted future semantic segmentation in video by taking the softmax pre-activations of past frames as input, and predicting the softmax pre-activations of future frames. While this approach is relevant for future semantic segmentation, where the softmax pre-activations provide a natural fixed-sized representation, it does not extend to instance segmentation since the instance-level labels vary in number between frames and are not consistent

across video sequences. To overcome this limitation, we develop predictive models for fixed-sized convolutional features, instead of making predictions directly in the label space. Our feature-based approach has many advantages over the previous approach: segmenting individual instances, working at a higher resolution and providing a framework that generalizes to other dense prediction tasks. In other work (Couprie et al., 2018), not presented here, we also investigate a method for predicting joint future semantic and instance segmentation, of moving objects by encoding instance and semantic segmentation information in a single representation based on distance maps. Object positions are extrapolated, and the resulting seeds are used to produce instance segmentation from the predictions using a watershed algorithm. While it requires shorter training time and incorporates a tracking algorithm, it is less accurate than the work we present here for future instance segmentation.

Our work also relates to concurrent works that aim to produce high resolution predictions, while preserving fine spatial structure. To preserve rich structure and complex motion, Xu et al., 2018 propose a two-stream architecture that treats low and high frequency components of each frame separately, relying on transformation learning in recurrent modules. However, they conduct experiments at a maximum resolution of 256×256, including for prediction of future semantic segmentation. Reda et al., 2018 propose spatially-displaced convolutions, that combine the strengths of vector- and kernel-based transformation learning, discussed in Section 2.3.3. This architecture is shown to yield visually pleasing results, but is evaluated either on multi-step prediction at a low resolution, or on single next frame prediction at a high resolution. Our task of predicting future instance segmentation allows us to take on these challenges jointly, producing visually pleasing segmentations at a high resolution of 1024×2048, for complex scenes involving a large number of instances, and with reasonable segmentation accuracy up to half a second ahead. To this end, our approach leverages a multi-scale, resolution-preserving architecture, relying on dilated convolutions (Chen et al., 2015), building on the architectures of Mathieu et al., 2016 and of the previous chapter. Additionally, our outputs are directly interpretable by downstream applications.

Instance segmentation approaches. Our approach can be used in conjunction with any deep network to perform instance segmentation. In Section 2.1.2, we described a variety of approaches for instance segmentation that have been explored in the past, including iterative object segmentation using recurrent networks (Romera-Paredes and Torr, 2016; Ren and Zemel, 2017), watershed transformation (Bai and Urtasun, 2017), object proposals (Hariharan et al., 2014; 2015; Dai et al., 2015; 2016; He et al., 2017), bottom-up grouping (Liu et al., 2017a) or clustering (Kong and Fowlkes, 2018) and dynamically instanciated CRFs (Arnab and Torr, 2017). In the work presented in this chapter, we build upon Mask R-CNN (He et al., 2017), which at the time of publication, had outperformed the state-of-the-art by a large margin. This method extends the Faster R-CNN object detector (Ren et al., 2015) by adding a network branch

to predict segmentation masks and extracting features for prediction in a way that allows precise alignment of the masks when they are stitched together to form the final output. In the distinction that we made Section 2.1.2 between detection-based approaches and segmentation-based approaches that learn using respectively object-centric and pixel-centric losses, this method belongs to the first class of approaches. This distinction also helps understand why the method proposed in Chapter 4 does not apply directly here. In this chapter, we show that it can be successfully extended to the class of detection-based approaches.

5.3 Predicting features for future instance segmentation

In Section 2.1.2, we gave a detailed review of the Mask R-CNN instance segmentation framework. In this section, we summarize it briefly and present how we can use it for anticipated recognition by predicting internal CNN features of future frames.

5.3.1 Instance segmentation with Mask R-CNN

The Mask R-CNN model of He et al., 2017 consists of three main stages. First, a CNN "backbone" architecture is used to extract high level feature maps. Second, a region proposal network (RPN) takes these features to produce regions of interest (ROIs), in the form of coordinates of bounding boxes susceptible of containing instances. The bounding box proposals are used as input to a *RoIAlign* layer, which interpolates the high level features in each bounding box to extract a fixed-sized representation for each box. Third, the features of each RoI are input to the detection branches, which produce refined bounding box coordinates, a class prediction, and a fixed-sized binary mask for the predicted class. Finally, the mask is interpolated back to full image resolution within the predicted bounding box and reported as an instance segmentation for the predicted class. We refer to the combination of the second and third stages as the "detection head".

He et al., 2017 use a Feature Pyramid Network (FPN) (Lin et al., 2017) as backbone architecture, which extracts a set of features at several spatial resolutions from an input image. The feature pyramid is then used in the instance segmentation pipeline to detect objects at multiple scales, by running the detection head on each level of the pyramid. Following Lin et al., 2017, we denote the feature pyramid levels extracted from an RGB image X by P_2 through P_5, which are of decreasing resolution $(H/2^l \times W/2^l)$ for P_l, where H and W are respectively the height and width of X. The features in P_l are computed in a top-down stream by up-sampling those in P_{l+1} and adding the result of a 1×1 convolution of features in a layer with matching resolution in a bottom-up convolutional stream pretrained for classification, *e.g.* ResNet (He

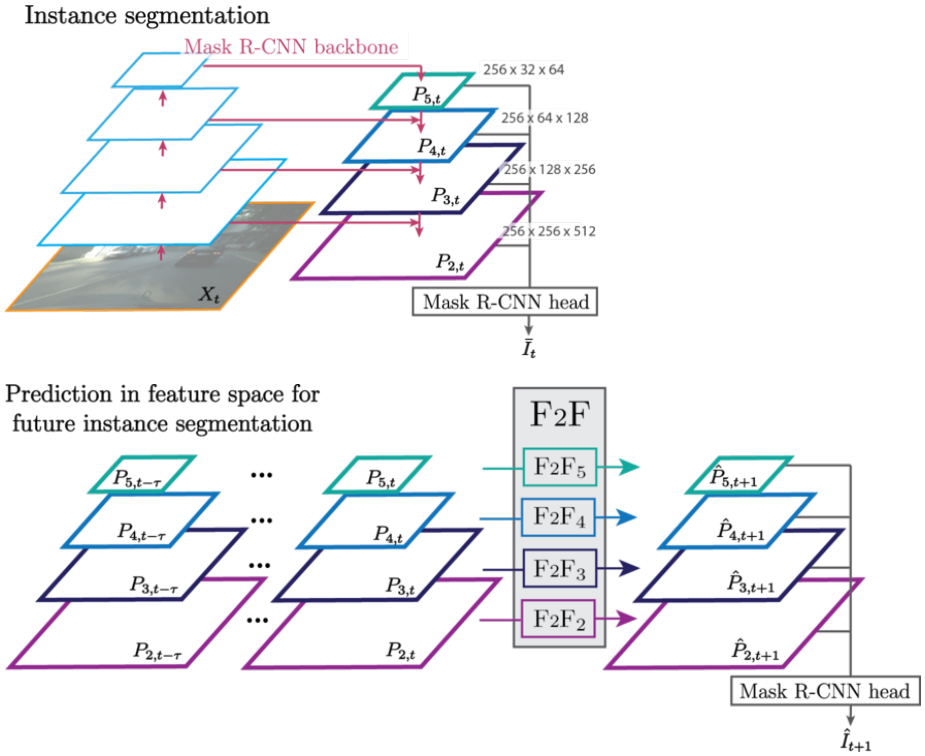


Figure 5.2: Top: Features in the FPN backbone are obtained by upsampling features in the top-down path, and combining them with features from the bottom-up path at the same resolution. Bottom: For future instance segmentation, we extract FPN features from frames $t-\tau$ to t, and predict the FPN features for frame $t+1$. We learn separate feature-to-feature prediction models for each FPN level: F2F$_l$ denotes the model for level l.

et al., 2016). We refer the reader to the top panel of Figure 5.2 for a schematic illustration, and to (He et al., 2017; Lin et al., 2017) for more details.

5.3.2 Forecasting convolutional features

Given a video sequence, our goal is to predict instance-level object segmentations for one or more future frames, *i.e.* for frames where we cannot access the RGB pixel values. Similar to previous work that predicts future RGB frames (Mathieu et al., 2016; Ranzato et al., 2014; Srivastava et al., 2015; Kalchbrenner et al., 2017) and future semantic segmentations, such as the approach presented in the previous chapter, we are interested in models where the input and output of the predictive model live in the same space, so that the model can be applied recursively to produce predictions for more than one frame ahead. The instance segmentations themselves, however, do not provide a suitable representation for prediction, since the instance-level labels vary in number between frames, and

are not consistent across video sequences. To overcome this issue, we instead resort to predicting the highest level feature volume in the Mask R-CNN architecture that describes the image as a whole. In particular, using the FPN backbone in Mask R-CNN, we want to learn a model that given the feature pyramids extracted from frames $X_{t-\tau}$ to X_t, predicts the feature pyramid for the unobserved RGB frame X_{t+1}.

Architecture. The features at the different FPN levels are trained to be input to a shared detection head, and are thus of similar nature. However, since the resolution changes across levels, the spatio-temporal dynamics are distinct from one level to another. Therefore, we propose a multi-scale approach, employing a separate network to predict the features at each level, of which we demonstrate the benefits in Section 5.4.1. The per-level networks are trained and function completely independently from each other. This allows us to parallelize the training across multiple GPUs. Alternative architectures in which prediction across different resolutions is tied are interesting, and could be the focus of future work. For each level, we concatenate the features of the input sequence along the feature dimension. We refer to the "feature to feature" predictive model for level l as $\mathrm{F2F}_l$. The overall architecture is summarized in the bottom panel of Figure 5.2.

Each of the $\mathrm{F2F}_l$ networks may be itself multi-scale, as in (Mathieu et al., 2016) and Chapter 4, depending on the scale hyperparameter chosen by cross-validation, to efficiently enlarge the field of view while preserving high-resolution details. More precisely, for a given level l, $\mathrm{F2F}_l$ consists of s_l subnetworks $\mathrm{F2F}_l^s$, where $s \in \{1, ..., s_l\}$, each implemented by a resolution-preserving CNN. The network $\mathrm{F2F}_l^{s_l}$ first processes the input downsampled by a factor of 2^{s_l-1}. Its output is up-sampled by a factor of 2, and concatenated to the input downsampled by a factor of 2^{s_l-2}. This concatenation constitutes the input of $\mathrm{F2F}_l^{s_l-1}$ which predicts a refinement of the initial coarse prediction. The same procedure is repeated until the final scale subnetwork $\mathrm{F2F}_l^1$. The design of subnetworks $\mathrm{F2F}_l^s$ is inspired by the S2S-dil model presented in Section 4.4.3, and recalled in Figure 5.3(a), leveraging dilated convolutions to further enlarge the field of view. In the following, since there is no ambiguity with other architectures, we drop the "-dil" notation and refer to it as S2S. Our architecture is shown in Figure 5.3(b). It differs in the number of feature maps per layer, the convolution kernel sizes and the dilation parameters, to make it more suited for the larger input dimension. Each subnetwork is fed with an input having a channel dimension $n \times p$, where n is the number of input frames, including the coarse prediction output by the previous subnetwork, and p is the channel dimension of the input and target feature space. In our experiments we have $n = 4$ (or $n = 5$ including the previous coarse prediction), and $p = 256$.

