Learning with weak supervision using deep generative networks

Mickaël Chen

THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ

Spécialité **Informatique**

École Doctorale Informatique, Télécommunications et Électronique (Paris)

Sujet de la thèse :

Learning with Weak Supervision Using Deep Generative Networks

Apprentissage en Supervision Faible par l'Emploi de Réseaux Génératifs Profonds

Par

Mickaël Chen

Présentée et soutenue publiquement le 02 juillet 2020
pour obtenir le grade de

DOCTEUR de SORBONNE UNIVERSITÉ

devant le jury composé de :

M. François Fleuret	Rapporteur
M. Jakob Verbeek	Rapporteur
M. Matthieu Cord	Examinateur
Mme. Elisa Fromont	Examinatrice
M. Thierry Artières	Encadrant
M. Ludovic Denoyer	Directeur de thèse

ABSTRACT

Many successes of deep learning rely on the availability of massive annotated datasets that can be exploited by supervised algorithms. Obtaining those labels at a large scale, however, can be difficult, or even impossible in many situations. Designing methods that are less dependent on annotations is therefore a major research topic, and many semi-supervised and weakly supervised methods have been proposed. Meanwhile, the recent introduction of deep generative networks provided deep learning methods with the ability to manipulate complex distributions, allowing for breakthroughs in tasks such as image edition and domain adaptation.

In this thesis, we explore how these new tools can be useful to further alleviate the need for annotations. Firstly, we tackle the task of performing stochastic predictions. It consists in designing systems for structured prediction that take into account the variability in possible outputs. We propose, in this context, two models. The first one performs predictions on multi-view data with missing views, and the second one predicts possible futures of a video sequence. Then, we study adversarial methods to learn a factorized latent space, in a setting with two explanatory factors but only one of them is annotated. We propose models that aim to uncover semantically consistent latent representations for those factors. One model is applied to the conditional generation of motion capture data, and another one to multi-view data. Finally, we focus on the task of image segmentation, which is of crucial importance in computer vision. Building on previously explored ideas, we propose a model for object segmentation that is entirely unsupervised.

RÉSUMÉ

Nombre des succès de l'apprentissage profond reposent sur la disponibilité de données massivement collectées et annotées, exploités par des algorithmes supervisés. Ces annotations, cependant, peuvent s'avérer difficiles à obtenir. La conception de méthodes peu gourmandes en annotations est ainsi un enjeu important, abordé dans des approches semi-supervisées ou faiblement supervisées. Par ailleurs ont été récemment introduit les réseaux génératifs profonds, capable de manipuler des distributions complexes et à l'origine d'avancées majeures, en édition d'image et en adaptation de domaine par exemple.

Dans cette thèse, nous explorons comment ces outils nouveaux peuvent être exploités pour réduire les besoins en annotations. En premier lieu, nous abordons la tâche de prédiction stochastique. Il s'agit de concevoir des systèmes de prédiction structurée tenant compte de la diversité des réponses possibles. Nous proposons dans ce cadre deux modèles, le premier pour des données multi-vues avec vues manquantes, et le second pour la prédiction de futurs possibles d'une séquence vidéo. Ensuite, nous étudions la décomposition en deux facteurs latents indépendants dans le cas où un seul facteur est annoté. Nous proposons des modèles qui visent à retrouver des représentations latentes sémantiquement cohérentes de ces facteurs explicatifs. Le premier modèle est appliqué en génération de données de capture de mouvements, le second, sur des données multi-vues. Enfin, nous nous attaquons au problème, crucial en vision par ordinateur, de la segmentation d'image. Nous proposons un modèle, inspiré des idées développées dans cette thèse, de segmentation d'objet entièrement non supervisé.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

MLP	Multi-Layer Perceptron
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long-Short Term Memory
VAE	Variational Auto-Encoder
GAN	Generative Adversarial Network
KLD	Kullback-Leibler Divergence
JSD	Jensen-Shannon Divergence
ELBO	Evidence Lower-Bound
ReLU	Rectified Linear Unit
ICA	Independant Component Analysis
MV-BiGAN	Multi-View Bidirectional GAN
SRVP	Stochastic Residual Video Prediction model
GMV	Generative Multi-View model
CGMV	Controlled Generative Multi-View model
ReDO	Redrawing Of Objects

CHAPTER 1

INTRODUCTION

Contents

1.1 Context

Over the last decade, Artificial Intelligence has experienced a major resurgence of interest both in academia and industry and even among the general public. Indeed, AI-powered systems have become an increasingly crucial part of the technology industry. Automated systems for recommendation and online content moderation are now prevalent on the internet, and companies have been investing heavily in the research and the deployment of novel AI products. At the heart of this renewed surge of interest, the field of Machine Learning has contributed to a wide array of new technologies that have the potential to profoundly affect the entire society. Deep Learning techniques, especially, have allowed breakthroughs in many AI tasks such as image recognition, machine translation or text, and image synthesis, and have impacts in domains of application as varied as dialog systems, facial recognition, sales and stock management, medical diagnosis, pharmaceutical research, and autonomous driving.

The different successes of Deep Learning, though, are mostly supported by a global increase in the availability of both computing power and data. Indeed, since the introduction of AlexNet (Krizhevsky et al. 2017) winning ILSVRC2012 (ImageNet Large Scale Visual Recognition Challenge 2012, Russakovsky et al. 2015), advances in Computer Vision have been mainly driven by the discovery of new ways to build deeper models, with the development or resurgence of techniques like residual connexions (He et al. 2016), batch-normalization (Ioffe et al. 2015), Rectified Linear Unit (ReLU) (Nair et al. 2010), weights initialization schemes (Glorot et al. 2010; He et al. 2015; Saxe et al. 2014), and adaptative gradient

descent methods (Kingma et al. 2015), and the release of large-scale annotated datasets such as LSUN (Large Scale Scene Understanding, Yu et al. 2015) or MS-COCO (Microsoft Common Objects in Context, Lin et al. 2014). Not limited to this one field, this trend has also been observed in other major applications of Deep Learning, such as Natural Language Processing and Reinforcement Learning for Games.

These considerations uncover a major limitation of Deep Learning. Indeed, the training Deep Neural Networks relies on the availability of large-scale datasets, often requiring annotations produced by humans. Creating those annotated datasets can be a costly and time-consuming endeavor, if possible at all. For instance, in image segmentation, natural images might be relatively easy to obtain, but the segmentation masks associated with the image need careful human annotations. Despite the good performances of Deep Neural Networks, this reliance on annotations hinder their relevance in cases where labels are unavailable or sensitive. Addressing this limitation is therefore of crucial importance, and entire subfields such as transfer learning, semi-supervised learning, and few-shot learning have been dedicated to that issue. In computer vision, the ability to partially re-use pre-trained Convolutional Neural Networks (CNNs) for learning models with fewer data has been instrumental in the adoption of deep networks by the industry. And very recently, self-supervised approaches have even achieved state-of-the-art performances compared to pre-trained classifiers (Bachman et al. 2019; He et al. 2019; Misra et al. 2019; Hénaff et al. 2019). Similarly in Natural Language Processing, contextual word embeddings trained on a large unlabelled text corpus are today at the heart of many advances in the field (Devlin et al. 2019). Another idea consists of taking advantage of easier-to-obtain information instead of direct supervision. For image segmentation, this can takes the form of weakly-supervised approaches that works solely on bounding boxes or image-level labeling instead of pixel-level annotations (Durand et al. 2017; Zhou et al. 2018).

One notable recent trend in deep unsupervised learning consists of uncovering a hidden structure using explicitly constrained neural networks. For instance, Ordered Neurons LSTM (Shen et al. 2019) encodes a sequence in a tree-like structure and has been successfully used for semantic tree discovery without supervision. In videos, invariance and difference through time (*i.e.* between frames) have been exploited to separate static content from the dynamic in video prediction (Denton et al. 2017; Sermanet et al. 2018). Capsule Networks (Sabour et al. 2017) aims to separate content and pose at each level of the latent hierarchy by using equivariant embeddings. In all those cases, architectural choices and the design of the objective function forces the training process towards desired solutions, without supervision.

The introduction of Deep Generative Models has also opened promising new perspectives. As a result, in the past few years, a number of studies that use some

form of deep generative modeling in diverse contexts with limited supervision have been published. Those works use Deep Generative Models as tools to design more flexible constraints and loss functions, fitting for lighter forms of supervision. Consider, for instance, the case of domain translation. The task is to learn a mapping between observations $\boldsymbol{x}$ from a source domain $\mathcal{X}$ and $\boldsymbol{y}$ from a target domain $\mathcal{Y}$. A simple supervised method would be to learn a mapping f_θ, where θ is the learnable parameters, so that for any pair $(\boldsymbol{x}, \boldsymbol{y})$ from the dataset, $f(\boldsymbol{x}) = \boldsymbol{y}$. This supervised formulation requires paired observations $(\boldsymbol{x}, \boldsymbol{y})$ to train the translation function. Using Deep Generative Models, it would be possible to instead specify as learning objective that the image of $\mathcal{X}$ by f_θ is $\mathcal{Y}$, removing any dependencies to annotated pairs. The latter objective can be seen as a relaxation of the pairwise supervision, as an optimal f_θ for the supervised problem is also optimal for the unpaired setting. Of course, as it is often the case when supervision is lighter, the new formulation of the problem is ill-posed and needs additional constraints. Different work tackles the issue of different manners, that could be trying to formulate the problem in a way where the solution is unique (Bézenac et al. 2019), by biasing explicitly the objective towards the desired solution (Gong et al. 2019b), or relying solely on implicit inductive biases of deep learning (Zhu et al. 2017).

1.2 Thesis Topics and Contributions

During this thesis, we aimed to extract knowledge from data when full supervision is not available. We explored how deep generative modeling has been used in configurations where annotations are lacking and proposed novel methods of our own. As other works presumed structural properties and designed neural architectures around them to uncover them, we tackled those problems by making assumptions about the generative processes behind the data and building systems in which an auxiliary generative process is able to provide training signals for an inductive network that lacked supervision.

We focused on three distinct setups that differ in the level and the nature of the supervision. Those tasks are performing stochastic prediction that consist of modeling multiple possible outcomes instead of a single prediction, the learning and use of factorized representations under different level of supervision, and the task of image segmentation without supervision.

In the following, we present those tasks and, for each topic, we succinctly introduce our contributions.

1.2.1 Performing Stochastic Predictions

While supervised learning is usually about predicting the most likely output for any given input, in many cases, multiple correct predictions are possible. For example, an image can be correctly inpainted in many different ways, and, in video prediction, the future is often inherently uncertain. In low dimensional settings, this variability can be captured using simple probabilistic models. An image classifier, for instance, can output the parameters of a categorical distribution over the different possible labels for a given input. But as the number of predicted variables grows, the dependencies between them can quickly become too complex to capture in a satisfactory fashion. For instance, systems predicting images are usually trained using a mean square error loss. The output of a function trained in that fashion can be interpreted as the mean of the possible distribution $p(\mathbf{y}|\mathbf{x})$ of outputs given the input $\boldsymbol{x}$, assuming p is using a Gaussian observation model and all pixels supposed independent to each other. In any case, methods of this type only provide partial information and don't account for the uncertainty and variability inherent to many tasks. This can be a problem when possible outputs are very diverse and motivates the need for systems that are better able to capture that variability. The difficulty however in training such models is twofold. First, the target distribution $p(y|x)$ is complex and difficult to describe, and, secondly, we need to predict all possibilities but can usually only observe one realization y for any given input condition x. While these difficulties have been tackled using Monte Carlo Markov Chains based techniques (Bengio et al. 2013b), stochastic predictions in high dimension have only recently emerged, after the introduction of deep generative methods.

We explore how these methods can be used to account for uncertainty, and how they can be used to perform stochastic predictions in Chapter 3. We present in that chapter our two contributions to this topic:

- Mickaël Chen and Ludovic Denoyer (2017). "Multi-view Generative Adversarial Networks". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part II*, pp. 175–188

- Jean-Yves Franceschi, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari (2020). "Stochastic Latent Residual Video Prediction". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, forthcoming*

Chen et al. 2017 introduces Multi-View Bidirectional GAN (MV-BiGAN), an early attempt at using Deep Generative Models for multi-view data that focuses on learning and constraining stochastic embeddings to perform predictions. Franceschi et al. 2020 presents Stochastic Residual Video Prediction model (SRVP), that lever-

ages recent techniques in deep generative modeling as well as advances in the modeling of dynamics to propose a state-of-the-art model for stochastic video prediction. SRVP, like other works (Denton et al. 2017), separates dynamics and static content as a crucial step. This leads us to our next topic, which is learning and exploiting factorized representations.

1.2.2 Adversarial Learning of Factorized Representations

The different successes of Deep Learning can be tied to the power of Deep Neural Networks to extract good representations automatically from the data. Some properties, such as smoothness and sparsity, have usually been considered of interest and have been enforced via regularization methods, for instance (Bengio et al. 2013a). In this regard, the ability of Deep Generative Models to impose priors on the distribution of latent representations allowed to constrain representation in ways that were not feasible before. For instance, it was discovered that disentangled representations, i.e. representation in which each dimension encodes for one and only one distinct factor of variation that explains the observations, emerges from the practical choice of isotropic Gaussian priors in Variational Auto-Encoders (VAEs) (Higgins et al. 2017). Other approaches, using adversarial training to constrain a representation to be independent to a sensitive variable, can effectively partition the information into different, complementary, latent representations, with applications for controlled data generation, domain adaptation, and fairness (Edwards et al. 2016; Ganin et al. 2015; Lample et al. 2017). More generally, the idea of splitting information into separable explanatory factors is enticing as it would allow building more interpretable models and might be a first step towards emulating causal modeling and symbolic reasoning while still retaining the expressiveness and end-to-end approach that made the success of neural approaches.

We discuss those methods and applications in more detail in Chapter 4. The work we present in that chapter has led to two conference publication and a journal extension:

- Qi Wang, Mickaël Chen, Thierry Artières, and Ludovic Denoyer (2018b). "Transferring style in motion capture sequences with adversarial learning". In: *26th European Symposium on Artificial Neural Networks, ESANN 2018, Bruges, Belgium, April 25-27, 2018*

- Qi Wang, Thierry Artières, Mickaël Chen, and Ludovic Denoyer (2018a). "Adversarial learning for modeling human motion". In: *The Visual Computer*, pp. 1–20

- Mickaël Chen, Ludovic Denoyer, and Thierry Artières (2018b). "Multi-View Data Generation Without View Supervision". In: *6th International Conference*

on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings

Wang et al. 2018b and Wang et al. 2018a discuss the use adversarial methods to separate style and action in Motion Capture modeling. Chen et al. 2018b then presents Controlled Generative Multi-View model (CGMV), a controllable generation method for multi-view data that is able to discover a consistent style encoding without using style labels. To design CGMV, we built a constrained generative model embedded with an assumption of independence between latent variables and combine it with a feature extractor. We train both modules simultaneously using adversarial learning. In the subsequent topic, we show how a similar method can be applied to a different topic by tackling the challenging task of unsupervised object segmentation.

1.2.3 Unsupervised Image Segmentation

Image segmentation is both a classic and prominent task in computer vision that consists in partitioning an image into meaningful regions. Over decades of research, a large number of different approaches have been proposed, mostly using prior knowledge about the regions one wishes to discover. A popular method is to build an energy function using handcrafted metrics, which is then minimized for each given image using a graph-cutting algorithm (Pham et al. 2018; Silberman et al. 2012). The introduction of large-scale annotated datasets allowed, instead, for a more data-driven approach. By casting the problem as a supervised problem in which each pixel needs to be classified in one of the possible regions, the adoption of deep segmentation methods resulted in leaps of performances on different benchmarks (Chen et al. 2018a; He et al. 2017; Zhao et al. 2018). But those large-scale datasets are also difficult to obtain as it requires pixel-level precision from annotators. Researchers have come up with a number of strategies to reduce the reliance on such supervision, mostly by combining classic image priors with deep learning techniques (Durand et al. 2017; Zhou et al. 2018).

In Chapter 5, we discuss those methods and present a novel approach for image segmentation without ground-truth annotations, that led to the following publication:

- Mickaël Chen, Thierry Artières, and Ludovic Denoyer (2019a). "Unsupervised Object Segmentation by Redrawing". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 12705–12716

As with CGMV, this model exploits an assumption of independence, but between regions instead of style and content factors, to build a constrained deep generative

model and combine it with a segmentation function. The system is constrained so that the generative model provides a learning signal for a segmentation function in the absence of labels.

1.2.4 Outline

This manuscript is organized as follows:

Firstly, Chapter 2 discusses critical notions for our work and provide a succinct overview of different usages of Deep Generative Models in contexts without full supervision.

It is followed by Chapters 3-5 that present our contributions to the three topics described here-above (performing stochastic predictions, learning factorized representations, and image segmentation).

Finally, Chapter 6 closes this manuscript with perspectives.

CHAPTER 2

DEEP GENERATIVE MODELS AND SUPERVISION

Contents

In this chapter, we introduce Deep Generative Models and, through examples of application, show how they can be useful in context without full supervision. In Section 2.1, we succinctly present the main families of Deep Generative Models and discuss their similarities and differences in usage. Section 2.2 aims to provide the reader with a landscape of weakly supervised problems and deep generative based solutions that have been proposed to tackle them. We especially discuss the topics of disentanglement, domain translation, and signal reconstruction. Finally, Section 2.3 explains how our work fits in this broader context.

2.1 Deep Generative Models

Modern Deep Generative Models, as introduced and popularized by Kingma et al. 2014, Rezende et al. 2014, and Goodfellow et al. 2014, are models that provide a way to generate new data points. Given a dataset $\mathcal{D}_{\mathcal{X}}$ of observations $\boldsymbol{x}$, and assuming that there is an underlying distribution $p(\boldsymbol{x})$ from which the observations have been sampled, these models try to generate new realizations of that distribution.

This problem has been tackled in multiple ways before, for instance by fitting a Gaussian mixture model or a graphical model, but it was difficult to scale up to high-dimensional data as modelizing more complex relations usually required

more accurate approximations of intractable marginal distributions. A first attempt to incorporate deep learning into generative modeling has been proposed by Bengio et al. 2013b and Bengio et al. 2014. Their Generalized Denoising Auto-Encoders parameterizes a Markov chain with neural networks that could be trained by computing exact gradients via back-propagation. However, the sequential nature of Monte-Carlo Markov Chains based methods makes them unpractical to train and use.

Subsequent approaches tried to simplify the process by adopting a common strategy that is, given an arbitrarily chosen distribution $p(\boldsymbol{z})$, to learn a function $G_\theta : \boldsymbol{z} \mapsto \boldsymbol{x}$ that maps $p(\boldsymbol{z})$ to $p(\boldsymbol{x})$, so that $p(G_\theta(\boldsymbol{z}))$ is an approximation of $p(\boldsymbol{x})$. It is then easy to generate new realizations of $p(\boldsymbol{x})$ by first sampling from $p(\boldsymbol{z})$, and then computing $G_\theta(\boldsymbol{z})$. A standard choice for $p(\boldsymbol{z})$ is a multivariate diagonal Gaussian $\mathcal{N}(0, I_d)$.

The difficult part is, of course, to train the generator function G_θ, especially since $p(\boldsymbol{x})$ is implicit and can only be estimated via samples from the dataset. Different families of models adopt different strategies that each have advantages and drawbacks. In this section, we present two of the main families of deep generative models: VAE and Generative Adversarial Network (GAN). For each, we discuss their strong points, limitations, and some preferred use-cases.

We also present conditional extensions of those models. Those variants aim at providing some control over the generated samples, the typical case being class-conditionnement in which the model is tasked to generate samples of a given class.

We focus especially on the image generation task, as it is the most documented and successful application of deep generative models, but they have also been used for other tasks, such as sequence modeling and natural language generation.

Also note that a third family of models, based on Normalizing Flows that have been popularized by Rezende et al. 2015 and Dinh et al. 2015, will not be discussed in this chapter. Very succinctly, they tackle the problem by forcing the learned transformation to be invertible so that learning an encoder that transforms an observation into a latent code is sufficient to recover the generator. This encoder can easily be trained via maximum likelihood as it is applied to the latent space instead of the observation space. In addition to being able to generate samples of good quality, this formulation comes with new capabilities that other models struggles with, such as the exact inference of the latent variable, and accurate likelihood estimation of an observation. Still, some challenges have yet to be overcome. In particular, two factors limit potential usages. Firstly, computation needed to enforce the invertibility constraint creates numerical instability during training. Secondly, current methods don't allow for a change of dimensionality between the space of latent distribution and the space of observations, reducing the impact of the exact inference capability. These two constraints limit the ways

those models they can applied, making them perhaps less revelant in the context of this thesis.

2.1.1 Variational Auto-Encoders

Introduced by Kingma et al. 2014 and Rezende et al. 2014, VAEs draw roots from the variational inference framework. We present here this framework and then discuss some strong points and limitations.

Formulation

Given a dataset of observations $\boldsymbol{x} \in \mathcal{D}_{\mathcal{X}}$, a prior distribution $p_\theta(\boldsymbol{z})$ on unobserved latent variables, and a parameterized model $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, variational inference provides an approximate q_ϕ of the often intractable true posterior probability $p_\theta(\boldsymbol{z}|\boldsymbol{x})$, as well as an Evidence Lower-Bound (ELBO) *i.e.* a lower-bound on $\log p_\theta(\boldsymbol{x})$ the marginal log-likelihood of the observed data. By maximizing the ELBO, variational inference tries to find a model $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ that explains the data the best.

In VAEs, the marginal log-likelihood $\log p_\theta(\boldsymbol{x})$ is decomposed as follows:

$$\begin{aligned} \log p_\theta(\boldsymbol{x}) &= D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})\right) + \mathcal{L}_{VAE} \\ \text{where } \mathcal{L}_{VAE} &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] - D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})\right). \end{aligned} \tag{2.1}$$

As the Kullback-Leibler Divergence (KLD) term is non-negative, $\mathcal{L}_{VAE}$ is indeed a lower-bound on $\log p_\theta(\boldsymbol{x})$. This ELBO is the VAE objective and can be maximized jointly w.r.t. variational parameters ϕ and generative model parameters θ using gradient descent techniques. To compute the ELBO and the gradients though, it is necessary to evaluate the log-likelihood $\log p_\theta(\boldsymbol{x}|\boldsymbol{z})$, the KLD term $D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})\right)$, and a stochastic gradient estimate for $q_\phi(\boldsymbol{z}|\boldsymbol{x})$.

Evaluating the log-likelihood requires, in turn, to choose an explicit family of parametrized distribution for $p_\theta(\boldsymbol{x}|\boldsymbol{z})$. For practical purposes, this distribution can be a set of independent Bernouilli or a multi-variate Gaussian with fixed variance, for instance. To compute the KLD term, VAEs model $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ as multivariate Gaussians, taking advantage of the closed-form formulation of the KLD between Gaussian distributions. If $p = \mathcal{N}(\mu_1, \sigma_1 \cdot I_d)$ and $q = \mathcal{N}(\mu_2, \sigma_2 \cdot I_d)$, then:

$$D_{\mathrm{KL}}\left(p||q\right) = \frac{1}{2}\sum_{i=1}^{d_z}\left(-1 - \log\left(\frac{\sigma_{1(i)}^2}{\sigma_{2(i)}^2}\right) + \frac{\sigma_{1(i)}^2}{\sigma_{2(i)}^2} + \frac{(\mu_{1(i)} - \mu_{2(i)})^2}{\sigma_{2(i)}^2}\right). \tag{2.2}$$

As for the stochastic gradient estimation for $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, instead of using the Monte Carlo estimator that has high variance, VAEs introduce a reparametrization trick.

The method relies on the fact that it is often possible to express a random variable $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})$ as a deterministic function of an auxiliary independent random variable $\epsilon \sim p(\epsilon)$. With $q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\mu_\phi, \sigma^2_\phi)$, it is easy to use the following reparametrization of $\boldsymbol{z}$:

$$\boldsymbol{z} = \mu_\phi + \sigma_\phi \cdot \epsilon, \text{ with } p(\epsilon) = \mathcal{N}(0, I_d). \tag{2.3}$$

The stochastic variable becomes ϵ, which is independent of distribution parameters μ_ϕ and σ_ϕ, making it possible to back-propagate the gradient from $\boldsymbol{z}$ to the trainable parameters ϕ.

Preferred Usages of VAEs

From a machine learning perspective, the ELBO can be seen as the combination of a reconstruction objective embodied in the log-likelihood term and a regularization over the latent space in the form of the KLD term. As in more traditional auto-encoders, the regularization creates a bottleneck and enforce some property in the latent code. In the case of a VAE with a diagonal Gaussian as a prior for the latent code, it has been observed that the KLD term promotes both sparsity and disentanglement of the latent code. These properties are useful in representation learning, making VAE a prime method for general feature extraction.

However, VAEs have to find an equilibrium between the two terms of the objective, *i.e.* between a good observation model $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ and a simple latent code $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. This leads VAEs to produce less realistic samples than the other Deep Generative Models. In image generation, for instance, VAEs are known for their blurry outputs and are therefore scarcely used when the restitution of the high-frequency features is a predominant criterion.

VAEs, therefore, are especially useful when the focus is on the interpretability of the latent space and less on the quality of the generation. They are commonly found, for instance, in video prediction and reinforcement learning, domains in which a good latent space is crucial. It is possible to bias VAEs into prioritizing a good reconstruction, as in Razavi et al. 2019 that uses a powerful observation model and a different prior on the latent space, but these methods tend to be very compute-intensive compared to others and results in latent spaces that are more difficult to exploit.

Conditional Variational Auto-Encoder

A Conditional Variational Auto-encoder, introduced by Sohn et al. 2015, extends the VAE for controlled generation. The goal here is to model the conditional probability $p(\boldsymbol{x}|\boldsymbol{c})$, where $\boldsymbol{c}$ is the condition. This is achieved by introducing $\boldsymbol{c}$ as an input of the encoder and the decoder, and writing the variational lower-bound on the log-likelihood $log p(\boldsymbol{x}|\boldsymbol{c})$ as follows:

$$\mathcal{L}_{CVAE} = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{c})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{c})\right] - D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{c})||p_\theta(\boldsymbol{z}|\boldsymbol{c})\right). \tag{2.4}$$

Notice that the prior on the latent variable $\boldsymbol{z}$ is now also conditioned by $\boldsymbol{c}$. Sohn et al. 2015 relaxes this requirement by modeling both variables as independent of one another ($p_\theta(\boldsymbol{z}|\boldsymbol{c}) = p(\boldsymbol{z})$). While this relaxation allow them to obtain convincing results in their class-conditionned experiments, it can be limiting in more complex cases. For instance, when the condition is an image instead of a category, the model often fails to learn a latent variable $\boldsymbol{z}$ that is independent of the input $\boldsymbol{c}$. Then, the decoder won't be able to generate properly at test time as it cannot process the independently sampled $\boldsymbol{z}$ and $\boldsymbol{c}$ at test time.

Instead, learning the conditional prior can be done by instantiating it as an additional neural network, as in Denton et al. 2018 in the context of video prediction. This solution, while effective, comes with its own limitations. Indeed, learning the prior reduces the regularizing effect imposed on the latent space by the KLD term and can result in a less interpretable latent variable.

2.1.2 Generative Adversarial Networks

GANs, that have been introduced by Goodfellow et al. 2014, leverage adversarial training for modeling complex distributions. We present the method hereafter, along with some popular variants before discussing practical uses.

Formulation

GANs learn a one-to-one mapping G, called the generator, between latent vectors $\boldsymbol{z}$ sampled from a noise distribution $p(\boldsymbol{z})$ (often chosen as $\mathcal{N}(0, I_d)$) and samples from the distribution of observations $p(\boldsymbol{x})$. To do so, the authors introduce a second learned function, the discriminator D, which is a classifier whose goal is to distinguish between real observations drawn from the dataset and generated samples $G(\boldsymbol{z}), \boldsymbol{z} \sim p(\boldsymbol{z})$. On the other hand, G is trained adversarially, with the opposite objective, so that D cannot distinguish between the generated and the real observations. The original formulation of the minimax adversarial objective is as follows:

$$\min_G \max_D \; \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})} \left[\log D(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})} \left[\log(1 - D(G(\boldsymbol{z})))\right]. \tag{2.5}$$

Under optimality conditions for D, this objective minimizes the Jensen-Shannon Divergence (JSD) between the two distributions. The GAN objective can, therefore, be seen as a distribution matching objective that ensures that the two types of inputs the discriminator can receive, either generated or real observations, are drawn from similar distributions.

Alternative Objectives

In the original publication, the authors point out that minimax adversarial formulation is unstable and often fails to learn. Over the years, many alternative objectives have been proposed. We present here some of the most prominent ones.

Non-Saturating Loss. A main cause of failure is that, as the objective is much easier for D, the discriminator can saturate, thus interrupting the gradient flow to the generator. To alleviate the problem, in addition to the minimax objective, the authors of Goodfellow et al. 2014 also introduced the following Non-Saturating GAN objective:

$$\begin{aligned} \min_D \quad & -\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\log D(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log(1 - D(G(\boldsymbol{z})))\right] \\ \min_G \quad & -\mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log D(G(\boldsymbol{z}))\right]. \end{aligned} \tag{2.6}$$

While in the minimax formulation, the generator's objective was the reverse of the discriminator's, *i.e.* minimizing the likelihood that $G(\boldsymbol{z})$ is classified as fake, the Non-Saturating formulation changes the generator's objective so that it maximizes the likelihood that $G(\boldsymbol{z})$ is classified as real instead. The goal of this change is to provide a stronger gradient to G.

f-GAN. f-divergences are a well-known family of functions, including the KLD and the JSD, that measures the difference between two probability distributions. If P and Q are two probability distributions, with respective absolutely continuous density functions p and q w.r.t. a base measure $d\boldsymbol{x}$ defined on domain $\mathcal{X}$, an f-divergence can be written $D_f(P||Q) =$ where f is a convex function such that $f(1) = 0$. f-divergences can, therefore, be seen as a weighted average of odds ratio and different choices of f recover different classic f-divergences. Nowozin et al. 2016 describes how to minimize a variety of f-divergences using f-GANs, which are variants of GANs with slight alterations of the objective functions. While these variations are scarcely used in practice are more stable alternatives exist, the work is interesting as it builds on the idea of interpreting a discriminator as an estimator of density ratio.

Wasserstein GAN. Arjovsky et al. 2017 makes the argument that f-divergences might be ill-suited for learning, as they diverge when the supports of the two compared distributions are different. While the argument is compelling, it should be noted that in practice, those f-divergences are learned via the discriminator using smooth approximations, which can alleviate the problem (Gretton et al. 2019). Still, the authors propose Wasserstein-GAN, that instead of minimizing an f-divergence, optimizes the Earth-Mover (or Wasserstein-1) distance, which

doesn't suffer the same issues. Using the Kantorovich-Rubinstein duality, they provide the following adversarial formulation:

$$\begin{aligned} &\max_{\substack{D \\ Lip(D)\leq 1}} \quad \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[D(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[D(G(\boldsymbol{z}))\right] \\ &\min_{G} \quad -\mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log D(G(\boldsymbol{z}))\right]. \end{aligned} \tag{2.7}$$

Here, the interpretation is that D is a critic that is used to estimate the Earth-Mover distance while G minimizes this estimate. The difficulty in this formulation is that they have to impose a Lipschitz constraint on D. The solution proposed by the authors was to clamp the weights so that they live in a compact space, but the method was found to be very unstable. It was improved upon by Gulrajani et al. 2017 who penalizes the norm of the gradients w.r.t. the inputs instead. The downside of using Wasserstein-GAN with Gradient Penalty (WGAN-GP), is that it requires second-order derivation, which is compute-intensive. While some state-of-the-art methods (Karras et al. 2019) make use of WGAN-GP, Lucic et al. 2018 reports on a large-scale study that the additional costs might not be worth the marginal improvement over some of the concurrent objectives.

Hinge Adversarial Loss. Another very commonly used objective for GANs is a hinge version of the adversarial loss (Lim et al. 2017; Tran et al. 2017), that writes as follows:

$$\begin{aligned} &\min_{D} \; -\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\min(0, -1 + D(\boldsymbol{x}))\right] - \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\min(0, -1 - D(G(\boldsymbol{z}))\right] \\ &\min_{G} \; -\mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[D(G(\boldsymbol{z}))\right]. \end{aligned} \tag{2.8}$$

As far as we know, no large-scale experiments have studied the properties of the hinge adversarial loss yet, but the formulation has been used multiple times (Miyato et al. 2018a; Zhang et al. 2019; Brock et al. 2019) to push the state-of-the-art in image generation, along with the WGAN-GP and the Non-Saturating objectives.

Conditional GANs

Generation conditioned on a variable $\boldsymbol{c}$ can be done using GANs by modeling the joint distribution $p(\boldsymbol{x}, \boldsymbol{c})$ as $p(\boldsymbol{x}|\boldsymbol{c})p(\boldsymbol{c})$, as proposed by Mirza et al. 2014. The conditional probability $p(\boldsymbol{x}|\boldsymbol{c})$ is implicitly captured by a conditional generator $G : (\boldsymbol{z}, \boldsymbol{c}) \mapsto \boldsymbol{x}$, with $\boldsymbol{z}$ being the stochastic noise input. The discriminator operates on the joint distributions induced by positive pairs $(\boldsymbol{x}, \boldsymbol{c})$ from the dataset and negative generated pairs $(G(\boldsymbol{z}, \boldsymbol{c}), \boldsymbol{c})$. This yield the following minimax formulation:

$$\min_{G}\max_{D} \; \mathbb{E}_{\boldsymbol{x},\boldsymbol{c}\sim p(\boldsymbol{x},\boldsymbol{c})}\left[\log D(\boldsymbol{x}, \boldsymbol{c})\right] + \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z}),\boldsymbol{c}\sim p(\boldsymbol{c})}\left[\log(1 - D(G(\boldsymbol{z}, \boldsymbol{c}), \boldsymbol{c}))\right]. \tag{2.9}$$

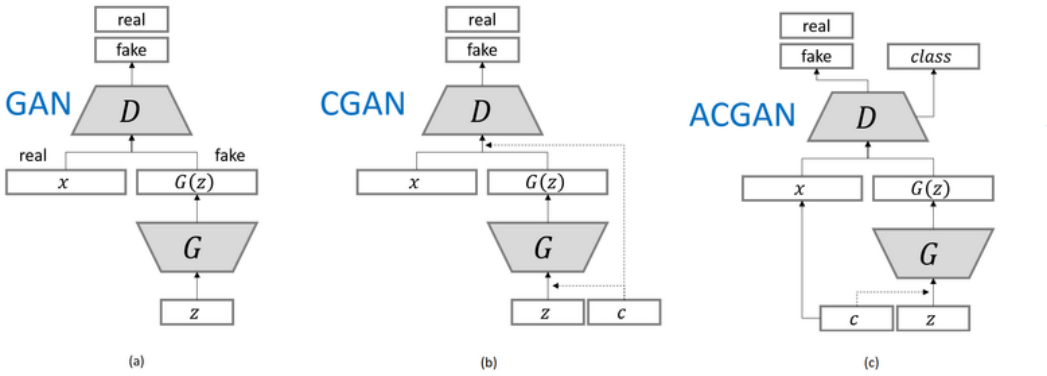


Figure 2.1 – (a) GAN (Goodfellow et al. 2014) and its conditional extensions (b) Conditional GAN (Mirza et al. 2014) and (c) Auxiliary Classifier GAN (Odena et al. 2017). Credits to Mino et al. 2018.

However, this model is very unstable and difficult to train.

Odena et al. 2017 obtains significant improvements by combining the conditional generation objective with an auxiliary classification objective, as illustrated in Figure 2.1. In image-to-image translation, i.e. $\boldsymbol{x}$ and $\boldsymbol{c}$ are both images, Isola et al. 2017 follows a similar approach and compensates for the instability by adding a ℓ_1 reconstruction loss. In both cases, it has been observed that the additional loss bias the model towards easy to classify samples in the case of Odena et al. 2017 or towards an average of the possible outputs for Isola et al. 2017, and greatly reduces the diversity of outputs.

To counteract the problem, Miyato et al. 2018b removes the auxiliary classifier and integrates it implicitly into the conditional generation objective via architectural constraints while Gong et al. 2019a adds a second additional classifier that is used to minimize the mutual information between $\boldsymbol{x}$ and $\boldsymbol{c}$. Both methods, however, are difficult to transpose to the case where the condition $\boldsymbol{c}$ is not a class label.

Preferred Usages of GANs

GANs minimizes a divergence between the distribution of the generated points and the distribution of the dataset. Their originality compared to other generative models is that they are likelihood free. While VAEs, for instance, have to compute the likelihood of the observation according to the learned model, which imposes an explicit form for that distribution, GANs can work with implicit distributions instead and only requires to be able to sample uniformly from $p(\boldsymbol{z})$ and $p(\boldsymbol{x})$. This removes the need to model any distribution explicitly, making GANS a flexible tool that can be incorporate in many deep pipelines. Originally developed for sampling new random data points, the concept has been applied in a variety of ways, such as constraining the latent space to follow a chosen distribution,

embedding inputs from different observation spaces into a shared latent space, aligning different domains, or learning factorized distributions. Unfortunately, they are also known to be unstable during training and prone to mode collapse and mode dropping. Also, the back-inference, *i.e.* retrieving the latent code z from a sample x, is not possible in the standard framework and is still an open problem. The two problems indicate that GAN generators tend to underfit and to ignore data points that don't belong to major modes.

Bidirectional Generative Adversarial Nets

One elegant solution for learning the reverse mapping from x to z in GANs was, independently, proposed by Donahue et al. 2017 and Dumoulin et al. 2017a. Building on the GAN framework, BiGAN introduces an encoder network E that is trained jointly with the generator G and the discriminator D. G models, as in standard GANs, the generative process $p_G(x|z)$, while E models the reverse $p_E(z|x)$. From both the encoder distribution and the generator distribution, we can model two joint distributions, respectively denoted $p_E(x,z)$ and $p_G(x,z)$:

$$\begin{aligned} p_G(x,z) &= p(z)p_G(x|z) \\ p_E(x,z) &= p(x)p_E(z|x). \end{aligned} \tag{2.10}$$

The discriminator network is then trained to determine whether a pair (x,z) is sampled from $p_G(x,z)$ or from $p_E(x,z)$, while E and G are both trained to fool D, resulting in the following learning problem:

$$\begin{aligned} \min_{G,E} \max_{D} \mathbb{E}_{x,z\sim p_E(x,z)} \left[\log D(x,z)\right] \\ + \mathbb{E}_{x,z\sim p_G(x,z)} \left[1 - \log D(x,z)\right]. \end{aligned} \tag{2.11}$$

Under optimality conditions for the discriminator, the objective recovers a generator function and an inverse encoder function that acts like an auto-encoder. However, the model suffers from the instability of adversarial training, especially since the model is never explicitly trained on auto-encoding. Originally, BiGANs were very limited in the features they were able to capture and reconstruct. The idea has been revisited by Donahue et al. 2019, by incorporating uses recent GAN training techniques. The latest version presents a learned generator that is much more expressive and an encoder that is competitive with state-of-the-art methods for feature extraction.

2.2 Constraining Distributions and Applications

After their first introduction, Deep Generative Models have been quickly adapted to a variety of different tasks and setup. A prominent trend was to

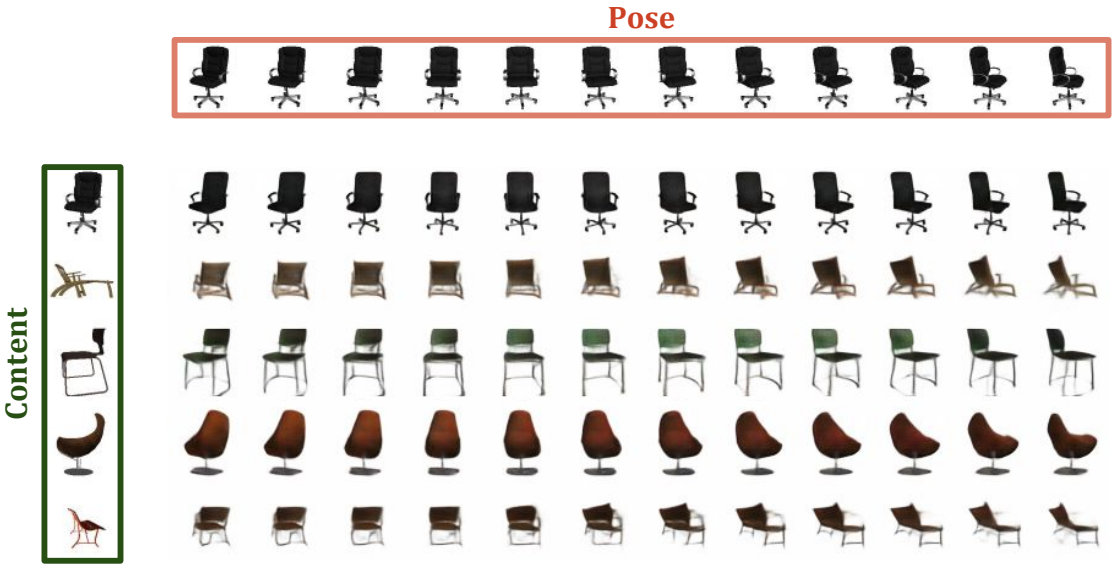


Figure 2.2 – Example of independent directions of variations we might want to capture in disentangled and factorized representations. Credits to Denton et al. 2017.

use those models in contexts where direct supervision is not available, achieving remarkable results that weren't possible before. Some of those results, and the mechanisms behind them, have yet to perfectly understood even today. Our own work in this thesis, that could be summarized as designing constrained generative processes that can be leveraged to extract knowledge from data without full supervision, fits humbly in this larger movement. To provide further context, we present in this section a few influential papers that are relevant in that regard. This section is split into four parts. First, we introduce and discuss unsupervised disentanglement with deep generative models. We then present methods for the related task of learning factorized representations. After that, the third and the fourth subsections deal respectively with adversarial domain adaptation and signal reconstruction.

2.2.1 Unsupervised Disentanglement

Representation learning, or Feature Learning, is the field that aims to extract meaningful representations from data, usually by leveraging the power of deep neural networks. One general type of representation that is of particular interest is disentangled representation. These representations aim to capture meaningful independent directions of variation in the data (Figure 2.2). In that sense, many disentangled representation learning methods can be interpreted as an implementation of non-linear Independant Component Analysis (ICA). Deep learning and

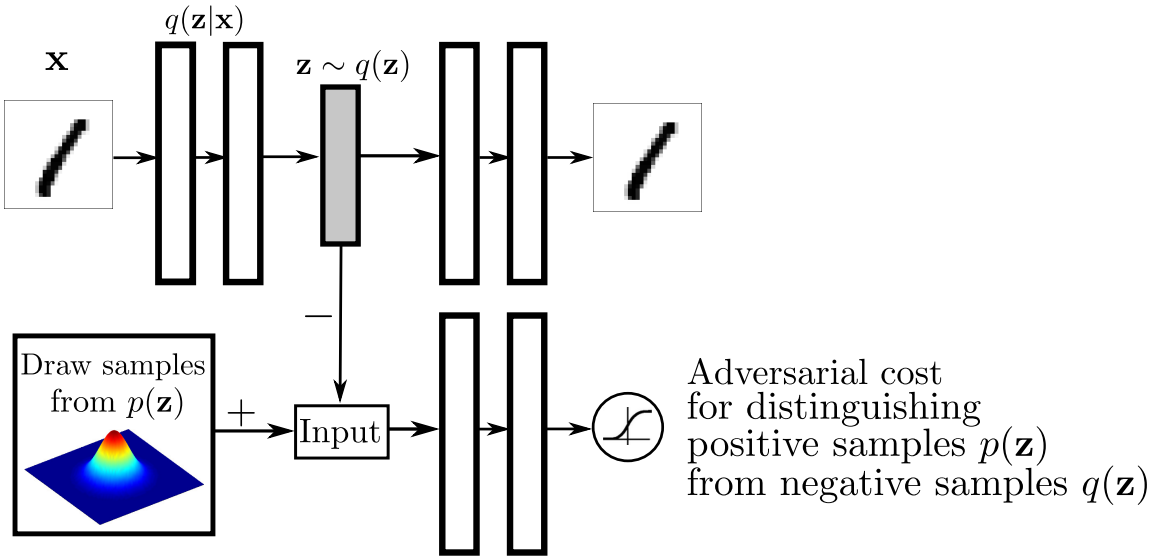


Figure 2.3 – Architecture of an adversarial autoencoder. The standard autoencoder (top part) is augmented with adversarial training using a discriminator (bottom part) that constrain the distribution of the codes. Credits to Makhzani et al. 2015.

in particular, deep generative models have provided many new tools for tackling this problem, and recent studies report that deep disentangled representations are useful for abstract visual reasoning (Steenkiste et al. 2019), or fair representation learning (Locatello et al. 2019a). We present in this subsection some promising approaches that were proposed to learn disentangled representations without supervision before discussing some open issues with the task.

Adversarial Auto-Encoders

One line of work in this direction, starting with Adversarial Auto-Encoders (Makhzani et al. 2015), tries to simply impose a known, structured, distribution on the latent space of a standard auto-encoder (Figure 2.3). Using the Non-Saturating loss on the latent code in this way results in the following objective, illustrated in Figure 2.3:

$$\begin{aligned} &\max_{D} \ \lambda \cdot \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})} \left[\log D(\boldsymbol{z})\right] + \lambda \cdot \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x})} \left[\log(1 - D(\text{enc}(\boldsymbol{x})))\right] \\ &\min_{\text{enc,dec}} \ \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x})} \left[||\boldsymbol{x} - \text{dec}(\text{enc}(\boldsymbol{x}))|| - \lambda \log D(\text{enc}(\boldsymbol{x}))\right] \end{aligned} \tag{2.12}$$

Such models could be seen as a form of unsupervised clustering. For instance, by imposing that the distribution of codes for images of hand-written digits to be a mixture of ten Gaussians, in addition to the auto-encoding objective, the model could hopefully learn to encode each class into its own cluster. This problem, of course, is ill-posed, as the decoder is also highly non-linear and many different

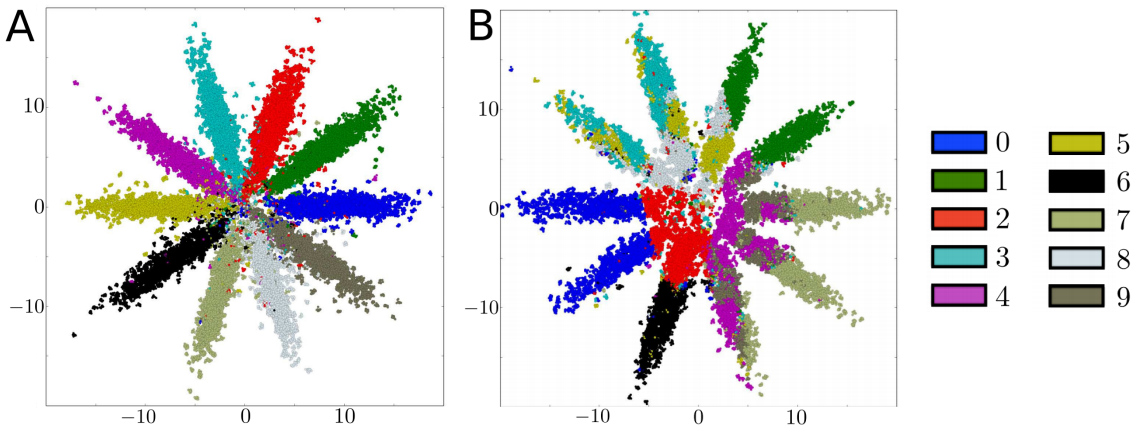


Figure 2.4 – Results obtained using an Adversarial Auto-Encoder to fit MNIST digits into a mixture of 10 2D Gaussians (a) in a semi-supervised setup with 10k labels, and (b) in an unsupervised setup. Colors represent the associated ground-truth labels. Credits to Makhzani et al. 2015.

arrangments could be decoded into digits, with no regard for the class label. Accordingly, while the model is able to fit the provided distribution, it doesn't discover the class labels in this fully unsupervised use-case. It is able to do so only in a semi-supervised setting (Figure 2.4).

Another way to leverage Adversarial Auto-Encoders for disentanglement would be to map the latent code to a multi-variate distribution in which each dimension is independent of the other dimensions. Preliminary studies in that direction have been presented in Rubenstein et al. 2018 using Wasserstein Auto-Encoders (Tolstikhin et al. 2018), an optimal transport based interpretation and generalization of Adversarial Auto-Encoders. However, while the approach seems promising, its usefulness for unsupervised disentanglement has yet to be demonstrated.

InfoGAN

Information Maximizing GAN (InfoGAN), proposed by Chen et al. 2016, suppose the existence of an unobserved code $\boldsymbol{c}$ that explains the observations $\boldsymbol{x}$. To uncover $\boldsymbol{c}$ from the dataset, InfoGAN proposes to learn a generator that transforms random codes $\boldsymbol{c}$ from chosen distributions $p(\boldsymbol{c})$ (and augmented with incompressible noise vectors $\boldsymbol{z}$), into realistic samples from the dataset distribution $p(\boldsymbol{x})$. To make sure that the generated outputs are correlated to the codes, the authors add an encoder that takes as input generated samples and is tasked to reconstruct the code $\boldsymbol{c}$ that generated it in the first place. This result in the objective:

$$\begin{aligned} &\max_{D} \ \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\log D(\boldsymbol{x})\right] + \cdot\mathbb{E}_{\boldsymbol{c}\sim p(\boldsymbol{c}),\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\log(1 - D(G(\boldsymbol{c},\boldsymbol{z})))\right] \\ &\min_{\text{enc},G} \ \mathbb{E}_{\boldsymbol{c}\sim p(\boldsymbol{c}),\boldsymbol{z}\sim p(\boldsymbol{z})}\left[||\boldsymbol{c} - \text{enc}(G(\boldsymbol{c}))||_2 - \lambda \cdot D(G(\boldsymbol{c},\boldsymbol{z}))\right] \end{aligned} \tag{2.13}$$

This objective learns a generator that maximizes the mutual information between codes $\boldsymbol{c}$ and generated samples $G(\boldsymbol{c})$. Using this setting, they extract features such as class information and rotation angle without supervision on multiple image disentanglement benchmarks.

β-VAE and Variants

Perhaps the most promising and studied line of work in unsupervised disentanglement is based on β-VAE (Higgins et al. 2017). While the phenomenon has yet to be fully understood, the authors noted that VAEs can be constrained to produce disentangled representations simply by imposing a stronger penalty on the KLD term of the ELBO. This slight alteration of the VAE objective by adding a hyperparameter β that controls the strength of the regularization gives the following objective:

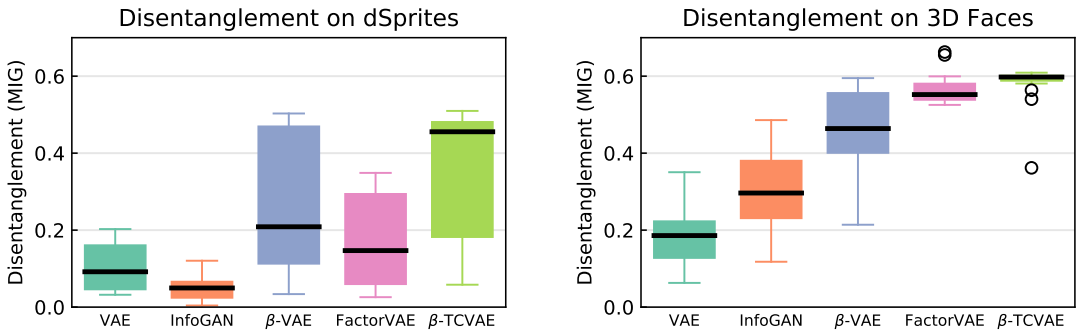
$$\mathcal{L}_{\beta-VAE} = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] - \beta \cdot D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})\right). \tag{2.14}$$

The authors provide comprehensive empirical evidence of successful disentanglement on many classic image datasets. It is hypothesized that the diagonal Gaussian prior encourages to discover dimensions that are independent of each other, resulting in a disentangled representation.

Building on these results, two concurrent works, FactorVAE (Kim et al. 2018) and β-TCVAE (Chen et al. 2018c) rewrites the KLD term as follows to isolate a Total Correlation term:

$$\begin{aligned}\mathbb{E}_{p(\boldsymbol{x})}\Big[D_{\mathrm{KL}}\big(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})\big)\Big] &= D_{\mathrm{KL}}\left(q(\boldsymbol{z},\boldsymbol{x})||q(\boldsymbol{z})p(\boldsymbol{x})\right) \\ &\quad + D_{\mathrm{KL}}\big(q(\boldsymbol{z})||\prod_j q(\boldsymbol{z}_j)\big) + \sum_j D_{\mathrm{KL}}\left(q(\boldsymbol{z}_j)||p(\boldsymbol{z}_j)\right)\end{aligned} \tag{2.15}$$

The second term in the decomposition corresponds to the Total Correlation term and measures the dependencies between each dimension of the aggregated posterior $q(\boldsymbol{z}) = \sum_{\boldsymbol{x}} q(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})$. Penalizing this term would promote independence between dimensions of the latent code while not affecting the two other terms that we might not want to penalize, as they represent the mutual information between the data $\boldsymbol{x}$ and the code $\boldsymbol{z}$, and the dimension-wise KLD between the prior and the posterior for $\boldsymbol{z}$. The main difference between the two models FactorVAE and β-TCVAE is in how they compute the Total Correlation term. FactorVAE uses adversarial training to estimate the term while β-TCVAE propose a weighted Monte Carlo approximation inspired by importance sampling. β-TCVAE is currently the state-of-the-art method for unsupervised disentanglement.



(a) dSprites (b) 3D Faces

Figure 2.5 – Disentanglement score (Mutual Information Gap) for different models on two benchmarks. The higher the better. Credits to Chen et al. 2018c.

Impossibility of Unsupervised Disentanglement

Despite the success of these methods, unsupervised learning of deep disentangled representations is still not well understood. Indeed, learning disentangled deep representation without supervision is fundamentally an ill-posed problem, as is non-linear ICA. This statement is supported by a result presented in Locatello et al. 2019b. Intuitively, the argument goes as follows: From a given factorized representation, the authors show how to build an infinite number of other equivalent factorized representations so that two cannot be disentangled at the same time. Any disentangling model would have to find the correct one in this infinite number of representations, and it cannot be distinguished using the unsupervised loss function alone. Therefore, some unclear inductive biases must account for the majority of the disentangling power behind those models. Uncovering those biases is a crucial challenge for a better understanding of both the unsupervised disentanglement as a task and the workings of neural networks.

It is noticeable that the models discussed in this subsection achieve very different performances (Figure 2.5), while also being very similar when viewed from the perspective of their objective functions. The InfoGAN objective is the same as that of Adversarial Auto-Encoder except that the role of the latent code c or z and that of the observation x are reversed. Adversarial Auto-Encoder generates a code and reconstructs the observation from the code where InfoGAN generates an observation and then retrieves the code. The β-VAE objective also resembles that of Adversarial Auto-Encoder. Both follow an encoder-decoder scheme where the latent codes are constrained to fit a specific distribution. Differences between all those models reside mostly in implementation choices (VAEs or GANs) and in differences in the relative compactness of the variables. Understanding the causes of the performance gap between those similar models might provide an insightful starting point.

2.2.2 Learning Factorized Representations

While previous methods try to encode each direction of variation into a single dimension, the concept of disentanglement can be extended to splitting information into distinct parts of the representation. A group of related features would be encoded into a, possibly multi-dimensional, variable, separated from another group of features if the two groups are considered independent. The overall representation is this way factorized into multiple parts that can be recomposed or mixed together into new combinations. Adversarial training is particularly suited to learn such kind of representations, with applications in domain adaptation, fairness, image synthesis, and video prediction. We present here four type of approaches to achieve such representation.

Censored Representations

Censoring, or removing a certain piece of information, from representations is a useful capability facilitated by adversarial training. From the fairness point a view, it can be desirable to learn a predictor f that doesn't use sensitive information encoded by a given variable $\boldsymbol{s}$. A widespread strategy, introduced by Edwards et al. 2016, is to learn a representation $\boldsymbol{z}$ of the data $\boldsymbol{x}$ that removes the sensitive variable $\boldsymbol{s}$ and then perform the prediction using only the censored representation $\boldsymbol{z}$. This same strategy is used in domain adaptation to learn representations that are shared across domains. This is easily done using adversarial training to censor the content of $\boldsymbol{z}$. From a similar setup, Fader Network (Lample et al. 2017) builds an image edition system in which the authors can control different attributes of an image. By censoring the attribute $\boldsymbol{s}$ from latent code $\boldsymbol{z}$ but performing prediction using both $\boldsymbol{s}$ and $\boldsymbol{z}$, the authors can then manipulate the output by modifying $\boldsymbol{s}$ as test time. Censored representations can be useful to build a system that learns factorized representations, as it gives the ability to control how the information is split in latent embeddings.

Representation Mixing

An intuitive way to use factorial representations is by mixing them. For instance, suppose $(\boldsymbol{z}_1, \boldsymbol{s}_1)$ encodes for two distinct features of $\boldsymbol{x}_1$, and $(\boldsymbol{z}_2, \boldsymbol{s}_2)$ for $\boldsymbol{x}_2$. One would expect $(\boldsymbol{z}_1, \boldsymbol{s}_2)$ and $(\boldsymbol{z}_2, \boldsymbol{s}_1)$ to encode for new examples that mix those features accordingly. Such constraints can be build using adversarial networks, as proposed by Mathieu et al. 2016. In their work, each data point $\boldsymbol{x}$ is annotated with a label $\boldsymbol{y}$ corresponding to its class. From this, they try to extract two independent representations, $\boldsymbol{z}$ that would encode for class information, and s that would encode for any other factor of variation. They train an encoder $\mathsf{enc} : \boldsymbol{x} \mapsto (\boldsymbol{z}, s)$ and a decoder $\mathsf{dec} : (\boldsymbol{s}, \boldsymbol{z}) \mapsto \boldsymbol{x}$ by mixing the latent representations, using ℓ_2 reconstruction loss for cases in which direct supervision is available, and

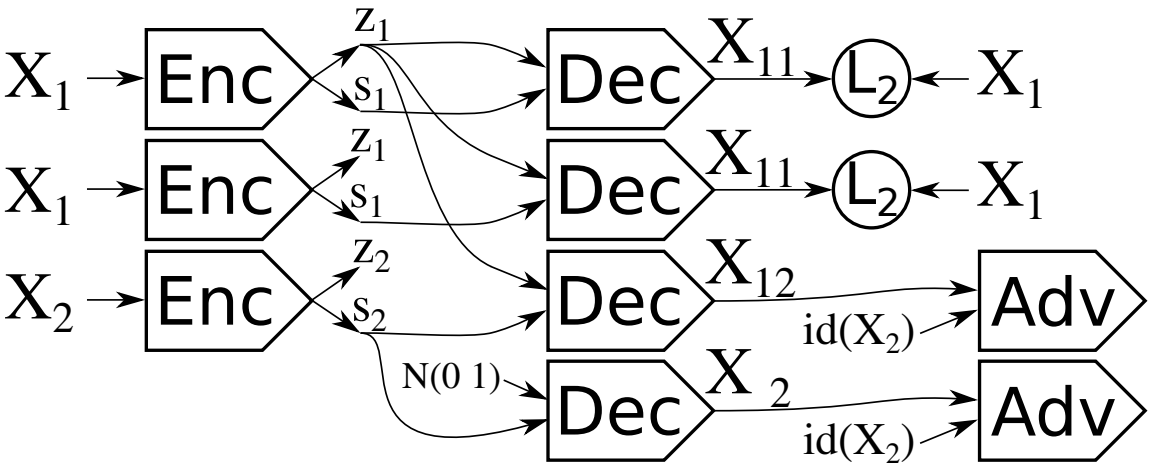


Figure 2.6 – Illustration of the different mixing pathways and associated objectives in Mathieu et al. 2016. Credits to Mathieu et al. 2016.

adversarial losses in the other cases to make sure that the output image is of the correct class. The system is illustrated in Figure 2.6.

Distribution Matching

Distribution matching can also be used to constrain two learned latent variables to be independent of one another. Indeed, suppose we want to learn an auto-encoder that split the code for data points $\boldsymbol{x} \in \mathcal{D}_{\mathcal{X}}$ into two distinct parts $\boldsymbol{z}$ and $\boldsymbol{s}$ that contain different information. Making sure $\boldsymbol{z}$ and $\boldsymbol{s}$ are independent variables can be done by adapting the Adversarial Auto-Encoder framework to the task of matching the joint distribution $q(\boldsymbol{z}, \boldsymbol{s})$ to the product distribution $p(\boldsymbol{z}, \boldsymbol{s}) = q(\boldsymbol{z})q(\boldsymbol{s})$. To do so, the adversarial framework only requires to be able to sample from $q(\boldsymbol{z}, \boldsymbol{s})$ and $p(\boldsymbol{z}, \boldsymbol{s})$. The former is immediate as it suffices to encode a random data point $\boldsymbol{z}_i, \boldsymbol{s}_i = \mathsf{enc}(\boldsymbol{x}_i)$. Sampling from $p(\boldsymbol{z}, \boldsymbol{s})$ can also be done easily by sampling two data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ independently, encoding both to recover $\boldsymbol{z}_i$, $\boldsymbol{s}_i$, $\boldsymbol{z}_j$, and $\boldsymbol{s}_j$, and mix them to obtain a pair of independently sampled representations $(\boldsymbol{z}_i, \boldsymbol{s}_j)$. This strategy is used by Denton et al. 2017 to separate static information and dynamic information in videos, and by Kim et al. 2018 in their estimation of the total correlation term in FactorVAE.

Removal of Redundancies by Regularizing

Another way to build independent representations is via careful regularization. Indeed, simply regularizing a variable might be enough to remove any redundant information. VAE's KLD regularization seems especially well suited to this approach and has been applied with great success in the context of video prediction to separate static information from the dynamic components in Denton et al. 2018 and Li et al. 2018.

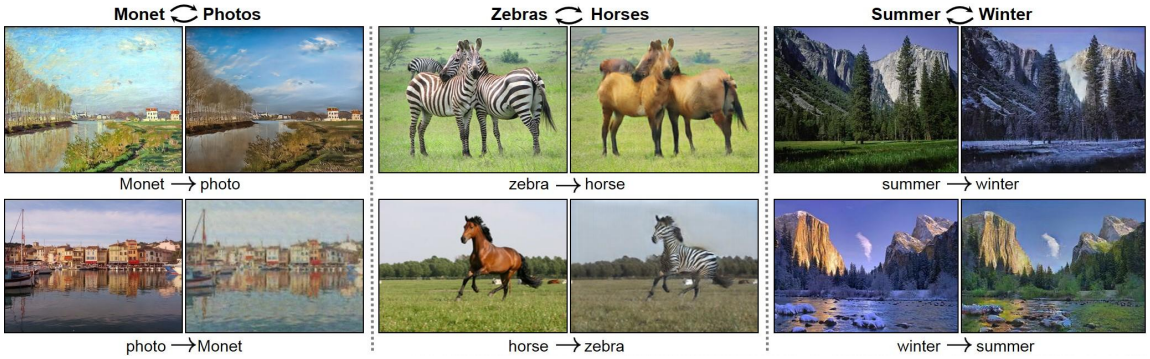


Figure 2.7 – Example of domain translations. Credits to Zhu et al. 2017.

2.2.3 Unpaired Domain Translation

Domain translation is a task that consists of projecting data points $\boldsymbol{x}$ that belong to the source domain $\mathcal{X}$ into a target domain $\mathcal{Y}$, as illustrated in Figure 2.7. A simple approach would be to frame the domain transfer as a multi-dimensional regression problem and learn a parametrized function f_θ by minimizing a reconstruction loss over pairs $(\boldsymbol{x}, \boldsymbol{y})$ from the dataset.

$$\mathcal{L}_{rec} = ||f_\theta(\boldsymbol{x}) - \boldsymbol{y}|| \tag{2.16}$$

This approach is for instance used for image-to-image translation by pix2pix (Isola et al. 2017) that augment this setup using adversarial training to make sure outputs $f_\theta(\boldsymbol{x})$ are realistic samples of the target domain $\mathcal{Y}$. However, this setup relies on the fact that the dataset contains paired examples $(\boldsymbol{x}, \boldsymbol{y})$. The task becomes more challenging when this supervision is lacking. In the unpaired setting, independent observations $\boldsymbol{x}$ from the source domain $\mathcal{X}$ and $\boldsymbol{y}$ from the target domain $\mathcal{Y}$ are readily available, but the correspondence between the two domains is unknown and no aligned pairs $(\boldsymbol{x}, \boldsymbol{y})$ are provided. There is, therefore, a need to find a mapping between those two domains while no example of such mapping is available.

Cycle-GAN

Cycle-GAN, a model that leverages adversarial training for image-to-image translation without paired data, have been proposed in three concurrent works (Kim et al. 2017a, Yi et al. 2017, Zhu et al. 2017). The idea is that by using adversarial training to learn a transformation $G : \mathcal{X} \to \mathcal{Y}$, it is possible to ensure the outputs of G belong to domain $\mathcal{Y}$. However, the problem is largely under-constrained, and the authors propose to train conjointly the inverse transformation $F : \mathcal{Y} \to \mathcal{X}$ using the same method. With the two functions, they can now able to enforce a cycle consistency loss that has to make sure that $F(G(\boldsymbol{x})) \approx \boldsymbol{x}$ and

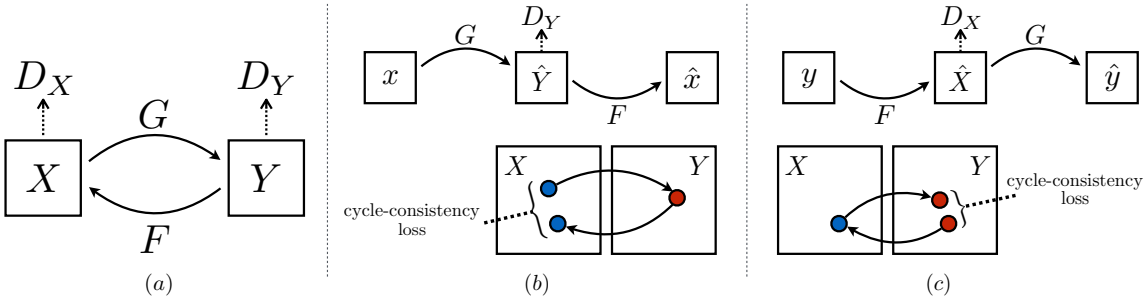


Figure 2.8 – Overview of Cycle-GAN. (a) Overall system. (b) Forward cycle-consistency. (c) Backward cycle-consistency. Credits to Zhu et al. 2017.

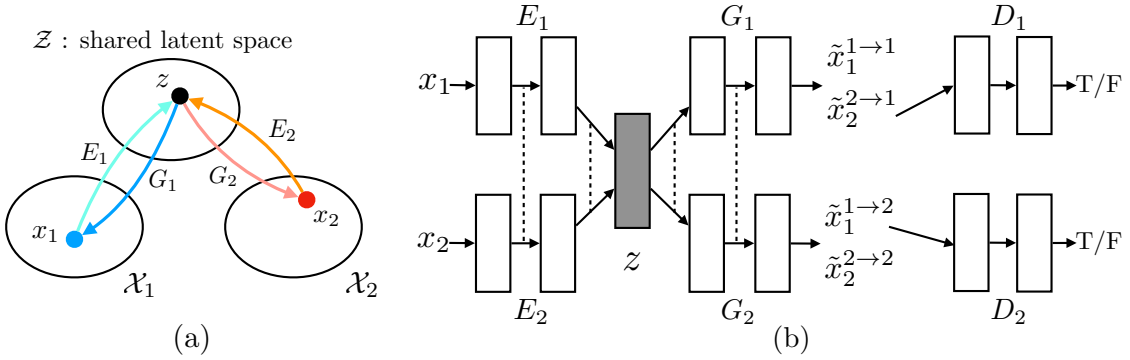


Figure 2.9 – Overview of UNIT. The encoder and the decoder for domain $\mathcal{X}_1$, and the encoder and the decoder for domain $\mathcal{X}_2$ are respectively denoted E_1, G_2, E_2, and G_2. (a) The shared latent space assumption. (b) The constraints: the dotted lines represent weights sharing and D_1 and D_2 are the discriminator for cross-domain losses. Credits to Liu et al. 2017.

$G(F(\boldsymbol{y})) \approx \boldsymbol{y}$ by minimizing the ℓ_1 distance between inputs and their respective reconstructions. The system is illustrated in Figure 2.8.

While the problem still lacks constraints, the authors demonstrate that they are nevertheless able to obtain good performances in the unpaired domain translation task between a variety of image domains. In this specific case, the good performances could be explained at least partially by the use of convolutional residual networks to implement G and F, as they are biased toward solutions that preserve the structure and only apply a change of texture instead.

UNIT

Another approach, UNIT, introduced by Liu et al. 2017, tries to discover a shared embedding space for both domains instead. UNIT uses two VAEs, one for

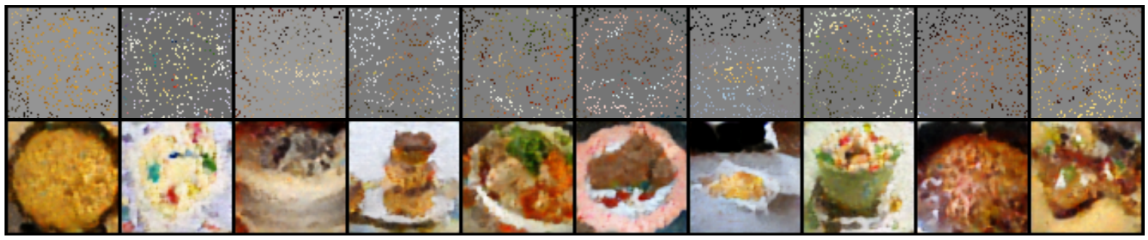


Figure 2.10 – Example of reconstructions (bottom row) of food images from a noisy observation (top row). Credits to Pajot et al. 2019

each domain $\mathcal{X}_1$ and $\mathcal{X}_2$. To promote the alignment of the latent spaces learned by the VAEs, they propose two additional constraints. First, the last layers of the encoders have shared weights, as well as the first layers of the decoders, limiting the capacity of the VAEs. Secondly, in addition to the VAE objectives, UNIT is also trained to translate between the two domains by decoding embeddings obtained from one domain using the decoder for the other domain. The outputs of those cross-domain pathways are constrained using adversarial training to belong to the desired target domain. The model and constraints are illustrated in Figure 2.9.

Unsupervised Machine Translation

Similar methods have been applied to natural language processing, for unsupervised machine translation without paired data. Zhang et al. 2017a, Zhang et al. 2017b, and Lample et al. 2018b demonstrate that Cycle-GAN's adversarial alignment method with cycle-consistency can be successfully applied between word embeddings of different languages for automated word translation, under simple orthogonality constraints for the translation functions. Artetxe et al. 2018 and Lample et al. 2018a later use a strategy more reminiscent of that of UNIT for sentence translation by using a shared Denoising Auto-Encoder on two corpora of different languages to learn aligned sentence embeddings. These strategies have been pushed to attain state-of-the-art performances even compared to fully supervised methods for machine translation.

2.2.4 Signal Reconstruction

In signal processing, a prominent problem is that of reconstructing a signal $\boldsymbol{x} \in \mathcal{X}$ from a set of measurements $\boldsymbol{y} \in \mathcal{Y}$, where $\boldsymbol{y}$ is degraded or noisy compared to the source signal $\boldsymbol{x}$. For instance, denoising consists of removing the noise from $\boldsymbol{y}$ to recover a clean image $\boldsymbol{x}$, super-resolution tries to retrieve a higher-resolution $\boldsymbol{x}$ from a low-resolution observation $\boldsymbol{y}$, and image inpainting is the task of completion of an image $\boldsymbol{y}$ where one region is missing. By defining a measurement function $f : \boldsymbol{x} \mapsto \boldsymbol{y}$ that maps the signal to its lossy observation, the

reconstruction problem is framed as an inverse problem that is to find the inverse mapping f^{-1} that for each given measurement $\boldsymbol{y}$ retrieves the corresponding signal $\boldsymbol{x}$. As the measurement is usually lossy, multiple $\boldsymbol{x}$ can fit the criterium and an additional regularization is usually imposed on $\boldsymbol{x}$. This problem can also be cast as the previously mentioned domain translation, but we present if the following some more specific approaches.

Deep Image Prior

Expressing the stated problem as energy minimization yield the following formulation:

$$\min_{\boldsymbol{x}} E(\boldsymbol{x}, \boldsymbol{y}) + R(\boldsymbol{x}), \tag{2.17}$$

where E is the task dependant data attachment term, and R the regularization term. A popular regularization, for instance, is the Total Variation to promote solutions with more uniform regions. Recently, Ulyanov et al. 2018 proposed to use a random deep CNNs as regularization prior, in the sense that they only consider solutions that can be generated by a fixed random CNN G. They changed the optimization problem from finding the optimal signal $\boldsymbol{x}$ to that of finding the optimal latent code $\boldsymbol{z}$, and remove the handcrafted regularization in favor of the CNN prior.

$$\min_{\boldsymbol{z}} E(G(\boldsymbol{z}), \boldsymbol{y}) \tag{2.18}$$

From this setting, Bora et al. 2017 proposes to learn the CNN prior by using the generator from a GAN, pre-trained on the uncorrupted images, instead. They also reintroduce a regularization, but on the latent space, as the generator has been pre-trained on $\boldsymbol{z} \sim \|0, I_d\|$ inputs.

$$\min_{\boldsymbol{z}} E(G(\boldsymbol{z}), \boldsymbol{y}) + ||\boldsymbol{z}||_2 \tag{2.19}$$

Note that contrary to other models that relied on implicit inductive biases to work, the formulations in these works explicitly acknowledge their existence, integrates the inductive biases into their assumptions and treat it as regularization.

Ambiant GAN

A variant of the signal reconstruction problem is when no clean signal $\boldsymbol{x}$ is available, and only $\boldsymbol{y}$ can be observed. In this case, Bora et al. 2018 demonstrates that it is possible to retrieve the clean data by using assumptions on the noise function instead. Their objective is to retrieve a clean dataset $\mathcal{X}$ given a noisy dataset $\mathcal{Y}$ and a simulated random noise function f_θ that needs to be differentiable. To do so, they use the GAN framework, with the difference that they add the noise layer f_θ after the generator, as depicted in Figure 2.11. In this process, the seed θ for the noise is assumed independent from the true signal $\boldsymbol{x}$. If the unobserved

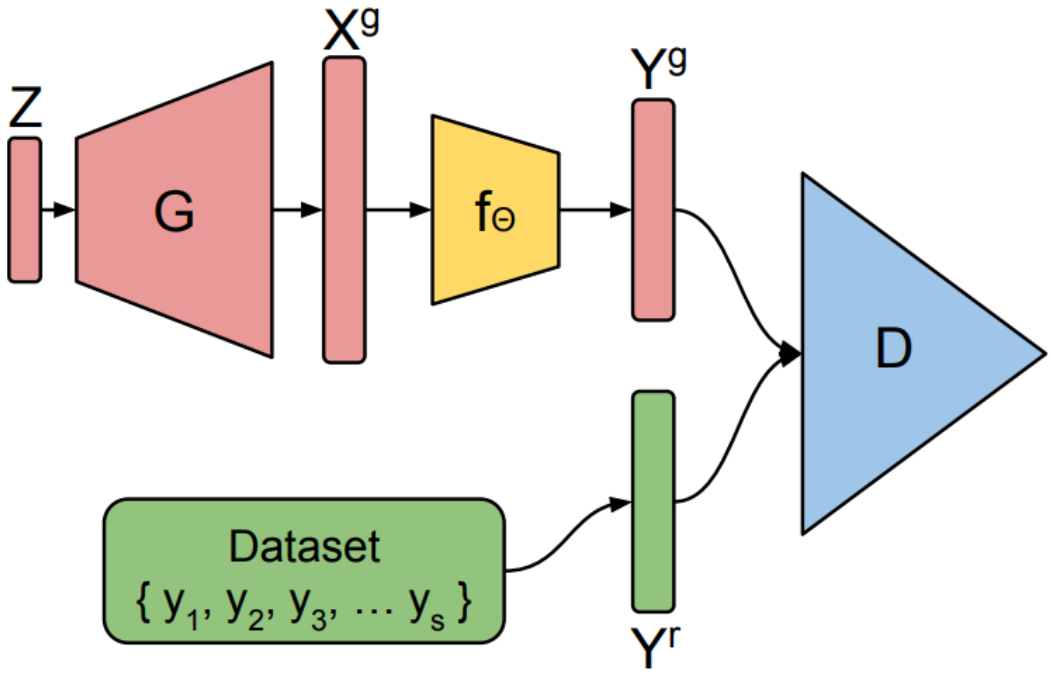


Figure 2.11 – AmbiantGAN training. The output of the generator is passed through a simulated noise function controlled by a random variable θ independently sampled for each output. Credits to Bora et al. 2018.