Training. We first train the $\mathrm{F2F}_5$ model to predict the coarsest features P_5, precomputed offline. Since the features of the different FPN levels are fed to the same recognition head network, the next levels are similar to the P_5 features. Hence, we initialize the weights of $\mathrm{F2F}_4$, $\mathrm{F2F}_3$, and $\mathrm{F2F}_2$ with the ones learned by $\mathrm{F2F}_5$, before fine-tuning them. For this, we compute features on the fly, due

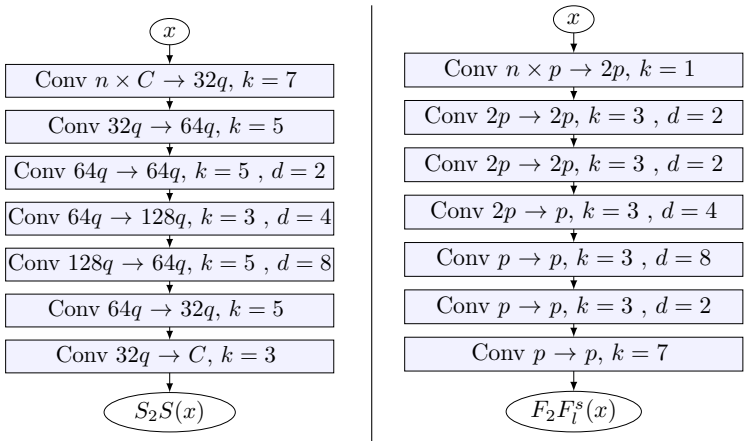


Figure 5.3: Architecture design of (a) S2S-dil introduced in Chapter 4, simply referred to as S2S in this chapter and (b) F2F$_l^s$. Each architecture takes the segmentations or features x for $n = 4$ frames in input, each respectively of channel dimension C or p. Each inner convolutional layer is followed by a ReLU. Each convolutional layer has a kernel of size $k \times k$ and dilation parameter d. In S2S, q is a hyperparameter used to scale the number of feature maps linearly for simple control over the capacity of the S2S model, set to 1.5 as in the previous chapter. Stride is always one and padding is chosen so as to maintain the size of the input.

to memory constraints. Each of the F2F$_l$ networks is trained using an ℓ_2 loss, since preliminary results showed improvement in this case over the ℓ_1 loss.

For multiple-time-step prediction, we can fine-tune each subnetwork F2F$_l$ autoregressively using backpropagation through time, similar to the previous chapter, to take into account error accumulation over time. In this case, given a single sequence of input feature maps, we train with a separate ℓ_2 loss on each predicted future frame. In our experiments, all models are trained in this autoregressive manner, unless specified otherwise.

5.4 Experimental evaluation

In this section we first present our experimental setup and baseline models, and then proceed with quantitative and qualitative results, that demonstrate the strengths of our F2F approach.

5.4.1 Experimental setup: dataset and evaluation metrics

Dataset. In our experiments, we use the Cityscapes dataset (Cordts et al., 2016) which contains 2,975 train, 500 validation and 1,525 test video sequences of 1.8 second each, recorded from a car driving in urban environments. Each sequence consists of 30 frames of resolution 1024×2048. Ground truth semantic and instance segmentation annotations are available for the 20-th frame of each sequence.

We employ a Mask R-CNN model pre-trained on the MS-COCO dataset (Lin et al., 2014) and fine-tune it in an end-to-end fashion on the Cityscapes dataset, using a ResNet-50-FPN backbone (He et al., 2016). The coarsest FPN level P5 has resolution 32×64, and the finest level P2 has resolution 256×512.

Following our setup for future semantic segmentation (Section 4.4), we temporally subsample the videos by a factor three, and take four frames as input. That is, the input sequence consists of feature pyramids for frames $\{X_{t-9}, X_{t-6}, X_{t-3}, X_t\}$. We denote by *short-term* and *mid-term* prediction respectively predicting X_{t+3} only (0.18 sec.) and predicting X_{t+3} through X_{t+9} (0.54 sec.). We additionally evaluate *long-term* predictions, corresponding predicting to X_{t+3} through X_{t+27} and 1.6 sec. ahead on the two long Frankfurt sequences of the Cityscapes validation set.

Conversion to semantic segmentation. For direct comparison to previous work, we also convert our instance segmentation predictions to semantic segmentation. To this end, we first assign to all pixels the *background* label. Then, we iterate over the detected object instances in order of ascending confidence score. For each instance, consisting of a confidence score c, a class k, and a binary mask m, we either reject it if it is lower than a threshold θ and accept it otherwise, where in our experiments we set $\theta = 0.5$. For accepted instances, we update the spatial positions corresponding to mask m with label k. This step potentially replaces labels set by instances with lower confidence, and resolves competing class predictions.

Evaluation metrics. To measure the instance segmentation performance, we use the standard Cityscapes metrics. The average precision metric AP50 counts an instance as correct if it has at least 50% of intersection over union (IoU) with the ground truth instance it has been matched with. The summary AP metric is given by average AP obtained with ten equally spaced IoU thresholds from 50% to 95%. Performance is measured across the eight classes with available instance-level ground truth: *person, rider, car, truck, bus, train, motorcycle,* and *bicycle*.

We measure semantic segmentation performance across the same eight classes. In addition to the IoU metric, computed w.r.t. the ground truth segmentation of the 20-th frame in each sequence, we also quantify the segmentation accuracy using three standard segmentation measures used by Yang et al., 2008, namely the probabilistic Rand index (RI) (Parntofaru and Hebert, 2005), global consistency error (GCE) (Martin et al., 2001), and variation of information (VoI) (Meilă, 2005). Good segmentation results are associated with high RI, low GCE

and low VoI. Since there is no ambiguity in terms of the ground truth used for the IoU metric, we drop the "-MO" notation in this chapter.

Implementation details. We cross-validate the number of scales, the optimization algorithm and hyperparameters per level of the pyramid. For each level of the pyramid a single scale network was selected, except for F_2F_2, where we employ 3 scales. The F_2F_5 network is trained for 60K iterations of SGD with Nesterov Momentum of 0.9, learning rate 0.01, and batch size of 4 images. It is used to initialize the other networks, which are trained for 80K iterations of SGD with Nesterov Momentum of 0.9, batch size of 1 image and learning rates of 5×10^{-3} for F_2F_4 and 0.01 for F_2F_3. For F_2F_2, which is much deeper than the models used at the other levels of the pyramid, due to the number of scales, we used Adam with learning rate 5×10^{-5} and default parameters.

5.4.2 Baseline models

As a performance upper bound, we report the accuracy of a Mask R-CNN oracle that has access to the future RGB frame. As a lower bound, we also use a trivial copy baseline that returns the segmentation of the last input RGB frame. In the following, we present two optical flow baselines, called *Warp* and *Shift*; a variant of the S2S model for discrete label maps; and Mask H2F, a baseline that repurposes and finetunes the Mask R-CNN architecture to predict future segmentation given the input frames. For completeness, we also include two weaker baselines, based on nearest neighbour search and on predicting the future RGB frames, and then segmenting them.

Optical flow baselines. We designed two baselines using the optical flow field computed from the last input RGB frame to the second last, as well as the instance segmentation predicted at the last input frame. As in Chapter 4, the flow fields are computed using Full Flow (Chen and Koltun, 2016) using the default parameters given by the authors on the MPI Sintel Flow Dataset (Butler et al., 2012).

The *Warp* approach consists in warping each instance mask independently using the flow field inside this mask. We initialize a separate flow field for each instance, equal to the flow field inside the instance mask and zero elsewhere. For a given instance, the corresponding flow field is used to project the values of the instance mask in the opposite direction of the flow vectors, yielding a new binary mask, using the t-centric method described in Section 4.4.2. To this predicted mask, we associate the class and confidence score of the input instance it was obtained from. To predict more than one time-step ahead, we also update the instance's flow field in the same fashion, to take into account the previously predicted displacement of physical points composing the instance. The predicted mask and flow field are used to make the next prediction, and so on. Maintaining separate flow fields allows competing flow values to coexist for the same spatial position, when they belong to different instances whose predicted trajectories lead them to overlap.

Prior to any post-processing, this baseline's predictions present some arti-

facts, as shown in Figure 5.4(a), in particular when objects are moving fast towards the camera. In this case, the optical flow should lead the predicted mask to become larger. But by construction, the number of pixels composing the masks can only stay equal or decrease in the warping process. Masks are therefore broken in parts corresponding to uniform areas of the flow field, and this phenomenon worsens with the number of steps. In order to remove these artifacts, we employ mathematical morphology operators to post-process the predictions. We employ a morphological closing, followed by a closing of holes on the masks. This addresses the problem in an effective manner, as shown in Figure 5.4(b).

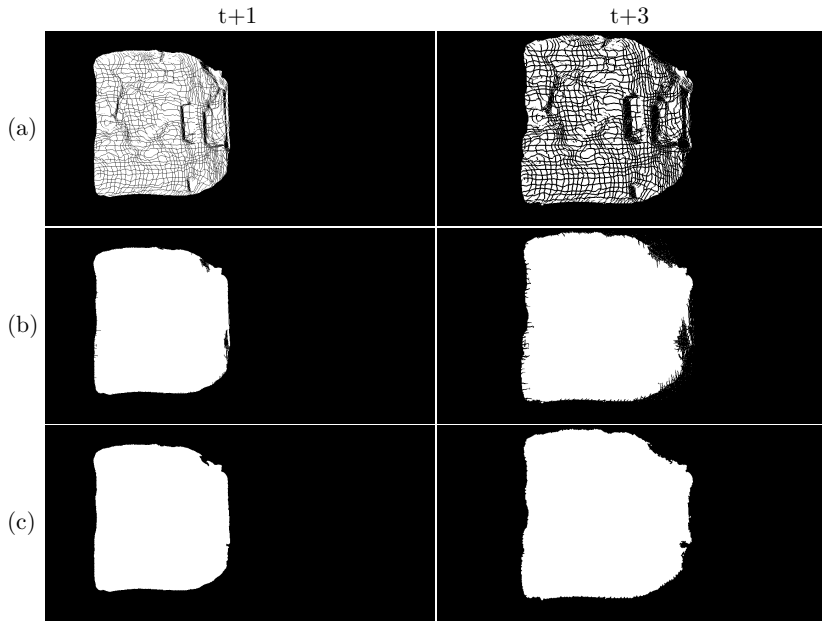


Figure 5.4: Qualitative comparison of future masks obtained using the *Warp* approach: (a) *w.o.* post-processing, (b) with closing operations, and (c) with full post-processing.

For mid-term predictions, we perform these operations on the output before it is used as input, at each time step. We use bilinear interpolation to estimate flow values at the added positions of the binary mask. This post-processing of the flow adds small spurious artifacts at the border of the the masks, visible in particular in Figure 5.4(b), right. These are easily removed using morphological openings, see Figure 5.4(c).

Warping the flow field when predicting multiple steps ahead suffers from error accumulation. To avoid this, we test another baseline, *Shift*, which shifts each mask with the average flow vector computed across the mask. To predict T time steps ahead, we simply shift the instance T times.

Future semantic segmentation using discrete label maps. For com-

parison with the future semantic segmentation approach presented in Chapter 4, which ignores instance-level labels, we train our S2S model on the label maps produced by Mask R-CNN. As before, we down-sample the Mask R-CNN label maps to 128×256. Unlike the soft label maps from the Dilation10 network (Yu and Koltun, 2016) used in the previous chapter, our converted Mask R-CNN label maps are discrete. For autoregressive prediction, we discretize our intermediate predictions by replacing the softmax network output with a one-hot encoding of the most likely class at each position. For autoregressive fine-tuning, we use a softmax activation with a low temperature parameter at the output of the S2S model, to produce near-one-hot probability maps in a differentiable way, enabling backpropagation through time.

Future segmentation using the Mask R-CNN architecture. As another baseline, we fine-tune Mask R-CNN to predict either short-term or mid-term future segmentation given the last 4 observed frames, denoted as the Mask H2F baseline. As initialization, we replicate the weights of the first layer learned on the COCO dataset across the 4 frames, and divide them by 4 to keep the features at the same scale.

Prediction in RGB space followed by segmentation. To show that prediction in the feature space is more effective than in the RGB space, we use Mask R-CNN to segment future RGB frames predicted using the X2X model presented in Section 4.3. We evaluate performance in two settings, with and without fine-tuning Mask R-CNN on the predicted (blurry) RGB frames of the training set, rather than the normal RGB frames. For fine-tuning, we keep the same optimization hyperparameters used for fine-tuning on the original Cityscapes dataset.

Nearest neighbour baseline. This baseline takes the P_5 features of the last observed frame, finds the nearest training frame in ℓ_2 distance on the features, and outputs the future segmentation of the matched frame. This segmentation corresponds to the ground-truth annotation if it is available, otherwise it is produced by the Mask R-CNN oracle. The searching set comprises the input frames used to train S2S and F2F, *i.e.* frames 2, 5, 8, 11 of each training sequence.

5.4.3 Results

Future instance segmentation. We first report results for the two weaker baselines we investigated, based on prediction of future RGB frames followed by segmentation, and on nearest neighbour search. When segmenting the predicted RGB frames, the resulting AP50 on the validation set of the Cityscapes dataset for short-term prediction is 6.9%, while the AP is 3.6%. This is much weaker than even the trivial copy baseline, which reaches 24.1% AP50 and 10.1% AP in the same setting. When fine-tuning Mask R-CNN on the predicted RGB frames, we obtain 19.2% AP50 and 8.6%, closer to, but still below the copy baseline. We show qualitative results in Figure 5.5.