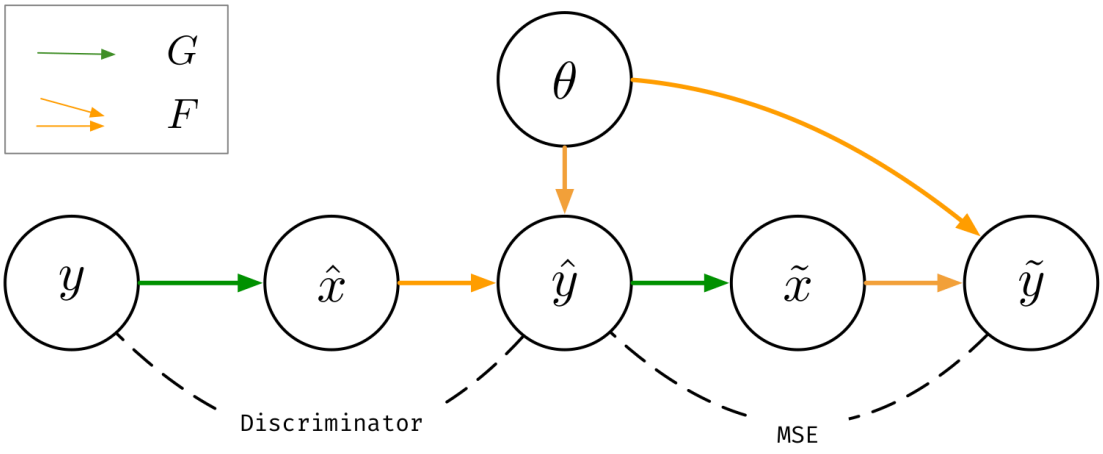


Figure 2.12 – Unsupervised Image Reconstruction model. The cycle-consistency between $\boldsymbol{y}$ and $f_\theta(G(\boldsymbol{y}))$ is enforced using pseudo-pairs $(\hat{\boldsymbol{y}}, \theta)$ that are produced using the generator. Credits to Pajot et al. 2019.

distribution $p(\boldsymbol{x})$ is unique, *i.e.* one and only one source distribution $p(\boldsymbol{x})$ can generate the given distribution of observations $p(\boldsymbol{y})$ given the random noise function, the method effectively recovers the correct distribution. The authors also show empirical experiments were the model still works even when the uniqueness assumption is violated. After training, the authors also propose to recover the correct $\boldsymbol{x}$ for a given $\boldsymbol{y}$ using gradient ascent on $\boldsymbol{z}$ as previously described in Bora et al. 2017.

Pajot et al. 2019 extends this model by directly learning a generator that takes $\boldsymbol{y}$ as input and outputs $\boldsymbol{x}$. The generator G is therefore also the inverse function $f^{-1} : \boldsymbol{y} \mapsto \boldsymbol{x}$. As they can't supervise directly on $\boldsymbol{x}$, they instead ensure that G is indeed the inverse of f by enforcing the cycle consistency $f_\theta(G(\boldsymbol{y})) = \boldsymbol{y}$. To do so, they need to know the random value of θ that produced the observation $\boldsymbol{y}$, but the information is not available from the dataset. Instead, they infer pseudo-pairs using the generator they are training. They first infer a signal $\boldsymbol{x}$, choose θ, then apply noise function f_θ to generate pseudo-pairs $(\theta, \boldsymbol{y})$, from which they can repeat the process to compute the cycle-consistency loss. The whole system is summarized in Figure 2.12.

2.3 Contributions

In the next chapters, we will present our contributions in this context.

In Chapter 3, we present a first contribution (Chen et al. 2017) for stochastic predictions with multi-view data. We introduce the task and confirm that deep generative models are suited in this setting. Our model draws inspiration from Bidirectional-GAN and uses probabilistic embeddings trained with the reparametrization trick, as well as a KLD-based regularization that is reminiscent of VAEs. We also present our subsequent work (Franceschi et al. 2020), which proposes a VAE-based state-space method for stochastic prediction of future frames in videos. The model takes advantage of separating a content embedding and a dynamics embedding and effectively uncouples the process of generating frames and that of predicting the latent dynamics to achieve state-of-the-art performances on multiple benchmarks.

Chapter 4 focuses more specifically on this idea of separation of information. We first show results from Wang et al. 2018b and Wang et al. 2018b, in which we apply information separation techniques in the context of Motion Capture Data. The presented models are based on Adversarial Auto-Encoders and use ideas from Fader Networks in a sequential setting. Those results highlight possible limitations with methods that use a single value or small vectors to encode for a feature. Then, based on those ideas, we present our work on factorized representation for multi-view image generation without view supervision (Chen et al. 2018b).

This setting is similar to that of Mathieu et al. 2016 we discussed earlier and we compare our approach to theirs.

Finally, Chapter 5 explains how we adapted concepts we learned in the previous chapter to the seemingly unrelated task of unsupervised image segmentation in Chen et al. 2019a. We propose a GAN-based generative model that separates the information of texture of different objects into independent vectors and we use it as a component in a system that allows training a segmentation function. The system is learned end-to-end and requires no supervision.

CHAPTER 3

PERFORMING STOCHASTIC PREDICTIONS

Contents

Many machine learning problems, such as least squares regression, aim at recovering either an average outcome or the most likely one. Algorithms built on such models would treat randomness, uncertainty, and the existence of different modalities, as noise that the model has to remove. However, in many cases, it would be beneficial to perform stochastic predictions and be able to consider possible alternative scenarios, building richer and more nuanced representations of the world. Probabilistic formulations can give such insight. The intuitive way to do so is posing the prediction problem as learning conditional probabilities, by estimating the distribution $p(\boldsymbol{y}|\boldsymbol{x})$ of possible outcomes $\boldsymbol{y}$ for any given input $\boldsymbol{x}$. This formulation is used in many standard setups, such as that of supervised classification. Indeed, for classification, the output of the neural network is often interpreted as the parameters of a categorical distribution over the possible class labels for the given input. In this example though, as the distribution is simple and easily summarized by its vector of parameters, sampling is not useful.

Sampling becomes more interesting when the output distribution is more complex, for instance when estimating a high-dimensional structured distribution. In general, there is no simple way to express such distribution via explicit parametrization. Instead, the different strategies rely on modeling the distributions implicitly and observing samples from that distribution. The task is challenging, though, as supervision is not perfect. Indeed, while we are trying to learn, for each input $\boldsymbol{x}$ a conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$, we only observe one

realization of $\boldsymbol{y}$. A pair $(\boldsymbol{x}, \boldsymbol{y})$ from the dataset provides a positive example but doesn't account for other unobserved possible realizations and therefore cannot directly serve to evaluate predictions.

Note that one strategy for sampling different outcomes, popularized by PixelCNN (Oord et al. 2016), is to treat the high-dimensional output as a sequence of scalar outputs. This changes the problem of generating one high-dimensional outcome to that of performing one (self-)supervised small prediction at a time for each dimension of the output space. This method still doesn't model the whole distribution explicitly, as it would require exploring all branches for each prediction step in the sequence, but are able to produce high-quality outputs in exchange for a steep computational cost. We focus instead on methods based on deep generative models that are faster by orders of magnitude than PixelCNN-based ones.

In this chapter, we discuss models that are able to perform stochastic predictions for high-dimensional structured data, such as images and videos. We present our contribution to two application domains with their own additional challenges and difficulties. The first domain, multi-view data, consists in combining multiple sources of information and have to deal with possibly missing views. In the second domain, videos, we have to capture the complex dynamics to correctly predict. We organize the chapter as follows: In Section 3.1, we present a first contribution that explored the feasibility of this approach for high-dimensional data. We introduce a framework for stochastic prediction, show how to account for uncertainty using stochastic embeddings and demonstrate the applicability of the concept on a high-dimensional dataset. In Section 3.2, we present with a second contribution a practical method for stochastic video prediction that takes advantage of recent developments in the field and demonstrated state-of-the-art performances on multiple challenging video datasets.

Those work have led to two publications:

- Mickaël Chen and Ludovic Denoyer (2017). "Multi-view Generative Adversarial Networks". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part II*, pp. 175–188

- Jean-Yves Franceschi, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari (2020). "Stochastic Latent Residual Video Prediction". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, forthcoming*

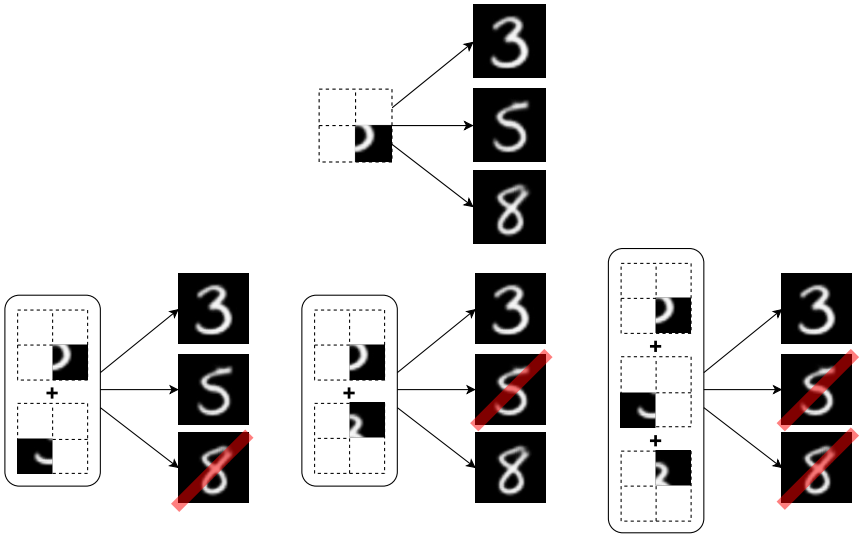


Figure 3.1 – Example of stochastic prediction behavior. For a given input, multiple predictions are possible. In MV-BiGAN, the complementarity of views is expressed under the following statement: As more information is available, the possibilities are narrowed down.

3.1 Multi-view Bidirectional-GAN

In a first contribution, we designed a method to sample diverse possible outputs using a stochastic latent space. Assuming that additional information should reduce the uncertainty over the outputs, we were able to build a system that can perform stochastic predictions and whose behavior is consistent when more information is added (Figure 3.1). We apply this model to multi-view data, as it allows us to tackle a set of interesting challenges in machine learning in general while having direct real-life impact and applications.

Indeed, many concrete applications involve multiple sources of information generating different views on the same object (Cesa-Bianchi et al. 2010). If we consider human activities, for example, GPS values from a mobile phone, navigation traces over the Internet, or even photos published on social networks are different views on a particular user. In multimedia applications, views can correspond to different modalities (Atrey et al. 2010) such as sounds, images, videos, and sequences of previous frames.

Challenges associated with multi-view settings are multiple. First, how to integrate and take advantage of the complementarity of views? Different works have explored how to effectively fusion views in a deep architecture (Ben-younes et al. 2019; Bordes et al. 2019; Ngiam et al. 2011; Srivastava et al. 2012; Wang et al. 2015). Secondly, how to handle the fact that some views might be missing? Ideally, a model should be robust to the absence of some views.

This section is organized as follows: Section 3.1.1 presents our model, Section 3.1.2 gives practical implementation choices, Section 3.1.3 shows our results and Section 3.1.4 discusses follow-ups of this work.

3.1.1 Model

In the following, we formalize our task and introduce notations and then present the general idea of our approach. We then translate the idea into a learning objective using adversarial losses, before discussing aggregation schemes for multi-view data. Finally, we explain how we stabilize our model by forcing a consistency of latent representations when views are missing.

Notations and task. Let us denote $\mathcal{X}$ the space of objects $\boldsymbol{x}$ on which different views will be acquired, V the number of possible views and, $\boldsymbol{x}_k$ the k-th view over $\boldsymbol{x}$. As some of the views can be missing, we represent a subset of views $\boldsymbol{v} = \{\boldsymbol{x}_k | \text{ the k-th is available for object } \boldsymbol{x}\}$. As each object $\boldsymbol{x}$ is associated with a target prediction $\boldsymbol{y} \in \mathcal{Y}$ in a dataset, the objective is to estimate the distributions $p(\boldsymbol{y}|\boldsymbol{v})$ for each available set of views $\boldsymbol{v}$. As all views $\boldsymbol{x}_k$ and target $\boldsymbol{y}$ are possibly high-dimensional, we use the stochastic prediction framework and aim to learn an implicit distribution from which we can sample diverse predictions. For instance, $\boldsymbol{v}$ could be a set of partial observations extracted from an image and the target $\boldsymbol{y}$ could be the full image, including unobserved parts.

General idea. The basic idea behind our approach is to jointly learn an encoder E and a decoder G that are respectively a mapping between the output space to a latent space and its inverse, as well as an encoder H that estimates a distribution over this latent space for a set of views $\boldsymbol{v}$ given as input. This setting would allow the sampling of different possible outcomes by decoding multiple latent representations sampled from the conditional distribution. To this setup, we add an assumption, that is that views are complementarity, or in other words, each views allow the reduction of the uncertainty over the output. These behaviors are illustrated in Figure 3.2.

Training objectives. To implement and train these functions, we need two distinct objectives. The first objective is to learn a bidirectional mapping between the latent space and the output space. To do so, we use the Bidirectional-GAN framework presented in Chapter 2. Accordingly, we introduce a discriminator D_1 that is used to match joint distributions $p_E(\boldsymbol{z}, \boldsymbol{y}) = p(\boldsymbol{y})p_E(\boldsymbol{z}|\boldsymbol{y})$ and $p_G(\boldsymbol{z}, \boldsymbol{y}) = p(\boldsymbol{z})p_G(\boldsymbol{y}|\boldsymbol{z})$ with the following minimax objective:

$$\min_{G,E} \max_{D_1} \mathbb{E}_{\boldsymbol{x},\boldsymbol{z}\sim p_E(\boldsymbol{x},\boldsymbol{z})} \left[\log D_1(\boldsymbol{x}, \boldsymbol{z})\right] + \mathbb{E}_{\boldsymbol{x},\boldsymbol{z}\sim p_G(\boldsymbol{x},\boldsymbol{z})} \left[1 - \log D_1(\boldsymbol{x}, \boldsymbol{z})\right]. \quad (3.1)$$

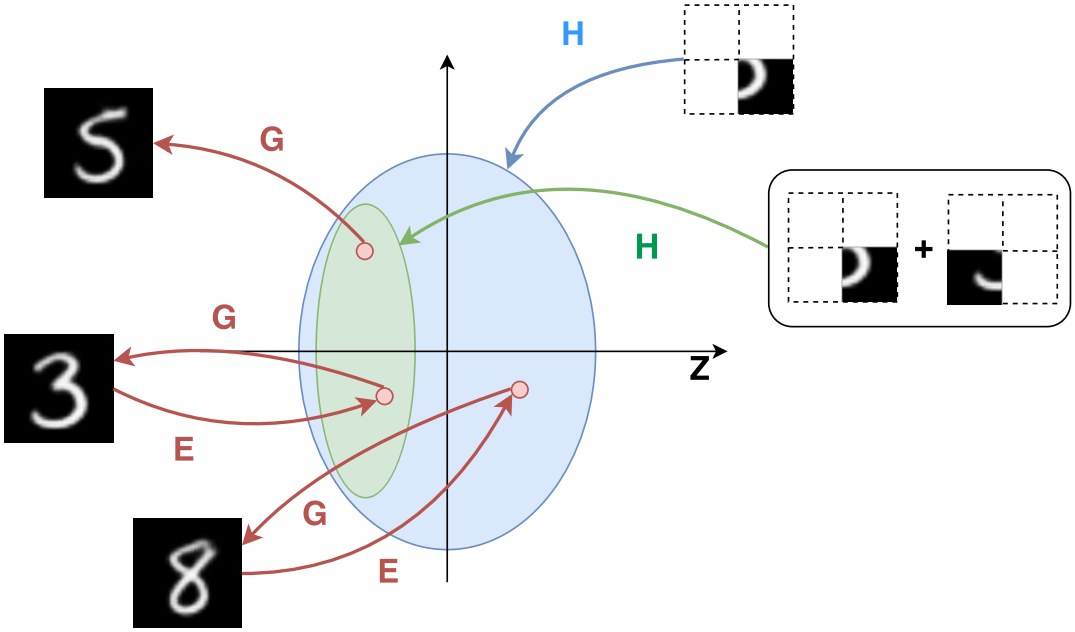


Figure 3.2 – Inputs (sets of views) and outputs are embedded as distributions in the same shared latent space. H is used to encode a set of views into a distribution in the latent space. It is then possible to sample multiple vectors $\boldsymbol{z}$ from this distribution that can then be decoded by G to the corresponding outputs.

The second objective is to learn the mapping between the input space (the views) and the latent space. This mapping needs to be consistent with the previously mentioned one: Given a pair $(\boldsymbol{v}, \boldsymbol{y})$ from the dataset, we wish a latent representation $\boldsymbol{z}$ sampled from $p_H(\boldsymbol{z}|\boldsymbol{v})$ to be similar to one from $p_E(\boldsymbol{z}|\boldsymbol{y})$. We define two joint distributions over $\boldsymbol{v}$ and $\boldsymbol{z}$:

$$\begin{aligned} p_H(\boldsymbol{v}, \boldsymbol{z}) &= p_H(\boldsymbol{z}|\boldsymbol{v})P(\boldsymbol{v}) \\ p_E(\boldsymbol{v}, \boldsymbol{z}) &= \sum_{\boldsymbol{y}} p_E(\boldsymbol{z}|\boldsymbol{y})p(\boldsymbol{v}, \boldsymbol{y}). \end{aligned} \tag{3.2}$$

Minimizing the Jensen-Shanon divergence between these two distributions is equivalent to solving the following adversarial problem against a second discriminator D_2:

$$\min_{E,H} \max_{D_2} \mathbb{E}_{\boldsymbol{v},\boldsymbol{z}\sim p_E(\boldsymbol{v},\boldsymbol{z})} \left[\log D_2(\boldsymbol{v}, \boldsymbol{z})\right] + \mathbb{E}_{\boldsymbol{v},\boldsymbol{z}\sim p_H(\boldsymbol{v},\boldsymbol{z})} \left[1 - \log D_2(\boldsymbol{v}, \boldsymbol{z})\right]. \tag{3.3}$$

Also, as we want H to output a distribution over the latent space, we model $p_H(\boldsymbol{z}|\boldsymbol{v})$ as a distribution $\mathcal{N}(\mu_H(\boldsymbol{v}), \sigma_H(\boldsymbol{v}))$ whose parameters $\mu_H(\boldsymbol{v})$ and $\sigma_H(\boldsymbol{v})$

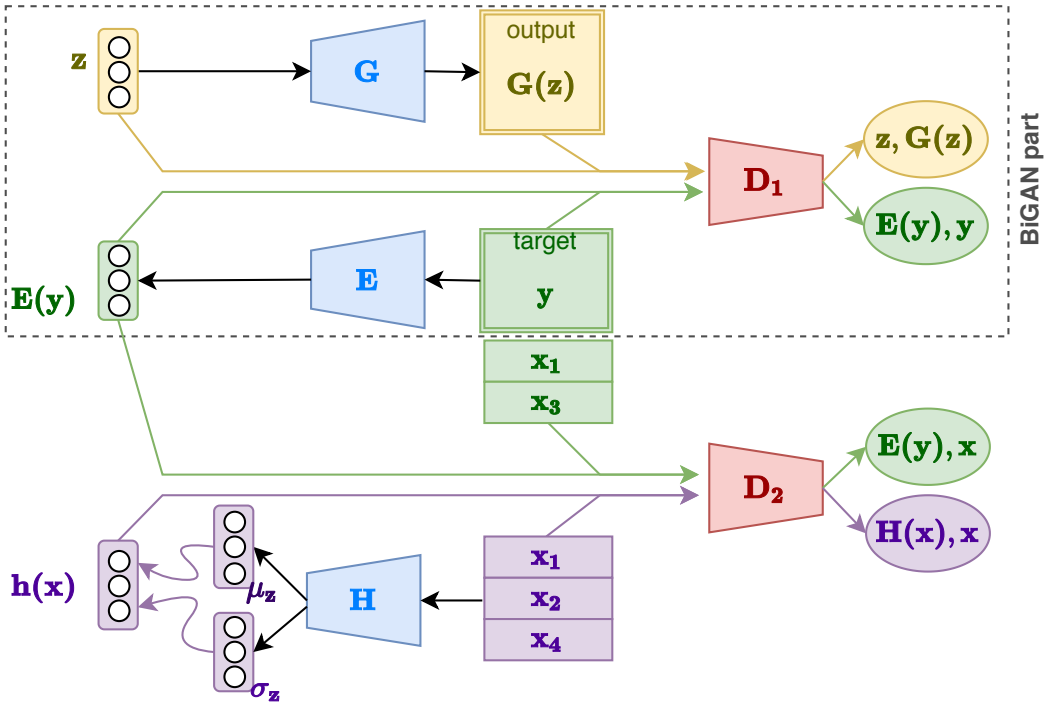


Figure 3.3 – Overview of the adversarial losses. The top part is similar to BiGAN and matches the output space to the latent space via the bidirectional mapping E and G. The bottom part matches the embedding for the views obtained by H to the embedding for the corresponding target obtained by E.

are obtained by forwarding input $\boldsymbol{v}$ through the learned function H. By merging the two objectives, we obtain the following learning problem:

$$\begin{aligned}\min_{G,E,H}\max_{D_1,D_2} \mathbb{E}_{\boldsymbol{y},\boldsymbol{z}\sim p_E(\boldsymbol{y},\boldsymbol{z})}\left[\log D_1(\boldsymbol{y},\boldsymbol{z})\right] + \mathbb{E}_{\boldsymbol{y},\boldsymbol{z}\sim p_G(\boldsymbol{y},\boldsymbol{z})}\left[1-\log D_1(\boldsymbol{y},\boldsymbol{z})\right] + \\ \mathbb{E}_{\boldsymbol{v},\boldsymbol{z}\sim p_E(\boldsymbol{v},\boldsymbol{z})}\left[\log D_2(\boldsymbol{v},\boldsymbol{z})\right] + \mathbb{E}_{\boldsymbol{v},\boldsymbol{z}\sim p_H(\boldsymbol{v},\boldsymbol{z})}\left[1-\log D_2(\boldsymbol{v},\boldsymbol{z})\right].\end{aligned} \tag{3.4}$$

This pipeline is illustrated in Figure 3.3.

Aggregation of views. As we consider the problem of computing an output distribution conditioned by multiple different views, encoder H needs to aggregate the different views into a single representation. Possible strategies can range from the very simple sum/average pooling to more complex schemes using recurrent networks for example. We opted for a basic sum-pooling solution detailed in Figure 3.4.

Note that our model is trained based on datasets where all the views are available for each data point. In order to generate examples where only subsets of views are available, the ideal procedure would be to consider all the possible subsets of views. Due to the number of data points that would be generated by such a procedure, we build random sequences of incremental sets of views and enforce the KLD regularization over successive sets.

Uncertainty reduction assumption. However, the previous idea suffers from a very high instability when learning, as it is usually noted with complex GANs architectures. In order to stabilize our model, we propose to add a regularization based on the idea that adding new views to an existing subset of views should reduce the uncertainty over the outputs. Indeed, under the assumption that views are consistent one another, adding a new view should allow to refine the predictions and reduce the variance of the distribution of the outputs.

Let us consider an object $\boldsymbol{x}$ observed via two sets of views, $\boldsymbol{v}$ and $\boldsymbol{v}'$, so that $\boldsymbol{v} \subset \boldsymbol{v}'$. Then, $\boldsymbol{v}'$ have more information than $\boldsymbol{v}$ over $\boldsymbol{x}$ has it contains at least all the views that are in subset $\boldsymbol{v}$. Intuitively, $p(\boldsymbol{x}|\boldsymbol{v}')$ should be "included" in $p(\boldsymbol{x}|\boldsymbol{v})$, in the sense that any $\boldsymbol{x}$ that is likely given $\boldsymbol{v}'$ is also likely given $\boldsymbol{v}$. In our model, since we assume a deterministic bidirectional mapping between y and x, this can be enforced at a latent level by minimizing $D_{\mathrm{KL}}(p(\boldsymbol{z}|\boldsymbol{v}')||p(\boldsymbol{z}|\boldsymbol{v}))$. By assuming those two distributions are diagonal gaussian distributions the KLD can be computed in close-form as in Equation 2.2 and differentiated w.r.t. the parameters of all involved functions. Note that this divergence is computed on the estimation made by the function H and will act as a regularization over the latent conditional distributions.

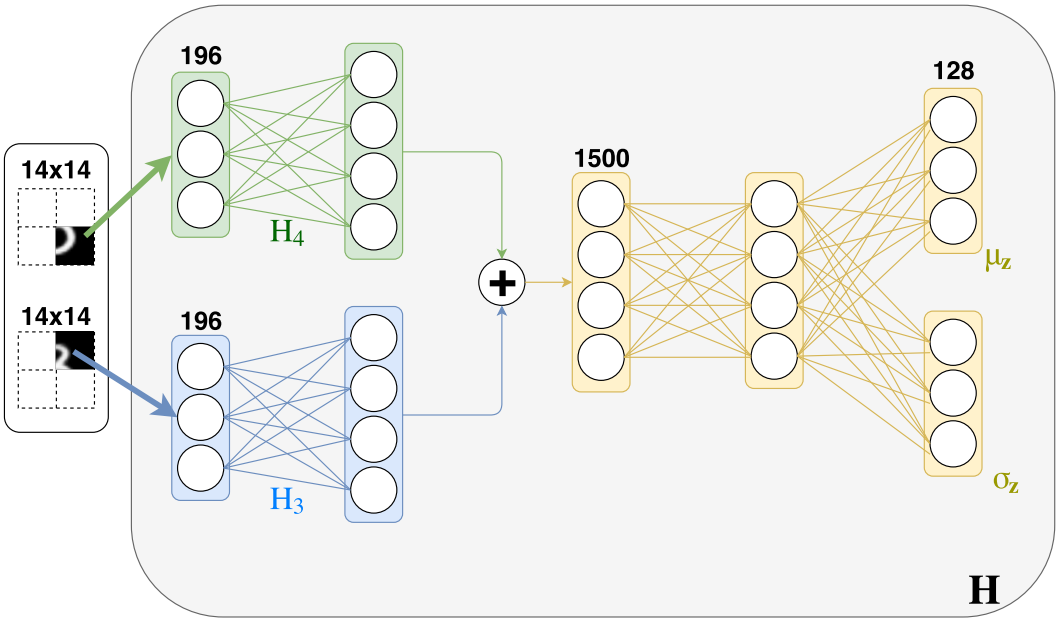


Figure 3.4 – Detail of the implementation used for H. View aggregation is a sum of linear transformations from each view's space. A Multi-Layer Perceptron (MLP) with one hidden layer and ReLU activations then transform the aggregate into the parameters of $p_H(\boldsymbol{z}|\boldsymbol{v})$. The hidden layer of this MLP also uses batch-normalization. The output layer has a tanh activation for μ and a negative exponential linear unit for σ^2.

The final objective function of the MV-BiGAN can be written as:

$$\begin{aligned}\min_{G,E,H}\max_{D_1,D_2} &\mathbb{E}_{\boldsymbol{y},\boldsymbol{z}\sim p_E(\boldsymbol{y},\boldsymbol{z})}\left[\log D_1(\boldsymbol{y},\boldsymbol{z})\right] + \mathbb{E}_{\boldsymbol{y},\boldsymbol{z}\sim p_G(\boldsymbol{y},\boldsymbol{z})}\left[1-\log D_1(\boldsymbol{y},\boldsymbol{z})\right] + \\ &\mathbb{E}_{\boldsymbol{v},\boldsymbol{z}\sim p_E(\boldsymbol{v},\boldsymbol{z})}\left[\log D_2(\boldsymbol{v},\boldsymbol{z})\right] + \mathbb{E}_{\boldsymbol{v},\boldsymbol{z}\sim p_H(\boldsymbol{v},\boldsymbol{z}}\left[1-\log D_2(\boldsymbol{v},\boldsymbol{z})\right] + \\ &\lambda\cdot\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\sum_{\boldsymbol{v}\subset\boldsymbol{v}'} D_{\mathrm{KL}}(p_H(\boldsymbol{z}|\boldsymbol{v}')||p_H(\boldsymbol{z}|\boldsymbol{v})),\end{aligned} \tag{3.5}$$

where λ controls the strength of the regularization.

3.1.2 Implementation choices

This section presents the different architectures we use to implement the different functions E, G, H, D_1 and D_2, and the hyperparameters for learning our models. Implementations were made using the LUA Torch7 framework and experiments conducted on an Nvidia Tesla K20m GPU device.

Neural architectures. Our experiments are mostly performed on variations of the MNIST handwritten digits dataset (LeCun et al. 1998). On those experiments, the generator function G is a MLP with three hidden layers. The second and the

Operation	Kernel	Stride	Pad	Feature maps	BN	Non-linearity
Convolution	4×4	2×2	1×1	64	$\times$	Leaky ReLU
Convolution	4×4	2×2	1×1	128	$\surd$	Leaky ReLU
Convolution	4×4	2×2	1×1	256	$\surd$	Leaky ReLU
Convolution	4×4	2×2	1×1	512	$\surd$	Leaky ReLU
Convolution	4×4	1×1		1000	$\times$	Linear
Transposed conv.	4×4	1×1		512	$\surd$	ReLU
Transposed conv.	4×4	2×2	1×1	256	$\surd$	ReLU
Transposed conv.	4×4	2×2	1×1	128	$\surd$	ReLU
Transposed conv.	4×4	2×2	1×1	64	$\surd$	ReLU
Transposed conv.	4×4	2×2	1×1	3	$\times$	Tanh

Table 3.1 – Convolution architectures used in our experiments on the CelebA dataset. The top part is used for encoding images into the aggregation space. The bottom part is used in G to generate images from a vector z.

third hidden layers are followed by batch normalizations and ReLUs while the output layer uses a sigmoid. The discriminator D_1 is composed of three fully connected layers with batch normalization at the third layer. A sigmoid is applied to the outputs. The vector z is concatenated to the representation at the second layer. The encoding network E and discriminator D_2 are very similar to the view encoder H that is detailed in Figure 3.4. The only differences are that D_2 uses a sigmoid as output activation and concatenates z after the aggregation step. Hidden layers in generator G or encoder networks E and H are of dimension 1500 and are followed by ReLUs non-linearities while leaky-ReLUs with slope 0.2 are used in the discriminator. Latent representations z are of size 128.

Optimization scheme. Models are optimized using a stochastic gradient descent method with mini-batches of size 128 over 300 epochs. In an adversarial fashion, we first update the discriminator networks D_1 and D_2, then we update the generator and encoders G, E and H with gradient steps in opposite directions. As with most other implementation of GAN-based models, we find that using the Non-Saturating adversarial objective proposed by Goodfellow et al. 2014 for E, G and H instead of the minimax objective leads to more stable training. For the gradient descent, we used Adam (Kingma et al. 2015) update procedure with standard hyperparameters $\beta_1 = 0.5$, $\beta_2 = 10^{-3}$ and a learning rate of $2 \cdot 10^{-5}$.

Convolutional variant. We also performed additional experiments using the CelebA dataset of faces (Liu et al. 2015). On that dataset, we used convolutional encoders, generators, discriminators similar to that popularized by DCGAN

(Radford et al. 2016), and described in table 3.1, to encode and decode images. For attribute vectors, we kept linear transformations. For discriminators, the convolution network is followed by a hidden fully connected layer before the output layer and z is concatenated at this hidden fully connected level. As in the MNIST experiments, the discriminator D_2 is similar to E and H, and z is concatenated directly at the aggregation level. Aggregation space is of size 1000. λ is set to 10^{-3}, and mini-batches size is 16. The model has been trained for 15 epochs.

3.1.3 Results

We evaluated our model qualitatively on three different types of experiments and two different datasets. The first dataset we experimented on is the MNIST dataset of handwritten digits. Experiments on MNIST are used to illustrate the ability of the MV-BiGAN to handle different subsets of views and to update its prediction when integrating new incoming views. The second dataset is the CelebA dataset, composed of both images of faces and corresponding attributes. We used to show how MV-BiGAN can deal with heterogeneous views. We present here our experiments and results.

MNIST, 4 views. We consider the problem where 4 different views can be available, each view corresponding to a particular quarter of the final image to predict – each view is a vector of $\mathbb{R}^{(14\times14)}$. The MV-BiGAN is used here to recover the original image. The model is trained on the MNIST training digits, and results are provided on the MNIST testing dataset.

Figure 3.5 illustrates the results obtained for some digits. In this figure, the second column displays the input (the subset of views), while the third shows predicted outputs sampled by MV-BiGAN. An additional view is added between each row. This experiment shows that when new views are added, the diversity in the predicted outputs decreases due to the KLD constraint introduced in the model, which is the desired behavior i.e. more information implied less variance. When the KLD constraint is removed (Figure 3.6), the diversity of outcomes remains important even if many views are provided to the model. This illustrates the importance of the KLD regularization term in the MV-BiGAN objective.

MNIST, sequence of incoming views. We present in Figure 3.7 another set of experiments in which the views correspond to images with missing values (missing values are replaced by 0.5). This can be viewed as a data imputation problem. Here also, the behavior of the MV-BiGAN exhibits interesting properties: the model is able to predict the desired output as long as enough information has been provided. When only non-informative views are provided, the model

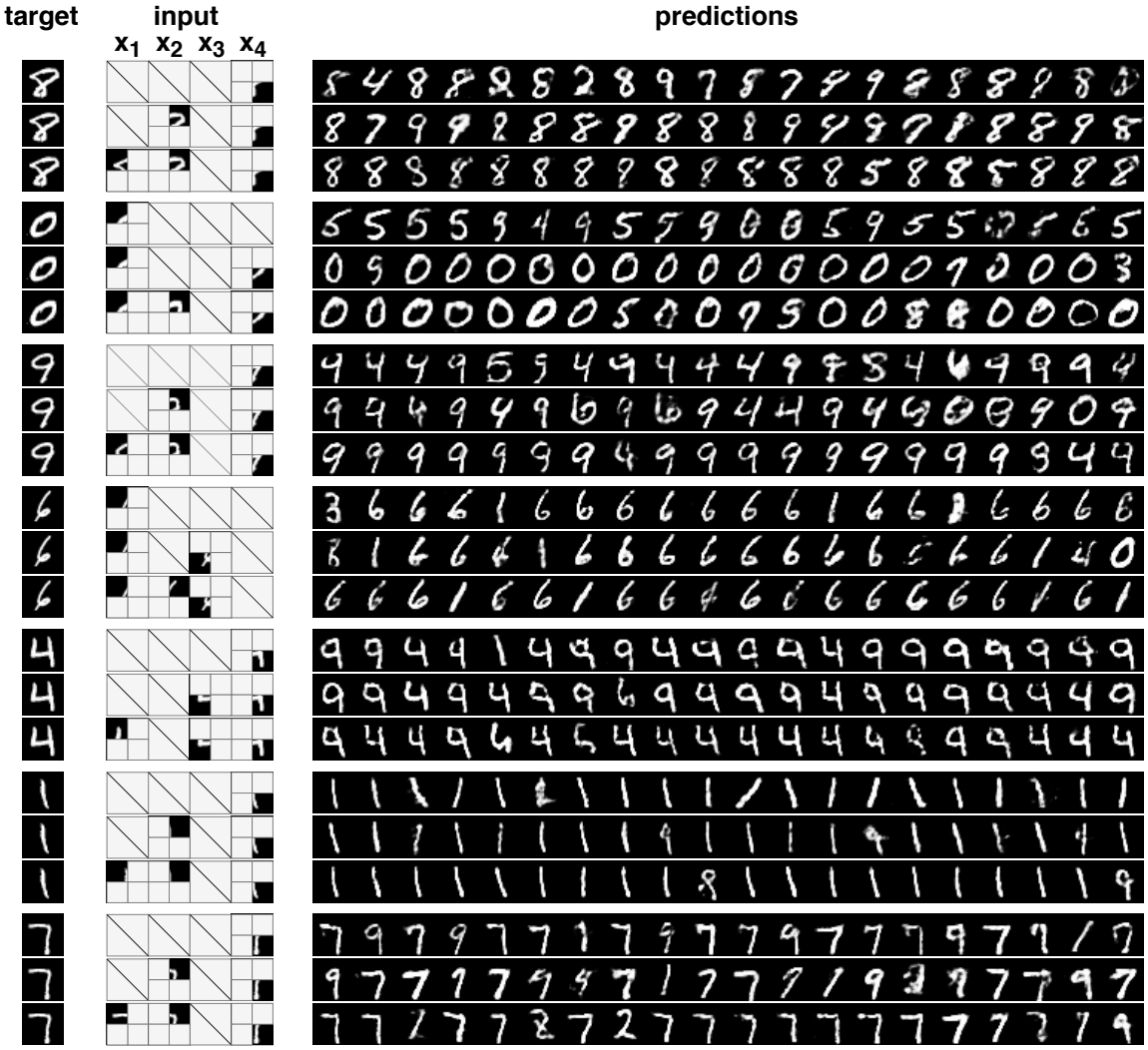


Figure 3.5 – Results of the MV-BiGAN on sequences of views. On each row, the first column corresponds to the ground-truth target, the second to the set of provided views, while the third columns correspond to possible predictions according to MV-BiGAN. In each block of three rows, the first row uses only one view as input, the second and the third rows each add one more view to the input from the previous row.

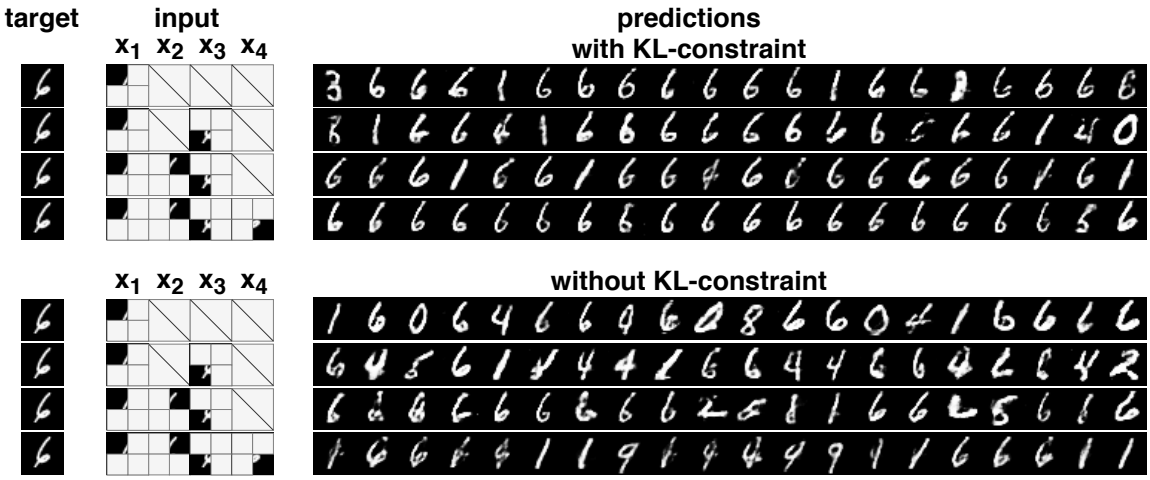


Figure 3.6 – Comparaison between MV-BiGAN with (top) and without (bottom) KL-constraint.

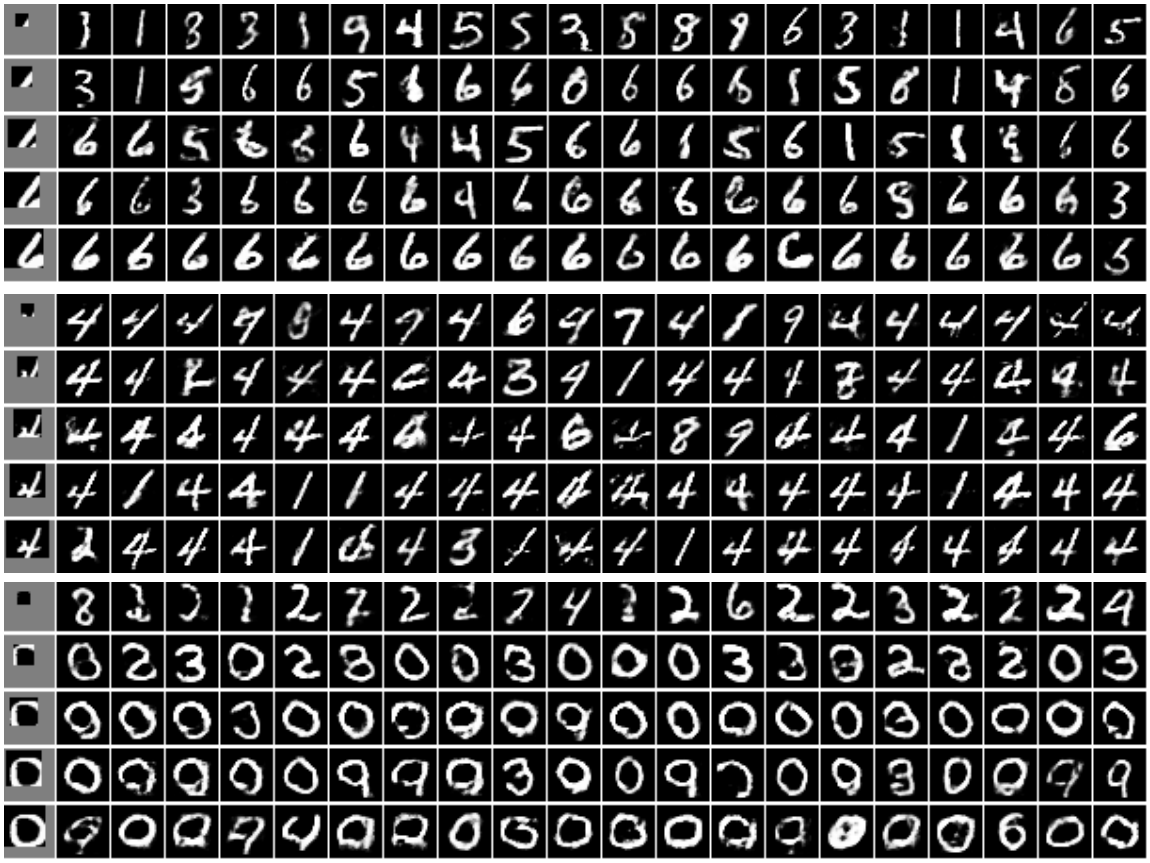


Figure 3.7 – MV-BiGAN ouputs (all columns except the first) with sequences of incoming views (first column). Here, each view is a 28×28 matrix. The successive views are added at each row. Values are between 0 and 1 with missing values replaced by 0.5.

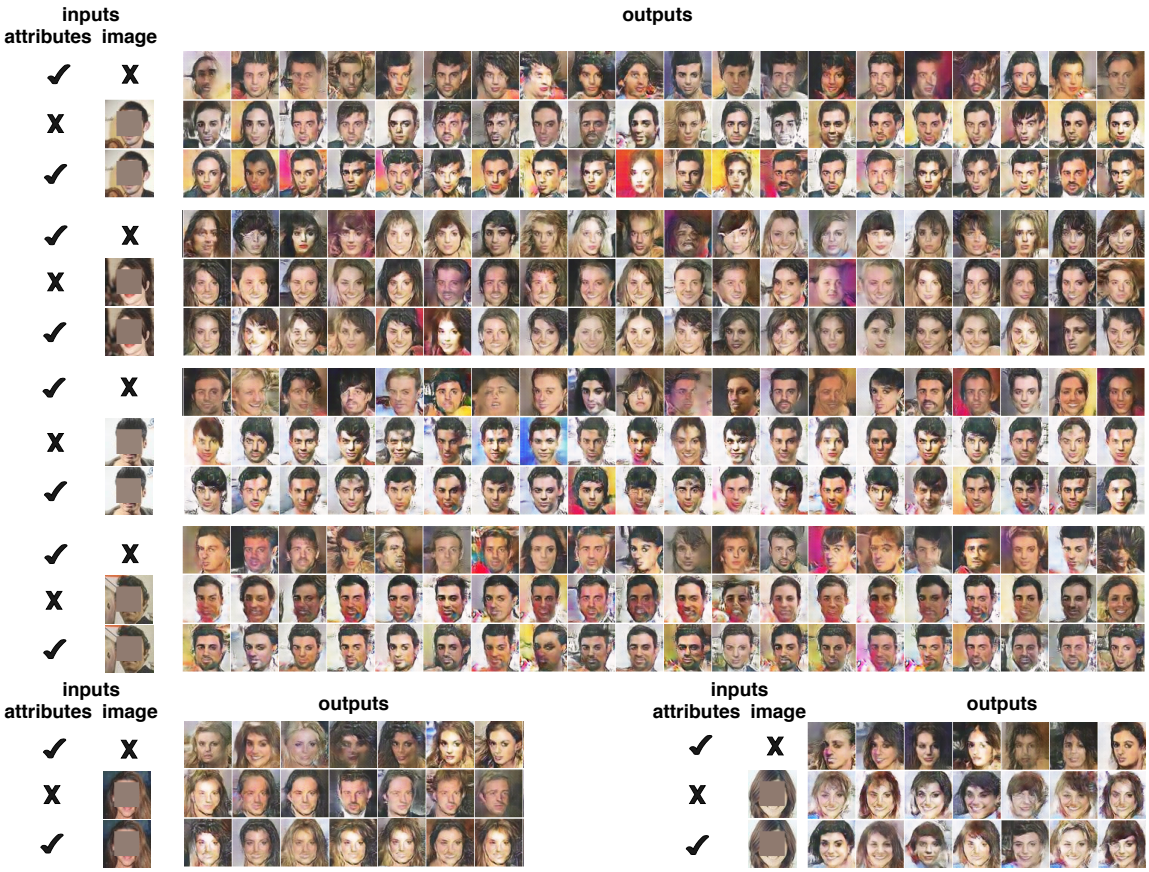


Figure 3.8 – Results obtained on the CelebA dataset. In each block, the first row corresponds to the images generated based on the attribute vector, the second row corresponds to images generated based on the incomplete face, the third row corresponds to the images generated based on the two views.

produces highly diverse digits. The diversity decreases when new information is added.

CelebA, integrating heterogeneous information. At last, the third experiment aims at measuring the ability of MV-BiGAN to handle heterogeneous inputs. We consider two views: (i) the attribute vector containing information about the person in the picture (such as hair color and gender), and (ii) an incomplete face picture. Figure 3.8 illustrates the results. The first line corresponds to the faces generated based on the attribute vector. One can see that the attribute information has been captured by the model: for example, the gender of the generated face looks correct showing that MV-GAN has captured this information from the attribute vector. The second line corresponds to the faces generated when using the incomplete face as an input. One can also see that the generated outputs are "compatible" with the incomplete information provided to the model. But the

attributes are not considered (for example, faces of both genders are generated). At last, the third line corresponds to images generated based on the two partial views (attributes and incomplete face) which are close to the ground-truth image (bottom left). However, in this set of experiments, MV-BiGAN's training is instable, and the quality of the generated faces is still not satisfying.

3.1.4 Follow-ups

We have presented the MV-BiGAN model for estimating conditional densities and handle multi-view inputs. The MV-BiGAN model is able to both handle subsets of views, but also to update its prediction when new views are added. It is based on the idea that the uncertainty of the prediction must decrease when additional information is provided. This idea is handled through a KLD constraint in the latent space. With this work, we showed that generative models could be used for stochastic predictions in a Multi-view setting. The obvious drawbacks of this method are that it is heavy as it requires 5 networks trained by combining two adversarial losses, which make it very unstable. Also, on the CelebA dataset, we can observe mismatches between input images and outputs that are comparable to those we can obtain using vanilla bidirection-GANs as an auto-encoder. This indicates that the BiGAN framework is not ideal for this task. Subsequent works in adversarial multi-view modeling have proposed lighter models instead (Sun et al. 2018; Xuan et al. 2018) and obtained significant improvement.

In this work, we also started to explore stochastic prediction from a stream of data. The natural next step was to tackle video data, and especially video prediction. We further investigate this theme in the next section. As the field gained in maturity, we were able to build on many recently introduced ideas to propose a model that achieve state-of-the-art for video prediction (Franceschi et al. 2020).

3.2 Stochastic Latent Residual Video Prediction

A natural application for stochastic prediction is for video data. The task, which consists in predicting the future of a video from a few conditioning frames in a self-supervised manner, has many applications in fields such as reinforcement learning (Gregor et al. 2019) or robotics (Babaeizadeh et al. 2018). It poses two distinct challenges, that are to correctly restitute high dimensional visual data and to capture complex dynamic representations of the world. The task has received a lot of attention in the computer vision community, but most of the proposed methods are deterministic and fail to account for the inherent uncertainty in video dynamics.

The combination of these two challenges with the inherent difficulty of stochastic prediction explains why stochastic video prediction has only very recently become an accessible task for researchers to tackle. A first family of work for stochastic video prediction adopts the PixelCNN approach designed for image generation, modeling frames as sequences of pixels (Oord et al. 2016; Kalchbrenner et al. 2017). In this framework, frame prediction is decomposed as a sequence of self-supervised classification tasks that consist in estimating a single pixel value knowing all previously generated pixels. This requires careful design of complex temporal generation schemes manipulating high-dimensional data, thus inducing a very high temporal generation cost.

More compute efficient continuous models rely on VAEs for the inference of low-dimensional latent state variables. Except for Xue et al. 2016 who learns a one-frame-ahead VAE using a stochastic representation of optical flow, those approaches model sequence stochasticity by incorporating a random latent variable per frame into a deterministic Recurrent Neural Network (RNN)-based model. Babaeizadeh et al. 2018, for instance, integrates stochastic variables into the ConvLSTM (Shi et al. 2015) architecture of Finn et al. 2016.

In those works, as in a standard VAE, the stochastic variable $\boldsymbol{z}$ is sampled from an approximated posterior $q_\psi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})$ that is given both the condition frames $\boldsymbol{x}$ and the target frames $\boldsymbol{y}$ during training time, while it is sampled from a fixed prior fixed distribution $p(\boldsymbol{z})$ to predict at test time. The VAE objective matches the two distributions by minimizing the KLD. However, this often results in a large discrepancy between the behavior of $\boldsymbol{z}$ during training and testing. In our previously mentioned work that used adversarial training, we avoid this discrepancy by introducing a learned distribution $p(\boldsymbol{z}|\boldsymbol{x})$ that minimizes the KLD. In a VAE framework, this simply means using a learned prior $p_\phi(\boldsymbol{z}|\boldsymbol{x})$ instead of a fixed prior $p(\boldsymbol{z})$.

In the context of prediction for low-dimensional time-series (such as speech and handwriting), Chung et al. 2015 had proposed VRNN, a sequential VAE where the prior is learned. Their model is based on a Long-Short Term Memory (LSTM) that learns an aggregated embedding of multiple frames and of the previous realizations of the stochastic latent variable. This embedding is used to compute all three components (the prior distribution $p_\phi(\boldsymbol{z}|\boldsymbol{x})$, the posterior distribution $q_\psi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})$, and the recognition model $p_\phi(\boldsymbol{x}|\boldsymbol{z})$) for the next prediction step. Stochastic video prediction using a learned prior was popularized by Denton et al. 2018, who proposed SVG, a sequential VAE with learned prior that decoupled the dynamics of the three components by using three distinct LSTMs. Lee et al. 2020 uses the VAE with a learned prior framework and combines it with both adversarial training and the use of convLSTM architectures, resulting in SAVP, a model that produces sharper predictions at the cost of a drop in their diversity.

In this context, we propose SRVP, a novel implementation of a sequential VAE with a learned prior. The main specificity of our work is that the dynamic model

is fully latent, which allows us to better separate the dynamic component and the image synthesis component. Indeed, while most models for video prediction handles separately the temporal dynamics modeling and image restitution parts of the task, the two components are still linked by the fact that they feed their predictions back into the latent space to predict the subsequent frames. This image-autoregressive process ties the frame synthesis and temporal processes together, as the dynamics now depends on the generated frames. Being fully latent, our process doesn't need any generated frames to compute the dynamics, but instead capture the complete state of the dynamic in the latent vector. Such state-space models have been shown to be useful for reinforcement learning applications (Gregor et al. 2019), and are more interpretable than image-autoregressive models (Rubanova et al. 2019). However, they are more difficult to train and requires non-trivial inference schemes (Krishnan et al. 2017) or a careful design of the dynamic model (Karl et al. 2017). This leads the most successful state-space models to only be evaluated on small or artificial toy tasks. Our work overcomes this issue by drawing from recent findings that links residual networks to first-order schemes for ordinary differential equations (Chen et al. 2018d). We design a dynamic component that determines the temporal evolution of the system through residual updates of the latent state, conditioned on learned stochastic variables. This dynamic component effectively governs the prior latent distribution of the stochastic variable as well as the reconstruction network. This formulation allows us to implement an efficient training strategy and process in an interpretable manner complex high-dimensional data such as videos, outperforming current state-of-the-art on multiple representative datasets.

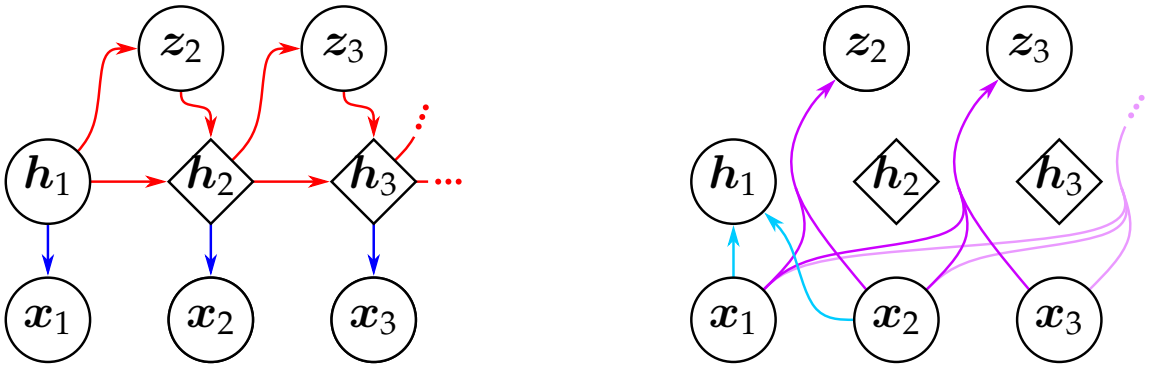
This section is organized as follows: We present our model for stochastic video prediciton in Section . The experimental setup is presented in Section , followed by results in Section .

3.2.1 Model

We consider the task of stochastic video prediction, consisting in estimating, given a number of conditioning video frames, the distribution of possible future frames given this conditioning. To this end, we propose a novel temporal latent residual model. This model has two components: one for the stochastic dynamic and the other for the static contents (such as background and shapes of objects). We describe in the following those two components before detailing the variational inference and neural architectures.

Stochastic Dynamic Model

Let $\boldsymbol{x}_{1:T}$ be a sequence of T video frames. We model their evolution by introducing latent variables $\boldsymbol{h}$ that are driven by a dynamic temporal model. Each



(a) Generative model p. (b) Inference model q.

Figure 3.9 – (a), (b) Proposed generative and inference models. Diamonds and circles represent, respectively, deterministic and stochastic states.

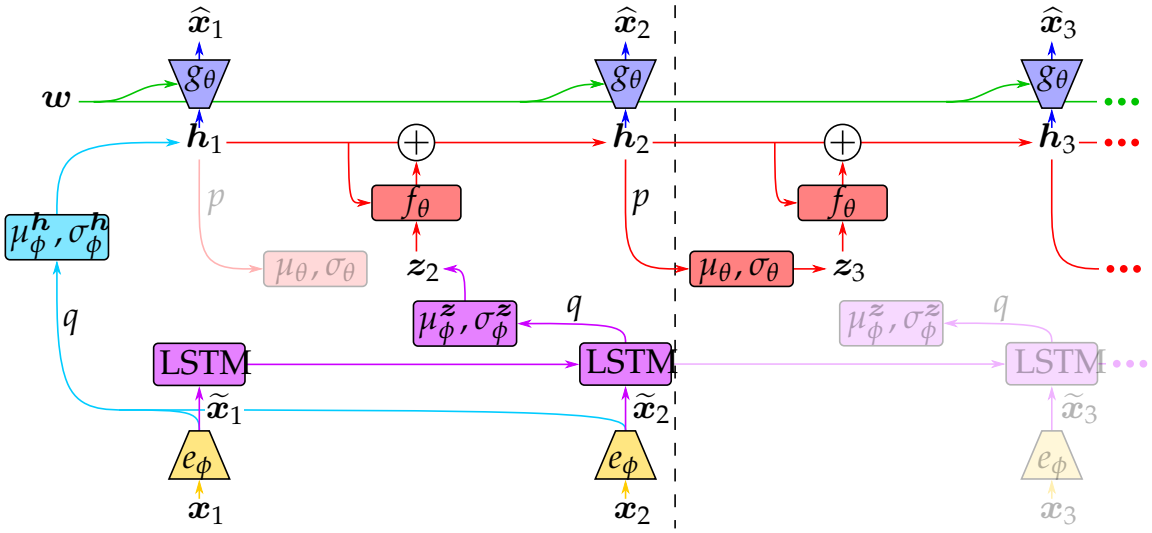


Figure 3.10 – Model and inference architecture on a test sequence. On the left, inference on conditioning frames is represented, with the transparent block representing the prior. On the right is the generation process for extrapolation, and transparent blocks correspond to the full inference performed at training time. e_ϕ and g_θ are deep CNNs, and other named networks are MLPs.

frame $\boldsymbol{x}_t$ is then generated from the corresponding latent state $\boldsymbol{h}_t$ only, making the dynamics independent from the previously generated frames.

We propose to model the transition function of the latent dynamic of $\boldsymbol{h}$ with a stochastic residual network. State $\boldsymbol{h}_{t+1}$ is chosen to deterministically depend on the previous state $\boldsymbol{h}_t$, conditionally to an auxiliary random variable $\boldsymbol{z}_{t+1}$. These auxiliary variables encapsulate the randomness of the video dynamics. They have a learned factorized Gaussian prior that depends on the previous state only. The model is depicted in Figure 3.9a and defined as follows:

$$\begin{cases} \boldsymbol{h}_1 \sim \mathcal{N}(\mathbf{0}, I), \\ \boldsymbol{z}_{t+1} \sim \mathcal{N}(\mu_\theta(\boldsymbol{h}_t), \sigma_\theta(\boldsymbol{h}_t) I), \\ \boldsymbol{z}_{t+1} \sim \mathcal{N}((\mu_\theta((\boldsymbol{h}_t), \sigma_\theta(\boldsymbol{h}_t) I), \\ \boldsymbol{h}_{t+1} = \boldsymbol{h}_t + f_\theta(\boldsymbol{h}_t, \boldsymbol{z}_{t+1}), \\ \boldsymbol{x}_t \sim \mathcal{G}(g_\theta(\boldsymbol{h}_t)), \end{cases} \tag{3.6}$$

where μ_θ, σ_θ, f_θ and g_θ are neural networks, and $\mathcal{G}(g_\theta(\boldsymbol{h}_t))$ is a probability distribution parameterized by $g_\theta(\boldsymbol{h}_t)$. In our experiments, $\mathcal{G}$ is a normal distribution with fixed diagonal variance and mean $g_\theta(\boldsymbol{h}_t)$. Note that $\boldsymbol{h}_1$ is assumed to have a standard Gaussian prior, and, in our VAE setting, will be inferred from conditioning frames for the prediction task.

The residual update rule takes inspiration in the Euler discretization scheme of differential equations, as the state of the system $\boldsymbol{h}_t$ is updated by its first-order movement, i.e., the residual $f_\theta(\boldsymbol{h}_t, \boldsymbol{z}_{t+1})$. Compared to a regular RNN, this simple principle makes our temporal model lighter and more interpretable. Note that equation 3.6, however, differs from a discretized ODE because of the introduction of the stochastic discrete-time variables $\boldsymbol{z}$. These considerations, outside our scope, are further discussed and validated in Franceschi et al. 2020.

Content Variable

Some components of video sequences can be static, such as the background or shapes of moving objects. They may not impact the dynamics; we, therefore, model them separately, in the same spirit as Denton et al. 2017 and Li et al. 2018. We compute a content variable $\boldsymbol{w}$ that remains constant throughout the whole generation process and is fed together with $\boldsymbol{h}_t$ into the frame generator. It enables the dynamical part of the model to focus only on movement, hence being lighter and more stable. Moreover, it allows us to leverage advances in neural network architectures, such as skip connections (Ronneberger et al. 2015),, to produce more realistic frames.

This content variable is a deterministic function c_ψ of a fixed number $k < T$ of frames $\boldsymbol{x}_c^{(k)}$:

$$\begin{cases} \boldsymbol{x}_c^{(k)} = \boldsymbol{x}_{i_1}, \dots, \boldsymbol{x}_{i_k}, \ \boldsymbol{w} = c_\psi(\boldsymbol{x}_c^{(k)}) \\ \boldsymbol{x}_t \sim \mathcal{G}(g_\theta(\boldsymbol{h}_t, \boldsymbol{w})). \end{cases} \tag{3.7}$$

During testing, $\boldsymbol{x}_c^{(k)}$ are the last k conditioning frames (usually between 2 and 5).

This content variable is not endowed with any probabilistic prior, contrary to the dynamic variables $\boldsymbol{h}$ and $\boldsymbol{z}$. Hence, the information it contains will not be constrained in the loss function, but only architecturally. To prevent temporal information from leaking in $\boldsymbol{w}$, we propose to uniformly sample these k frames within $\boldsymbol{x}_{1:T}$ during training. We also design c_ψ as a permutation-invariant function, which is done by using a MLP fed with the sum of individual frame representations.

This absence of prior and its architectural constraint allows $\boldsymbol{w}$ to contain as much non-temporal information as possible while preventing it from containing dynamic information. On the other hand, due to their strong standard Gaussian priors, $\boldsymbol{h}$ and $\boldsymbol{z}$ are encouraged to discard any unnecessary information. Therefore, $\boldsymbol{h}$ and $\boldsymbol{z}$ should only contain temporal information that could not be captured by $\boldsymbol{w}$.

Variational Inference and Architecture

Following the generative process depicted in Figure 3.9a, the conditional joint probability of the full model, given a content variable $\boldsymbol{w}$, can be written as:

$$p(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{2:T}, \boldsymbol{h}_{1:T} | \boldsymbol{w}) = p(\boldsymbol{h}_1) \prod_{t=2}^{T} p(\boldsymbol{z}_t, \boldsymbol{h}_t | \boldsymbol{h}_{t-1}) \prod_{t=1}^{T} p(\boldsymbol{x}_t | \boldsymbol{h}_t, \boldsymbol{w}), \tag{3.8}$$

with

$$p(\boldsymbol{z}_t, \boldsymbol{h}_t | \boldsymbol{h}_{t-1}) = p(\boldsymbol{z}_t | \boldsymbol{h}_{t-1}) p(\boldsymbol{h}_t | \boldsymbol{h}_{t-1}, \boldsymbol{z}_t). \tag{3.9}$$

According to the expression of $\boldsymbol{h}_{t+1}$ in Equation 3.6, $p(\boldsymbol{h}_t | \boldsymbol{h}_{t-1}, \boldsymbol{z}_t) = \delta(\boldsymbol{h}_t - \boldsymbol{h}_{t-1} - f_\theta(\boldsymbol{h}_{t-1}, \boldsymbol{z}_t))$, where δ is the Dirac delta function centered on $\boldsymbol{0}$. Thus, to optimize the likelihood of the observed videos $p(\boldsymbol{x}_{1:T} | \boldsymbol{w})$, we need to infer latent variables $\boldsymbol{h}_1$ and $\boldsymbol{z}_{2:T}$. This is done by deep variational inference using the inference model parameterized by ϕ and shown in Figure 3.9b, which comes down to considering a variational distribution $q_{Z,H}$ defined and factorized as follows:

$$q_{Z,H} \triangleq q(\boldsymbol{z}_{2:T}, \boldsymbol{h}_{1:T} | \boldsymbol{x}_{1:T}, \boldsymbol{w}) = q(\boldsymbol{h}_1 | \boldsymbol{x}_{1:k}) \prod_{t=2}^{T} q(\boldsymbol{z}_t | \boldsymbol{x}_{1:t}) \underbrace{q(\boldsymbol{h}_t | \boldsymbol{h}_{t-1}, \boldsymbol{z}_t)}_{= p(\boldsymbol{h}_t | \boldsymbol{h}_{t-1}, \boldsymbol{z}_t)}, \tag{3.10}$$

with $q(\boldsymbol{h}_t|\boldsymbol{h}_{t-1}, \boldsymbol{z}_t) = p(\boldsymbol{h}_t|\boldsymbol{h}_{t-1}, \boldsymbol{z}_t)$ being the aforementioned Dirac delta function. This yields the following ELBO:

$$\log p(\boldsymbol{x}_{1:T}|\boldsymbol{w}) \geq \mathcal{L}(\boldsymbol{x}_{1:T}; \boldsymbol{w}, \theta, \phi) \triangleq -D_{\mathrm{KL}}(q(\boldsymbol{h}_1|\boldsymbol{x}_{1:k})||p(\boldsymbol{h}_1)) + \mathbb{E}_{(\tilde{\boldsymbol{z}}_{2:T}, \tilde{\boldsymbol{h}}_{1:T}) \sim q_{Z,H}} \left[\sum_{t=1}^{T} \log p(\boldsymbol{x}_t|\tilde{\boldsymbol{h}}_t, \boldsymbol{w}) - \sum_{t=2}^{T} D_{\mathrm{KL}}(q(\boldsymbol{z}_t|\boldsymbol{x}_{1:t})||p(\boldsymbol{z}_t|\tilde{\boldsymbol{h}}_{t-1})) \right]. \quad (3.11)$$

The sum of KLD expectations implies to consider the full past sequence of inferred states for each time step, due to the dependence on conditionally deterministic variables $\boldsymbol{h}_{2:T}$. However, optimizing $\mathcal{L}(\boldsymbol{x}_{1:T}; \boldsymbol{w}, \theta, \phi)$ with respect to model parameters θ and variational parameters ϕ can be done efficiently by sampling a single full sequence of states from $q_{Z,H}$ per example, and computing gradients by backpropagation through all inferred variables, using the reparameterization trick. We classically choose $q(\boldsymbol{h}_1|\boldsymbol{x}_{1:k})$ and $q(\boldsymbol{z}_t|\boldsymbol{x}_{1:t})$ to be factorized Gaussians so that all KLDs can be computed analytically.

We include an ℓ_2 regularization term on residuals f_θ which stabilizes the temporal dynamics of the residual network, as noted by Behrmann et al. 2019 and Rousseau et al. 2019. Given a set of videos $\mathcal{X}$, the full optimization problem, where $\mathcal{L}$ is defined as in Equation 3.11, is then given as:

$$\arg\max_{\theta,\phi,\psi} \sum_{\boldsymbol{x} \in \mathcal{X}} \left[\mathbb{E}_{\boldsymbol{x}_{\mathrm{c}}^{(k)}} \mathcal{L}(\boldsymbol{x}_{1:T}; c_\psi(\boldsymbol{x}_{\mathrm{c}}^{(k)}), \theta, \phi) - \lambda \cdot \mathbb{E}_{(\boldsymbol{z}_{2:T}, \boldsymbol{h}_{1:T}) \sim q_{Z,H}} \sum_{t=2}^{T} ||f_\theta(\boldsymbol{h}_{t-1}, \boldsymbol{z}_t)||_2 \right]. \quad (3.12)$$

Figure 3.10 depicts the full architecture of our temporal model, corresponding to how the model is applied during testing. The first latent variables are inferred with the conditioning framed and are then predicted with the dynamic model. In contrast, during training, each frame of the input sequence is considered for inference, which is done as follows. Firstly, each frame $\boldsymbol{x}_t$ is independently encoded into a vector-valued representation $\tilde{\boldsymbol{x}}_t$, with $\tilde{\boldsymbol{x}}_t = e_\phi(\boldsymbol{x}_t)$. $\boldsymbol{h}_1$ is then inferred using a MLP on the first k encoded frames $\tilde{\boldsymbol{x}}_{1:k}$. Each $\boldsymbol{z}_t$ is inferred in a feed-forward fashion with a LSTM on the encoded frames. Inferring $\boldsymbol{z}$ this way experimentally performs better than inferring them from the whole sequence $\boldsymbol{x}_{1:T}$. We hypothesize that this follows from the fact that this filtering scheme is closer to the prediction setting, where the future is not available.

3.2.2 Experimental Setup

We evaluate our model on four metrics and four standard stochastic video prediction datasets. We present in this section, in order, the different metrics, the baseline models we compare against, the datasets of videos we use in our evaluation, and all implementation details and hyperparameters.

Metrics

The stochastic nature and novelty of the task of stochastic video prediction make it challenging to evaluate (Lee et al. 2020): since videos and models are stochastic, comparing the ground truth and a predicted video is not adequate. We thus adopt the standard approach (Denton et al. 2018; Lee et al. 2020) consisting in, for each test sequence, sampling from the tested model a given number (here, 100) of possible futures and reporting the best performing sample against the true video. We report this discrepancy for three commonly used metrics: Peak Signal-to-Noise Ratio (PSNR, *higher is better*), Structured Similarity (SSIM, *higher is better*), and Learned Perceptual Image Patch Similarity (LPIPS, *lower is better*) (Zhang et al. 2018). PSNR greatly penalizes errors in predicted dynamics, as it is a pixel-level measure derived from the ℓ_2 distance, but might also favor blurry predictions. SSIM rather compares local frame patches to circumvent this issue but loses some dynamics information. LPIPS compares images through a learned distance between activations of deep CNNs trained on image classification tasks, and have been shown to better correlate with human judgment on real images. Finally, the recently proposed Fréchet Video Distance (FVD, *lower is better*) (Unterthiner et al. 2018) aims at directly comparing the distribution of predicted videos with the ground truth distribution through the representations computed by a deep CNN trained on action recognition tasks. It has been shown, independently from LPIPS, to better capture the realism of predicted videos than PSNR and SSIM. We treat all four metrics as complementary, as they capture different scales and modalities.

Baselines

We compare our method against several state-of-the-art models in video prediction. Our baseline models are SV2P (Babaeizadeh et al. 2018), SVG (Denton et al. 2018), SAVP (Lee et al. 2020), and StructVRNN (Minderer et al. 2019). SV2P is the first to use the reparametrization trick and KLD to learn stochastic video generation models. In essence, it alters a sequential generative model proposed by Bayer et al. 2014 and adapts it for image processing by replacing the RNN with ConvLSTMs (Shi et al. 2015) and adding a specific masking module for handling static backgrounds. The other works adopt the VAE with a learned prior in different implementations: SVG decouple the dynamics of each component (prior, posterior and decoding) by using three LSTMs, SAVP combines VRNN with an adversarial loss and a ConvLSTM, and StructVRNN applies a transformation from pixel space to the much smaller key-points space in which they train the VRNN Chung et al. 2015 dynamic model. Our own approach aims to learn better dynamics by having the prior and the prediction models share a single residual dynamic model.

Datasets

Stochastic Moving MNIST (SM-MNIST). This dataset consists in one or two MNIST digits (LeCun et al. 1998) of size 28×28 moving linearly within a 64×64 frame and randomly bounce against its border, sampling a new direction and velocity at each bounce (Denton et al. 2018). We use the same settings as Denton et al. 2018, train all models on 15 timesteps, condition them at test time on 5 frames, and predict at 20 frames. Note that we adapted the dataset to sample more coherent bounces: the original dataset computes digit trajectories that are dependent on the chosen framerate, unlike our corrected version of the dataset. We consequently retrained SVG on this dataset, obtaining comparable results as those originally presented by Denton et al. 2018. Test data were produced by generating two trajectories for each digit, and pairwise combining these trajectories to produce 10 000 testing sequences that each contains two digits.

KTH Action dataset (KTH). This dataset is composed of real-world 64×64 videos of 25 people performing one of six actions (walking, jogging, running, boxing, handwaving and, handclapping) in front of different backgrounds (Schüldt et al. 2004). Uncertainty lies in the appearance of subjects, the action they perform and how they are performed. We use the same settings as Denton et al. 2018, train all models on 20 timesteps, condition them at test time on 10 frames, and predict 30 frames. The training set is formed with actions from the first 20 subjects, the remaining five being used for testing. Training is performed by sampling sub-sequences of size 20 in the train set. The test set is composed of 1000 randomly sampled sub-sequences of size 40.

Human3.6M. This dataset is also made of videos of subjects performing various actions (Ionescu et al. 2011; Ionescu et al. 2014). While there are more actions and details to capture with less training subjects than in KTH, the video backgrounds are less varied, and subjects always remain within the frames. We use the same settings as Minderer et al. 2019, train all models on 16 timesteps, condition them at test time on 8 frames, and predict 45 frames. Videos used in our experiment are subsampled from the original videos at 6.25Hz, center-cropped from 1000×1000 to 800×800 and resized using the Lanczos of the Pillow library filter to 64×64. The training set is composed of videos of subjects 1, 5, 6, 7, and 8, and the testing set is made from subjects 9 and 11; videos showing more than one action, marked by "ALL" in the dataset, are excluded. The test set is composed of 1000 randomly sampled sub-sequences of size 40 from the testing videos.

BAIR robot pushing dataset (BAIR). This dataset contains 64×64 videos of a Sawyer robotic arm pushing objects on a tabletop (Ebert et al. 2017). It is highly stochastic as the arm can change its direction at any moment. We use the same

settings as Denton et al. 2018, train all models on 12 timesteps, condition them at test time on 2 frames, and predict 28 frames. Training and testing sets are the same as those used by Denton et al. 2018.

Implementation Details

Framework and material. We used Python 3.7.6 and PyTorch 1.4.0 (Paszke et al. 2019) to implement our model. Each model was trained on a Nvidia GPUs with CUDA 10.1 in mixed-precision training using Apex. [1]

Encoder and decoder architecture. Both g_θ and e_ϕ are chosen to have different architectures depending on the dataset. We used the same architectures as in Denton et al. 2018: a DCGAN discriminator and generator architecture (Radford et al. 2016) for Moving MNIST, and a VGG16 (Simonyan et al. 2015) architecture (mirrored for e_ϕ) for BAIR and KTH. In both cases, the output of e_ϕ (i.e., $\widetilde{\boldsymbol{x}}$) is a vector of size 128, and g_θ and e_ϕ weights are initialized using a centered normal distribution with a standard deviation of 0.02.

For the Moving MNIST dataset, the content variable $\boldsymbol{w}$ is obtained directly from $\widetilde{\boldsymbol{x}}$ and is thus a vector of size 128. For KTH, Human3.6M, and BAIR, we supplement this vectorial variable with skip connections from all layers of the encoder g_θ that are then fed to the decoder e_ϕ to handle complex backgrounds. For Moving MNIST, the number of frames k used to compute the content variable is 5; for KTH and Human3.6M, it is 3; for BAIR, it is 2.

LSTM architecture. The LSTM used for all datasets has a single layer of LSTM cells with a hidden state size of 256.

MLP architecture. All MLPs used in inference (with parameters ϕ) have three linear layers with hidden size 256 and ReLU activations. All MLPs used in the forward model (with parameters θ) have four linear layers with hidden size 512 and ReLU activations. Weights of f_θ, in particular, are orthogonally initialized with a gain of 1.2 for KTH and Human3.6M, and 1.41 for the other datasets, while the other MLPs are initialized with default weight initialization of PyTorch.

Sizes of latent variables. The sizes of the latent variables in our model are the following: for Moving MNIST, $\boldsymbol{h}$ and $\boldsymbol{z}$ have size 20; for KTH, Human3.6M, and BAIR, $\boldsymbol{h}$ and $\boldsymbol{z}$ have size 50.

Optimization scheme. All models are trained using the Adam optimizer (Kingma et al. 2015) with learning rate 3×10^{-4} and regularization coefficient $\lambda = 1$. The

1. https://github.com/nvidia/apex.

batch size is chosen to be 128 for Moving MNIST, 100 for KTH and Human3.6M, and 192 for BAIR.

For the Moving MNIST dataset specifically, we follow Higgins et al. 2017 and reweight the KLD term (last term in Equation 3.11) by multiplying it by a factor of 2.

Fixed variance of the observation. The variance ν used in the observation probability distribution $\mathcal{G}(g_\theta(\boldsymbol{h})) = \mathcal{N}(g_\theta(\boldsymbol{h}), \nu I)$ is chosen as follows:

- for Moving MNIST, $\nu = 1$;
- for KTH and Human3.6M, $\nu = 4 \times 10^{-2}$;
- for BAIR, $\nu = \frac{1}{2}$.

Number of optimization steps. The number of optimization steps is the following for the different datasets:

- Stochastic Moving MNIST: 1 000 000 steps with additional 100 000 steps where the learning rate is linearly decreased to 0;
- KTH: 150 000 steps with additional 50 000 steps where the learning rate is linearly decreased to 0, the final model being chosen among several checkpoints as the one having the best evaluation PSNR (which differs from the test score as we extract from the train set an evaluation set);
- Human3.6M: 325 000 steps with additional 25 000 steps where the learning rate is linearly decreased to 0, the final model being chosen as for KTH;
- BAIR: 1 000 000 steps, with additional 500 000 steps where the learning rate is linearly decreased to 0.

The evaluation sets of KTH and Human3.6M are chosen by randomly selecting 5% of the training videos from the training set.

3.2.3 Results

This section exposes the experimental results of our method on four standard stochastic video prediction datasets. We compare our method with state-of-the-art baselines on stochastic video prediction. Furthermore, we qualitatively study the dynamics and latent space learned by our model. Animated video samples are available at https://sites.google.com/view/srvp/.

All baseline results are presented only on the datasets on which they were tested in the original articles. They were obtained with pre-trained models released by

Table 3.2 – Evaluation scores for our model and baselines on the KTH, Human3.6M and BAIR datasets with their 95%-confidence intervals over five different samples from the models. Bold scores indicate the best performing method and, where appropriate, scores whose means lie in the confidence interval of the best performing method.

Dataset	Models	FVD	PSNR	SSIM	LPIPS
stochastic MMNIST	SVG	—	14.45 ± 0.06	0.7070 ± 0.0021	—
	Ours	—	$\mathbf{16.90 \pm 0.09}$	$\mathbf{0.7789 \pm 0.0025}$	—
BAIR	SV2P	965 ± 17	$\mathbf{20.39 \pm 0.42}$	$\mathbf{0.8169 \pm 0.0110}$	0.0912 ± 0.0063
	SAVP	$\mathbf{152 \pm 9}$	18.44 ± 0.40	0.7886 ± 0.0117	$\mathbf{0.0634 \pm 0.0048}$
	SVG	255 ± 4	18.95 ± 0.41	0.8057 ± 0.0116	$\mathbf{0.0609 \pm 0.0046}$
	Ours	163 ± 4	19.59 ± 0.46	$\mathbf{0.8196 \pm 0.0110}$	$\mathbf{0.0574 \pm 0.0046}$
KTH	SV2P	636 ± 1	28.18 ± 0.39	0.8141 ± 0.0068	0.2049 ± 0.0080
	SAVP	374 ± 3	26.51 ± 0.36	0.7560 ± 0.0083	0.1120 ± 0.0058
	SVG	377 ± 6	28.06 ± 0.34	0.8438 ± 0.0067	0.0923 ± 0.0057
	Ours	$\mathbf{222 \pm 3}$	$\mathbf{29.69 \pm 0.37}$	$\mathbf{0.8697 \pm 0.0057}$	$\mathbf{0.0736 \pm 0.0036}$
Human 3.6M	StructVRNN	556 ± 9	24.46 ± 0.22	0.8868 ± 0.0031	0.0557 ± 0.0019
	Ours	$\mathbf{416 \pm 5}$	$\mathbf{25.30 \pm 0.25}$	$\mathbf{0.9074 \pm 0.0028}$	$\mathbf{0.0509 \pm 0.0019}$

the authors, except those of StructVRNN for which we trained a model using the code and hyperparameters provided by the authors.