The nearest neighbour baseline obtains even poorer results, with an AP50

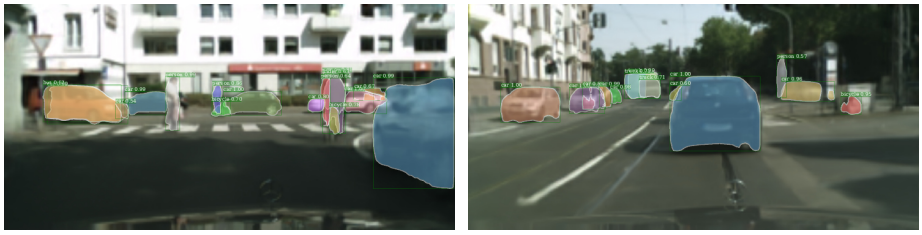


Figure 5.5: Short-term prediction in RGB space, followed by instance segmentation prediction using a Mask R-CNN model fine-tuned to this setting.

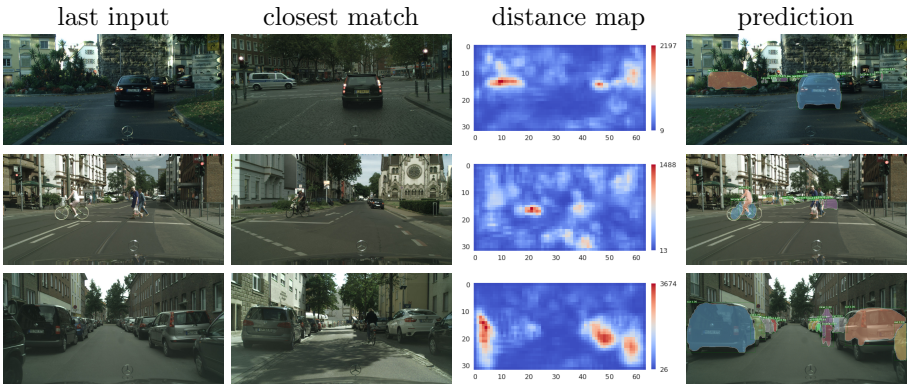


Figure 5.6: Nearest neighbour baseline. For each example, we show the last input frame, its closest match, the corresponding squared pixelwise ℓ_2 distance heat map and the predicted instance segmentations, visualized over the actual future frame.

of 0.3% and IoU of 7.9%. This is due to the limited size of the dataset, and the large number of instances present in each frame: each image contains on average 7 humans and 12 vehicles (Cordts et al., 2016). Although the nearest neighbour baseline sometimes accurately matches large instances, the other objects lead to a great number of false positives and false negatives, severely degrading the performance. We show examples where this occurs in Figure 5.6.

In Table 5.1 we report the performance of the *Warp* baseline corresponding to the illustrations in Figure 5.4: (a) before any post-processing is applied (*Warp w.o.* post-processing), (b) with closing operations only (*Warp w.o.* opening), and (c) with full post-processing (*Warp*). These results show that the post processing operations we employ significantly improve performance.

The *Shift* baseline leads to qualitatively better masks than the *Warp* baseline in cases where the optical flow field is not accurate enough. This approach, however, is unable to scale the objects, and is therefore unsuitable for long-term prediction when objects significantly change in scale as their distance to the camera changes. We illustrate this in Figure 5.7, in an example where a train is approaching the camera. At the first prediction, the mask predicted by

	Mid-term		
	AP50	AP	IoU
Warp w.o. post-processing	5.7	1.6	32.2
Warp w.o. opening	10.9	4.0	40.6
Warp	**11.1**	**4.1**	**41.4**

Table 5.1: Ablation study on the Cityscapes validation set for the post-processing operations employed by the *Warp* optical flow baseline.

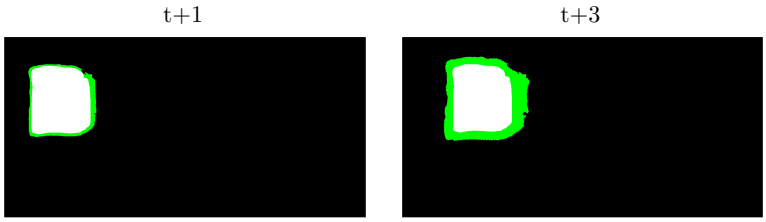


Figure 5.7: Comparison between the masks predicted by *Shift*, in white, and *Warp*, the union of the white and green zones. Predictions are shown for short and mid-term.

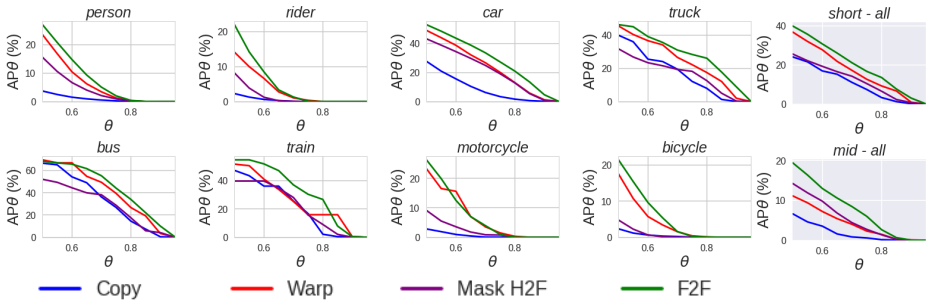
Shift has nicer contours than that of *Warp*. However, one can already see that the *Warp* mask is a bit larger. By the third prediction, we see that this has become much more accentuated. Disentangling the camera motion from that of the instances and incorporating additional geometric priors to additionally scale masks might improve the results of the *Shift* approach, and could be the basis for future work.

In Table 5.2 we present instance segmentation results of our future feature prediction approach (F2F) and compare it to the performance of the oracle, copy, optical flow and Mask H2F baselines. The copy baseline performs very poorly (24.1% in terms of AP50 *vs*. 65.8% for the oracle), which underlines the difficulty of the task. The two optical flow baselines perform comparably for short-term prediction, and are both much better than the copy baseline. As expected, for mid-term prediction, the *Warp* approach outperforms *Shift*. The Mask H2F baseline performs poorly for short-term prediction, but its results degrade slower with the number of time steps predicted, and it outperforms the *Warp* baseline for mid-term prediction. As Mask H2F outputs a single time step prediction, either for short or mid-term predictions, it is not subject to accumulation of errors, but each prediction setting requires training a specific model. Our F2F approach gives the best results overall, reaching more than 37% of relative improvement over our best mid-term baseline. While our F2F autoregressive fine-tuning makes little difference in case of short-term prediction (40.2% *vs*. 39.9% AP50 respectively), it gives a significant improvement for mid-term prediction (17.5% *vs*. 19.4% AP50 respectively).

In Figure 5.8(a), we show how the AP metric varies with the IoU threshold,

	Short-term		Mid-term	
	AP50	AP	AP50	AP
Mask R-CNN oracle	65.8	37.3	65.8	37.3
Copy last segmentation	24.1	10.1	6.6	1.8
Optical flow – *Shift*	37.0	16.0	9.7	2.9
Optical flow – *Warp*	36.8	16.5	11.1	4.1
Mask H2F *	25.5	11.8	14.2	5.1
F2F w/o ar. fine tuning	**40.2**	19.0	17.5	6.2
F2F	39.9	**19.4**	**19.4**	**7.7**

Table 5.2: Instance segmentation accuracy on the Cityscapes validation set for short-term (0.18 sec.) and mid-term (0.54 sec.) prediction. * Separate models were trained for short-term and mid-term predictions.



(a) Short-term - Individual classes scores (b) Overall

Figure 5.8: Instance segmentation APθ across different IoU thresholds θ. (a) Short-term prediction per class; (b) Average across all classes for short-term (top) and mid-term prediction (bottom).

for short-term prediction across the different classes and for each method. For individual classes, F2F gives the best results across thresholds, except for very few exceptions. In Figure 5.8(b), we show average results over all classes for short-term and mid-term prediction. We see that F2F consistently improves over the baselines across all thresholds, particularly for mid-term prediction.

Table 5.3 shows the positive impact of using each additional feature level, denoted by P_i–P_5 for $i = 2, 3, 4$. We also report performance when using all features levels, predicted by a model trained on the coarsest P_5 features, shared across levels, denoted by P_5 //. The drop in performance w.r.t. the column P_2-P_5 underlines the importance of training specific networks for each feature level.

We provide instance segmentation results on the Cityscapes test set in Table 5.4 for mid-term prediction, as obtained from the online evaluation server. For reference, we also computed the Mask R-CNN oracle results and the results

Levels	P_5	P_4–P_5	P_3–P_5	P_2–P_5	P_5 //
IoU	15.5	38.5	54.7	**60.7**	38.7
AP50	2.2	10.2	24.8	**40.2**	16.7

Table 5.3: Ablation study: short-term (0.18 sec.) prediction on the Cityscapes val. set.

of baselines *Warp* and Mask H2F. The results are comparable to those on the validation set, and we again observe that the predictions of our F2F model are far more accurate than those of the baselines.

	Mid-term	
	AP50	AP
Mask R-CNN oracle	58.1	31.9
Optical flow – *Warp*	11.8	4.3
Mask H2F	12.2	4.6
F2F	**17.5**	**6.7**

Table 5.4: Mid-term (0.54 sec.) instance segmentation results on Cityscapes test set.

	Short-term				Mid-term			
	IoU ↑	RI ↑	VoI ↓	GCE ↓	IoU ↑	RI ↑	VoI ↓	GCE ↓
Oracle (Chapter 4)	64.7	—	—	—	64.7	—	—	—
S2S (Chapter 4)	55.3	—	—	—	40.8	—	—	—
Oracle	73.3	94.0	20.8	2.3	73.3	94.0	20.8	2.3
Copy	45.7	92.2	29.0	3.5	29.1	90.6	33.8	4.2
Shift	56.7	92.9	25.5	2.9	36.7	91.1	30.5	3.3
Warp	58.8	**93.1**	25.2	3.0	41.4	91.5	31.0	3.8
Mask H2F *	46.2	92.5	27.3	3.2	30.5	91.2	31.9	3.7
S2S	55.4	92.8	25.8	2.9	**42.4**	91.8	29.7	3.4
F2F	**61.2**	**93.1**	**24.8**	**2.8**	41.2	**91.9**	**28.8**	**3.1**

Table 5.5: Short-term (0.18 sec.) and mid-term (0.54 sec.) semantic segmentation of moving objects (8 classes) performance on the Cityscapes validation set. * Separate models were trained for short-term and mid-term predictions.

Future semantic segmentation. We additionally provide a comparative evaluation on semantic segmentation in Table 5.5. First, we observe that our

discrete implementation of the S2S model performs slightly better than the one we evaluated in the previous chapter, thanks to our better underlying segmentation model (Mask R-CNN *vs*. the Dilation10 model of Yu and Koltun, 2016). Second, we see that the Mask H2F baseline performs weakly in terms of semantic segmentation metrics for both short and mid-term prediction, especially in terms of IoU. This may be due to frequently duplicated predictions for a given instance, see Section 5.4.3. Third, the advantage of *Warp* over *Shift* appears clearly again, with a 5% boost in mid-term IoU. Finally, we find that F2F obtains clear improvements in IoU over all methods for short-term segmentation, ranking first with an IoU of 61.2%. Our F2F mid-term IoU is comparable to those of the S2S and *Warp* baseline, while being much more accurate in depicting contours of the objects as shown by consistently better RI, VoI and GCE segmentation scores.