Comparisons with Baselines

Evaluation results of our model and baseline are provided in Table 3.2. Overall, our model outperforms comparable methods in the literature on tested datasets.

On the stochastic Moving MNIST dataset, our model outperforms SVG on both PSNR and SSIM; LPIPS and FVD are not reported as they are not relevant for this synthetic task. Decoupling dynamics from image synthesis allows our method to

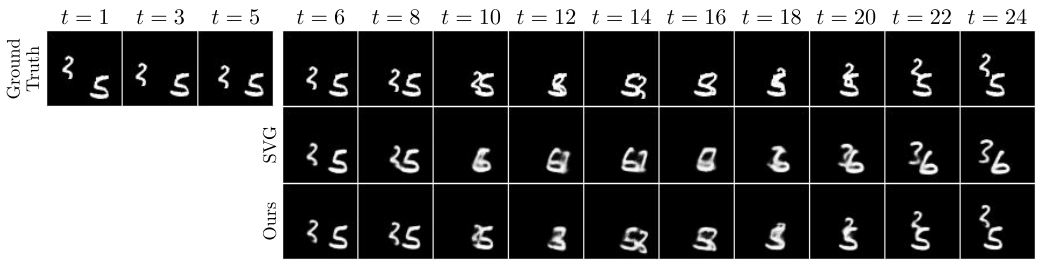


Figure 3.11 – Conditioning frames and corresponding ground truth and best samples with respect to PSNR from SVG and our method for an example of the SM-MNIST dataset.

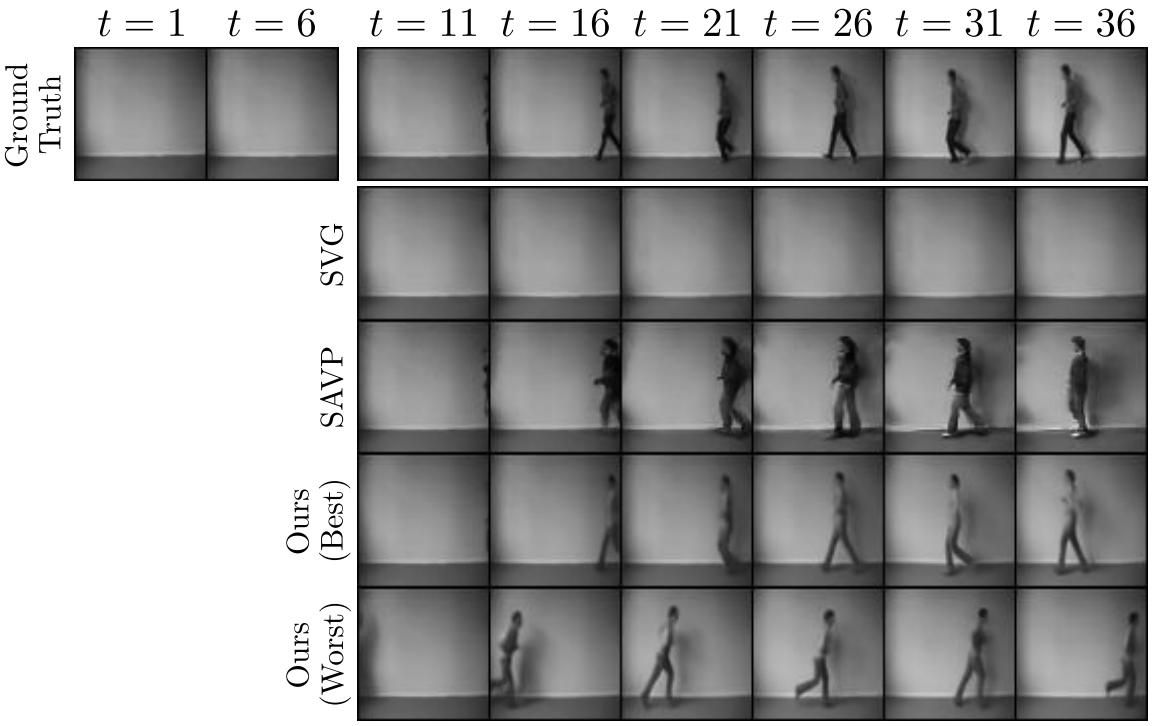


Figure 3.12 – Conditioning frames and corresponding ground truth, best samples from SVG, SAVP and our method, and worst sample from our method, for a video of the KTH dataset. Samples are chosen according to their LPIPS with respect to the ground truth. SVG fails to make a person appear unlike SAVP and our model, and the latter better predicts the subject pose.

maintain temporal consistency despite high-uncertainty frames where crossing digits become indistinguishable. For instance in Figure 3.11, the digits shape changes after they cross in the SVG prediction, while our model predicts the correct digits.

On the KTH Action dataset, we substantially outperform every considered baseline for each metric. In some videos, the subject only appears after the conditioning frames, requiring the model to sample the moment and location of the subject appearance, as well as its action. This critical case is illustrated in Figure 3.12. There, SVG fails to even generate a moving person; only SAVP and our model manage to do so, and our best sample is closer to the subject's poses compared to SAVP. Moreover, the worst sample of our model demonstrates that it captures the diversity of the dataset by making a person appear at different time steps and different speeds.

On the Human3.6M dataset, we significantly outperform the state-of-the-art StructVRNN model on all metrics. Figure 3.13 shows the dataset challenges; in particular, both methods do not capture well the subject's appearance. Nonetheless, our model better captures its movements and produces more realistic frames.

Figure 3.13 – Conditioning frames and corresponding ground truth, best samples from StructVRNN and our method, and worst and random samples from our method, with respect to LPIPS, for a video of the Human3.6M dataset. Our method better captures the dynamic of the subject and produces fewer artifacts than in StructVRNN predictions.

On the BAIR robot pushing dataset, our method achieves similar or better results compared to state-of-the-art models in terms of PSNR, SSIM and LPIPS, except for SV2P that produces very blurry samples, as seen in Figures 3.27-3.29, yielding good PSNR but prohibitive LPIPS scores. Our method's FVD score is competitive with SAVP whose adversarial loss enables it to better model small objects and outperforms SVG, whose variational architecture is closest to ours, demonstrating the advantage of non-autoregressive methods.

We provide more qualitative comparisons in Figures 3.16-3.29. In particular, Figure 3.18 shows on stochastic MMNIST dataset SVG changing a digit shape in the course of a prediction even though it does not cross another digit, whereas ours maintain the digit shape. We assume that this advantage of ours comes from the latent nature of the dynamic of our model and the use in our of a static content variable that is prevented from containing temporal information. Indeed, even when the best sample from our model is not close to the ground truth of the dataset, like in Figure 3.19, the shapes of the digits are still maintained by our model.

Interpolation of Dynamics

Our state-space structure allows us to learn semantic representations in $\boldsymbol{h}_t$. To highlight this feature, we test whether two Moving MNIST trajectories can be interpolated by linearly interpolating their inferred latent initial conditions. We begin by generating two trajectories $\boldsymbol{x}^{\mathrm{s}}$ and $\boldsymbol{x}^{\mathrm{t}}$ of a single moving digit. We infer their respective latent initial conditions $\boldsymbol{h}_1^{\mathrm{s}}$ and $\boldsymbol{h}_1^{\mathrm{t}}$. We then use our model to gen-

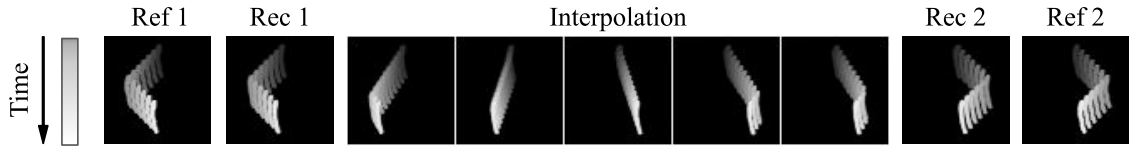


Figure 3.14 – From left to right, $\boldsymbol{x}^s$, $\hat{\boldsymbol{x}}^s$ (reconstruction of $\boldsymbol{x}^s$ by the VAE of our model), results of the interpolation in the latent space between $\boldsymbol{x}^s$ and $\boldsymbol{x}^t$, $\hat{\boldsymbol{x}}^t$ and $\boldsymbol{x}^t$. Each trajectory is materialized in shades of grey in the frames.

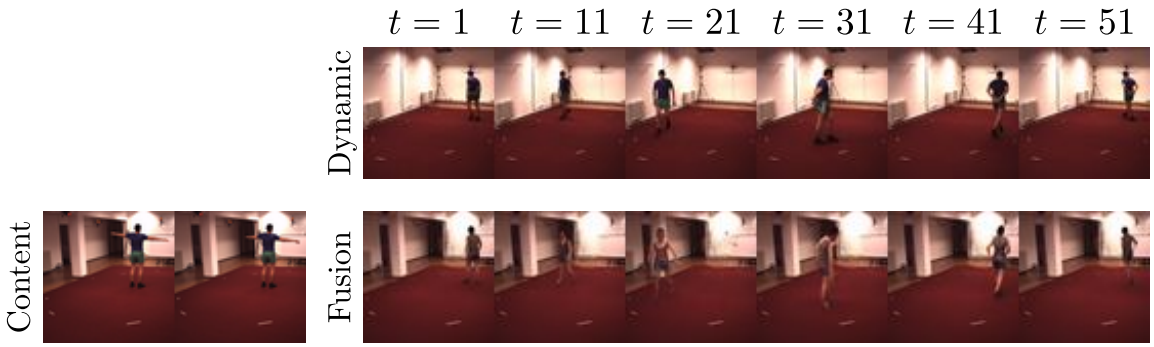


Figure 3.15 – Video (bottom right) generated from the dynamic latent state $\boldsymbol{h}$ inferred with a video (top) and the content variable $\boldsymbol{w}$ computed with the conditioning frames of another video (bottom left). The generated video keeps the same background as the bottom left frames, while the subject moves accordingly to the top frames.

erate frame sequences from latent initial conditions linearly interpolated between $\boldsymbol{h}_1^s$ and $\boldsymbol{h}_1^t$. If it learned a meaningful latent space, the resulting trajectories should also be smooth interpolations between the directions of reference trajectories $\boldsymbol{x}^s$ and $\boldsymbol{x}^t$, and this is what we observe in Figure 3.14. Additional examples are presented in Figures 3.36 and 3.37.

Disentangling Dynamics and Content

As discussed in Section 3.2.1, our model is supposed to separate content from dynamics. We validate this behavior with an additional set of experiments. To this end, two sequences $\boldsymbol{x}^s$ and $\boldsymbol{x}^t$ are drawn from the Human3.6M test set. While $\boldsymbol{x}^s$ is used for extracting our content variable $\boldsymbol{w}^s$, dynamic states $\boldsymbol{h}^t$ are inferred with our model from $\boldsymbol{x}^t$. New frame sequences $\hat{\boldsymbol{x}}$ are finally generated from the fusion of the content vector and the dynamics. This should result in a content corresponding to the first sequence $\boldsymbol{x}^s$ while moving according to the dynamics of the second sequence $\boldsymbol{x}^t$. Such samples are shown in Figure 3.15. More samples for KTH, Human3.6M, and BAIR can be seen in Figures 3.30-3.35.

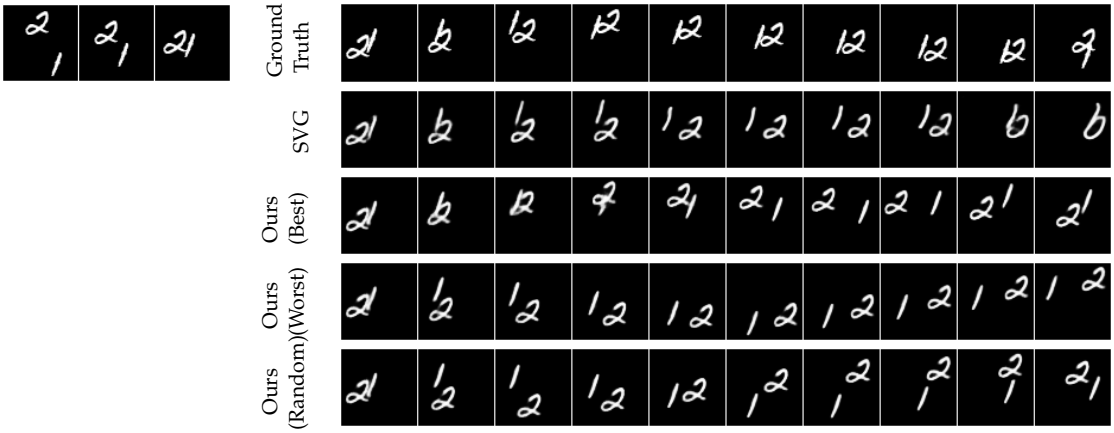


Figure 3.16 – Conditioning frames and corresponding ground truth and best samples with respect to PSNR from SVG and our method, and worst and random samples from our method, for an example of the SM-MNIST dataset.

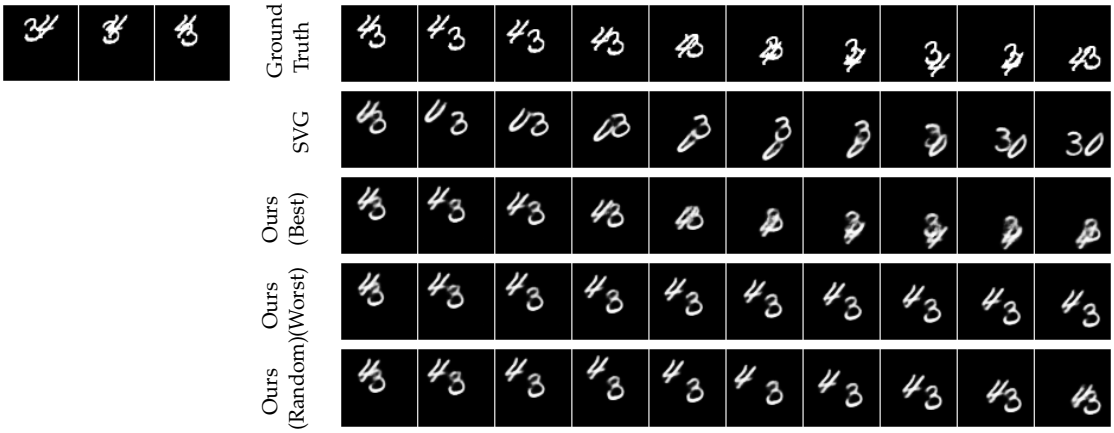


Figure 3.17 – Additional samples for the SM-MNIST dataset (cf. Figure 3.16).

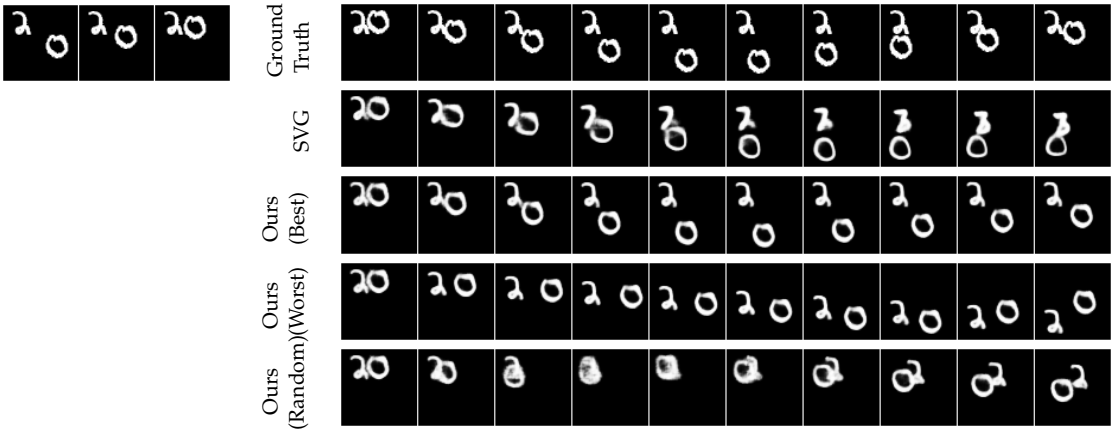


Figure 3.18 – Additional samples for the SM-MNIST dataset (cf. Figure 3.16). SVG fails to maintain the shape of a digit, while ours is temporally coherent.

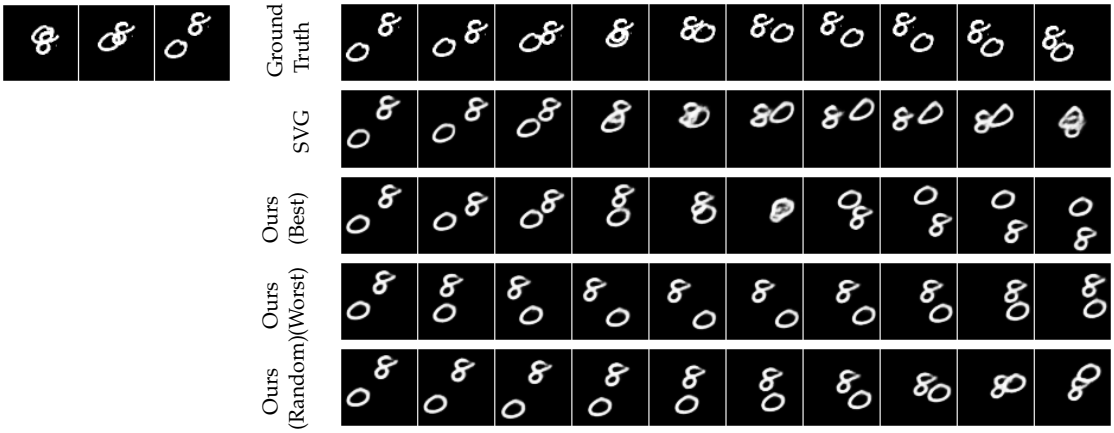


Figure 3.19 – Additional samples for the SM-MNIST dataset (cf. Figure 3.16). This example was chosen in the worst 1% test examples of our model with respect to PSNR. Despite this criterion, our model maintains temporal consistency as digits are not deformed in the course of the video.

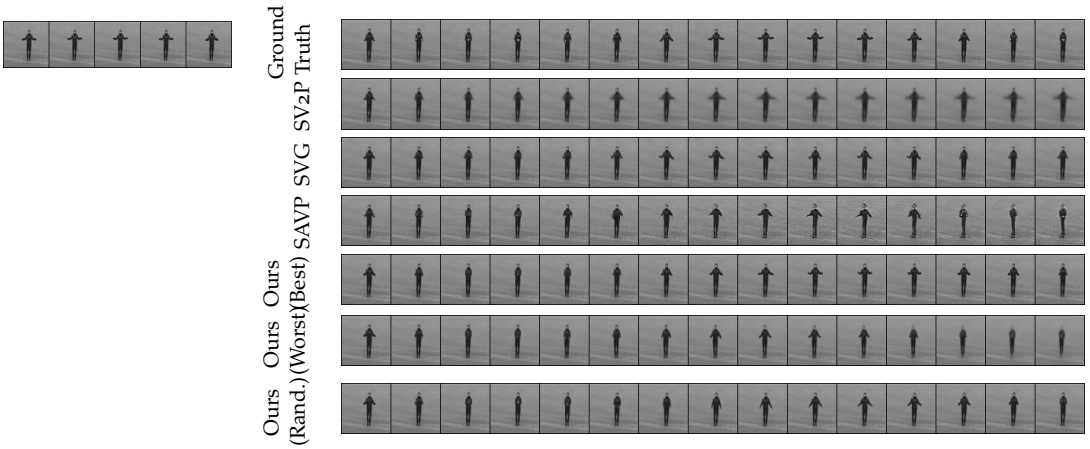


Figure 3.20 – Conditioning frames and corresponding ground truth, best samples from SVG, SAVP and our method, and worst and random samples from our method, for an example of the KTH dataset. Samples are chosen according to their LPIPS with respect to the ground truth. On this specific task (clapping), all methods but SV2P (which produce blurry predictions) perform well, even though ours stays closer to the ground truth.

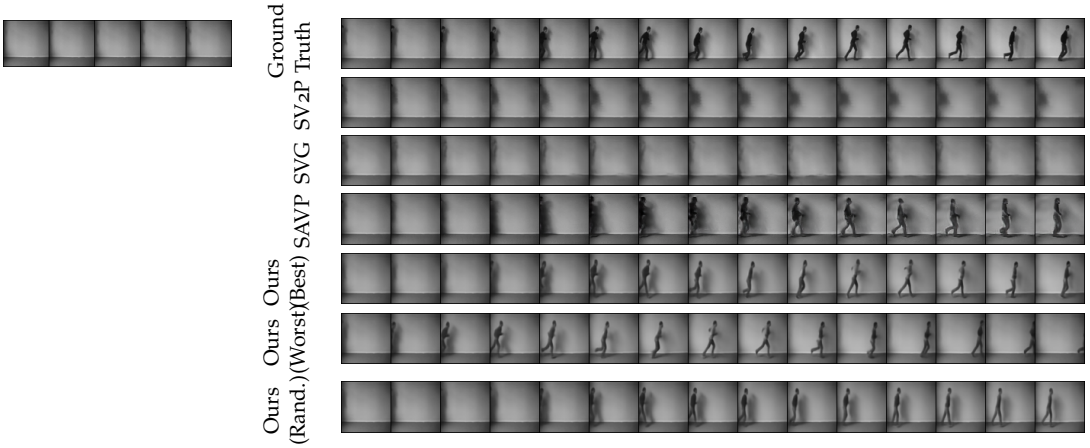


Figure 3.21 – Additional samples for the KTH dataset (cf. Figure 3.20). In this example, the shadow of the subject is visible in the last conditioning frames, foreshadowing its appearance. This is a failure case for SVG and SAVP which only produce an indistinct shadow, whereas SAVP and our model make the subject appear. Yet, SAVP produces the wrong action and an inconsistent subject in its best sample, while ours is correct.

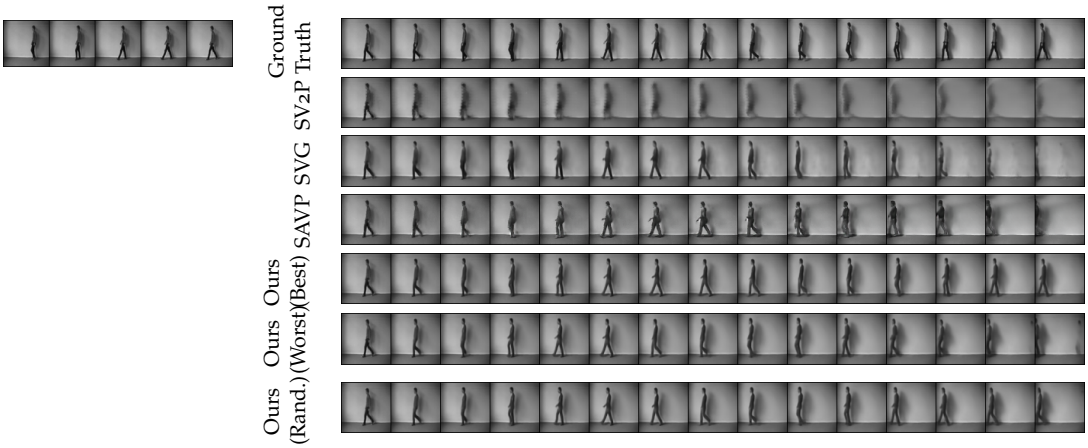


Figure 3.22 – Additional samples for the KTH dataset (cf. Figure 3.20). This example is a failure case for each method: SV2P produce blurry frames, SVG and SAVP are not consistent (change of action or subject appearance in the video), and our model produces a ghost image at the end of the prediction on the worst sample only.

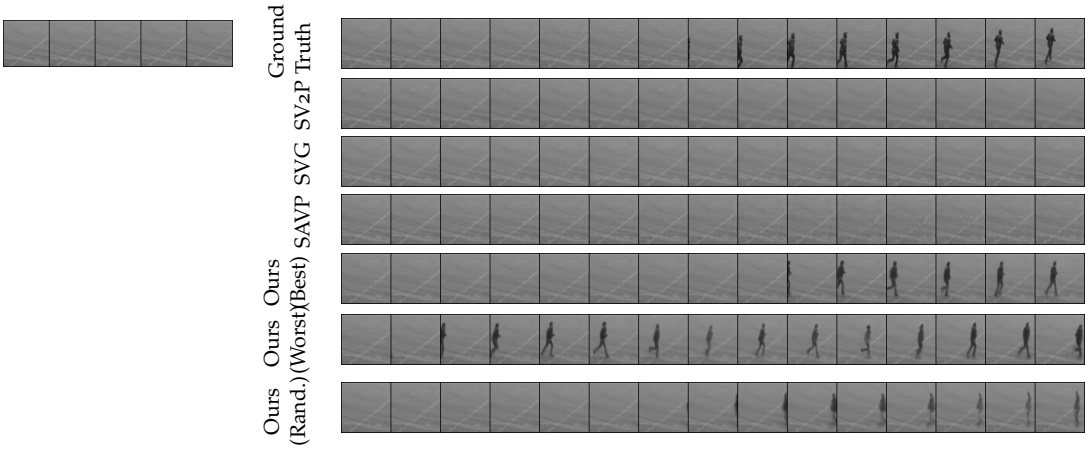


Figure 3.23 – Additional samples for the KTH dataset (cf. Figure 3.20). Our model is the only one to make a subject appear in the ground truth.

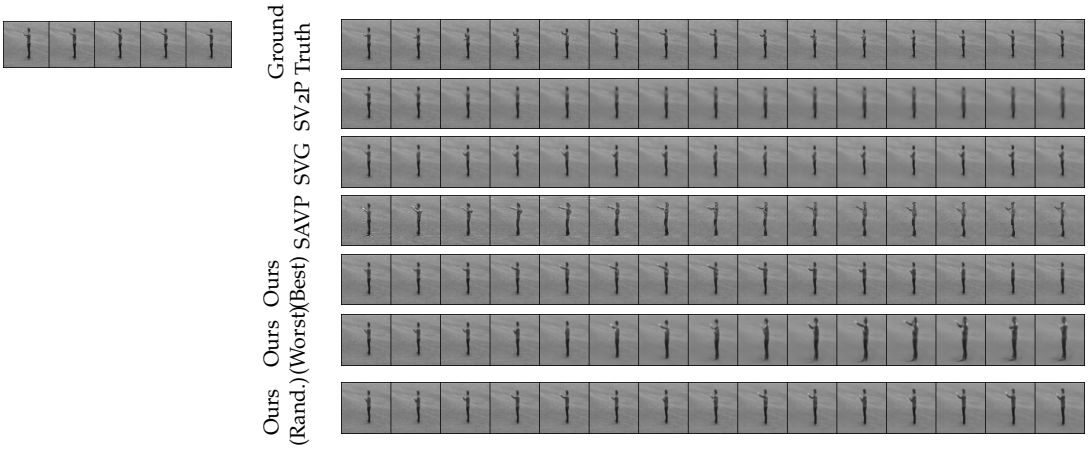


Figure 3.24 – Additional samples for the KTH dataset (cf. Figure 3.20). The subject in this example is boxing, which is a challenging action in the dataset as all methods are far from the ground truth, even though ours remain closer in this case as well.

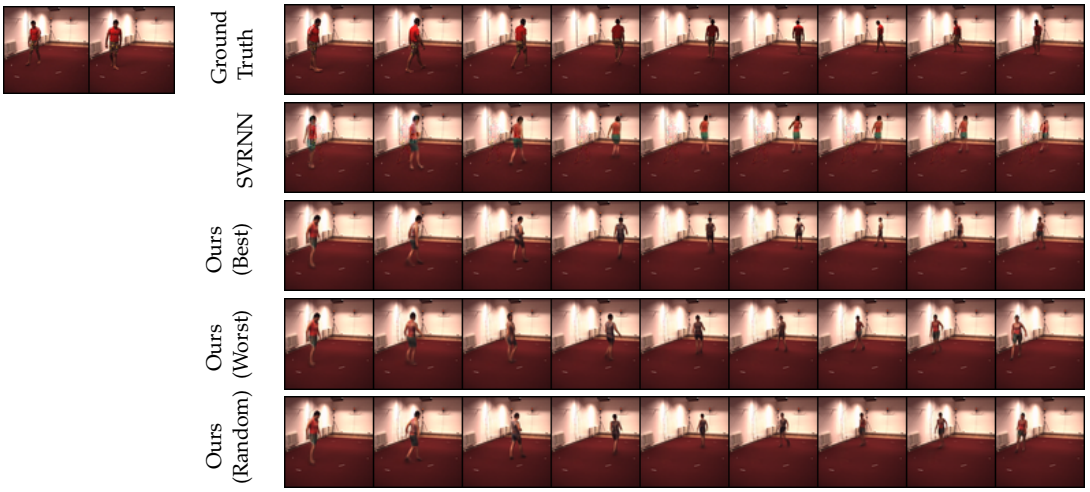


Figure 3.25 – Conditioning frames and corresponding ground truth, best samples from StructVRNN and our method, and worst and random samples from our method, for an example of the Human3.6M dataset. Samples are chosen according to their LPIPS with respect to the ground truth. We better capture the movements of the subject as well as their diversity, predict more realistic subjects, and present frames with less artefacts.

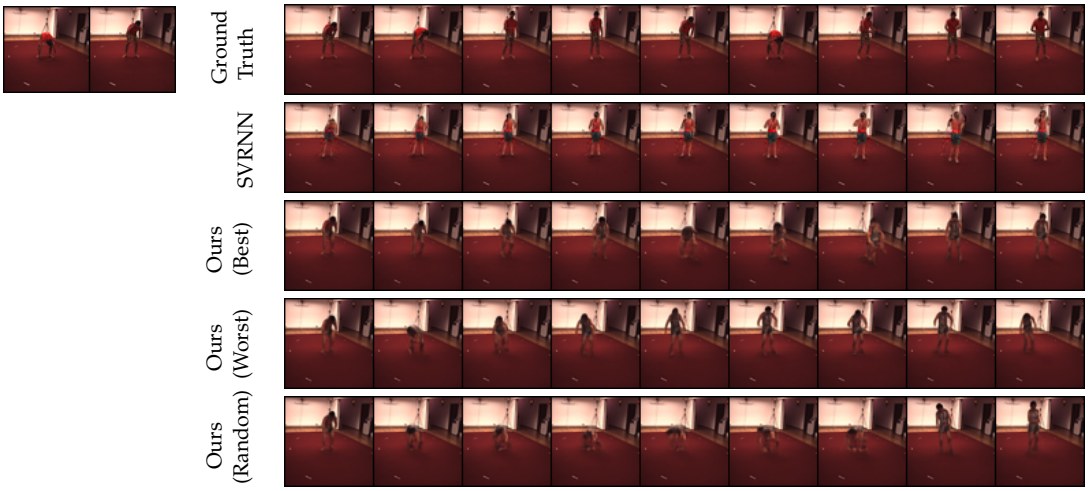


Figure 3.26 – Additional samples for the Human3.6M dataset (cf. Figure 3.25). This action is better captured by our model, which is able to produce diverse realistic predictions.

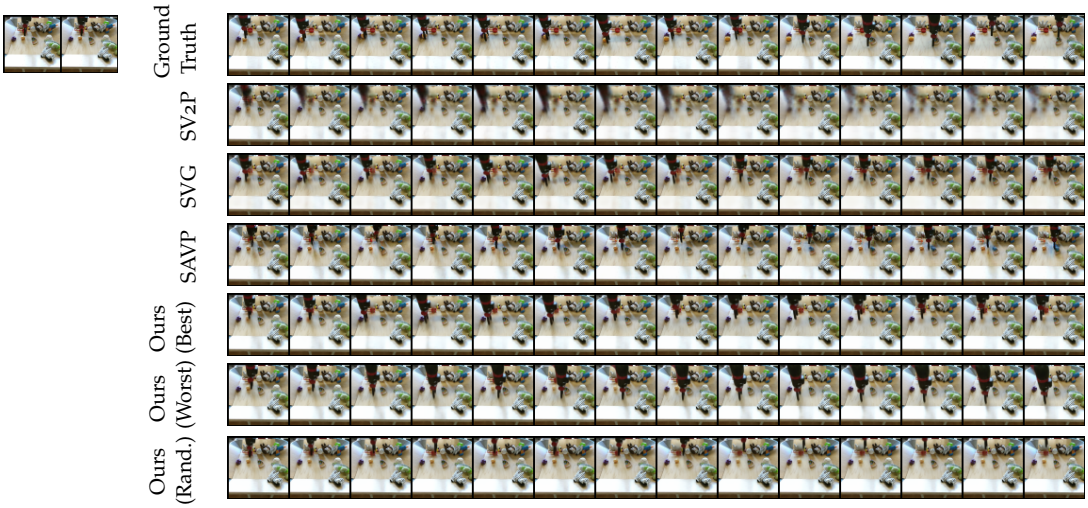


Figure 3.27 – Conditioning frames and corresponding ground truth, best samples from SVG, SAVP and our method, and worst and random samples from our method, for an example of the BAIR dataset. Samples are chosen according to their LPIPS with respect to the ground truth.

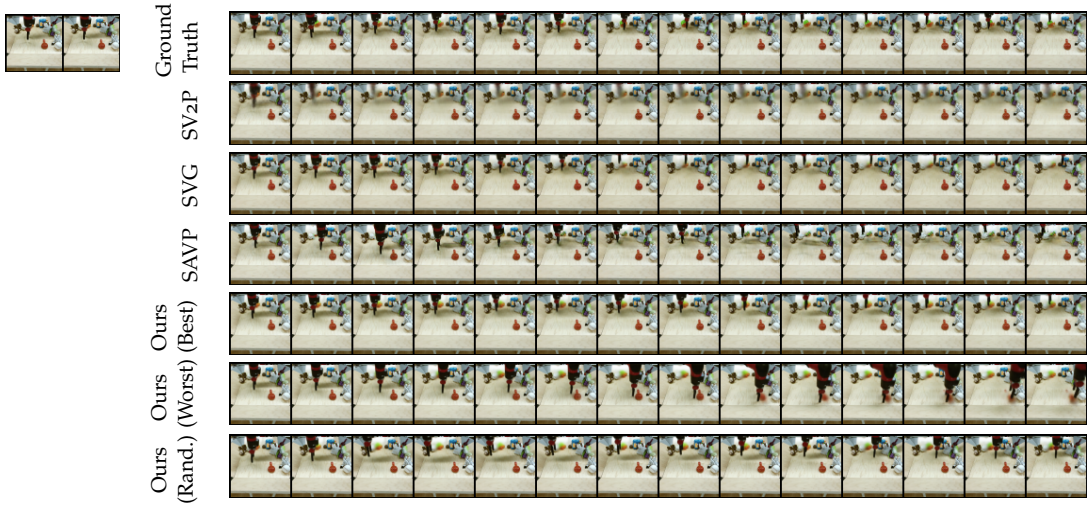


Figure 3.28 – Additional samples for the BAIR dataset (cf. Figure 3.27).

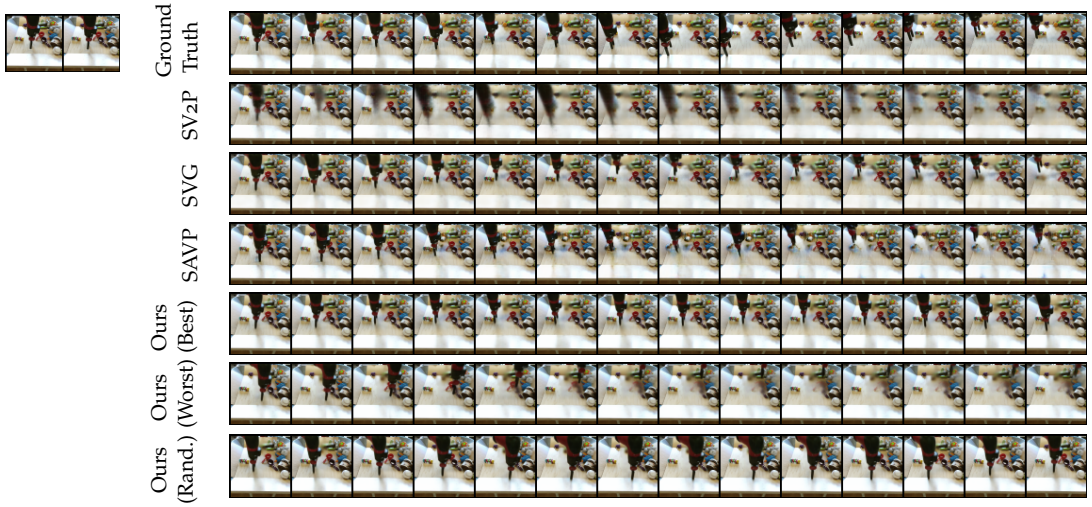


Figure 3.29 – Additional samples for the BAIR dataset (cf. Figure 3.27).

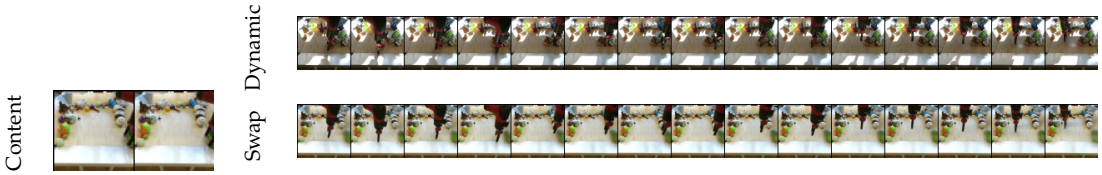


Figure 3.30 – Video (bottom right) generated from the combination of dynamic variables ($\boldsymbol{h}$, $\boldsymbol{z}$) inferred with a video (top) and the content variable ($\boldsymbol{w}$) computed with the conditioning frames of another video (bottom left).

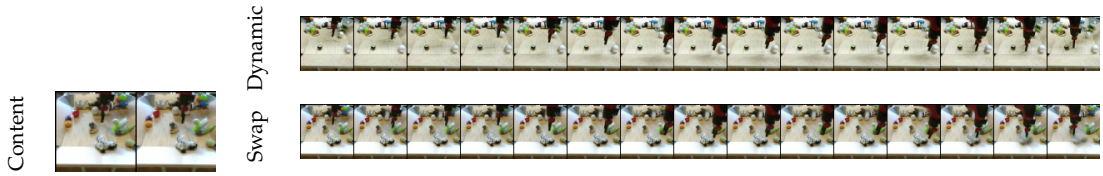


Figure 3.31 – Additional example of content swap (cf. Figure 3.30).

Figure 3.32 – Additional example of content swap (cf. Figure 3.30). In this example, the extracted content is the video background, which is successfully transferred to the target video.

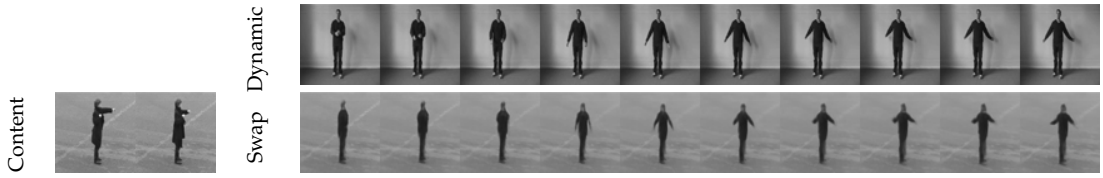


Figure 3.33 – Additional example of content swap (cf. Figure 3.30). In this example, the extracted content is the video background and the subject appearance, which are successfully transferred to the target video.

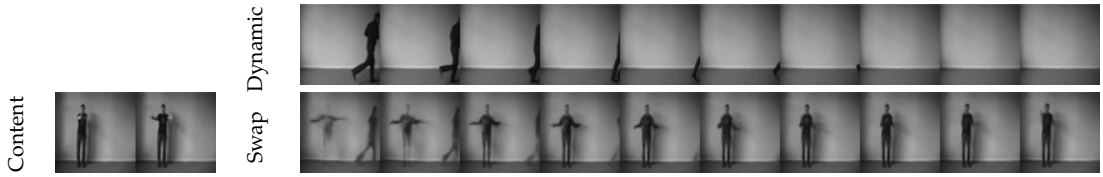


Figure 3.34 – Additional example of content swap (cf. Figure 3.30). This example shows a failure case of content swapping.

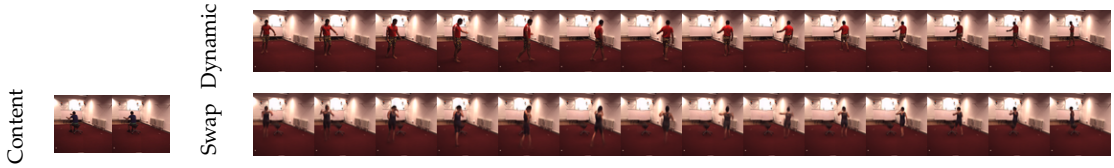
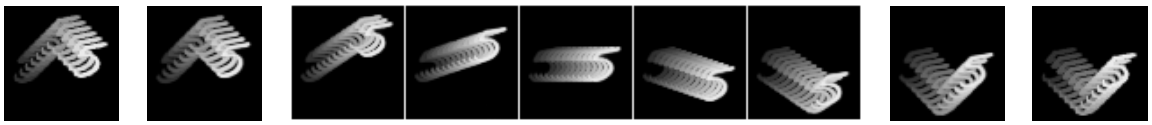
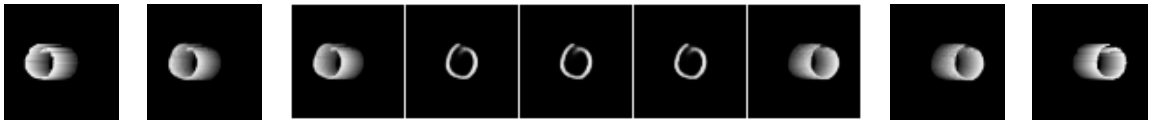


Figure 3.35 – Additional example of content swap (cf. Figure 3.30).



(a) Ref 1 (b) Rec 1 (c) Interpolation (d) Rec 2 (e) Ref 2

Figure 3.36 – From left to right, $\boldsymbol{x}^{\mathrm{s}}$, $\widehat{\boldsymbol{x}}^{\mathrm{s}}$ (reconstruction of $\boldsymbol{x}^{\mathrm{s}}$ by the VAE of our model), results of the interpolation in the latent space between $\boldsymbol{x}^{\mathrm{s}}$ and $\boldsymbol{x}^{\mathrm{t}}$, $\widehat{\boldsymbol{x}}^{\mathrm{t}}$ and $\boldsymbol{x}^{\mathrm{t}}$. Each trajectory is materialized in shades of grey in the frames.



(a) Ref 1 (b) Rec 1 (c) Interpolation (d) Rec 2 (e) Ref 2

Figure 3.37 – Additional example of interpolation in the latent space between two trajectories (cf. Figure 3.36).

3.2.4 Conclusion

We introduced a novel dynamic latent model for stochastic video prediction. Our model is based on a VAE with a learned prior, but unlike other image-autoregressive models, effectively decouples frame synthesis and stochastic dynamics using a fully latent dynamic model. We experimentally demonstrate the performances and advantages of the proposed model, which outperforms prior state-of-the-art methods for stochastic video prediction. This work is, to the best of our knowledge, the first to propose a latent dynamic model scaling for video prediction. In addition to the experiments presented in this manuscript, we have also designed experiments that aim to validate the good behavior of our update rule and exposes its ties with the neural ODE literature. Those experiments are beyond the scope of this manuscript but results can be accessed in Franceschi et al. 2020.

From another perspective, we also believe our work highlights the importance of uncovering meaningful latent spaces. Denton et al. 2017 showed how separating static content from the dynamic content was useful to learn a stochastic dynamic. Our work builds on that idea, by removing the static content from the dynamic variable and designing a more interpretable stochastic dynamic model, our model was able to exploit the good properties of the latent space and achieve state-of-the-art performances.

In the next chapter, we further explore how, in a different context, to learn and use embeddings that separate information that is commonly shared across a set of data points and factors of variations.

CHAPTER 4

ADVERSARIAL METHODS FOR LEARNING FACTORIZED REPRESENTATIONS

Contents

In this chapter, we explore different methods to learn factorized representations, that is, to learn a latent representation that is split into independent components. Such representation has recently become of interest in deep learning to build more efficient and interpretable models. Indeed, such representation would be cleaned from confounding factors, allowing to build potentially better prediction systems. Once factorized representations are learned, an immediate application is to perform style transfer between two observations, by mixing the different components extracted from those observations. A generative variant, for which the marginal distribution of each latent component is known, would also be useful for conditional synthesis. By mixing a known representation with a randomly sampled one, such a model would be able to generate new samples *in the style of* a given example.

We set our work in the context of generation using two components, and in which information about only one component is known while the other not annotated. This setting applies to many real datasets and covers, for instance, the case of a dataset labeled by class from which we would like to uncover an embedding of the factors of variations intra-class. It is, however, more challenging as it is unclear if the embeddings for the unlabeled factors that the different

models discover can be meaningful across multiple classes despite not being provided any knowledge of such alignment.

We apply our work to two different settings. In Section 4.1, we compare some of those approaches for controlled generation and style transfer in the context of motion capture sequences. In addition, we were also interested in a specific question. Many methods of disentanglement we mentioned in Chapter 2 encode a factor of variation into a single scalar value or a categorical label. We hypothesized that this might hinder the ability of the models to capture complex factors and highlight this effect via experiments. In Section 4.2, we explore controllable image synthesis in a multi-view setting with a model that factorizes style and content into latent vectors. The work presented in this chapter have led to the publication of two conference papers and a journal extension:

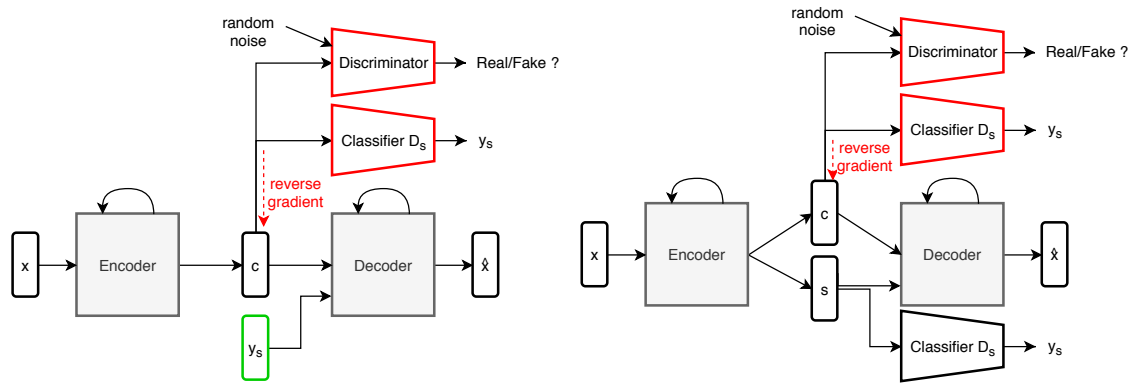
- Mickaël Chen, Ludovic Denoyer, and Thierry Artières (2018b). "Multi-View Data Generation Without View Supervision". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*

- Qi Wang, Mickaël Chen, Thierry Artières, and Ludovic Denoyer (2018b). "Transferring style in motion capture sequences with adversarial learning". In: *26th European Symposium on Artificial Neural Networks, ESANN 2018, Bruges, Belgium, April 25-27, 2018*

- Qi Wang, Thierry Artières, Mickaël Chen, and Ludovic Denoyer (2018a). "Adversarial learning for modeling human motion". In: *The Visual Computer*, pp. 1–20

4.1 Factorized Human Motion Model

Motion capture data are wildly used today in the industry, notably to create 3D animations. However, creating new animations is a time-consuming and labor-expensive process as it requires capturing motions. Generating realistic motion capture data has then become a relevant research topic. For practical purposes, such a generator also needs to grant the human designer a level of control about what kind of motion is generated. This control might take the form of an additional input to the generator, with a high-level interpretation so that these may be set by the designer (e.g., enabling a designer to ask for an animation of a man walking with joy). Another way to control the synthesis is to provide an example of what we want, and expecting the generator to produce a sequence similar, in some aspect, to the input. Fundamentally, the difference between the two setups is that, in the first one, style is considered a categorical variable while, in the second one, style is more complex and needs to be extracted from an example.



(a) ASFSEA (b) DASEA

Figure 4.1 – Architecture for our models.

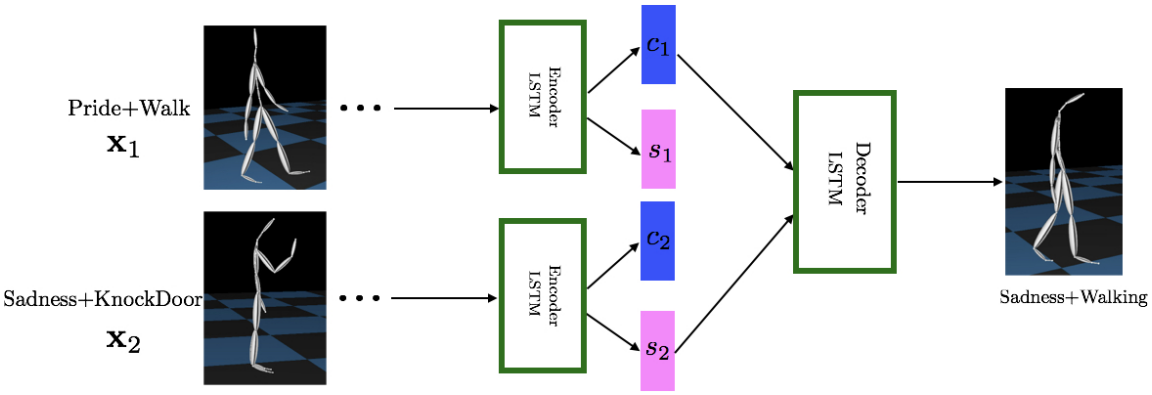


Figure 4.2 – Using DASEA for transfer.

Exploiting recently introduced adversarial learning techniques, we propose two models for controlled generation of motion capture data, one conditioned by a class label, and the other by an example observation. This section is organized as follows: Section 4.1.1 presents our proposed models. Section 4.1.2 describes our experimental setups, and results are exposed in Section 4.1.3.

4.1.1 Models

To build our generative model, we follow the approach of Adversarial Auto-Encoder (Makhzani et al. 2015) we presented in Chapter 2: using a sequential encoder and decoder plugged in a sequence-to-sequence auto-encoding fashion (Sutskever et al. 2014), we add a discriminator D_c whose task is to constrain the marginal distribution of the latent codes c to be similar to a Gaussian $\mathcal{N}(0, I)$ distribution.

From this simple generative setup, we derive two variants, one for class label conditioned generation and the other for example-based generation.

Adversarial Style-Free Sequence Auto-Encoder (ASFSEA). To provide control in the form of a style label input $\boldsymbol{y_s}$, we follow the approach of Lample et al. 2017 and censor the latent code $\boldsymbol{c}$ using an adverse classifier $D_{\boldsymbol{s}}$. $D_{\boldsymbol{s}}$ is trained to find the style label $\boldsymbol{y_s}$ from the latent code $\boldsymbol{c}$, but the objective is reversed for the encoder so that $\boldsymbol{y_s}$ cannot be retrieved from $\boldsymbol{c}$. The decoder, though, still have to reconstruct the input sequence x from the censored variable $\boldsymbol{c}$ and would be lacking style information to do so. Therefore, we provide the associated label $\boldsymbol{y_s}$ along with $\boldsymbol{c}$ to the decoder. This yields the following objective, illustrated in Figure 4.1a:

$$\begin{aligned}\min_{\text{enc,dec}} \max_{D_{\boldsymbol{c}},D_{\boldsymbol{s}}} \; & \mathcal{L}_{\boldsymbol{s}}(\boldsymbol{y_s}, D_{\boldsymbol{s}}(\text{enc}(\boldsymbol{x})))+ \\ & \mathbb{E}_{\boldsymbol{x}\sim p_{data}}\left[||\boldsymbol{x} - \text{dec}(\text{enc}(\boldsymbol{x}), \boldsymbol{y_s})||\right] + \\ & \mathbb{E}_{\boldsymbol{x}\sim p_{data}}\left[\log D_{\boldsymbol{c}}(\text{enc}(\boldsymbol{x}))\right] + \mathbb{E}_{\boldsymbol{c}\sim p_{noise}}\left[\log(1 - D_{\boldsymbol{c}}(\boldsymbol{c}))\right],\end{aligned} \tag{4.1}$$

where $\mathcal{L}_{\boldsymbol{s}}(\boldsymbol{y_s}, \hat{\boldsymbol{y}}_{\boldsymbol{s}})$ is the log-likelihood of $\boldsymbol{y_s}$ evaluated by the model with parameters $\hat{\boldsymbol{y}}_{\boldsymbol{s}}$, a classical classification objective. To generate at test time, we choose $\boldsymbol{c}$ by either extracting it from an input $\boldsymbol{x}$ or sampling it from $\mathcal{N}(0, I)$, set the desired label for $\boldsymbol{y_s}$ and use the decoder to obtain a new sequence $\text{dec}(c, y_{\boldsymbol{s}})$.

Disentangling Adversarial Sequence Auto-Encoder (DASAE). We observe that this method is limited by the fact that style is considered solely as a categorical variable, forcing any variation of style to be encoded in the supposedly censored variable $\boldsymbol{c}$. We postulate that a richer representation of style would be beneficial and our second variant is aimed at testing that hypothesis. Therefore, we design it to extract an embedding of style from an example instead of simply using the label $y_{\boldsymbol{s}}$. To do so, we simply modify the encoder to also produce a style embedding s, and feed it to the decoder instead of $y_{\boldsymbol{s}}$. To prevent s to encode information redundant with $\boldsymbol{c}$, we keep the size of the embedding very small and bias it towards storing style by using an additional classifier $C_{\boldsymbol{s}}$ to predict $y_{\boldsymbol{s}}$ using s. This yields the following objective, illustrated in Figure 4.1b:

$$\begin{aligned}\min_{\text{enc,dec},C_{\boldsymbol{s}}} \max_{D_{\boldsymbol{c}},D_{\boldsymbol{s}}} \; & \mathcal{L}_{\boldsymbol{s}}(y_{\boldsymbol{s}}, D_{\boldsymbol{s}}(\text{enc}_{\boldsymbol{c}}(\boldsymbol{x}))) - \mathcal{L}_{\boldsymbol{s}}(\boldsymbol{y_s}, C_{\boldsymbol{s}}(\text{enc}_{\boldsymbol{s}}(\boldsymbol{x})))+ \\ & \mathbb{E}_{\boldsymbol{x}\sim p_{data}}\left[||\boldsymbol{x} - \text{dec}(\text{enc}_{\boldsymbol{c},\boldsymbol{s}}(\boldsymbol{x}))||\right] + \\ & \mathbb{E}_{\boldsymbol{x}\sim p_{data}}\left[\log D_{\boldsymbol{c}}(\text{enc}_{\boldsymbol{c}}(\boldsymbol{x}))\right] + \mathbb{E}_{\boldsymbol{c}\sim p_{noise}}\left[\log(1 - D_{\boldsymbol{c}}(\boldsymbol{c}))\right],\end{aligned} \tag{4.2}$$

To generate, we choose $\boldsymbol{c}$ as in ASFSEA, and extract $\boldsymbol{s}$ from another sequence of the dataset. Then, we use the decoder to create a new sequence $\text{dec}(\boldsymbol{c}, \boldsymbol{s})$. Obtaining both $\boldsymbol{c}$ and $\boldsymbol{s}$ from different sequences of the dataset results is a process akin to style transfer (Figure 4.2).

4.1.2 Experimental setup

Dataset. We performed experiments with the Emilya Dataset (Fourati et al. 2014). It includes motion capture sequences performed by 11 actors corresponding to 8 activities performed under 8 emotions. Each actor was recorded while performing each of the activities under each of the 8 emotions. Data are captured by the inertial motion capture system Xsens and are recorded at a rate of 120 Hz. Each sequence in the dataset corresponds to a single activity which is performed by an actor under a single emotion. We select 60% of the data for the training set, 20% of the data 123 for the validation set and the remaining 20% for the test set. We split the dataset across (activity, emotion, actor) pairs to make sure that they are distributed evenly in training, validation and test set. For global translation information, we remove the horizontal position coordinates and normalize the vertical position with respect to 11 actors. We keep all the three global orientations and other joint angles and normalize them using the Max-Min method to the range (0, 1). For training our models, we cut the sequences in windows of size 200 (1.7 s approximately). In the end, we get a training set with 78,982 sequences, the validation set includes 21,614 sequences and the test set 38,637. Each of these sequences includes two hundred frames, i.e., feature vectors (one sequence is about 1.7s length). The main reason for dealing with short sequences (200 frames) is that we want to capture the short-term dynamics of motion but our architectures are not dedicated to learning from fixed-length sequences. Our modeling would learn with varying length sequences as well. Finally, note that each frame is a 69-dimensional real vector. It represents the body position at a given time through the values of the Euler angles of 23 key joints (in three dimensions).

Baselines. We compare our models to a few learned baselines. Seq2Seq (Sutskever et al. 2014) is an auto-encoder that uses a sequential model for encoding and decoding. SeqVAE (Bowman et al. 2016) is a variational extension of Seq2Seq). ASAE designates an adversarial auto-encoder with a Seq2Seq architecture. ERD (Fragkiadaki et al. 2015) uses a two-layered LSTM adapted to human motion. Note that to generate sequences with non-generative models (Seq2Seq and ERD), we suppose embeddings follow Gaussian. We first estimate, after training, the distribution of the latent codes computed from training sequences. Then, we use this distribution and the decoder part to define a generator as we did for adversarially learned models.

Ablation studies. To evaluate the effectiveness of the adverse classifier D_s in censoring the representation c, we remove this component from ASFSAE and compute the metric on this ablated model that we refer to as ASFSAE w/o D_s. Also, as our motivation for building DASAE was that a single label might not be

able to capture style appropriately, we want to evaluate the impact of different sizes for its style vector s. We test vectors of dimension 3, 5 and 8.

Evaluation metrics. We validate our model in three ways. First, we need to verify that models generates realistic sequences. Secondly, we have to evaluate the diversity and the completeness of generated sequences, (i.e., do the generated data cover the whole variety of true sequences?). Finally, we have to evaluate how disentangle the latent representation is. We present our three metrics in the following.

We compare their performances as generative models through the computation of the likelihood of test data. The higher this quantity the better the generative model. Of course, one cannot exactly compute such a likelihood and we rely on the method used by Goodfellow et al. 2014 and Denton et al. 2015 that relies on a Gaussian Parzen estimator. To compute the estimated likelihood of test data by a generative model, the method consists in first using the generative model to generate a large number of data, then using these data to fit a Gaussian Parzen estimator in order to get an estimated probability density function (PDF) on data that correspond to the generative model. Finally, this PDF is used to compute the likelihood of a separate test dataset. To exploit this idea on sequences, we consider fixed-length generated sequences that we reshape as vectors. We randomly select 10,000 synthesized sequences with a generative model and use these data to learn a Parzen estimator. Then, we compute the log-likelihood of a random subset of 10,000 test sequences under this estimator. Note that to get a generative model from the standard learning of a sequence autoencoder, we first estimate the distribution of encodings, assuming a Gaussian distribution and generate sequences by first sampling a noise vector using this estimated noise distribution then feeding this to the decoder part of the autoencoder.

To evaluate the diversity and the completeness of generated sequences, statistics are computed as follows for each generative model: We generated a set of 60 k sequences. For each generated sequence, we computed its minimum distance to a true sequence from the validation and test set ($\approx$ 60 k sequences). We report this average minimum distance as G2T criterion (Generated to True). We compute similarly T2G (True to Generated), the minimum distance from each true sequences from the dataset to the set of generated sequences.

Finally, we evaluate the disentanglement by evaluating the capacity to perform style transfer. To do so, we train an emotion classifier operating on sequences (with an architecture similar to the encoder in our models) using the training data. We compare the accuracy of this classifier for sets of data generated by each model, which should give us indications about how well the emotion has been transferred.

Model	Estimated Likelihood	G2T	T2G	Acc. Emotion Classification
Seq2Seq	1730 ± 10.5	0.6925	0.6070	—
ERD	1369 ± 123	0.4152	0.6947	—
SeqVAE	1719 ± 19	0.5724	0.5900	—
ASAE	1797 ± 4.6	0.5205	0.5205	—
ASFSEA w/o D_s	1804 ± 7.2	0.5058	0.5058	41.78%
ASFSAE	1805 ± 6.4	0.4990	0.4990	45.33%
DASAE-d3	1815 ± 11	0.5145	0.5145	48.02%
DASAE-d5	1808 ± 10	0.5225	0.5225	55.52%
DASAE-d8	1762 ± 10.4	0.5619	0.5619	67.00%

Table 4.1 – Results computing our metrics on different models. The first group of models is deterministic models, the second group is generative but not conditional, the third group contains conditional generative models. Mean and standard deviations are computed for 10 runs of the Parzen estimator. The emotion classification accuracy has to be put in perspective by the accuracy of our classifier on the real test set (82%).

Implementation details. Baseline models are implemented as in original publications when possible. For other models, we grid-searched the learning rate, the optimization strategy, the network structure, etc. The detailed structures and hyperparameters used to learn of all our models are provided in the supplementary material file of the publication associated with this work [1].

4.1.3 Results

Results are reported in Table 4.1. Qualitative results in the video format are also available [2]. Firstly, we can observe a clear advantage of adversarial models compared to the others on the three metrics that evaluate the quality of the generative model (likelihood, G2T, and T2G). Secondly, the additional information provided by conditioning with style labels does seem to be used effectively by conditional models as they improve over their unconditioned counterpart (ASAE).

As for the disentanglement evaluated through the emotion transfer task, we confirm our two hypotheses. 1) The adversarial censoring is effective, as ASFSEA w/o D_s performs noticeably worse than the other models. 2) Using a single label is not sufficient to encode for the emotion, as ASFSAE performs worse than DASAE.

1. https://link.springer.com/article/10.1007/s00371-018-1594-7
2. https://link.springer.com/article/10.1007/s00371-018-1594-7

Moreover, the higher the number of dimensions in s, the better the accuracy of the classifier, which indicates that a large vector is necessary in this case. However, we also notice a drop in generation quality as the size of s increases. This can be explained by the fact that in DASAE, nothing prevents s to contain information redundant with c except for the fact that s has a very small capacity. At test time, c and s are drawn independently, which means that they can contain contradictory information to the decoder, which in turn can generate wrong sequences.

4.1.4 Conclusion

We presented a modeling framework for building generative models of motion capture data trajectories. It is based on sequence autoencoders implemented with recurrent neural networks and on adversarial learning. Within this framework, we were able to design various models that fit different animation tasks, giving control over the animation synthesis. We proposed two variants of conditional models that allow synthesizing motion under a specific context where the context may be chosen by hand at synthesis time. Finally, we were able to design a motion sequence editing model that allows transforming an input motion sequence to match the style of a second sequence. This work both demonstrates the potential of neural architectures trained within the adversarial framework for the synthesis of realistic motion capture sequences and is a step toward highly flexible synthesis systems allowing a designer to synthesize sequences with a high level of monitoring.

From a machine learning perspective, we also validated that the usual practice of disentangling factors in a single label or dimension might limit the ability of the model to capture complex factors of variations, especially for non-trivial generation tasks. Our model, however, was limited in the capacity of its latent component as the bottleneck on the representation ensured its disentangling power. In the next section, we build on that finding and propose a different way to leverage adversarial training to split information in large representation vectors.

4.2 Multi-view Generation Without View Labels

In this section, we follow up on the idea of splitting information into different components by considering image datasets composed of different objects that each is depicted in multiple views. For instance, in a human face dataset, a particular data sample may be thought as a mix of content information (e.g. related to its class label, in this case, the identity of the person) and of side information, the view, which accounts for the, often unlabeled, factors of variability (e.g. exposure, viewpoint, wearing glasses). From this point of view, all samples of a given class share the same content information while they differ on the view information. A

number of approaches have been proposed to disentangle the content from the view, also referred to as style in some papers (Denton et al. 2017; Mathieu et al. 2016). For instance, different models have been built to extract and separate, from a photo of any object, the characteristics of the object and the camera position. Once such a disentanglement is learned, one may build various applications like predicting how an object looks like under different viewpoints (Mathieu et al. 2016; Zhao et al. 2018).

More concretely, we propose two models to tackle this particularly difficult setting: a generative model Generative Multi-View model (GMV) that generates objects under various views (multi-view generation), and a conditional extension CGMV that generates a large number of views of any input object (conditional multi-view generation). These two models are based on the adversarial training schema of GANs. The simple but strong idea is to focus on distributions over pairs of examples (e.g. images representing a same object in different views) rather than distribution on single examples.

Learning generative models using sets of latent variables to describe the objects have been tackled through methods such as inter-battery factor analysis (IBFA) (Tucker 1958; Klami et al. 2013). Those methods are very related to Canonical Correlation Analysis (CCA) and have been used to deal with multiview data to infer one view from another one and/or to improve classification systems. More recently, non-linear variants (Tang et al. 2017; Li et al. 2015) have been proposed, that rely on regularization terms and on specific factorization functions to capture these factors (e.g. Damianou et al. 2012). Our approach is based on the same assumptions as these methods i.e. each observation is generated based on one common latent factor describing the content of the object and with a view latent factor responsible for the difference between two observations. Our model makes use of a discriminator function to capture common and specific information based on a pair of observations, making it scalable to large datasets without relying on handcrafted priors.

Compared to other recent approaches for image synthesis that we discussed in Chapter 2, this strategy of factorizing into latent components is often more flexible. For instance, editing the view of a picture can be cast as attribute manipulation (Lample et al. 2017, Zhao et al. 2018 and Kim et al. 2017b) or domain transfer (Kim et al. 2017a; Liu et al. 2017; Yi et al. 2017; Zhu et al. 2017). In both cases, they assume that the number of different domains (or attributes) is very limited as additional networks need to be trained for each new domain or attribute. Uncovering continuous embeddings for each component overcomes this limitation as views don't need to be discrete or sparse. As established in the previous section, it should be beneficial to embed the components as vectors instead of reducing a complex factor to a single value. Also, all those approaches use supervision on the component we want to alter (the view), while we suppose this information is

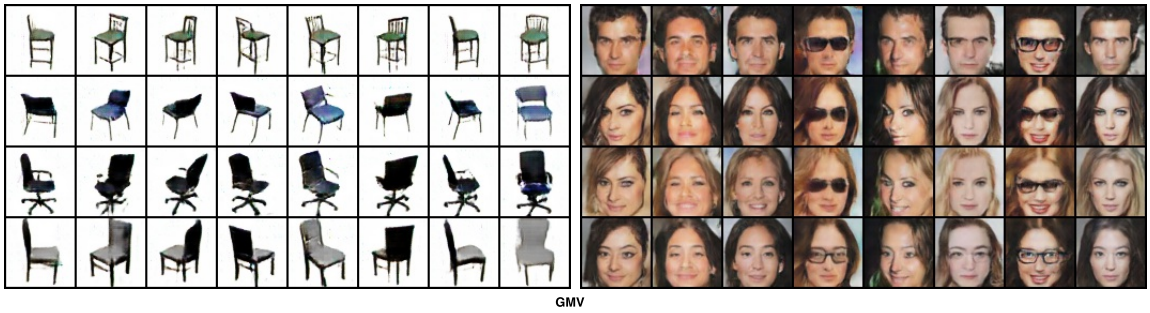


Figure 4.3 – Samples generated by our model GMV on the 3D-Chairs and the CelebA datasets. All images in a row have been generated with the same content vector, and all images in a column have been generated with the same view vector.

unavailable and only use information on the invariant component (the content), which is often easier to obtain.

Our contributions are the following: (i) We propose a new generative model able to generate data with various content and high view diversity using supervision on the content information only. (ii) We extend the model to a conditional model that allows generating new views over any input sample. (iii) We report experimental results on four different image datasets that show the ability of our models to generate realistic samples and to capture (and generate with) the diversity of views.

The section is organized as follows. Section 4.2.2 details our proposal for a generative multi-view model, Section 4.2.3 follows-up with its conditional extension, and finally, we report experimental protocols in Section 4.2.4 and results on the various generative tasks allowed by our models in Section 4.2.5. Section 4.2.6 closes the chapter.

4.2.1 Objective and Notations

We consider the multi-view setting where data samples represent an object that has been observed in various views. The distribution of the data $\boldsymbol{x} \in \mathcal{X}$ is assumed to be driven by two latent factors: a content factor, denoted $\boldsymbol{c}$, that corresponds to the invariant proprieties of the object, and a view factor, denoted $\boldsymbol{v}$, that corresponds to the variations. Typically, if $\mathcal{X}$ is the space of people's faces, $\boldsymbol{c}$ stands for the intrinsic features of a person's face while $\boldsymbol{v}$ stands for the transient features and the viewpoint of a particular photo of the face, including the photo exposure and additional elements like a hat, glasses, etc. Importantly, we assume that these two components $\boldsymbol{c}$ and $\boldsymbol{v}$ are independent.

We focus on two different tasks: (i) *Multi-View Generation*: we want to be able to sample over $\mathcal{X}$ by controlling the two factors $\boldsymbol{c}$ and $\boldsymbol{v}$. Said otherwise, we want

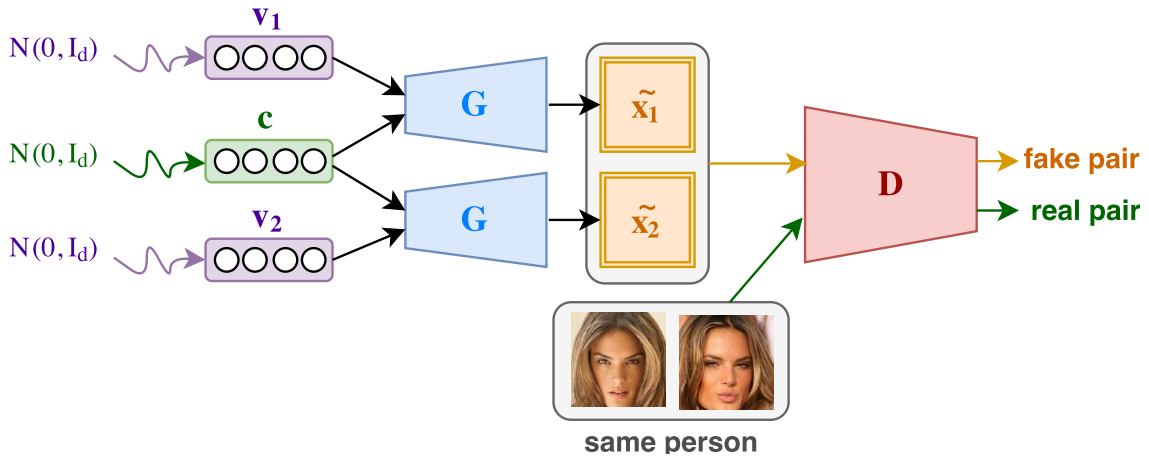


Figure 4.4 – Overview of the GMV model. The generator G produces an image given a content vector $\boldsymbol{c}$ and view vector $\boldsymbol{v}$. A pair of images is generated by sampling a common content factor $\boldsymbol{c} \sim p_c$ but two different views factors $\boldsymbol{v}_1 \sim p_{\boldsymbol{v}}$ and $\boldsymbol{v}_2 \sim p_{\boldsymbol{v}}$. The discriminator D is learned to distinguish between such pairs of generated images and real pairs of samples corresponding to the same object under different views. Real pairs are built by sampling uniformely from views of the same object in the training set. No information on the views is used.

to be able to generate different views of the same object, or the same view of different objects. Given two distributions, $p(\boldsymbol{c})$ and $p(\boldsymbol{v})$, this sampling will be possible if we are able to estimate $p(\boldsymbol{x}|\boldsymbol{c}, \boldsymbol{v})$ on a training set. (ii) *Conditionnal Multi-View Generation*: the second objective is to be able to sample different views of a given object. For instance, it can be different views of a particular person based on one of his photos. Given $p(v)$, this sampling will be achieved by learning the probability $p(c|\boldsymbol{x})$ in addition to $p(\boldsymbol{x}|c, v)$.

The key issue for tackling these two tasks lies in the ability to accurately learn generative models able to generate from a disentangled latent space where the content factor and the view factor are encoded (and thus sampled) separately. This would allow controlling the sampling on the two different axes, the content and the view. Note that we are only using content annotations and not any view information.

4.2.2 Generative Multi-view Model (GMV)

Let us consider two distributions over the content and view factors denoted as $p_{\boldsymbol{c}}$ and $p_{\boldsymbol{v}}$, these distributions typically being isotropic Gaussian distributions, and a target distribution $p(\boldsymbol{x})$ that we can simulate by sampling observations from the dataset. Our objective is to learn a generator G that satisfies two criteria. Firstly,

the distribution $p_G(x)$ of random variable $G(\boldsymbol{c}, \boldsymbol{v})$ has to approximate $p(\boldsymbol{x})$, i.e. generated samples are indistinguishable from real observations. Secondly, we want that its first input $\boldsymbol{c}$ controls the content of the generated sample while its second input v controls its view. Doing so would allow us to tune the output sample by modifying its content $\boldsymbol{c}$ or its view $\boldsymbol{v}$.

Adversarial learning is well suited for our objective but a standard GAN approach would not be able to disentangle the latent space as desired. Without constraint, the content and view factors are going to be diluted and mixed in the input noise vectors with no possibility to know which dimensions capture the content and which capture the view. Instead, we propose a novel way to achieve this desired feature. The key idea of our model is to focus on the distribution of pairs of inputs rather than on the distribution over individual samples. We explain now which pairs we are talking about and why considering pairs might be useful.

From the dataset, we build a pair (x_1, x_2) that consists of two independent samples of a given object under two different views. We also build another type of pair using the generator. We construct those by using, for each pair, a single random content vector $\boldsymbol{c} \sim p_{\boldsymbol{c}}$ shared across the pair and two view vectors $v_1, v_2 \sim p_{\boldsymbol{v}}$, one per generated sample. Note that no information on view is necessary for this approach.

Following the adversarial training of GANs, the pair (x_1, x_2) from the dataset is given to a discriminator D as a real pair, while the generated pair $(G(\boldsymbol{c}, \boldsymbol{v}_1), G(\boldsymbol{c}, \boldsymbol{v}_2))$ is provided as fake. To be able to fool the discriminator, the generator then has to achieve three goals. (i) As in a regular GAN, each sample generated by G needs to look realistic. (ii) Moreover, because real pairs are composed of two views of the same object, the generator should generate pairs of the same object. Since the two sampled view factors v_1 and v_2 are different, the only way this can be achieved is by encoding the invariant features into the content vector $\boldsymbol{c}$. (iii) Finally, it is expected that the discriminator should easily discriminate between a pair of samples corresponding to the same object under different views from a pair of samples corresponding to the same object under the same view. Because the pair shares the same content factor $\boldsymbol{c}$, this should force the generator to use the view factors v_1 and v_2 to produce diversity in the generated pair.

Compared to conditional variants of GANs that use class labels directly, this pairwise approach is likely to work better in cases in which there is a very large number of classes, or in which the model can take advantage of a smooth continuity between classes.

The GMV architecture is detailed in Figure 4.4. It is learned by optimizing the following non-saturating adversarial objective:

$$\begin{aligned}\min_D \; & -\mathbb{E}_{\boldsymbol{x}_1,\boldsymbol{x}_2\sim p(\boldsymbol{x}_1,\boldsymbol{x}_2)}\left[\log D(\boldsymbol{x}_1,\boldsymbol{x}_2)\right] \\ & -\mathbb{E}_{\boldsymbol{v}_1,\boldsymbol{v}_2\sim p(\boldsymbol{v}),c\sim p(\boldsymbol{c})}\left[\log(1-D(G(\boldsymbol{c},\boldsymbol{v}_1),G(\boldsymbol{c},\boldsymbol{v}_2)))\right] \\ \min_G \; & -\mathbb{E}_{\boldsymbol{v}_1,\boldsymbol{v}_2\sim p(\boldsymbol{v}),c\sim p(\boldsymbol{c})}\left[\log D(G(\boldsymbol{c},\boldsymbol{v}_1),G(\boldsymbol{c},\boldsymbol{v}_2))\right].\end{aligned} \tag{4.3}$$

where $p(\boldsymbol{x}_1,\boldsymbol{x}_2)$ is the distribution of pairs of the same content under two independently sampled views. Note that, since the proposed model can be seen as a particular GAN architecture over pairs of inputs, the global minimum of the learning criterion is obtained when the model is able to sample pairs of views over a similar object.

As discussed above, the training of the discriminator on pairs of samples introduce useful constraints on how the content and the view information are used to generate samples. Once the model is learned, we are left with a generator G that generates single samples by first sampling $\boldsymbol{c}$ and $\boldsymbol{v}$ following $p_{\boldsymbol{c}}$ and $p_{\boldsymbol{v}}$, then by computing $G(\boldsymbol{c},\boldsymbol{v})$. By freezing $\boldsymbol{c}$ or $\boldsymbol{v}$, one may then generate samples corresponding to multiple views of any particular content, or corresponding to many contents under a particular view. It is also possible to make interpolations between two given views over a particular content or between two contents using a particular view.

4.2.3 Conditional Generative Model (CGMV)

The GMV model allows one to sample objects with different views. However, many applications, such as image editing, would benefit from being also able to change the view of a given object that would be provided as an input to the model. More generally, the problem is to infer the content representation $\boldsymbol{c}$ from a real image, which GANs struggle with. This section aims at extending our generative model the ability to extract this content factor from any given input and to use this extracted content in order to generate new views of the corresponding object.