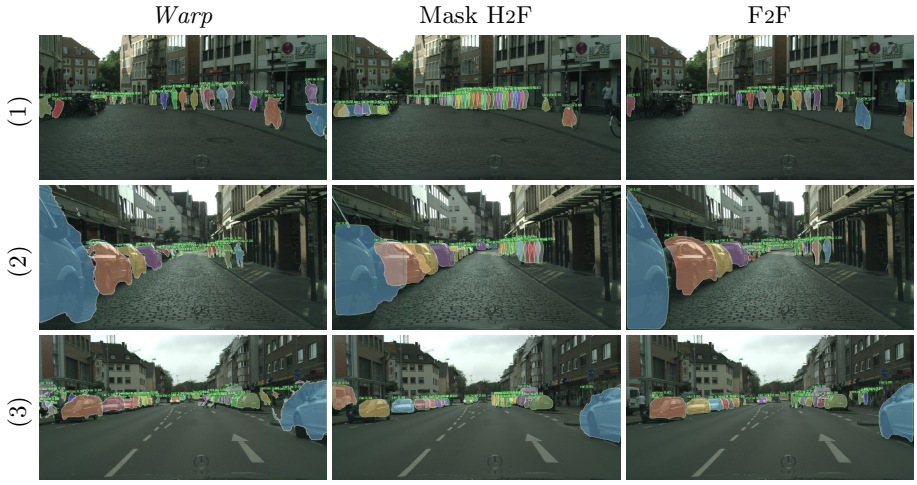


Figure 5.9: Mid-term instance segmentation predictions (0.54 sec.) for 3 sequences, from left to right: *Warp* baseline, Mask H2F baseline and F2F.

Qualitative Results. Figures 5.9 and 5.10 show representative results of our approach, both in terms of instance and semantic segmentation prediction, as well as results from the *Warp* and Mask H2F baselines for instance segmentation and S2S for semantic segmentation. We visualize predictions with a threshold of 0.5 on the confidence of masks. The Mask H2F baseline frequently predicts several masks around objects, especially for objects with ambiguous trajectories, like pedestrians, and less so for more predictable categories like cars. We speculate that this is due to the object-centric loss that the network is optimizing, which does not discourage this behaviour, and due to which the network is learning to predict several plausible future positions, as long as they overlap sufficiently with the ground-truth position. This does not occur with the other methods, which are either optimizing a per-pixel loss or are not learned at all. F2F results are often better aligned with the actual layouts of the objects

than the *Warp* baseline, showing that our approach has learned to model dynamics of the scene and objects more accurately than the baseline. As expected, the predicted masks are also much more precise than those of the S2S model, which is not instance-aware.

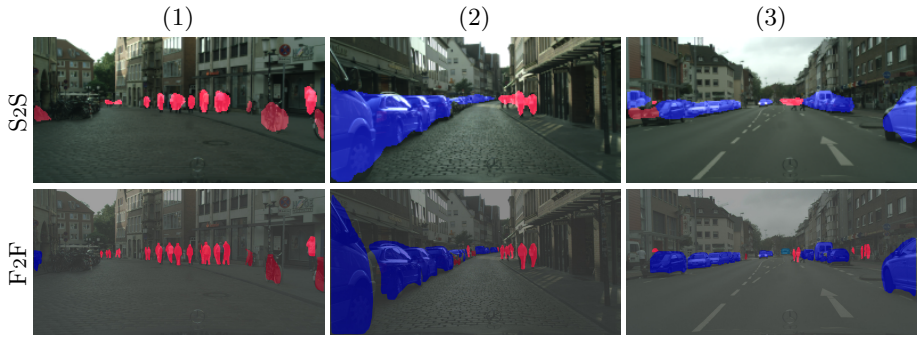


Figure 5.10: Mid-term semantic segmentation predictions (0.54 sec.) for 3 sequences. For each case we show from top to bottom: S2S model and F2F model.

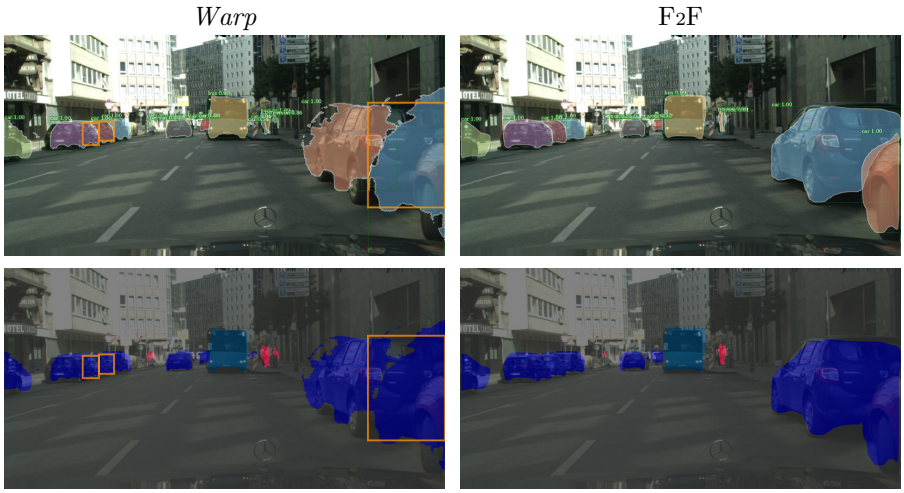


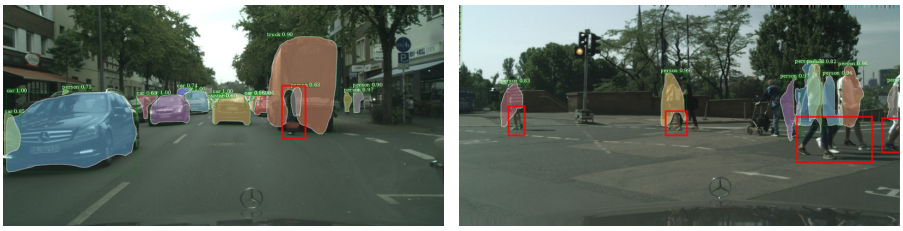
Figure 5.11: Mid-term predictions of instance and semantic segmentation with the *Warp* baseline and our F2F model. Inaccurate instance segmentations can result in accurate semantic segmentation areas; see orange rectangle highlights.

In Figure 5.11 we provide additional examples to better understand why the difference between F2F and the *Warp* baseline is smaller for semantic segmentation metrics than for instance segmentation metrics. When several instances of the same class are close together, inaccurate estimation of the instance masks

may still give acceptable semantic segmentation. This typically happens for groups of pedestrians and rows of parked cars. If an instance mask is split across multiple objects, this will further affect the AP measure than the IoU metric. The same example also illustrates common artifacts of the *Warp* baseline that are due to error accumulation in the propagation of the flow field.

5.4.4 Discussion

Failure cases. To illustrate some of the remaining challenges in predicting future instance segmentation we present several failure cases of our F2F model in Figure 5.12. In Figure 5.12(a), the masks predicted for the truck and the person are incoherent, both in shape and location. More consistent predictions might be obtained with a mechanism for explicitly modeling occlusions. Certain motions and shape transformations are hard to predict accurately due to the inherent ambiguity in the problem. This is, *e.g.*, the case for the legs of pedestrians in Figure 5.12(b), for which there is a high degree of uncertainty on the exact pose. Since the model is deterministic, it predicts a rough mask due to averaging over several possibilities. This may be addressed by modeling the intrinsic variability using GANs, VAEs, or autoregressive models (Kalchbrenner et al., 2017; Goodfellow et al., 2014; Kingma and Welling, 2014), presented in Section 2.2.

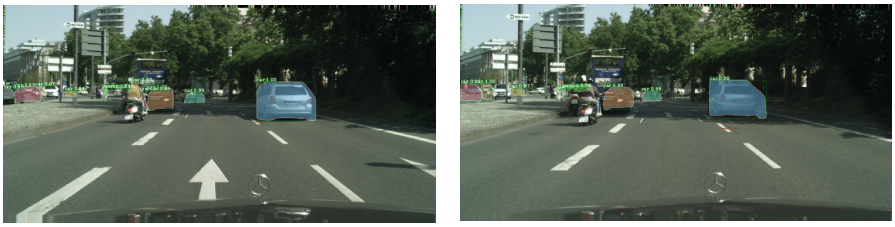


(a) (b)

Figure 5.12: Failure modes of mid-term prediction with the F2F model, highlighted with the red boxes: (a) incoherent masks and (b) lack of detail in highly deformable object regions, such as legs of pedestrians.

Long term prediction. In Figure 5.13, we show a prediction of F2F up to 1.5 sec. in the future in a sequence of the long Frankfurt video of the Cityscapes validation set, where frames are temporally subsampled by a factor three as before, resulting in a framerate of 5.6 Hz. To allow temporal consistency between predicted objects, we apply an adapted version of the method of Gkioxari and Malik, 2015 as a post-processing step. We define the linking score as the sum of confidence scores of subsequent instances and of their IoU. We then greedily compute the paths between instances which maximize these scores using the Viterbi algorithm. We thereby obtain object tracks along the (unseen) future video frames. Some object trajectories are forecasted reasonably well up to a second, such as the motorbike rider, while others are lost by that time such

as the motorbike. We also compute the AP with the ground truth of the long Frankfurt video. For each method, we give the best result of either predicting 9 frames with a frame interval of 3, or the opposite. For Mask H2F, only the latter is possible, as there are no such long sequences available for training. We obtain an AP of 0.5 for the flow and copy baseline, 0.7 for F2F and 1.5 for Mask H2F. All methods lead to very low scores, highlighting the severe challenges posed by this problem.



1 sec. 1.5 sec.

Figure 5.13: Long-term predictions (1.5 seconds) from our F2F model.

5.5 Conclusion

We introduced a new anticipated recognition task: predicting instance segmentation of future video frames. This task is defined at a semantically meaningful level rather the level of raw RGB values, and adds instance-level information as compared to predicting future semantic segmentation. We proposed a generic and self-supervised approach for anticipated recognition based on predicting the convolutional features of future video frames. In our experiments we apply this approach in combination with the Mask R-CNN instance segmentation model. We predict the internal "backbone" features which are of fixed dimension, and apply the "detection head" on these features to produce a variable number of predictions. Our results show that future instance segmentation can be predicted much better than naively copying the segmentations from the last observed frame, and that our future feature prediction approach significantly outperforms two strong baselines, the first one relying on optical-flow-based warping and the second on repurposing and fine-tuning the Mask R-CNN architecture for the task. When evaluated on the more basic task of semantic segmentation without instance-level detail, our approach yields performance quantitatively comparable to earlier approaches, while having qualitative advantages.

Our work shows that with a feed-forward network we are able to obtain surprisingly accurate results. More sophisticated architectures have the potential to further improve performance. Predictions may be also improved by explicitly modeling the temporal consistency of instance segmentation, and predicting

multiple possible futures rather than a single one.

We invite the reader to watch videos of our predictions at http://thoth.inrialpes.fr/people/pluc/instpred2018, and we provide the code for our Pytorch implementation at https://github.com/facebookresearch/instpred.

Chapter 6

Conclusion

Video prediction has been increasingly studied in recent years as a particular case of predictive learning, with broad applications in robotics and navigation systems. However, the task of predicting future RGB frames is excruciatingly challenging. Besides the challenge of handling the inherent uncertainty of the task, predicting future RGB frames involves disentangling all factors of variations of videos, anticipating each factor's evolution and combining them adequately to form RGB frames. Furthermore, the predicted RGB frames cannot be directly interpreted by downstream applications. These applications will in general extract high-level features prior to prediction, which will be invariant to some of the modeled factors. Such a pipeline therefore seems suboptimal.

For this reason, in this thesis, our goal was to study video prediction directly in high-level feature spaces, where prediction would be easier and directly interpretable by downstream algorithms. We also required that these feature spaces should remain rich in information and spatially detailed, so that they could be sufficient to model important concepts, relating to object and scene dynamics, as well as interactions between objects. These concepts are in turn necessary and sufficient for decision-making and planning in a wide variety of scenarios.

We considered semantic segmentation, as one of the most complete forms of visual understanding, presenting the intended characteristics. In Chapter 3, we first proposed a discriminative approach based on adversarial training for the task of semantic segmentation on still images. Observing that state-of-the-art segmentation models are trained with a per-pixel loss, and inspired by the success of generative adversarial networks, the motivation for our approach was that the discriminator could serve as a learned loss to regularize the segmentation model, by enforcing forms of higher-order consistency at training time, without adding complexity to the model used at test time.

Next, in Chapter 4, we introduced the task of predicting the semantic segmentation of future frames and proposed an autoregressive convolutional model to address. Leveraging the recent progress in the field, we used a state-of-the-art system to segment frames in a video dataset, so that the segmentations could serve as the inputs and targets of our predictive model.

While it is spatially detailed, semantic segmentation does not distinguish between distinct object instances belonging to the same class. Instance segmentation overcomes this shortcoming by additionally associating with each pixel an instance label. This additional level of detail is crucial to allow predictions relating to trajectories or interactions with objects, such as those met in robotics and autonomous navigation systems. For this reason, in Chapter 5, we extended our method to the more challenging problem of predicting future instance segmentation. To deal with a varying number of output labels per image, we developed a predictive model in the space of high-level convolutional image features of the Mask R-CNN instance segmentation model, and we use the predictions to produce the instance segmentation of future frames.

6.1 Summary of contributions

We summarize our contributions here.

1. In Chapter 3, we present, to the best of our knowledge, the first application of adversarial training to semantic segmentation. This approach enforces long-range spatial label contiguity, without adding complexity to the model used at test time. We propose and evaluate three input schemes for the discriminators, two of them designed to reduce the disparity between discrete ground truth segmentations and continous predicted segmentations. Our experimental results on the Stanford Background and Pascal VOC 2012 dataset show that our approach leads to improved labeling accuracy, with more pronounced gains for the smaller Stanford Background dataset. This work was published at the NIPS Workshop on Adversarial Training (Luc et al., 2016), and inspired a large body of works (190 citations at the time of writing).

2. In Chapters 4 and 5, we introduce the tasks of predicting future semantic and instance segmentation, which yield predictions that are spatially detailed, directly interpretable by downstream applications and semantically richer than any video prediction task we are aware of. Both tasks can be evaluated using metrics, inherited from the original recognition tasks on still images, and over which there is more consensus than over the metrics used in video prediction and image generation. We also observed that instance segmentation metrics were better at evaluating trajectories than semantic segmentation metrics, when multiple instances of the same class are close to each other.

3. In Chapter 4, we showed that error accumulation was slower for prediction of future semantic segmentation than for its RGB counterpart, supporting the fact that this task is better suited for recursive prediction and for prediction further into the future. We also found that when training a predictive segmentation model for single time-step prediction, adding an auxiliary task of predicting future RGB frames significantly hurt the

future segmentation performance. This suggests that rather than helping the model with extra supervision, the video prediction task is instead compromising some of the model capacity, supporting our initial hypothesis that the severe challenges associated with such low-level prediction are hindering the model's ability to learn the concepts that are more important for decision-making and planning.

4. In Chapters 4 and 5, we introduced strong models for these two tasks, based on advances in semantic segmentation, instance segmention and video prediction. These models are able to produce convincing predictions up to half a second into the future, on the Cityscapes dataset for urban scene understanding, that consists of videos of much higher complexity than have been attempted by video prediction approaches, especially in terms of number of instances per frame. We also showed important gains compared with a number of baselines relying on optical flow, repurposing and fine-tuning a state-of-the-art instance segmentation network. In Chapter 5, our multiscale approach allowed us to produce segmentations at a high resolution and up to half a second into the future, while video prediction approaches at comparable resolutions only evaluate the immediate next frame setting.

5. For both tasks introduced in Chapters 4 and 5, we showed that predicting directly in these semantically rich spaces is dramatically more accurate than predicting future RGB frames and then segmenting these. In the case of future instance segmentation, we showed that this holds when the segmentation model is fine-tuned on the predicted frames, due both to the restricted low resolution setting that can be afforded by RGB prediction and to the severe degradation of frames.

6. Finally, the methods presented in Chapters 4 and 5 do not require extremely costly, temporally dense video annotation, and instead rely on state-of-the-art segmentation models trained using image-level only annotations. Chapter 4 demonstrates the effectiveness of our general self-supervised approach in the case where the underling segmentation network produces unified image-level feature volumes (in the case of semantic segmentation), while Chapter 5 extends it to the more involved case where it produces object-level feature volumes (in the case of detection-based semantic segmentation). This demonstrates the wide applicability of our general approach, which allows different architectures for future segmentation prediction and still image segmentation to be swapped in, whether they are detection-based or segmentation-based. Additionally, the approach presented in Chapter 5 supports models that are able to produce multiple scene interpretations, such as surface normals, object bounding boxes, and human part labels, like the model proposed by (Kokkinos, 2017), without having to design appropriate encoders and loss functions for all these tasks to drive the future prediction. Other anticipated recognition tasks are also possible, such as human pose estimation. The work

presented in Chapter 4 was published at ICCV 2017 (Luc et al., 2017) and has already attracted significant interest (70 citations at the time of writing). Finally, the work presented in Chapter 5 was published at ECCV 2018 (Luc et al., 2018) and has been cited 7 times.

6.2 Perspectives

In this section, we propose some future perspectives based on the work presented in this thesis and recent advances in the field of computer vision and machine learning.

Handling uncertainties

We demonstrated the ability of our models to predict half a second ahead at a high resolution, in complex scenes that could occur in the context of autonomous driving. Predicting half a second ahead leaves relatively little uncertainty in general, which is why our deterministic models are able to produce convincing segmentations. Nevertheless, as described in Section 2.3.3, video prediction is inherently uncertain, and the degree of uncertainty naturally increases with the length of the offset at which we wish to predict. To improve the quality of the predictions and handle longer term prediction, it is necessary to learn a model that allows us to approximately sample from the different modes of the condition distribution over the future output sequence.

As described in Section 2.3.3, an important line of work in video prediction has focused on applying and extending ideas from generative modeling. Variational autoencoders (VAEs) (Kingma and Welling, 2014) have been shown to produce diverse results (Babaeizadeh et al., 2018; Denton and Fergus, 2018), while Generative adversarial networks (GANs) (Goodfellow et al., 2014) usually lead to sharper predictions (Vondrick et al., 2016b; Jang et al., 2018; Tulyakov et al., 2018) but struggle with diversity, in the setting where more than one input frame is used. At this point, it is useful to recall that in Chapter 4, the use of an adversarial loss had little impact in the setting of predicting future semantic segmentation, presumably due to the fact that blur in the predicted segmentation maps do not significantly affect the final segmentations, obtained by taking for each spatial position the label with highest probability. As a consequence, at least for short- and mid- term prediction, a VAE-based approach could be of more interest. Alternatively, exploring the application in the context of video prediction of work aiming to reduce mode collapse and mode dropping of GANs would be an interesting direction, *e.g.* (Lucas et al., 2018; Elfeki et al., 2018; Shmelkov et al., 2019).

Even when ignoring considerations relating to multi-modality, the integration of predictive models such as ours in critical applications like autonomous driving would require a quantification of the uncertainty of the model's predictions, so that downstream applications could modulate their reliance on the predictions. In some cases, inputs will be very similar to the training samples,

in which case the uncertainty should be small; while in others, inputs will be far from the samples the model has been trained on, and the uncertainty should be large. While the first source of uncertainty, due to the multiple modes, is independent of how much data the model has been trained on, the second could be decreased by increasing the size and the diversity of the dataset, as well as the model's capacity. Bhattacharyya et al., 2019 propose a Bayesian learning framework for our future semantic segmentation task, that is able to address both types of uncertainties. This framework leads to state-of-the-art results, with good diversity properties. Prediction of future instance segmentation could also largely benefit from the application of such methods.

Finally, while being extremely powerful models for image generation, from a scene understanding perspective, VAEs and GANs are limited in the sense that they learn an unstructured representation that lacks the explicit modeling of fundamental notions such as objects and their appearance, occlusion, and placement in the scene. While supervised (deep) learning models are able to locate and recognize objects with great precision, discovering and harnessing object-level structure of scenes in unsupervised generative models is a still a challenge, despite recent efforts (Eslami et al., 2016; Yang et al., 2017). The framework proposed by Eslami et al., 2016, described in Section 2.2.2, is particularly interesting, as the inference network learns without supervision to model a number of objects that is unknown and that can vary from image to image. This is at the cost of a relatively high-variance training procedure, even on a rather simplistic setting for 2D images, where occlusions between objects are not modeled and where objects are placed against a uniform black background. Kosiorek et al., 2018 propose a temporal extension of this framework, under the same restrictive assumptions. It is also important to improve the image generative model, imposing less restrictive assumptions, such as uniform backgrounds. Our preliminary experiments indicate that a separate background and foreground model can be learned using a masking schedule, that exploits the fact that objects are usually localized in a limited part of the image, while an important proportion of the background can be roughly inferred, given the rest of the image. At the beginning of training, the background model learns to reconstruct the image provided only half of it, while the foreground model has access to the whole image. Second, occlusion handling could involve predicting a permutation on the objects, specifying an order in which the objects can be decoded and pasted on the image canvas. The discrete permutation variable modeling the occlusion ordering can be modeled using the Gumbel-Sinkhorn approach of Mena et al., 2018. Finally, combinations with supervised approaches would be interesting. For exemple, we could replace the recurrent inference network with a recurrent instance segmentation method such as the works of Romera-Paredes and Torr, 2016; Ren and Zemel, 2017 or with a Region Proposal Network (RPN) (Ren et al., 2015), in both cases pretrained, to see how far image modeling capabilities can be pushed, and whether this could allow the RPN to learn to detect objects of unseen classes, without bounding box annotations.

Predicting multiple instance tracks

Downstream algorithms could benefit from a yet a richer type of prediction, consisting of instance tracks for the whole video, rather than per-image lists of instances. In Chapter 5, as a preliminary step towards this goal, we explored a simple tracking algorithm originally proposed by Gkioxari and Malik, 2015. At the introduction of the video object detection challenge ImageNet Vid (Russakovsky et al., 2015), the dominant approach consisted in performing image detection, followed by tracking and/or box-level post-processing (Han et al., 2016; Kang et al., 2016; Lee et al., 2016). Similar approaches could be used on top of our future instance segmentations, together with instance segmentations extracted from the input frames.

While deep learning tracking methods initially required learning a discriminative classifier online (Ma et al., 2015; Wang et al., 2015; Danelljan et al., 2015), very recently, siamese networks have provided a very efficient and performing paradigm (Bertinetto et al., 2016; Tao et al., 2016). In the approach proposed by Bertinetto et al., 2016, an embedding function is used on both the reference crop and each of the input frames. At each time step, the feature maps are cross-correlated, leading to a similarity map for each input frame, indicating where the target object's likely localization. The embedding function is learned discriminatively on positive and negative pairs. This framework has seen considerable improvements, with the works of Li et al., 2018a; Zhu et al.; He et al., 2018a; Yang et al., 2018; Wang et al., 2019. Of particular interest to us, Li et al., 2018a train a RPN (Ren et al., 2015) to propose regions that are likely to contain the target object, based on the embeddings of the reference crop and the input image. The very recent framework proposed by Wang et al., 2019 extends this with a parallel branch predicting a binary mask for each proposal, obtaining state-of-the-art performance. While this framework does not rely on intermediate detections, investigating the applicability of our method provides another exciting avenue, using this time the learned embedding space as inputs and targets, and relying on the predicted features to obtain future mask proposals. Conditioning the feature prediction on the RPN intermediate outputs ("sampled" by choosing the top scoring proposal) could help improve the consistency of trajectories.

Other approaches for video object detection instead extend two-stage detection pipelines to video volumes: *video tube proposals* are first produced, and then given as input to recognition branches (Kang et al., 2017; Vu et al., 2018); in a similar spirit to previous work for action detection (Oneata et al., 2014; Gemert et al., 2015; Kalogeiton et al., 2017). Features up to a certain level are still extracted per frame, before being aggregated temporally, and used for the tube proposal prediction and recognition. These methods could also be the basis for an extension of our method, where prediction would occur in the highest image feature level. Temporal aggregation over the input and predicted features would then allow video level prediction.

List of publications

The work presented in this thesis led to the following publications:

Predicting Future Instance Segmentation by Forecasting Convolutional Features, Pauline Luc, Camille Couprie, Yann LeCun and Jakob Verbeek, In ECCV 2018, Munich, Germany

Predicting Deeper into the Future of Semantic Segmentation, Pauline Luc, Natalia Neverova, Camille Couprie, Jakob Verbeek and Yann LeCun, In ICCV 2017, Venice, Italy

Semantic Segmentation using Adversarial Networks, Pauline Luc, Camille Couprie, Soumith Chintala and Jakob Verbeek, In NIPS Workshop on Adversarial Training 2016, Barcelona, Spain

Additional publications:

Joint Future Semantic and Instance Segmentation Prediction, Camille Couprie, Pauline Luc and Jakob Verbeek, In ECCV Workshop on Anticipating Human Behavior 2018, Munich, Germany

Appendix A

Conditional Random Fields

Let X and Y be two sets of random variables, with each random variable in Y taking its values in a finite set of labels $\mathcal{Y}$. We assume that X is observed, and our goal is to infer Y.