To achieve such a goal, we must add to our generative model an encoder function, denoted E, that will map any input in $\mathcal{X}$ to the content space (see Figure 4.5). To do so, we encode an input sample $\boldsymbol{x}$ in the content space using an encoder function (implemented again as a neural network). This encoder serves to generate a content vector $\boldsymbol{c} = \mathrm{E}(x)$ that will be combined with a randomly sampled view $v \sim p_v$ to generate an artificial example. The artificial sample is then combined with the original input $\boldsymbol{x}$ to form a negative pair. This is illustrated in the extreme right part of Figure 4.5 and is very similar to that of Conditional GAN as introduced by Mirza et al. 2014.

Yet, CGANs have severe weaknesses and are known to easily miss modes of the underlying distribution. In our case, the generator enters a state where it

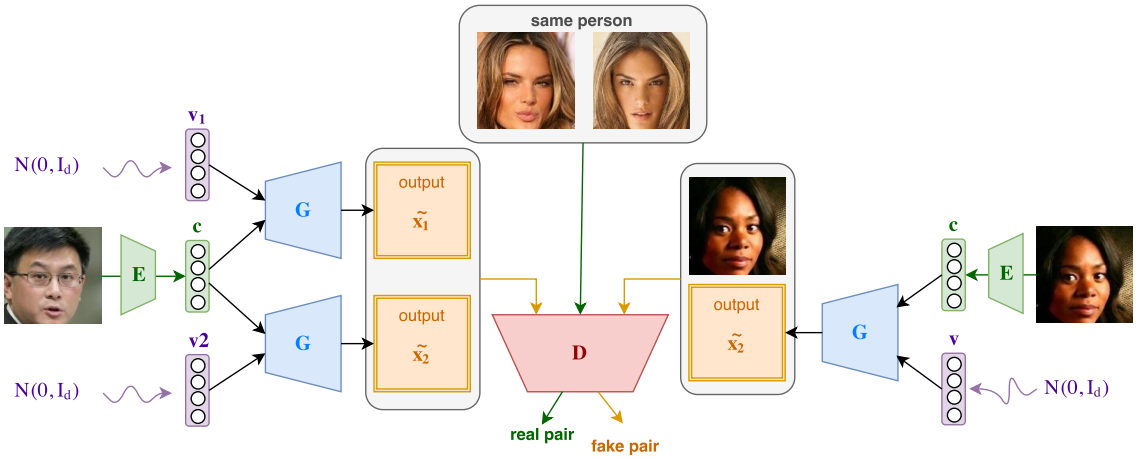


Figure 4.5 – The conditional generative model CGMV. Content vectors are not randomly sampled but are inferred from real inputs through an encoder. The discriminator is provided two types of negative examples: examples of the first type are pairs of generated samples using the same content factor but with two different views (left). The second type of negative examples is composed of pairs of a real sample $\boldsymbol{x}$ and of a generated sample built from $\boldsymbol{x}$ using a CGAN-like approach (right). This artificial sample corresponds to the same content as in the input $\boldsymbol{x}$ but under a different view.

ignores the noisy component v (see results in Figures 4.9 and 4.9). To overcome this phenomenon, we use the same idea as in GMV. We build negative pairs $(G(c, v_1), G(c, v_2))$ by randomly sampling two views v_1 and v_2 that we combine to a single content $\boldsymbol{c}$. This time, however, $\boldsymbol{c}$ is not sampled according to a noise distribution but it is computed from an observation $\boldsymbol{x}$ using the encoder E, i.e. $\boldsymbol{c} = E(\boldsymbol{x})$. By doing so, we preserve the ability of our approach to generate pairs with view diversity. Since this diversity can only be captured by taking into account the two different view vectors provided to the model (v_1 and v_2), this will encourage $G(c, v)$ to generate samples containing both the content information $\boldsymbol{c}$ and the view $\boldsymbol{v}$. As it was done for the GMV model, positive pairs are sampled from the training set and correspond to two views of a given object.

Dataset	number of samples		number of objects		views per object		
	train	test	train	test	min	mean	max
3D-Chairs	80600	5766	1300	93	62	62	62
CelebA	198791	3808	9999	178	1	19.9	35
MVC-Cloth	159128	2132	37004	495	4	4.3	7
102-Flowers	8189	—	102	—	40	80.3	258

Table 4.2 – Datasets Statistics: Train and Test data include samples corresponding to different objects. GMV and CGMV are optimized on the train set. The test set is used as inputs of CGMV and CGAN for evaluation.

In this setting, applying the non-saturating adversarial approach yields the following objective:

$$\begin{aligned}
\min_D \; & -\mathbb{E}_{\boldsymbol{x}_1,\boldsymbol{x}_2\sim p(\boldsymbol{x}_1,\boldsymbol{x}_2)}\left[\log D(\boldsymbol{x}_1,\boldsymbol{x}_2)\right] \\
& -\frac{1}{2}\mathbb{E}_{\boldsymbol{v}_1,\boldsymbol{v}_2\sim p(\boldsymbol{v}),x\sim p(\boldsymbol{x})}\left[\log(1-D(G(E(\boldsymbol{x}),\boldsymbol{v}_1),G(E(\boldsymbol{x}),\boldsymbol{v}_2)))\right] \\
& -\frac{1}{2}\mathbb{E}_{\boldsymbol{v}\sim p(\boldsymbol{v}),x\sim p(\boldsymbol{x})}\left[\log(1-D(\boldsymbol{x},G(E(\boldsymbol{x}),\boldsymbol{v})))\right] \\
\min_G \; & -\frac{1}{2}\,\mathbb{E}_{\boldsymbol{v}_1,\boldsymbol{v}_2\sim p(\boldsymbol{v}),\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\log D(G(E(\boldsymbol{x}),\boldsymbol{v}_1),G(E(\boldsymbol{x}),\boldsymbol{v}_2))\right] \\
& -\frac{1}{2}\,\mathbb{E}_{\boldsymbol{v}\sim p(\boldsymbol{v}),\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\log D(\boldsymbol{x},G(E(\boldsymbol{x}),\boldsymbol{v}))\right].
\end{aligned} \tag{4.4}$$

Note that these functions are trained end-to-end (in opposition to being pre-trained using the GMV approach). At test time we are interested in getting the encoder E and the generator G. As with GMV, these models may be used to generate new views of any object which is observed as an input sample $\boldsymbol{x}$ by computing its content vector $E(\boldsymbol{x})$, then sampling $\boldsymbol{v} \sim p_{\boldsymbol{v}}$ and, finally by computing the output $G(E(\boldsymbol{x}), \boldsymbol{v})$.

4.2.4 Experimental Protocol

Datasets. In order to evaluate the quality of our two models, we have performed experiments over four image datasets of various domains. The 3D-Chairs dataset (Aubry et al. 2014) are 3D models of chairs that can be rotated and rendered with a camera placed at two different azimuths. The CelebA dataset (Liu et al. 2015) is a dataset of faces labeled by identity and attributes. We consider that faces with the same identity label shares the same content, and do not use the other attributes during training. This dataset has more variability in views, but contents are relatively well structured. The MVC-Cloth dataset (Liu et al. 2016) depicts multiple articles of clothing from multiple angles. This dataset has canonical

views, but contents can be very different (for instance, leggings and shirts). The 102-Flowers dataset (Nilsback et al. 2008) is a dataset of photos of different types of flowers. We consider every flower from a category to share the same content. The datasets also varies in terms of the number of objects and of views per object. Those statistics are given in Table 4.2. The only supervision that we use is if two samples correspond or not to the same object (or belong to the same category in case of 102-Flowers).

Model architecture. We have used the same architectures for every dataset. The images were rescaled to $3 \times 64 \times 64$ tensors. The generator G and the discriminator D follow the DCGAN implementation proposed in Radford et al. 2016. For the conditional model, the encoder E is similar to D except for the first layer that includes a batch normalization and the last layer that doesn't have a non-linearity. An implementation of our algorithms is available [3].

Optimization. Learning has been made using classical GAN learning techniques: we used Adam optimizer (Kingma et al. 2015) with batches of size 128. Following standard practice, the learning rates in the GMV experiments are set to $1 \cdot 10^{-3}$ for G and $2 \cdot 10^{-4}$ for D. For the CGMV experiments, learning rates are set to $5 \cdot 10^{-5}$. The adversarial objectives are optimized by alternating gradient descent over the generator/encoder, and over the discriminator.

Baselines. We compare our proposal with recent state-of-the-art techniques. However, most existing methods are learned on datasets with view labeling. To fairly compare with alternative models we have built baselines working in the same conditions as our models.

For pure multi-view generative setting, we compared GMV with standard GANs that are learned to approximate the joint distribution of multiple samples: GANx2 is learned to output pairs of views over the same object, GANx4 is trained on quadruplets, and GANx8 on eight different views. For instance, the generator of a GANx2 model takes as input a sampled noise vector and outputs 2 images, while its discriminator is learned to distinguish these pairs of images as negative samples from positive samples which are built as for the GMV model, i.e. pairs of samples in the dataset that corresponds to the same object. The main difference with GMV is that the above GANx methods do not explicitly distinguish content and view factors as it is done in GMV.

Likewise, for conditional generation, we compared our approach CGMV to the CGAN baseline. In addition, we compare our model to the one from Mathieu et al. 2016 we presented in Chapter 2.2.2. For the sake of completeness and transparency, we report results obtained with two implementations. The first

3. https://github.com/mickaelChen/GMV

one based on the implementation provided by the authors[4] (denoted Mathieu et al. 2016), and the second one (denoted Mathieu et al. 2016+) that implements the same model using architectures inspired from DCGAN (Radford et al. 2016), which is more stable and that we have carefully tuned to allow a fair comparison with our approach.

Metrics. Aside from the qualitative judgments we use to show that our model is able to generate realistic samples of the same content under different views and of different content under the same view, and can correctly interpolate between samples, we also use a few metrics for the quantitative evaluation of our models using the multiple resources available for the CelebA dataset.

Firstly, using the CelebA dataset, we propose a set of experiments to measure the ability of our model to generate multiple views of the same image i.e. *to preserve the identity of the person* in two generated images. In order to evaluate if two images belong to the same person, we extracted features using VGG Face descriptor (Parkhi et al. 2015), which has been proposed for the person reidentification task. The proximity between the face descriptors computed from two images reflects the likeness of the persons in these images. We report a reidentification score that corresponds to the probability that a positive pair (i.e. a pair with two images of the same person) is associated with a lower distance than a negative pair (AUC). A score of 100% would correspond to a perfect model. We compute the AUC using 15 000 generated positive pairs and 15 000 generated negative pairs as a global estimation of the quality of our model. We also compute the AUC in two additional settings to better characterize the behavior of the models. Comparing generated positive pairs with negative pairs sampled from the dataset tells us how well the model can perform the task of generating the same person. Comparing real positive pairs with generated negative pairs allow us to asses the diversity of the generated individuals, as a model with low AUC under this setting indicates that the model generates very similar individuals.

Secondly, we take advantage of the large number of unused labels in the CelebA dataset and use those labels as a proxy to evaluate how well our model approximates the underlying distribution of the train set $p(\boldsymbol{x})$. Indeed, if our model captured $p(\boldsymbol{x})$ accurately, then the distribution of the labels in a generated dataset would be similar to the distribution of labels in the dataset. We estimate the distribution of the labels in the generated datasets using classifiers pre-trained on the train set and compare it to the distribution of labels in the dataset estimated on the test set with the same classifiers. As a first approximation, we assume independence between all labels and compute the distance between the estimated distribution q_{labels} for the generated set and the estimated distribution p_{labels} for the dataset as follows: labels distance $d = \sum_{l \in labels} |q_l - p_l|$, where q_l and p_l

4. https://github.com/MichaelMathieu/factors-variation

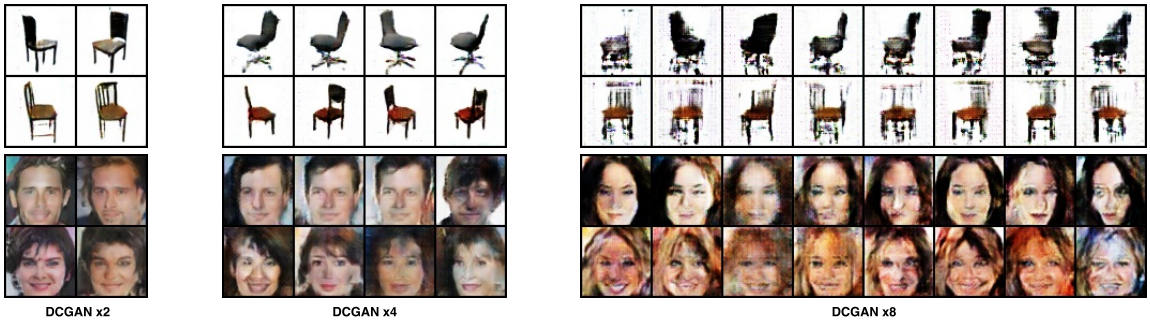


Figure 4.6 – Samples generated by the GANx2, GANx4 and GANx8 models. These samples have to be compared to the ones presented in Figure 4.3.

Figure 4.7 – Samples generated by the GMV model on the MVC-Cloth and the 102-Flowers datasets. All images in a row have been generated with the same content vector, and all images in a column have been generated with the same view vector.

correspond to the estimated Bernouilli parameters for the distribution of the label l.

This score, however, misses the dependencies between labels. To verify if our model generates rare combinations of labels, we also compute the number of unique combinations of the 40 attributes that occur in a set of samples and compare them with the number of unique combinations in the dataset, using again labels estimated by the pre-trained classifiers. We computed those values on sets of 3800 images, equal to the number of images in our test set.

Anecdotally, the celebA dataset also contains human evaluation for the "blurry" attribute. We also report the proportion of generated image considerer to be blurry by our estimator, as it gives an indication about the quality of generated images.

4.2.5 Results

Generating Multiple Contents and Views

We first assess the ability of our model to generate a large variety of objects and views. Figure 4.3 shows examples of generated images by our model and Figure 4.6 shows images sampled by GANx models (GANx2, GANx4, and GANx8).

For GMV generated images, a row shows a multiple samples that have been generated with the same sampled content factor $\boldsymbol{c} \sim p_{\boldsymbol{c}}$ but with various sampled view factors $v \sim p_v$, while the same view factor v is used for all samples in a column. Figure 4.7 shows generated samples for the two other datasets using the same presentation. Our approach allows to accurately generate the same views of multiple objects, or alternatively, the multiple views of a single object. The generated images are of quality, and the diversity of the generated views is high, showing the ability of our model to capture the diversity of the training dataset in terms of possible views.

For GANx baselines (4.6), images generated by these models a row corresponds to the multiple images produced by the model for a given sampled noise vector. When comparing GMV generated images to those generated by GANx models, one can see that the quality of images produced by GANx2 is comparable to the ones we obtain with GMV showing our approach has the same ability to generate realistic outputs. But GANx2 is only able to generate two views of each object since it does not distinguish content and view factors. For the same reason, the images in the same column (for different objects) do not necessarily match the same view. While GANx4 or GANx8 could have the ability to generate more views for each object, the learning problem is more difficult due to the very high-dimensionality of the observation space, and the visual qualities of the generation degrade.

Figure 4.8 shows generated samples obtained by interpolation between two different view factors (left) or two content factors (right). It allows us to have a better idea of the underlying view/content structure captured by GMV. We can see that our approach is able to smoothly move from one content/view to another content/view while keeping the other factor constant. This also illustrates that content and view factors are well independently handled by the generator i.e. changing the view does not modify the content and vice versa. Additional samples are shown in Figure 4.11 and interpolations in Figures 4.12 and 4.13.

Generating Multiple Views of a Given Object

The second set of experiments assess the ability of CGMV to capture a particular content from an input sample and to use this content to generate multiple views of the same object. Figure 4.9 and 4.10 illustrate the diversity of views in samples generated by our model and compare our results with those obtained with the

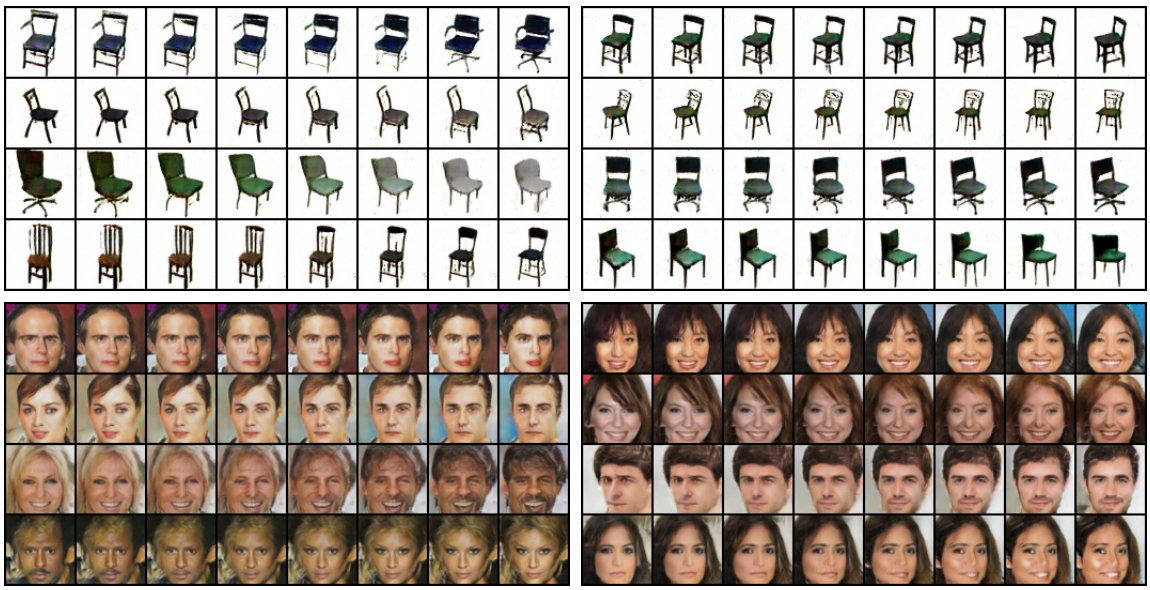


Figure 4.8 – Samples generated by the GMV model by using interpolation on the content (left) or on the view (right) for the 3D-Chairs (top) and the CelebA datasets (bottom). Within each of the four boxes, each row is independent of the others. For the two left boxes: The left and right column correspond to generated samples with two sampled content factors, while the middle images correspond to the samples generated by using linear interpolated content factors between the two extreme content factors while the view factor remains fixed. The two right boxes are built the same way by exchanging the roles of view and content.

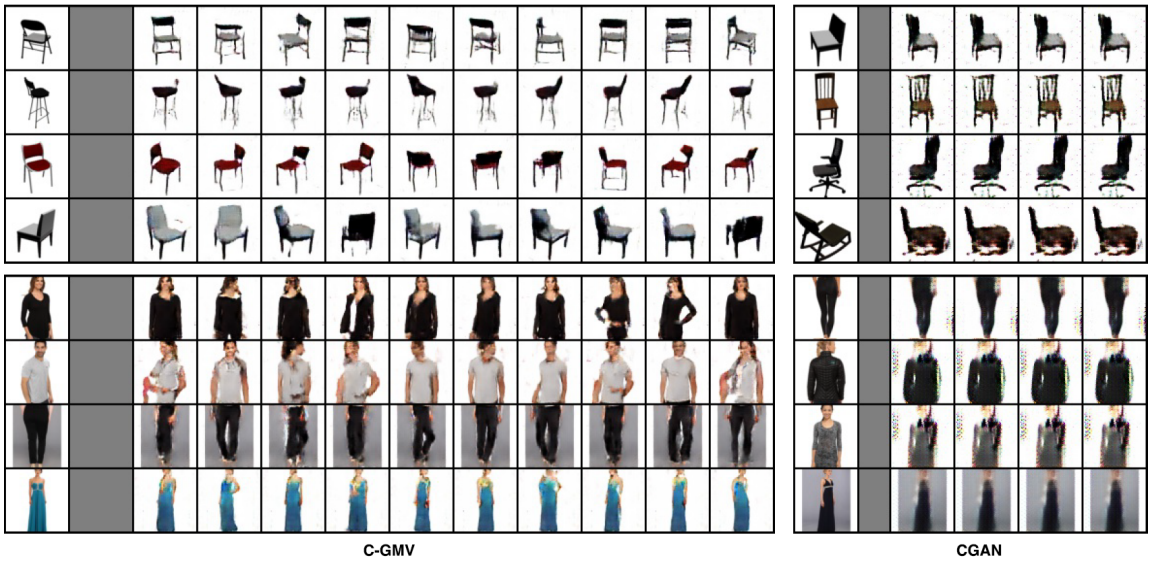


Figure 4.9 – Samples generated by conditional models. The images on the left are generated by CGMV while the images on the right are generated by a single CGAN. The leftmost column corresponds to the input example from which the content factor is extracted, while other columns are images generated with randomly sampled views.

CGAN model and to models from Mathieu et al. 2016. For each row, the input sample is shown in the left column. New views are generated from that input and shown on the right. Concerning the CGAN approach, the mode collapse phenomenon that we previously described occurs: the model does not take into account the view factor and always generate similar samples without any view diversity. The CGMV model demonstrates here that it is able to extract the content information and to generate a large variety of views of any object for which we have one image. Also, comparison with Mathieu et al. 2016 shows that images generated by the CGMV model are of better quality and more diverse. Additional samples are shown in Figures 4.14 and 4.15.

Identity Preservation

We measure here the ability of GMV and CGMV to preserve the identity of the person in pairs of generated images. First, when pairs of images are sampled directly from the dataset (Table 4.3, first column), the VGG Face classifier obtains an AUC of 93.3% (and not 100 %). This is due to the imperfection of the identity matching model that we use. This value can thus be seen as an upper bound of the expected performance of the different models. When comparing generative models (Table 4.3, columns 2 to 5), the GMV model obtains an AUC of 76.3 % which significantly outperforms the other generative models (GANx2, GANx4, and GANx8) that are less able to generate images of the same person. It means

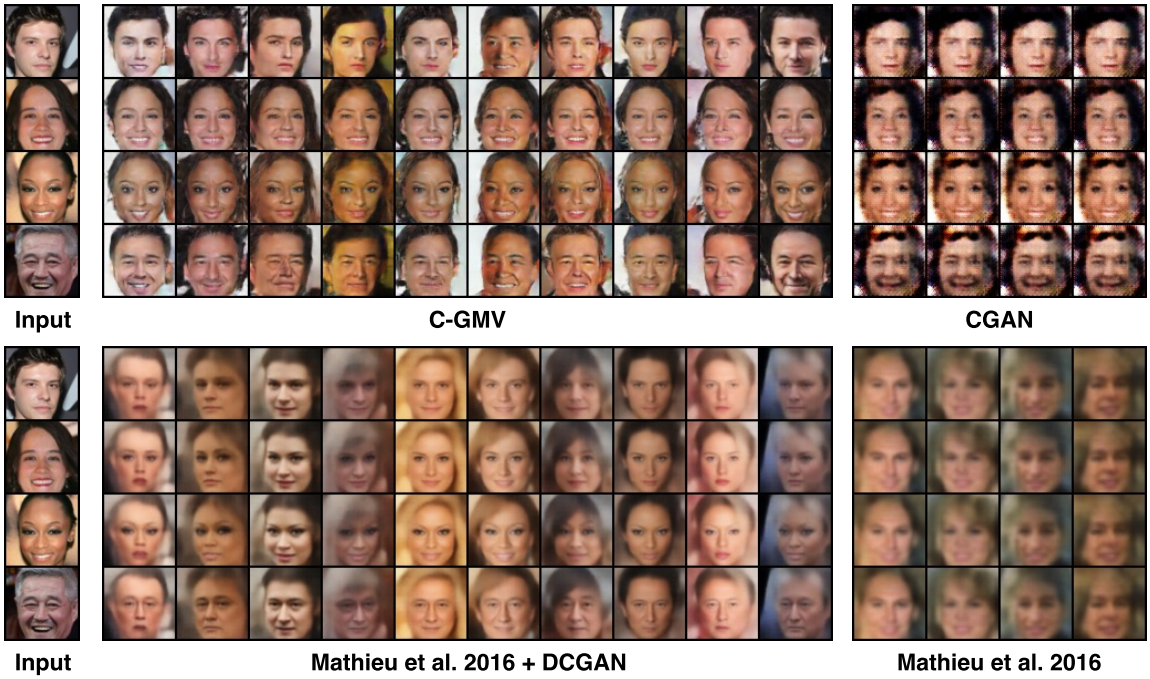


Figure 4.10 – Samples generated by conditional models: CGMV and CGAN (top), Mathieu et al. 2016 and + variant (bottom). The figure shows samples generated from four input images (leftmost column) by computing their content factors and by randomly sampling one view factor per column.

positive pair		negative pair	CelebA	GMV	GANx2	GANx4	GANx8
generated	vs	generated	93.3 %	76.3 %	72.0 %	74.2 %	57.5%
generated	vs	real	93.3 %	92.9 %	98.2 %	99.1 %	99.9%
real	vs	generated	93.3 %	78.7 %	51.5 %	47.8 %	13.3%

Table 4.3 – Identity preservation AUC of generative models. The CelebA column is the AUC obtained when positive pairs correspond to two images of the same person sampled from the test set, and negative pairs to images of two different persons.

positive pair		negative pair	CGMV	Mathieu et al. (2016)	Mathieu et al. (2016) +	CGAN
mixed	vs	real	82.0%	71.7%	87.6%	81.8%
mixed	vs	generated	67.2%	50.6%	77.3%	69.8%
generated	vs	generated	75.1%	61.2%	84.2%	100%
generated	vs	real	98.0%	99.5%	98.8%	100%
real	vs	generated	60.1%	14.0%	60.1%	36.7%

Table 4.4 – Identity preservation AUC of conditional models. Mixed positive pairs correspond to pairs in which one image is from the dataset and the other is generated using this real image as an input.

that 76.3 % of the pairs generated with the same content vector $\boldsymbol{c}$ (but random view vectors) have a lower distance than pairs generated with two randomly sampled content vectors $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$. More specifically, while all models are able to consistently generate positive pairs (Table 4.3 line 2), only the GMV can reliably generate negative pairs (Table 4.3 line 3). This hints at a lack of diversity in the image generated by the other models.

When comparing conditional models, the pair of images is generated based on a given input image. The AUC thus measures the ability of the models to generate new views while preserving the identity of the input person. We compare the CGMV model with the two implementations of the model by Mathieu et al. 2016 that we mentioned in Section 4.2.4. In addition to the settings we already used for purely generative models, we also report results for mixed positive pairs that consist of one real image from the dataset and one image generated using this real image as input. These mixed pairs are more representative of the way conditional models will be actually used. Results are reported on Table 4.4.

Firstly, as with the generative settings, all models are able to generate positive pairs easily (fourth line). On generative negative pairs, our model CGMV and Mathieu et al. 2016+ perform similarly, and largely better than CGAN (last line). This indicates that CGAN is unable to generate diversity in identity. Overall, Mathieu et al. 2016+ holds an advantage over CGMV, especially in the mixed settings that measure how well the model is able to preserve the identity of the input image. However, it should be noted that this metric doesn't account for the diversity of generated faces. For instance, CGAN also obtains suspiciously high scores in all settings except for the negative pair generation. These scores can be explained by the fact that the CGAN model is unable to generate diversity not only in identities but also in views. It thus always produce pairs of images that are exactly the same. It is then very easy to classify them as belonging to the same person. Our following experiments aim at evaluating the different models in

Model	Label Distance	Unique Combinations %	blurry %
GANx2	0.074	—	52.5 %
GANx4	0.092	—	82.4 %
GANx8	0.179	—	98.6 %
GMV	0.036	46.3 %	28.9 %
CGAN	0.139	08.7 %	99.8 %
Mathieu et al. 2016	0.133	03.2 %	99.9 %
Mathieu et al. 2016+	0.086	18.8 %	74.0 %
CGMV	0.043	42.6 %	37.2 %
Dataset	0.000	47.2 %	8.61 %

Table 4.5 – Diversity and blurriness scores for the different models. They have to be compared with the dataset estimate reported on the last line (the closer to the Dataset, the better).

terms of diversity of views (i.e. the generated views are different, and accurately capture the dataset distribution).

Quality and Diversity of the Generated Images

To evaluate the quality of the generative models, we estimated the labels of generated images using pre-trained classifiers. The idea is that a good generative model should generate samples with a distribution on attributes that is close to the one in the training set (while it is not actually included in the objective criterion). We report our estimation of the distribution of labels in generated sets in Table 4.6. We observed that many rare attributes were completely ignored (i.e. no occurrence in generated samples) by models such as Mathieu et al. 2016, CGAN, GANx8. Globally, there seems to be strong differences between models, with GMV and CGMV appearing to be better at capturing the dataset distribution. To quantify those differences, we aggregate them into a label distance score, a number unique combinations of labels score and, a blurriness score. We report those in Table 4.5.

GMV and CGMV perform better than all baselines on those three metrics. Our label distance is very low (below 0.05), and the estimated number of unique combinations of labels is high. These results show that CGAN and Mathieu et al. 2016 models ignore many rare attributes (see also Table 4.6) hence yield very poor diversity. While Mathieu et al. 2016+ is better than the original Mathieu et al. 2016 model, it does not reach the diversity obtained with either GMV and CGMV models which are close to the estimated diversity of the dataset. Our models are also less susceptible to produce samples qualified as "blurry" by our pretrained detector.

Even if the evaluation method is imperfect, those results support the fact that the CGMV and GMV methods are better able to generate interesting outputs.

To conclude on this set of experiments, the GMV and CGMV models seem to be the best trade-off between identity preservation and diversity in the generated views. While Mathieu et al. 2016+ tends to have a better identity preservation ability, it is at the price of generating samples with a much lower diversity than our approach which better captures the dataset distribution. Concurrently to our work, Donahue et al. 2018 proposed a very similar model and obtained comparable findings when evaluating with different metrics against different baselines, further supporting our observations.

4.2.6 Perspectives

In this chapter, we explored further the use of factorized representation. In the context of motion capture models, we introduced how such models could be used for motion capture synthesis. We also established that it could be beneficial to separate our different components into large vectors instead of a single label or scalar value as it is often done in disentangling models. We have then proposed, in the context of image synthesis, a generative model operating on a disentangled latent space which may be learned from multiview data without any view supervision, allowing its application to many datasets. Our model generates realistic data with a rich diversity of views. We also proposed a conditional version of this model which can generate new views of an input image which may again be learned without view supervision. Our experimental results show the quality of the produced outputs, and the ability of the model to capture content and view factors. They also illustrate the ability of GMV and CGMV to capture diversity on the CelebA dataset and to generate more realistic samples than baseline models.

Aside from the general advantages that come with using good representations in machine learning, we were also interested in investigating the use of such an approach for data augmentation. Indeed, if an object is underrepresented in the dataset, our model allow to generate new data of the object. This can possibly reduce the bias in class-imbalanced datasets. While our own preliminary experiments were met with little success, this line of work has been pursued with promosing results by Antoniou et al. 2018.

In our next chapter, we go a different direction and try to apply our method for the seemingly unrelated task of object segmentation in images. Indeed, to design our models, we first assumed the existence of a generative process in which natural images were explained by the mixing of two independent factors. Building a neural architecture to implement this process, we separated the component as desired and were able to exploit it in CGMV to extract the content factor. We show that this general framework can be applied, on very similar grounds and with

some additional assumptions, to extract object segmentation masks from images without supervision.

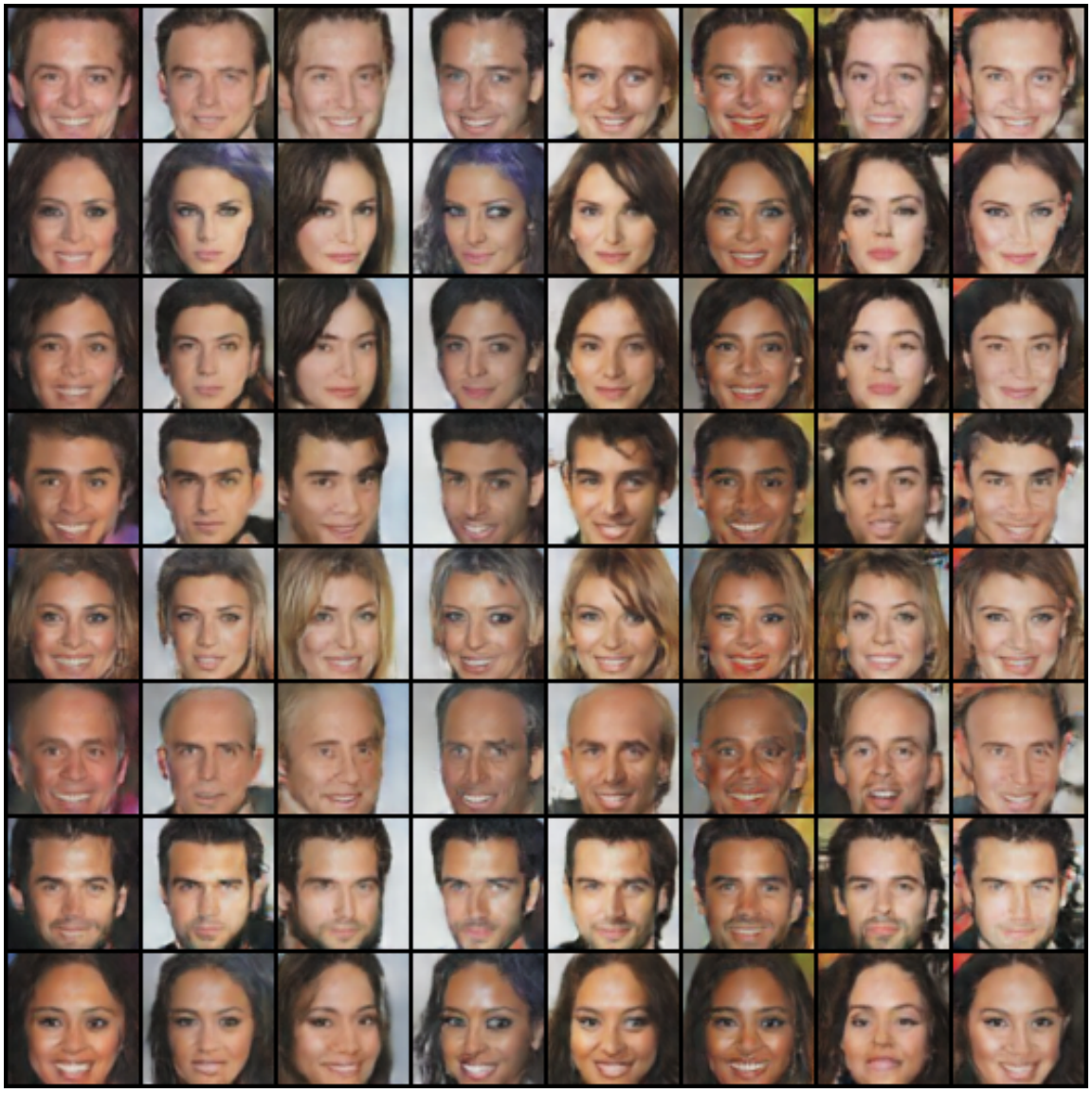


Figure 4.11 – Additional samples using GMV: All images in a row have been generated with the same content vector, and all images in a column have been generated with the same view vector.

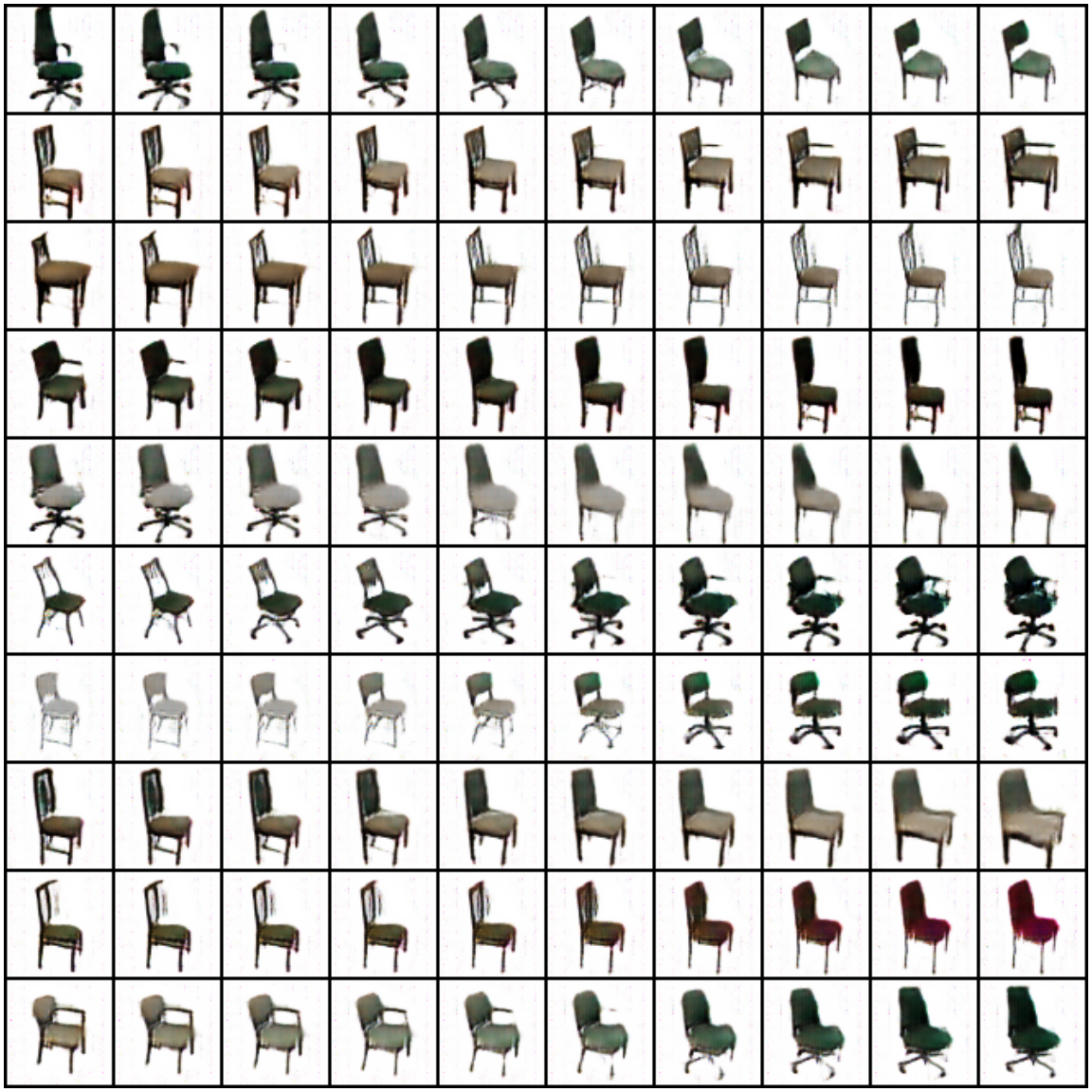


Figure 4.12 – Additional interpolations with GMV latent space: The view is shared among every images. Content is interpolated on a row.