A very straight-forward definition can be given in terms of factor graphs, and motivates intuitively the use of CRFs in vision applications. A factor graph is a bipartite graph $G = (V, F, E)$ – *i.e.* a graph whose nodes are partitioned into two subsets V and F, and whose edges each connect one node of V to a node in F – used to describe of family of distributions over the multivariate random variable Z. Such a family is called an *undirected graphical model*.

We say that a distribution $p(\boldsymbol{z})$ factorizes according to the factor graph $G = (V, F, E)$ if each *variable node* in V indexes one of the components of Z and if there exists a set of *local functions* $\{\Psi_a\}_a \in F$ such that

$$\forall z, p(z) = \frac{1}{\mathcal{Z}} \prod_{a \in F} \Psi_a(z_{\mathcal{N}(a)}), \tag{A.1}$$

where $\mathcal{N}(a) \subseteq V$ denotes the neighbours of a in G. Hence the nodes belonging to F are called *factor nodes*. Here, Z is the *partition function* or *normalization constant*, ensuring the fact that $\sum_{z \in \mathcal{Z}} p(z) = 1$. From this we can deduce its expression:

$$\mathcal{Z} = \sum_{z} \prod_{a \in F} \Psi_a(z_{\mathcal{N}(a)}). \tag{A.2}$$

Let us now partition Z into the set of observed variables X and the set of variables to predict Y. Let $G = (V, F, E)$ be a factor graph, such that we can partition V into V_X and V_Y to index respectively the components of X and Y, and such that

$$\forall a \in F, \mathcal{N}(a) \not\subset V_X. \tag{A.3}$$

G defines a *conditional random field* over (X, Y) if the conditional distribution $p(y|x)$ factorizes according to G, *i.e.* if

$$\forall x, y \in \mathcal{X} \times \mathcal{Y}, p(y|x) = \frac{1}{\mathcal{Z}(x)} \prod_{a \in F} \Psi_a(x_a, y_a), \tag{A.4}$$

where $(x_a, y_a) = \{x_s; s \in \mathcal{N}(a)\} \cup \{y_s; s \in \mathcal{N}(a)\})$ and where again, we can express $\mathcal{Z}(x)$ as

$$\mathcal{Z}(x) = \sum_y \prod_{a \in F} \Psi_a(x_a, y_a)). \tag{A.5}$$

The idea behind a conditional random field is to avoid altogether the difficulties arising from modeling the complex relationships between observed variables, and instead focus on modeling directly what is of interest to us, *i.e.* predicting y from x by modeling $p(y|x)$. This is why we restrict ourselves to the factor graphs that do contains factors involving only variables in X and use them to represent $p(y|x)$ instead of the joint distribution $p(x, y)$.

Note that a resulting difference with a general undirected probabilistic model defined by a factor graph is that the partition function now depends on x, and is computed by summing over the y variables only. In some cases, this means that its computation will be tractable, where in contrast, a corresponding generative model might have been either too complex for inference to be tractable or too simplistic to be accurate.

Instead of parametrizing the factors directly though, in vision it is common to define instead *potentials*, which together sum to the *energy* $E(x, y)$ of the configuration y. The Gibbs distribution of this energy gives us the conditional distribution we are modeling:

$$P(Y|x) = \frac{e^{-\beta E(x,Y)}}{\mathcal{Z}(x)}, \tag{A.6}$$

where

$$\mathcal{Z}(x) = \sum_y e^{-\beta E(x,y)}. \tag{A.7}$$

The use of a classifier suggests a first obvious set of potentials, called the *unary potentials* or the *data terms*, corresponding to the initial classifier's predictions for each individual label y_s given the set of features extracted from x. Typically, we use the log-probabilities given by the classifier (*e.g.* when using multinomial logistic regression, we can directly use the softmax pre-activations). This ensures that the predictions will be those given by the original classifier if no other potentials were used.

Several additional sets of potentials and their parametrization have been proposed, as we will detail in the following. Commonly, pairwise potentials are used, *e.g.* between two neighbouring labels in an image. HIgher-order potentials

are more rarely used. These model interactions among larger numbers of labels, *e.g.* to model that most labels in a superpixel take the same value.

The parameters of a CRF are usually estimated using the maximum likelihood framework. Usually, computing the partition function Z is intractable. Indeed, consider one of the most basic connectivity pattern for a graph over the spatial positions, called the *4N-neighbourhood*, where pairwise interactions are defined between each site and its four closest sites. In this case, even the most sophisticated procedures for exact inference *e.g.* Junction Tree Algorithm cannot be applied, since it would require building a tree equivalent to the graph, by grouping the variables in the leaves. Doing this in a grid-like structure gives us a one-node tree containing the whole graph, so does not reduce the complexity of the problem. Approximate inference is therefore necessary, and can be used in one of two ways, as developed by Sutton and McCallum, 2012. The first way is by defining an easier objective, called a *surrogate*, ideally such that the maximum of this objective and of the true log likelihood will match. In this case, inference is necessary to approximate the partition function. The second way is by directly approximating the gradients to the log likelihood using approximated marginals. Indeed, the latter can be expressed as :

$$\nabla_{\theta_a} \log p(y|x) = \sum_{a \in F} \nabla_{\theta_a} \log \psi_a(x_a, y_a; \theta_a) - \nabla_{\theta_a} \log \mathcal{Z}(x), \tag{A.8}$$

and

$$\begin{aligned}
\nabla_{\theta_a} \log \mathcal{Z}(x) &= \frac{1}{\mathcal{Z}(x)} \nabla_{\theta_a} \sum_y \prod_{a' \in F} \psi'_a(x'_a, y'_a; \theta'_a) && \text{(A.9)} \\
&= \frac{1}{\mathcal{Z}(x)} \sum_y \prod_{a' \in F \setminus a} \psi'_a(x'_a, y'_a; \theta'_a) \nabla_{\theta_a} \psi_a(x_a, y_a; \theta_a) && \text{(A.10)} \\
&= \sum_y \frac{p(y|x)}{\psi_a(x_a, y_a; \theta_a)} \nabla_{\theta_a} \psi_a(x_a, y_a; \theta_a) && \text{(A.11)} \\
&= \sum_y p(y|x) \nabla_{\theta_a} \log \psi_a(x_a, y_a; \theta_a) && \text{(A.12)} \\
&= \sum_{y_a} p(y_a|x) \nabla_{\theta_a} \log \psi_a(x_a, y_a; \theta_a). && \text{(A.13)}
\end{aligned}$$

We see that we can substitute approximate marginals in Eq A.13 to obtain approximate gradients to the log likelihood, and use them in gradient based optimization methods. Note that when parameters are shared across cliques of the same type – as is common – we simply aggregate the obtained gradients across cliques.

Formulating a surrogate objective has the advantage that we know exactly what we are optimizing, which is necessary for certain optimization algorithms and useful from an analysis perspective. On the other hand, direct approximation of the gradients is more flexible because it allows the use of any approximate

inference algorithms, but conversely the learning procedure cannot in general be formulated as the optimization of a surrogate likelihood function.

Appendix B

Semantic Segmentation Metrics

Mean Intersection-over-Union The most common metric for evaluating performance is the mean intersection over union (mIoU), which is most clearly expressed in function of the confusion matrix M. It is defined as:

$$\text{mIOU} = \frac{1}{|\mathcal{Y}|} \sum_c \frac{M_{cc}}{\sum_i M_{ic} + \sum_i M_{ci} - M_{cc}}. \tag{B.1}$$

It has a very simple interpretation. Consider a class c, and the set of pixels predicted to belong to that class, versus the set of pixels actually belonging to that class according to the ground truth. The ratio of the sizes of the intersection and the union of these two sets can be at most 1, if the two sets are equal. Hence the closer it is to 1, the more accurate the predictions are. One averages this ratio over classes to obtain the final performance metric.

The reasons there is a consensus on using this metric instead of a simpler one, such as the accuracy, are the following. First, it is important to average the performance over the classes, because in most datasets, there is large imbalance between a *catch-all-remaining* class (*e.g.* "background" or "other") and the other classes. Second, when considering the average per-class accuracy, which would be the next obvious candidate, the problem is now that the metric is now biased towards methods that make coarse predictions, because that will in most cases significantly increase the accuracy of all classes, except for the single *catch-all-remaining* class, which will only be slightly decreased. Replacing accuracy with IoU solves this issue.

BF measure The BF measure, introduced by Csurka et al., 2013 is another metric of interest to us. It is complementary to the IoU, in that it focuses specifically on the quality of the contour, which significantly contributes to the perceived segmentation quality in some applications. This metric extends

a popular metric used in segmentation called the Berkeley contour matching score of Martin et al., 2004. Specifically, it measures how close the two contours are, by computing the proportion of points in the predicted boundary that are close enough to the ground truth boundary as a measure of precision, and the proportion of points in the ground truth boundary that are close enough to the predicted segmentation as a measure of recall, and summarize these into a $F1$ measure, which is aggregated across classes and images. Formally, for an image indexed by i, consider the union S_c^i of classes present in either the ground truth or the predicted segmentations. For each class c in S_c^i, calling $B_{gt}^{i,c}$ the boundary of the ground truth map for class c in image i, and $B_{ps}^{i,c}$ that of the binarized predicted segmentation, and given an distance error tolerance parameter θ^i, the precision $P^{i,c}$ and recall $R^{i,c}$ are defined as:

$$P^{i,c} = \frac{1}{|B_{ps}^{i,c}|} \sum_{p \in B_{ps}^{i,c}} [d(p, B_{gt}^{i,c} < \theta^i)], \tag{B.2}$$

$$R^{i,c} = \frac{1}{|B_{gt}^{i,c}|} \sum_{p \in B_{gt}^{i,c}} [d(p, B_{ps}^{i,c} < \theta^i)], \tag{B.3}$$

where d is the Euclidean distance, and $[x]$ denotes the Iverson brackets, equal to 1 if x is true and 0 otherwise. The tolerance in the distance error θ^i, used to decide whether a point has a match or not, is a computed as a fixed factor θ times the length of the image diagonal, so that the metric does not depend on the image size and can be computed across images of different sizes.

These can be used to compute a $\mathrm{F1}^{i,c}$ score:

$$\mathrm{F1}^{i,c} = \frac{2P^{i,c}R^{i,c}}{P^{i,c} + R^{i,c}}. \tag{B.4}$$

These scores are averaged over the set of classes S_c^i to obtain a $\mathrm{F1}^i$ score for each image, which are again averaged over the images to obtain the final performance metric BF.

Bibliography

Y. Abu Farha, A. Richard, and J. Gall. When will you do what? - anticipating temporal occurrences of activities. In *CVPR*, 2018.

P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *ICCV*, 2015.

A. Alahi, V. Ramanathan, and L. Fei-Fei. Socially-aware large-scale crowd forecasting. In *CVPR*, 2014.

P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *PAMI*, 33(5):898–916, 2011.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. In *ICML*, 2017.

A. Arnab and P.H.S. Torr. Pixelwise instance segmentation with a dynamically instantiated network. In *CVPR*, 2017.

A. Arnab, S. Jayasumana, S. Zheng, and P. Torr. Higher order conditional random fields in deep neural networks. In *ECCV*, 2016.

L. Ba and R. Caruana. Do deep nets really need to be deep? In *NIPS*, 2014.

M. Babaeizadeh, C. Finn, D. Erhan, R. H. Campbell, and S. Levine. Stochastic variational video prediction. In *ICLR*, 2018.

V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *PAMI*, 39(12):2481–2495, 2017.

M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In *CVPR*, 2017.

R. Bellman. A markovian decision process. *Indiana University Mathematics Journal*, 6(4):679–684, 1957.

S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. arXiv:1506.03099, 2015.

Y. Bengio and S. Bengio. Modeling high-dimensional discrete data with multi-layer neural networks. In *NIPS*, 1999.

L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr. Fully-convolutional siamese networks for object tracking. In *ECCV Workshops*, 2016.

E. De Bezenac, A. Pajot, and P. Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. In *ICLR*, 2018.

P. Bhattacharjee and S. Das. Temporal coherency based criteria for predicting video frames using deep multi-stage generative adversarial networks. In *NIPS*, 2017.

A. Bhattacharyya, M. Fritz, and B. Schiele. Long-term on-board prediction of people in traffic scenes under uncertainty. In *CVPR*, 2018.

A. Bhattacharyya, M. Fritz, and B. Schiele. Bayesian prediction of future street scenes using synthetic likelihoods. In *ICLR*, 2019.

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

G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *ECCV*, 2008.

A. Bubic, D. Y. Von Cramon, and R. Schubotz. Prediction, cognition and the brain. *Frontiers in Human Neuroscience*, 4:25, 2010.

D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.

W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki. Scene labeling with lstm recurrent neural networks. In *CVPR*, 2015.

W. Byeon, Q. Wang, R. K. Srivastava, and P. Koumoutsakos. Contextvp: Fully context-aware video prediction. In *ECCV*, 2018.

T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li. Mode regularized generative adversarial networks. In *ICLR*, 2017.

L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015.

Q. Chen and V. Koltun. Full flow: Optical flow estimation by global optimization over regular grids. In *CVPR*, 2016.

X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel. Variational lossy autoencoder. In *ICLR*, 2017.

K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. 2014.

Andy Clark. Whatever next? predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, 36:1–24, 2013.

D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24:603–619, 2002.

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.

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International Conference on Computer and Games*, 2006.

C. Couprie, P. Luc, and J. Verbeek. Joint future semantic and instance segmentation prediction. In *ECCV Workshop on Anticipating Human Behavior*, 2018.

G. Csurka, D. Larlus, and F. Perronnin. A simple high performance approach to semantic segmentation. In *BMVC*, 2008.

G. Csurka, D. Larlus, and F. Perronnin. What is a good evaluation measure for semantic segmentation? In *BMVC*, 2013.

J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *CVPR*, 2015.

J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016.

M. Danelljan, G. Häger, F. Shahbaz Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCV Workshops*, 2015.

D. de Geus, P. Meletis, and G. Dubbelman. Panoptic segmentation with a joint semantic and instance segmentation network. arXiv:1809.02110, 2018.

M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 72(7-9):1508–1524, 2009.

E. Denton and V. Birodkar. Unsupervised learning of disentangled representations from video. In *NIPS*, 2017.

E. Denton and R. Fergus. Stochastic video generation with a learned prior. In *ICML*, 2018.

E. Denton, S. Chintala, A. Szlam, and R. Fergus. Deep generative image models using a Laplacian pyramid of adversarial networks. In *NIPS*, 2015.

C. Doersch. Tutorial on variational autoencoders. arXiv:1606.05908, 2016.

A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *NIPS*, 2016.

A. Dosovitskiy and V. Koltun. Learning to act by predicting the future. In *ICLR*, 2017.

A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015a.

A. Dosovitskiy, J. Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In *CVPR*, 2015b.

M. Elfeki, C. Couprie, M. Rivière, and M. Elhoseiny. GDPP: learning diverse generations using determinantal point process. arXiv:1812.00068, 2018.

S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, K. Kavukcuoglu, and G. E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *NIPS*, 2016.

M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 111(1):98–136, 2015.

C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *PAMI*, 35(8):1915–1929, 2013.

C. Finn and S. Levine. Deep visual foresight for planning robot motion. In *ICRA*, 2017.

C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *NIPS*, 2016.

D. A. Forsyth, J. Malik, M. M. Fleck, H. Greenspan, T. K. Leung, S. J. Belongie, C. Carson, and C. Bregler. Finding pictures of objects in large collections of images. In *Object Representation in Computer Vision*, 1996.

D. Fouhey and C. Zitnick. Predicting object dynamics in scenes. In *CVPR*, 2014.

A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez. A review on deep learning techniques applied to semantic segmentation. arXiv:1704.06857, 2017.

L. A. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *NIPS*, 2015.

L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016.

J. Gauthier. Conditional generative adversarial nets for convolutional face generation. Unpublished, http://www.foldl.me/uploads/2015/conditional-gans-face-generation/paper.pdf, 2015.

J. Gemert, M. Jain, E. Gati, and C. G. Snoek. Apt: Action localization proposals from dense trajectories. In *British Machine Vision Association*, 2015.

S. Gershman and N. D. Goodman. Amortized inference in probabilistic reasoning. In *Cognitive Science Society*, 2014.

A. Ghosh, V. Kulharia, V. P. Namboodiri, P. H. S. Torr, and P. K. Dokania. Multi-agent diverse generative adversarial networks. In *CVPR*, 2018.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and segmentation. *PAMI*, 38(1):142–158, 2016.

G. Gkioxari and J. Malik. Finding action tubes. In *CVPR*, 2015.

P. W. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33(10):75–84, 1990.

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.

S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

D. Grangier, L. Bottou, and R. Collobert. Deep convolutional networks for scene parsing. In *ICML Deep Learning Workshop*, 2009.

L.-Y. Gui, Y.-X. Wang, X. Liang, and J. M. F. Moura. Adversarial geometry-aware human motion prediction. In *ECCV*, 2018.

I. Gulrajani, K. Kumar, F. Ahmed, A. Ali Taiga, F. Visin, D. Vazquez, and A. Courville. Pixelvae: A latent variable model for natural images. In *ICLR*, 2017.

W. Han, P. Khorrami, T. Le Paine, P. Ramachandran, M. Babaeizadeh, H. Shi, J. Li, S. Yan, and T. S. Huang. Seq-nms for video object detection. arXiv:1602.08465, 2016.

B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *ICCV*, 2011.

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

B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015.

A. He, C. Luo, X. Tian, and W. Zeng. Towards a better match in siamese network based visual object tracker. In *ECCV Workshops*, 2018a.

K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

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

K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, 2017.

K. He, R. Girshick, and P. Dollár. Rethinking imagenet pre-training. arXiv:1811.08883, 2018b.

X. He and R. S. Zemel. Learning hybrid models for image annotation with partially labeled data. In *NIPS*, 2009.

X. He, R. S. Zemel, and M. Á. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *CVPR*, 2004.

G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008.

M. Henaff, A. Canziani, and Y. LeCun. Model-predictive policy learning with uncertainty regularization for driving in dense traffic. In *ICLR*, 2019.

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning Workshop*, 2014.

M. Hoai and F. De la Torre. Max-margin early event detectors. *IJCV*, 107(2): 191–202, 2014.

Q. Hoang, T. D. Nguyen, T. Le, and D. Phung. MGAN: Training generative adversarial nets with multiple generators. In *ICLR*, 2018.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

J. Hohwy. *The predictive mind.* Oxford University Press, New-York, NY, US, 2013.

M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015.

Y. Jang, G. Kim, and Y. Song. Video prediction with appearance and motion conditions. In *ICML*, 2018.

D. Jayaraman, F. Ebert, A. A. Efros, and S. Levine. Time-agnostic prediction: Predicting predictable video frames. In *ICLR*, 2019.

X. Jin, X. Li, H. Xiao, X. Shen, Z. Lin, J. Yang, Y. Chen, J. Dong, L. Liu, Z. Jie, J. Feng, and S. Yan. Video scene parsing with predictive feature learning. 2017a.

X. Jin, H. Xiao, X. Shen, J. Yang, Z. Lin, Y. Chen, Z. Jie, J. Feng, and S. Yan. Predicting scene parsing and motion dynamics in the future. In *NIPS*, 2017b.

J. Johnson, A. Alahi, and F.-F. Li. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

N. Kalchbrenner, A. van den Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu. Video pixel networks. In *ICML*, 2017.

V. Kalogeiton. *Localizing spatially and temporally objects and actions in videos*. PhD thesis, University of Edinburgh ; INRIA Grenoble, 2017.

V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid. Action tubelet detector for spatio-temporal action localization. In *ICCV*, 2017.

K. Kang, W. Ouyang, H. Li, and X. Wang. Object detection from video tubelets with convolutional neural networks. In *CVPR*, 2016.

K. Kang, H. Li, T. Xiao, W. Ouyang, J. Yan, X. Liu, , and X. Wang. Object detection in videos with tubelet proposal networks. In *CVPR*, 2017.

T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018.

D. Kingma and M. Welling. Auto-encoding variational Bayes. In *ICLR*, 2014.

A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. arXiv:1801.00868, 2017.

A. Kirillov, R. B. Girshick, K. He, and P. Dollár. Panoptic feature pyramid networks. arXiv:1901.02446, 2019.

K. Kitani, B. Ziebart, J. Bagnell, and M. Hebert. Activity forecasting. In *ECCV*, 2012.

P. Kohli, L. Ladický, and P. Torr. Robust higher order potentials for enforcing label consistency. *IJCV*, 82(3):302–324, 2009.

I. Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, 2017.

S. Kong and C. Fowlkes. Recurrent pixel embedding for instance grouping. In *CVPR*, 2018.

A. R. Kosiorek, H. Kim, I. Posner, and Y.-W. Teh. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in Neural Information Processing Systems*, 2018.

P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*, 2011.

A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

M. P. Kumar and D. Koller. Efficiently selecting regions for scene understanding. In *CVPR*, 2010.

S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. 2003.

L. Ladicky, C. Russell, P. Kohli, and P. H. S. Torr. Associative hierarchical crfs for object class image segmentation. In *ICCV*, 2009.

L. Ladicky, P. Sturgess, K. Alahari, C. Russell, and P. H. S. Torr. What, where & how many? combining object detectors and crfs. In *ECCV*, 2010.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

A. Lamb, V. Dumoulin, and A. C. Courville. Discriminative regularization for generative models. arXiv:1602.03220, 2016.

T. Lan, T.C. Chen, and S. Savarese. A hierarchical representation for future action prediction. In *ECCV*, 2014.

D. Larlus and F. Jurie. Combining appearance models and markov random fields for category level object segmentation. In *CVPR*, 2009.

H. Larochelle and I. Murray. The neural autoregressive distribution estimator. In *AISTATS*, 2011.

A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016.

A. X. Lee, R. Zhang, F. Ebert, P. Abbeel, C. Finn, and S. Levine. Stochastic adversarial video prediction. arXiv:1804.01523, 2018.

B. Lee, E. Erdenee, S. Jin, M. Y. Nam, Y. G. Jung, and P. K. Rhee. Multi-class multi-object tracking using changing point detection. In *ECCV*, 2016.

N. Lee, W. Choi, P. Vernaza, C. Choy, P. Torr, and M. Chandraker. DESIRE: distant future prediction in dynamic scenes with interacting agents. In *CVPR*, 2017.

V. Lempitsky, A. Vedaldi, and A. Zisserman. Pylon model for semantic segmentation. In *NIPS*, 2011.

B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018a.

J. Li, A. Raventos, A. Bhargava, T. Tagawa, and A. Gaidon. Learning to fuse things and stuff. arXiv:1812.01192, 2018b.

K. Li, B. Hariharan, and J. Malik. Iterative instance segmentation. In *CVPR*, 2016.

Q. Li, A. Arnab, and P.H.S. Torr. Weakly- and semi-supervised panoptic segmentation. In *ECCV*, 2018c.

Y. Li, X. Chen, Z. Zhu, L. Xie, G. Huang, D. Du, and X. Wang. Attention-guided unified network for panoptic segmentation. arXiv:1812.03904, 2018d.

Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang. Flow-grounded spatial-temporal video prediction from still images. In *ECCV*, 2018e.

G. Lin, C. Shen, A. van den Hengel, and I. Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *CVPR*, 2016.

T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014.

T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

Z. Lin, A. Khetan, G. Fanti, and S. Oh. Pacgan: The power of two samples in generative adversarial networks. In *NIPS*, 2018.

S. Liu, J. Jia, S. Fidler, and R. Urtasun. SGN: sequential grouping networks for instance segmentation. In *CVPR*, 2017a.

W. Liu, D. Lian W. Luo, and S. Gao. Future frame prediction for anomaly detection – a new baseline. In *CVPR*, 2018.

Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *ICCV*, 2017b.

J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

P. Luc, C. Couprie, S. Chintala, and J. Verbeek. Semantic segmentation using adversarial networks. In *NIPS Workshop on Adversarial Training*, 2016.

P. Luc, N. Neverova, C. Couprie, J. Verbeek, and Y. LeCun. Predicting deeper into the future of semantic segmentation. In *ICCV*, 2017.

P. Luc, C. Couprie, Y. LeCun, and J. Verbeek. Predicting future instance segmentation by forecasting convolutional features. In *ECCV*, 2018.

T. Lucas and J. Verbeek. Auxiliary guided autoregressive variational autoencoders. In *ECML-PKDD*, 2018.

T. Lucas, C. Tallec, Y. Ollivier, and J. Verbeek. Mixed batches and symmetric discriminators for GAN training. In *ICML*, 2018.

Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. In *CVPR*, 2017.

C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015.

D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001.

D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *PAMI*, 26(5):530–549, 2004.

M. Mathieu. *Unsupervised Learning Under Uncertainty*. PhD thesis, New York University, 2017.

M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016.

M. Meilă. Comparing clusterings: An axiomatic view. In *ICML*, 2005.

G. Mena, D. Belanger, S. Linderman, and J. Snoek. Learning latent permutations with gumbel-sinkhorn networks. In *ICLR*, 2018.

L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. In *ICLR*, 2017.

P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell. Learning to navigate in complex environments. 2017.

M. Mirza and S. Osindero. Conditional generative adversarial nets. In *NIPS Deep Learning Workshop*, 2014.

A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. In *ICML*, 2014.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. 2013.

S. Mohamed and D. J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *NIPS*, 2015.

D. Munoz, A. Bagnell, and M. Hebert. Stacked hierarchical labeling. In *ECCV*, 2010.

P. Chris A. Courville N. Ballas, L. Yao. Delving deeper into convolutional networks for learning video representations. In *ICLR*, 2016.

S. S. Nabavi, M. Rochan, and Y. Wang. Future semantic segmentation with convolutional LSTM. 2018.

R. M. Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 56 (1):71–113, 1992.

T. Nguyen, T. Le, H. Vu, and D. Phung. Dual discriminator generative adversarial nets. In *NIPS*, 2017.

D. Nilsson and C. Sminchisescu. Semantic video segmentation by gated recurrent flow propagation. In *CVPR*, 2018.

H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *ICCV*, 2015.

J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. P. Singh. Action-conditional video prediction using deep networks in atari games. In *NIPS*, 2015.

M. Oliu, J. Selva, and S. Escalera. Folded recurrent neural networks for future video prediction. In *ECCV*, 2018.

D. Oneata, J. Revaud, J. Verbeek, and C. Schmid. Spatiotemporal object detection proposals. In *ECCV*, 2014.

P.-Y. Oudeyer and F. Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers in neurorobotics*, 1(6), 2007.

C. Parntofaru and M. Hebert. A comparison of image segmentation algorithms. Technical Report CMU-RI-TR-05-40, Carnegie Mellon University, 2005.

D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

V. Patraucean, A. Handa, and R. Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *ICLR Workshop Track*, 2016.

D. Pavllo, D. Grangier, and M. Auli. Quaternet: A quaternion-based recurrent model for human motion. 2018.

D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli. Modeling human motion with quaternion-based neural networks. arXiv:1901.07677, 2019.

M. Pei, Y. Jia, and S.C. Zhu. Parsing video events with goal inference and intent prediction. In *ICCV*, 2011.

P. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *ICML*, 2014.

P.O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

M.A. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra. Video (language) modeling: a baseline for generative models of natural videos. arXiv:1412.6604, 2014.

F. A. Reda, G. Liu, K. J. Shih, R. Kirby, J. Barker, D. Tarjan, A. Tao, and B. Catanzaro. Sdc-net: Video prediction using spatially-displaced convolution. In *ECCV*, 2018.

S. Reed, Y. Zhang, Y. Zhang, and H. Lee. Deep visual analogy-making. In *NIPS*, 2015.

S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *ICML*, 2016.

M. Ren and R.S. Zemel. End-to-end instance segmentation with recurrent attention. In *CVPR*, 2017.

S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, 2015.

D.J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

B. Romera-Paredes and P. Torr. Recurrent instance segmentation. In *ECCV*, 2016.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, 2015.

S. Roweis, L. Saul, and G. Hinton. Global coordination of local linear models. In *NIPS*, 2002.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. In *IJCV*, 2015.

R. Salakhutdinov and G. Hinton. Deep boltzmann machines. In *AISTATS*, 2009.

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *NIPS*, 2016.

S. Saxena and J. Verbeek. Convolutional neural fabrics. In *NIPS*, 2016.

F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

A. Schwing and R. Urtasun. Fully connected deep structured networks. arXiv:1503.02351, 2015.

P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine. Time-contrastive networks: Self-supervised learning from video. 2018.

S. Shalev-Shwartz and A. Shashua. On the sample complexity of end-to-end training vs. semantic abstraction training. arXiv:1604.06915, 2016.

S. Shalev-Shwartz, N. Ben-Zrihem, A. Cohen, and A. Shashua. Long-term planning by short-term prediction. arXiv:1602.01580, 2016.

X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, 2015.

K. Shmelkov, T. Lucas, K. Alahari, C. Schmid, and J. Verbeek. Coverage and quality driven training of generative image models. arXiv:1901.01091, 2019.

J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

P. J. Silvia. *Curiosity and motivation*, pages 157—-167. Oxford University Press, New-York, NY, US, 2012.

K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *NIPS*, 2017.

N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using LSTMs. In *ICML*, 2015.

M. F. Stollenga, W. Byeon, M. Liwicki, and J. Schmidhuber. Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation. In *NIPS*, 2015.

M. Sun, B. Kim, P. Kohli, and S. Savarese. Relating things and stuff via object property interactions. *PAMI*, 36(7):1370–1383, 2014.

C. Sutton and A. McCallum. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2012.

R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

R. S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bulletin*, 2(4):160–163, 1991.

G. Synnaeve, Z. Lin, J. Gehring, D. Gant, V. Mella, V. Khalidov, N. Carion, and N. Usunier. Forward modeling for partial observation strategy games - A starcraft defogger. In *NIPS*, 2018.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

R. Tao, E. Gavves, and A. W. M. Smeulders. Siamese instance search for tracking. In *CVPR*, 2016.

D. Tarlow and R. Zemel. Structured output learning with high order loss functions. In *AISTATS*, 2012.

A. W. Terwilliger, G. Brazil, and X. Liu. Recurrent flow-guided semantic forecasting. In *WACV*, 2019.

G. Tesauro. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3), 1995.

J. Tighe and S. Lazebnik. Superparsing - scalable nonparametric image parsing with superpixels. *IJCV*, 101(2):329–349, 2013.

I. O. Tolstikhin, S. Gelly, O. Bousquet, C.-J. Simon-Gabriel, and B. Schölkopf. Adagan: Boosting generative models. In *NIPS*, 2017.

D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. In *CVPR*, 2018.

A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *ICML*, 2016.

J. Verbeek and W. Triggs. Region classification with markov field aspect models. In *CVPR*, 2007.

R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. In *ICLR*, 2017a.

R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee. Learning to generate long-term future via hierarchical prediction. In *ICML*, 2017b.

O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, W. M. Czarnecki, A. Dudzik, A. Huang, P. Georgiev, R. Powell, T. Ewalds, D. Horgan, M. Kroiss, I. Danihelka, J. Agapiou, J. Oh, V. Dalibard, D. Choi, L. Sifre, Y. Sulsky, S. Vezhnevets, J. Molloy, T. Cai, D. Budden, T. Paine, C. Gulcehre, Z. Wang, T. Pfaff, T. Pohlen, Y. Wu, D. Yogatama, J. Cohen, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, C. Apps, K. Kavukcuoglu, D. Hassabis, and D. Silver. Alphastar: Mastering the real-time strategy game starcraft ii. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, 2019. Accessed: 2019-03-24.

F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho, Y. Bengio, M. Matteucci, and A. Courville. Reseg: A recurrent neural network-based model for semantic segmentation. In *CVPR*, 2016.

C. Vondrick and A. Torralba. Generating the future with adversarial transformers. In *CVPR*, 2017.

C. Vondrick, P. Hamed, and A. Torralba. Anticipating the future by watching unlabeled video. In *CVPR*, 2016a.

C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *NIPS*, 2016b.

C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy. Tracking emerges by colorizing videos. In *ECCV*, 2018.

S. Vora, R. Mahjourian, S. Pirk, and A. Angelova. Future semantic segmentation leveraging 3d information. arXiv:1811.11358, 2018.

T.-H. Vu, W. Choi, S. Schulter, and M. Chandraker. Memory warps for learning long-term online video representations. arXiv:1803.10861, 2018.

N. Wahlström, T. B. Schön, and M. P. Deisenroth. From pixels to torques: Policy learning with deep dynamical models. In *ICML Deep Learning Workshop*, 2015.

J. Walker, A. Gupta, and M. Hebert. Dense optical flow prediction from a static image. In *ICCV*, 2015.

J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *ECCV*, 2016.

J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014.

J. Y. A. Wang and E. H. Adelson. Layered representation for motion analysis. In *CVPR*, 1993.

L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *ICCV*, 2015.

Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019.

W. Zhou Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

Y. Wang, M. Long, J. Wang, Z. Gao, and P. S. Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. In *NIPS*, 2017.

C. J. C. H. Watkins and P. Dayan. Technical note q-learning. *Machine Learning*, 8:279–292, 1992.

M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *NIPS*, 2015.

T. Weber, S. Racanière, D. P. Reichert, L. Buesing, A. Guez, D. J. Rezende, A. P. Badia, O. Vinyals, N. Heess, Y. Li, R. Pascanu, P. Battaglia, D. Hassabis, D. Silver, and D. Wierstra. Imagination-augmented agents for deep reinforcement learning. In *NIPS*, 2017.

P. Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, 1(4):339–356, 1988.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.

J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *CVPR*, 2006.

Chang Xiao, Peilin Zhong, and Changxi Zheng. Bourgan: Generative networks with metric embeddings. In *NIPS*, 2018.

W. Xiong, W. Luo, L. Ma, W. Liu, and J. Luo. Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks. In *CVPR*, 2018.

Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun. Upsnet: A unified panoptic segmentation network. arXiv:1901.03784, 2019.

J. Xu, B. Ni, Z. Li, S. Cheng, and X. Yang. Structure preserving video prediction. In *CVPR*, 2018.

T. Xue, J. Wu, K. Bouman, and W. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *NIPS*, 2016.

A. Yang, J. Wright, Y. Ma, and S. Sastry. Unsupervised segmentation of natural images via lossy data compression. *CVIU*, 110(2):212–225, 2008.

C. Yang, Z. Wang, X. Zhu, C. Huang, J. Shi, and D. Lin. Pose guided human video generation. In *ECCV*, 2018.

J. Yang, A. Kannan, D. Batra, and D. Parikh. LR-GAN: Layered recursive generative adversarial networks for image generation. In *ICLR*, 2017.

Tien-Ju Yang, M. D. Collins, Y. Zhu, J.-J. Hwang, T. Liu, X. Zhang, V. Sze, G. Papandreou, and L.-C. Chen. Deeperlab: Single-shot image parser. arXiv:1902.05093, 2019.

F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

H. Zhang, T. Xu, and H. Li. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017.

L. Zhang, L. Lu, R. M. Summers, E. Kebebew, and J. Yao. Convolutional invasion and expansion networks for tumor growth prediction. *IEEE Transactions on Medical Imaging*, 37(2):638–648, 2018.

L. Zhang, L. Lu, R. Zhu, M. Bagheri, R. M. Summers, and J. Yao. Spatial-temporal convolutional lstms for tumor growth prediction by learning 4d longitudinal patient data. arXiv:1902.08716, 2019.

S. Zheng, M.-M. Cheng, J. Warrell, P. Sturgess, V. Vineet, and C. Rother P. H. S. Torr. Dense semantic image segmentation with objects and attributes. In *CVPR*, 2014.

S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015.

Y. Zhou and T. Berg. Learning temporal transformations from time-lapse videos. In *ECCV*, 2016.

Y. Zhou, X. Hu, and B. Zhang. Interlinked convolutional neural networks for face parsing. In *International Symposium on Neural Networks*, 2015.

G. Zhu, Z. Huang, and C. Zhang. Object-oriented dynamics predictor. In *NIPS*, 2018a.

H. Zhu, F. Meng, J. Cai, and S. Lu. Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation. *Journal of Visual Communication and Image Representation*, 34:12–27, 2016.

Y. Zhu, Y. Tian, D. N. Metaxas, and P. Dollár. Semantic amodal segmentation. In *CVPR*, 2017.

Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. D. Newsam, A. Tao, and B. Catanzaro. Improving semantic segmentation via video propagation and label relaxation. 2018b.

Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking.

A. Ziat, E. Delasalles, L. Denoyer, and P. Gallinari. Spatio-temporal neural networks for space-time series forecasting and relations discovery. In *International Conference on Data Mining*, 2017.