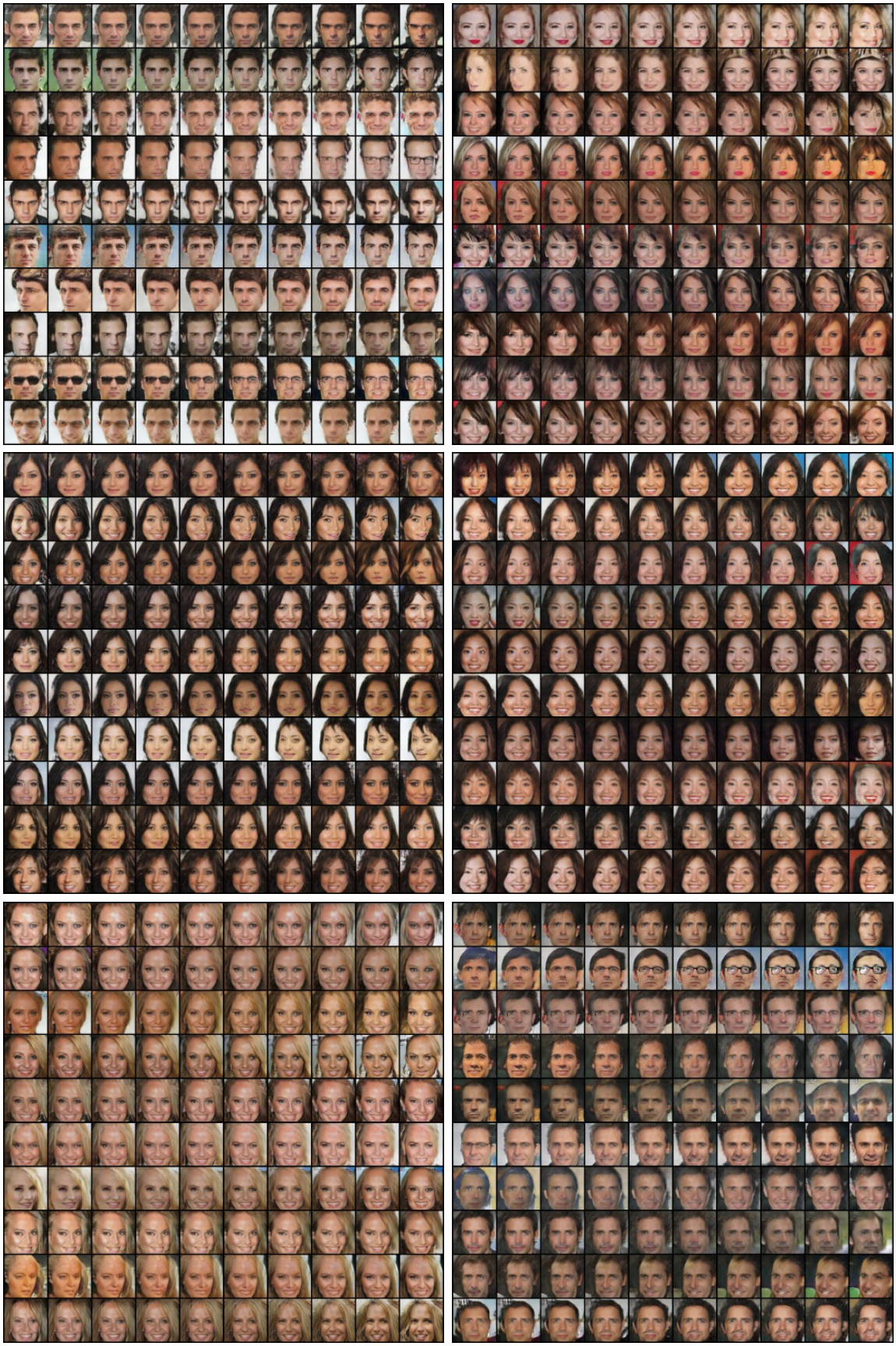


Figure 4.13 – Additional interpolations with GMV latent space: Each block is generated with the same content vector. The left-most and right-most columns are generated views. Images in between are interpolated.

Figure 4.14 – Additional samples of CGMV on the MVC-Cloth dataset. Left-most column shows inputs. Views are not aligned.

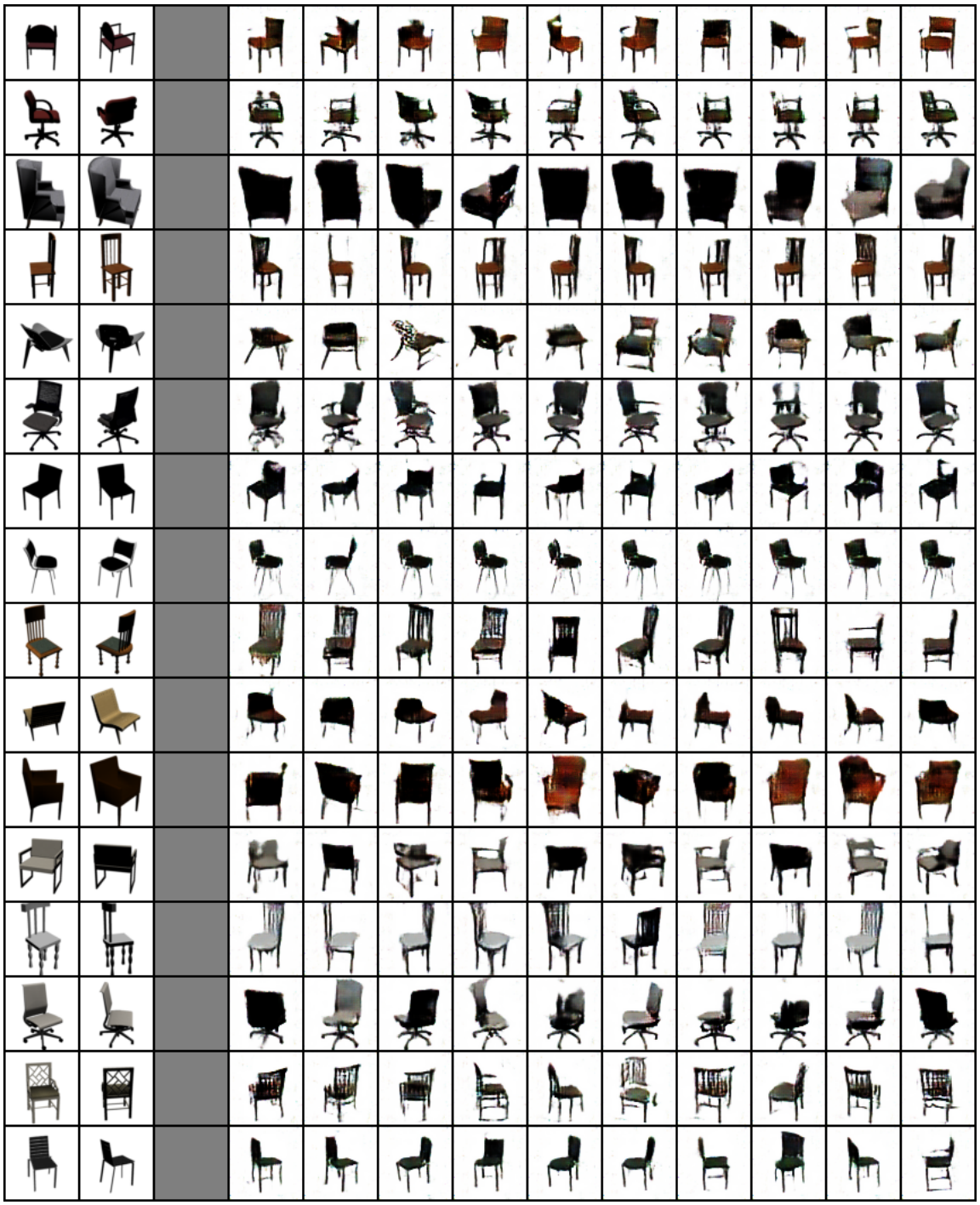


Figure 4.15 – Additional results of CGMV on the 3d-Chairs dataset. Left-most column shows inputs. Views are not aligned.

Attribute	CGAN	GANx2	GANx4	GANx8	Mathieu et al.	Mathieu et al. +	GMV	CGMV	CelebA estimate
50 Clock Shadow	0.0820	0.0041	0.0098	0.2302	0.0000	0.0003	0.0077	0.0048	0.0103
Arched Eyebrows	0.0040	0.1196	0.0608	0.0000	0.0003	0.1540	0.2152	0.2159	0.2182
Attractive	0.0008	0.2726	0.1484	0.0000	0.0016	0.2254	0.4721	0.4423	0.5961
Bags Under Eyes	0.0050	0.0051	0.0188	0.0139	0.0000	0.0011	0.0191	0.0111	0.0374
Bald	0.0000	0.0065	0.0029	0.0000	0.0005	0.0069	0.0074	0.0021	0.0053
Bangs	0.1542	0.0767	0.0728	0.0755	0.0000	0.0177	0.1128	0.1534	0.1903
Big Lips	0.0005	0.0104	0.0069	0.0000	0.0000	0.0011	0.0206	0.0238	0.0347
Big Nose	0.0209	0.0119	0.0316	0.0405	0.0000	0.0013	0.0254	0.0138	0.0437
Black Hair	0.4733	0.1131	0.1704	0.1615	0.0246	0.1302	0.2297	0.2296	0.3937
Blond Hair	0.0119	0.1201	0.0761	0.0000	0.0071	0.0698	0.1456	0.1074	0.1008
Blurry	0.9984	0.5249	0.8238	0.9857	0.9995	0.7405	0.2895	0.3722	0.0861
Brown Hair	0.0013	0.0014	0.0104	0.0000	0.0000	0.0000	0.0272	0.0225	0.0589
Bushy Eyebrows	0.0135	0.0156	0.0296	0.0000	0.0000	0.0042	0.0338	0.0228	0.0647
Chubby	0.0003	0.0011	0.0024	0.0000	0.0000	0.0013	0.0040	0.0011	0.0118
Double Chin	0.0000	0.0004	0.0010	0.0000	0.0000	0.0000	0.0020	0.0011	0.0029
Eyeglasses	0.0029	0.0393	0.0502	0.0004	0.0000	0.0034	0.0381	0.0257	0.0384
Goatee	0.0045	0.0070	0.0244	0.0038	0.0000	0.0003	0.0225	0.0061	0.0282
Gray Hair	0.0000	0.0023	0.0070	0.0005	0.0000	0.0016	0.0089	0.0011	0.0021
Heavy Makeup	0.0013	0.1304	0.0976	0.0000	0.0003	0.1317	0.3336	0.3085	0.3897
High Cheekbones	0.0243	0.2491	0.2169	0.0000	0.1098	0.3209	0.3494	0.3892	0.3374
Male	0.4862	0.3711	0.4102	0.8821	0.4032	0.2135	0.3054	0.2354	0.3071
Mouth Slightly Open	0.0614	0.3391	0.2964	0.2042	0.0841	0.2206	0.3926	0.3918	0.3621
Mustache	0.0013	0.0056	0.0126	0.0000	0.0000	0.0000	0.0182	0.0053	0.0163
Narrow Eyes	0.0159	0.0214	0.0276	0.0000	0.0212	0.0595	0.0786	0.0944	0.0816
No Beard	0.6198	0.9513	0.8631	0.1651	0.9952	0.9976	0.9360	0.9669	0.9408
Oval Face	0.0000	0.0079	0.0008	0.0000	0.0000	0.0042	0.0416	0.0296	0.1074
Pale Skin	0.0040	0.0111	0.0115	0.0000	0.0000	0.0061	0.0195	0.0352	0.0303
Pointy Nose	0.0000	0.0166	0.0520	0.0000	0.0013	0.0280	0.1700	0.1085	0.1363
Receding Hairline	0.0000	0.0472	0.0290	0.0000	0.1270	0.0839	0.0521	0.0217	0.0292
Rosy Cheeks	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0034	0.0011	0.0050
Sideburns	0.0442	0.0024	0.0178	0.1187	0.0000	0.0000	0.0123	0.0032	0.0155
Smiling	0.1418	0.4043	0.3956	0.0829	0.3151	0.4058	0.4606	0.4979	0.3953
Straight Hair	0.0003	0.0864	0.0138	0.0000	0.0132	0.0804	0.0984	0.0672	0.2882
Wavy Hair	0.2595	0.0044	0.1663	0.0984	0.0000	0.0000	0.1111	0.1082	0.0939
Wearing Earrings	0.0000	0.0013	0.0001	0.0000	0.0000	0.0101	0.0209	0.0071	0.0321
Wearing Hat	0.0021	0.0261	0.0312	0.0000	0.0000	0.0029	0.0324	0.0283	0.0426
Wearing Lipstick	0.0127	0.2923	0.2287	0.0000	0.0132	0.3156	0.4738	0.4971	0.5569
Wearing Necklace	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0011	0.0008	0.0003
Wearing Necktie	0.0000	0.0080	0.0024	0.0000	0.0000	0.0053	0.0095	0.0026	0.0153
Young	0.9183	0.9068	0.8170	0.4186	0.9307	0.9214	0.8941	0.9328	0.9292

Table 4.6 – Distribution of the different attributes over generated samples. For example, 3.8% of the samples generated by the GMV model exhibit the "Eyeglasses" attribute.

CHAPTER 5

UNSUPERVISED OBJECT SEGMENTATION

Contents

In the previous chapter, we discussed how to build and exploit a generative model based on an assumption of independence between latent factors. Using similar techniques, we now tackle the challenging task of unsupervised image segmentation. Image segmentation aims at splitting a given image into a set of non-overlapping regions corresponding to the main components in the image. It has been studied for a long time in an unsupervised setting using prior knowledge on the nature of the region one wants to detect using e.g. normalized cuts and graph-based methods. Recently the rise of deep neural networks and their spectacular performances on many difficult computer vision tasks have led to revisit the image segmentation problem using deep networks in a fully supervised setting (Chen et al. 2018a; He et al. 2017; Zhao et al. 2017), a problem referred to as semantic image segmentation.

Although such modern methods allowed learning successful semantic segmentation systems, their training requires large-scale labeled datasets with usually a need for pixel-level annotations. This feature limits the use of such techniques for many image segmentation tasks for which no such large scale supervision is available. To overcome this drawback, we follow here a very recent trend that aims at revisiting the unsupervised image segmentation problem with new tools and new ideas from the recent history and success of deep learning (Xia et al.

2017) and the recent results of supervised semantic segmentation (Chen et al. 2018a; He et al. 2017; Zhao et al. 2017).

Building on the idea of scene composition (Burgess et al. 2019; Eslami et al. 2016; Greff et al. 2019; Yang et al. 2017) and the adversarial learning principle (Goodfellow et al. 2014), we propose to address the unsupervised segmentation problem in a new way. As with the work presented in the previous chapter, we start by postulating an underlying generative process for images. This time, this process relies on an assumption of independence between regions of an image we want to detect. This means that replacing one object in the image with another one, e.g. a generated one, should yield a realistic image. We use such a generative model as a backbone for designing an object segmentation model we call ReDO (ReDrawing of Objects), which outputs are then used to modify the input image by redrawing detected objects. Following ideas from adversarial learning, the supervision of the whole system is provided by a discriminator that is trained to distinguish between real images and fake images generated accordingly to the generative process. Despite being a simplified model for images, we find this generative process effective for learning a segmentation model.

The chapter is organized as follows. We present related work in image segmentation in Section 5.1, then we describe our method in Section 5.2. We first define the underlying generative model that we consider in Section 5.2.2 and detail how we translate this hypothesis into a neural network architecture to learn a segmentation module in Section 5.2.3. Then we describe our experimental setup, including implementation details, in Section 5.3. Finally, we present experimental results on three datasets in Section 5.4 that explore the feasibility of unsupervised segmentation within our framework and compare its performance against a baseline supervised with few labeled examples.

The work presented in this chapter has led to the following publication:

- Mickaël Chen, Thierry Artières, and Ludovic Denoyer (2019a). "Unsupervised Object Segmentation by Redrawing". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 12705–12716.

5.1 Context for Segmentation And Scene Composition

Image segmentation is a very active topic in deep learning that boasts impressive results when using large-scale labeled datasets. Those approaches can effectively parse high-resolution images depicting complex and diverse real-world scenes into informative semantics or instance maps. State-of-the-art methods use clever

architectural choices or pipelines tailored to the challenges of the task (Chen et al. 2018a; He et al. 2017; Zhao et al. 2017).

However, most of those models use pixel-level supervision, which can be unavailable in some settings, or costly to acquire in any case. Some works tackle this problem by using fewer labeled images or weaker overall supervision. One common strategy is to use image-level annotations to train a classifier from which class saliency maps can be obtained. Those saliency maps can then be exploited with other means to produce segmentation maps. For instance, WILDCAT (Durand et al. 2017) uses a Conditional Random Field (CRF) for spatial prediction in order to post-process class saliency maps for semantic segmentation. PRM (Zhou et al. 2018), instead, finds pixels that provoke peaks in saliency maps and uses these as a reference to choose the best regions out of a large set of proposals previously obtained using MCG (Arbeláez et al. 2014), an unsupervised region proposal algorithm. Both pipelines use a combination of a deep classifier and a method that takes advantage of spatial and visual handcrafted image priors.

Co-segmentation, introduced by Rother et al. 2006, addresses the related problem of segmenting objects that are shared by multiple images by looking for similar data patterns in all those images. Like the aforementioned models, in addition to prior image knowledge, deep saliency maps are often used to localize those objects (Hsu et al. 2019). Unsupervised co-segmentation (Hsu et al. 2018), i.e. the task of covering objects of a specific category without additional data annotations, is a setup that resembles ours. However, unsupervised co-segmentation systems are built on the idea of exploiting features similarity and can't easily be extended to a class-agnostic system. As we aim to ultimately be able to segment very different objects, our approach instead relies on independence between the contents of different regions of an image which is a more general concept.

Fully unsupervised approaches have traditionally been more focused on designing handcrafted features or energy functions to define the desired property of *objectness*. Impressive results have been obtained when making full use of depth maps in addition to usual RGB images (Pham et al. 2018; Silberman et al. 2012) but it is much harder to specify good energy functions for purely RGB images. W-NET (Xia et al. 2017) extracts latent representations via a deep auto-encoder that can then be used by a more classic CRF algorithm. Kanezaki 2018 further incorporate deep priors and train a neural network to directly minimize their chosen intra-region pixel distance. A different approach is proposed by Ji et al. 2019 whose method finds clusters of pixels using a learned distance invariant to some known properties. Unlike ours, none of these approaches are learned entirely from data.

Our work instead follows a more recent trend by inferring scene decomposition directly from data. Stemming from DRAW (Gregor et al. 2015), many of those approaches (Burgess et al. 2019; Eslami et al. 2016) use an attention network to read a region of an image and a VAE to partially reconstruct the image in an

iterative process in order to flesh out a meaningful decomposition. LR-GAN (Yang et al. 2017) is able to generate simple scenes recursively, building object after object, and Sbai et al. 2018 decompose an image into single-colored strokes for vector graphics. While iterative processes have the advantage of being able to handle an arbitrary number of objects, they are also more unstable and difficult to train. Most of those can either only be used in generation (Yang et al. 2017), or only handle very simple objects (Burgess et al. 2019; Eslami et al. 2016; Greff et al. 2019). As a proof of concept, we decided to first ignore this additional difficulty by only handling a set number of objects but our model can naturally be extended with an iterative composition process. Closer to our setup, the very recent IODINE Greff et al. 2019 proposes a VAE adapted for multi-objects representations. Their learned representations include a scene decomposition, but they need a costly iterative refinement process whose performance has only been demonstrated on simulated datasets and not real images.

Like ours, some prior work has tried to find segmentation masks by recomposing new images. SEIGAN (Ostyakov et al. 2018) and Cut & Paste (Remez et al. 2018) learn to separate object and background by moving the region corresponding to the object to another background and making sure the image is still realistic. These methods, however, need to have access to background images without objects, which might not be easy to obtain.

5.2 Method

5.2.1 Overview

A segmentation process F splits a given image $I \in \mathbb{R}^{W \times H \times C}$ into a set of non-overlapping regions. F can be described as a function that assigns to each pixel coordinate of I one of n regions. The problem is then to find a correct partition F for any given image I. Lacking supervision, a common strategy is to define properties one wants the regions to have, and then to find a partition that produces regions with such properties. This can be done by defining an energy function and then finding an optimal split. The challenge is then to accurately describe and model the statistical properties of meaningful regions as a function one can optimize.

We address this problem differently. Instead of trying to define the right properties of regions at the level of each image, we make assumptions about the underlying generative process of images in which the different regions are explicitly modeled. Then, by using an adversarial approach, we learn the parameters of the different components of our model so that the overall distribution of the generated images matches the distribution of the dataset. We detail the generative process in Section 5.2.2, while the way we learn F is detailed in Section 5.2.3.

5.2.2 Generative Process

We consider that images are produced by a generative process that operates in three steps: first, it defines the different regions in the image i.e the organization of the scene (composition step). Then, given this segmentation, the process generates the pixels for each of the regions *independently* (drawing step). At last, the resulting regions are assembled into the final image (assembling step).

Let us consider a scene composed of $n-1$ objects and one background we refer to as object n. Let us denote $M^k \in \{0,1\}^{W \times H}$ the mask corresponding to object k which associates one binary value to each pixel in the final image so that $M^k_{x,y} = 1$ iff the pixel of coordinate (x, y) belongs to object k. Note that, since one pixel can only belong to one object, the masks have to satisfy $\sum_{k=1}^{n} M^k_{x,y} = 1$ and the background mask M^n can therefore easily be retrieved computed from the object masks as $M^n = 1 - \sum_{k=1}^{n-1} M^k$.

The pixel values of each object k are denoted $V^k \in \mathbb{R}^{W \times H \times C}$. Given that the image we generate is of size $W \times H \times C$, each object is associated with an image of the same size but only the pixels selected by the mask will be used to compose the output image. The final composition of the objects into an image is computed as follows:

$$I \leftarrow \sum_{k=1}^{n} M^k \odot V^k. \tag{5.1}$$

To recap, the underlying generative process described previously can be summarized as follow: i) first, the masks M^k are chosen together based on a mask prior $p(M)$. ii) Then, for each object independently, the pixel values are chosen based on a distribution $p(V^k|M^k, k)$. iii) Finally, the objects are assembled into a complete image.

This process makes an assumption of independence between the colors and textures of the different objects given the scene composition. The assumption is naive, as colorimetric values such as exposition, brightness, or even the colors of two objects, are often related, but it results in a constrained model that can provide a learning signal to a segmentation function, as we will show hereafter.

5.2.3 From Generative Process to Object Segmentation

Instead of considering a purely generative process where the masks are generated following a prior $p(M)$, we consider the inductive process where the masks are extracted directly from any input image I through the function F which is the object segmentation function described previously. The role of F is thus to output a set of masks given any input I. The new generative process acts as follows: i) it takes a random image in the dataset and computes the masks using

$F(I) \rightarrow M_1, \ldots, M_n$, and ii) it generates new pixel values for the regions in the image according to a distribution $p(V^k|M^k, k)$. iii) It aggregates the objects as before.

In order for output images to match the distribution of the training dataset, all the components (i.e. F and $p(V^k|M^k, k)$) are learned adversarially following the GAN approach. Let us define $D : \mathbb{R}^{W \times H \times C} \rightarrow \mathbb{R}$ a discriminator function able to classify images as fake or real. Let us denote $G_F(I, z_1, \ldots, z_n)$ our generator function able to compose a new image given an input image I, an object segmentation function F, and a set of vectors $z_1, \ldots, z_n$ each sampled independently following a prior $p(z)$ for each object k, background included. Since the pixel values of the different regions are considered as independent given the segmentation, our generator can be decomposed in n generators denoted $G_k(M^k, z_k)$, each one being in charge of deciding the pixel values for one specific region. The complete image generation process thus operates in three steps:

$$\begin{aligned} &1)\ M^1, \ldots, M^n \leftarrow F(I) && \text{(composition step)} \\ &2)\ V^k \leftarrow G_k(M^k, z_k) \text{ for } k \in \{1, \ldots, n\} && \text{(drawing step)} \\ &3)\ G_F(I, z_1, \ldots, z_n) = \sum_{k=1}^{n} M^k \odot V^k && \text{(assembling step).} \end{aligned} \tag{5.2}$$

Provided the functions F and G_k are differentiable, they can be learned by solving the following adversarial problem:

$$\begin{aligned} \min_{G_F} \max_{D} \mathcal{L} = \ & \mathbb{E}_{I \sim p_{data}} \big[\log D(I) \big] + \\ & \mathbb{E}_{I \sim p_{data}, z_1, \ldots z_n \sim p(z)} \big[\log(1 - D(G_F(I, z_1, \ldots, z_n))) \big]. \end{aligned} \tag{5.3}$$

Therefore, in practice, we have F output soft masks in $[0, 1]$ instead of binary masks. Also, in line with recent GAN literature (Brock et al. 2019; Miyato et al. 2018a; Tran et al. 2017; Zhang et al. 2019), we choose to use the hinge version of the adversarial loss (Lim et al. 2017; Tran et al. 2017) instead, and obtain the following formulation:

$$\begin{aligned} \max_{G_F} \mathcal{L}_G &= \mathbb{E}_{I \sim p_{data}, z_1, \ldots, z_n \sim p(z)} \big[D(G_F(I, z_1, \ldots, z_n)) \big] \\ \max_{D} \mathcal{L}_D &= \mathbb{E}_{I \sim p_{data}} \big[\min(0, -1 + D(I)) \big] + \\ & \qquad \mathbb{E}_{I \sim p_{data}, z_1, \ldots, z_n \sim p(z)} \big[\min(0, -1 - D(G_F(I, z_1, \ldots, z_n))) \big]. \end{aligned} \tag{5.4}$$

Still, as it stands, the learning process of this model may fail for two reasons. Firstly, this model naturally converges to a trivial extractor F that puts the whole image into a single region, the other regions being empty. Secondly, the model can ignore the input image I and still produce realistic images. We now detail those two problems and present our solutions.

Figure 5.1 – Illustration of the effect of the conservation of information constraint. The unconstraint model can generate realistic images from trivial masks (left). But in that case, the conservation of information constraint is violated (right).

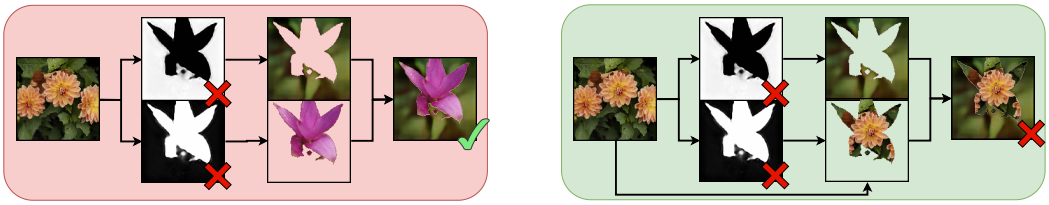


Figure 5.2 – Illustration of the effect of redrawing a single mask at a time. The previous model can generate realistic images while ignoring the input (left). By redrawing only one region, the model cannot generate realistic images anymore (right).

Conservation of region information. Our first problem is that F can map all pixels to one region, resulting in a trivial but valid solution, as the pipeline then collapses into a regular GAN. To prevent this from happening, we impose a first constraint that is that given a region k generated from a latent vector z_k, the final image $G_F(I, z_k, k)$ must contain information about z_k. Indeed, if region k is empty, i.e. $M^k_{x,y} = 0$ for all x, y, then z_k cannot be retrieved from the final image. Equivalently, if z_k can be retrieved, then region k is not empty. This problem and its solution are illustrated in Figure 5.1. This information conservation constraint is implemented through an additional reconstruction term in the loss function: $\mathcal{L}_z = \sum_{k=1}^{n} ||\delta_k(G_F(I, z)) - z_k||$, where δ_k is the function which objective is to infer the value of z_k given any image I. One can learn such a function simultaneously with the generator to promote the conservation of information. This strategy is similar to the mutual information maximization used in InfoGAN (Chen et al. 2016).

Constraining mask extraction by redrawing a single region. The second problem is that F can ignore the input I. Indeed, if F produces random valid segmentation masks but meaningless w.r.t. its input I, the generative model can still build

realistic outputs. Therefore, we need an additional constraint that conditions the output to the input and forces the model to extract meaningful region masks instead of ignoring the image. To this end, we take advantage of the assumption that the different objects are independently generated. We can, therefore, replace only one region at each iteration instead of regenerating all the regions. Since the generator now has to use original pixel values from the image in the reassembled image, it cannot make arbitrary splits. The generation process becomes as follows:

$$\begin{aligned} &1)\ M^1, \dots, M^n \leftarrow F(I) && \text{(composition step)} \\ &2)\ V^k \leftarrow I \text{ for } k \in \{1, \dots, n\} \setminus \{i\} \\ &\quad V^i \leftarrow G_i(M^i, z_i) && \text{(drawing step)} \\ &3)\ G_F(I, z_i, i) = \sum_{k=1}^{n} M^k \odot V^k && \text{(assembling step),} \end{aligned} \tag{5.5}$$

where i designates the index of the only region to redraw and is sampled from $\mathcal{U}(n)$, the discrete uniform distribution on $\{1, \dots, n\}$. This problem and its solution are illustrated in Figure 5.2.

The final complete process is illustrated in Figure 5.3 and corresponds to the following learning objectives:

$$\begin{aligned} \max_{G_F, \delta} \mathcal{L}_G &= \mathbb{E}_{I \sim p_{data}, i \sim \mathcal{U}(n), z_i \sim p(z)} \big[D(G_F(I, z_i, i)) \big] - \lambda_z \mathcal{L}_z \\ \max_{D} \mathcal{L}_D &= \mathbb{E}_{I \sim p_{data}} \big[\min(0, -1 + D(I) \big] + \\ &\qquad \mathbb{E}_{I \sim p_{data}, i \sim \mathcal{U}(n), z_i \sim p(z)} \big[\min(0, -1 - D(G_F(I, z_i, i)) \big], \end{aligned} \tag{5.6}$$

where λ_z is a fixed hyper-parameter that controls the strength of the information conservation constraint. Note that the constraint is necessary for our model to find non-trivial solutions, as otherwise, putting the whole image into a single region is both optimal and easy to discover for the neural networks. The learning algorithm follows classical GAN schema (Brock et al. 2019; Goodfellow et al. 2014; Miyato et al. 2018a; Zhang et al. 2019) by updating the generator and the discriminator alternatively.

5.3 Experimental Setup

5.3.1 Datasets

We present results on three natural image datasets and one toy dataset. All images have been resized and then cropped to 128×128.

The Flowers dataset (Nilsback et al. 2007; Nilsback et al. 2008) is composed of 8189 images of flowers. The dataset is provided with a set of masks obtained via

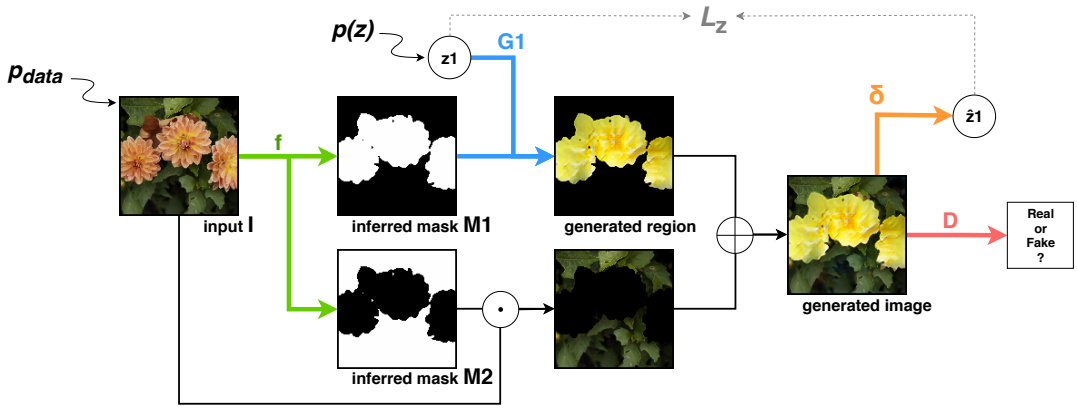


Figure 5.3 – Example generation with $G_f(I, z_i, i)$ with $i = 1$ and $n = 2$. Learned functions are in color.

an automated method built specifically for flowers (Nilsback et al. 2007). We split into sets of 6149 training images, 1020 validation and 1020 test images and use the provided masks as ground-truth for evaluation purpose only.

The Labeled Faces in the Wild (LFW) dataset (Huang et al. 2007b; Learned-Miller 2014) is a dataset of 13233 faces. A subpart of the funneled version (Huang et al. 2007a) has been segmented and manually annotated (Kae et al. 2013), providing 2927 ground-truth masks. We use the non-annotated images for our training set. We split the annotated images between validation and testing sets so that there is no overlap in the identity of the persons between both sets. The test set is composed of 1600 images, and the validation set of 1327 images.

The Caltech-UCSD Birds 200 2011 (CUB) dataset (Wah et al. 2011) is a dataset containing 11788 photographs of birds. We use 10000 images for our training split, 1000 for the test split, and the rest for validation.

As a sanity check, we also build a toy dataset (colored-2-MNIST) in which each sample is composed of an uniform background on which we draw two colored MNIST (LeCun et al. 1998) digits: one odd digit and one even digit. Odd and even numbers have colors sampled from different distributions so that our model can learn to differentiate them. For this dataset, we set $n = 3$ as there are three components.

As an additional experiment, we also build a new dataset by fusing Flowers and LFW datasets. This new Flowers+LFW dataset has more variability, and contains different type of objects. We used this dataset to demonstrate that ReDO can work without label information on problems with multiple categories of objects.

5.3.2 Metrics

To evaluate our method ReDO, we use two metrics commonly used for segmentation tasks. The pixel classification accuracy (Acc) measures the proportion of pixels that have been assigned to the correct region. The intersection over union (IoU) is the ratio between the area of the intersection between the inferred mask and the ground-truth over the area of their union. In both cases, higher is better. Because ReDO is unsupervised and we can't control which output region corresponds to which object or background in the image, we compute our evaluation based on the regions permutation that matches the ground-truth the best. We also used IoU for model selection, computed on a held out labeled validation set. When available, we present our evaluation on both the training set and a test set as, in an unsupervised setting, both can be relevant depending on the specific use case. Using those metrics, we compared the performance of ReDO, which is unsupervised, with a supervised method, keeping the same architecture for F in both cases, to analyze how many training samples are needed to reach the performance of the unsupervised model.

5.3.3 Implementation Details

We now provide some information about the architecture of the different components. The code is also available open-source [1]. As usual with GAN-based methods, the choice of a good architecture is crucial. We have chosen to build on the GAN and the image segmentation literature and to take inspiration from the neural network architectures they propose.

Distribution and size of z_i. We sample noise vectors z_i of size 32 (except for MNIST where we used vectors of size 16) from $\mathcal{N}(0, I_d)$ distribution. Those have been chosen smaller than what is usually found in GAN literature so that the vectors could be reasonably retrieved by δ.

Neural architectures. For the mask generator F, we use an architecture inspired by PSPNet (Zhao et al. 2017). The proposed architecture is a fully convolutional neural network similar to one used in image-to-image translation (Zhu et al. 2017), to which we add a Pyramid Pooling Module (Zhao et al. 2017) whose goal is to gather information on different scales via pooling layers. The final representation of a given pixel is thus encouraged to contain local, regional, and global information at the same time.

The region generators G_k, the discriminator D and the network δ that reconstructs z are based on SAGAN (Zhang et al. 2019) that is frequently used in recent

1. https://github.com/mickaelChen/ReDO

mask network $f(I)$	nonlinearities	output size
Image I		3x128x128
Conv 7x7 (reflect. pad 3)	Instance Norm, ReLU	16x128x128
Conv 3x3 (stride 2, pad 1)	Instance Norm, ReLU	32x64x64
Conv 3x3 (stride 2, pad 1)	Instance Norm, ReLU	64x32x32
Residual Bloc (Instance Norm, ReLU)		64x32x32
Residual Bloc (Instance Norm, ReLU)		64x32x32
Residual Bloc (Instance Norm, ReLU)		64x32x32
Pyramid Pooling Module		68x32x32
Upsample		68x64x64
Conv 3x3 (pad 1)	Instance Norm, ReLU	34x64x64
Upsample		34x128x128
Conv 3x3 (pad 1)	Instance Norm, ReLU	17x128x128
Conv 3x3 (reflect. pad 3)	sigmoid (if $n = 2$) or softmax	nx128x128

Table 5.1 – Architecture of mask network f. The architecture is similar to CycleGAN for image translation, except with less residual blocs but a Pyramid Pooling Module introduced by PSPNet.

discriminator network D and encoder δ	output size
image input I	3x128x128
Down Res Bloc (ReLU)	64x64x64
Self-Attention Bloc	
Down Res Bloc (ReLU)	128x32x32
Down Res Bloc (ReLU)	256x16x16
Down Res Bloc (ReLU)	512x8x8
Down Res Bloc (ReLU)	1024x4x4
Res Bloc (ReLU)	1024x4x4
Spatial sum pooling	1024x1x1
Linear	1 for D, 32 for δ

Table 5.2 – Architecture of discriminator network D and encoder δ.

region generator network $G_k(z_k, M^k)$	output size
noise vector input z_k	32
Linear, Conditional Instance Norm, ReLU	16ch.x4x4
Up Res Bloc (CINorm, ReLU, concat 1x4x4 M^k)	16ch.x8x8
Up Res Bloc (CINorm, ReLU, concat 1x8x8 M^k)	8ch.x16x16
Up Res Bloc (CINorm, ReLU, concat 1x16x16 M^k)	4ch.x32x32
Up Res Bloc (CINorm, ReLU, concat 1x32x32 M^k)	2ch.x64x64
Self-Attention Bloc	
Up Res Bloc (CINorm, ReLU, concat 1x64x64 M^k)	ch.x128x128
Conditional Instance Norm, ReLU, concat 1x128x128 M^k	
Conv 3x3 (padding 1), Tanh	3x128x128

Table 5.3 – Architecture of region generator network G_k. Main differences compared to other popular implementations are that mask input is concatenated at each layer and the noise vector is used as seed input but also fed into the network via instance norm conditioning. For the LFW and MNIST dataset, we set ch=64. For other datasets, ch=32 performed more consistently.

GAN literature (Brock et al. 2019; Lucic et al. 2019). Notably, we use self-attention (Zhang et al. 2019) in G_k and D to handle non-local relations, and spectral normalization (Miyato et al. 2018a) for weight regularization in all networks except for the mask provider F (on which we apply a weight decay of 10^{-4} instead).

To both promote stochasticity in our generators and encourage our latent code z to encode for texture and colors, we also use conditional batch-normalization in G_k. The technique has emerged from style modeling for style transfer tasks (Dumoulin et al. 2017b; Perez et al. 2018) and has since been used for GANs as a means to encode for style and to improve stochasticity (Almahairi et al. 2018; Chen et al. 2019b; Wang et al. 2016). All parameters of the different δ_k functions are shared except for their last layers.

Optimization scheme. As it is standard practice for GANs (Brock et al. 2019), we use orthogonal initialization (Saxe et al. 2014) for our networks and Adam (Kingma et al. 2015) with $\beta = (0, .9)$ as optimizer. Learning rates are set to 10^{-4} except for the mask network F which uses a smaller value of 10^{-5}. We used mini-batches of size 20, as much as we could fit on our GPU. Initialization gain of 0.8 was used.

Strength of the conservation of information. We use $\lambda_z = \frac{5}{n*size_z}$ for all datasets except LFW, for which $\lambda_z = \frac{15}{n*size_z}$. Despite our conservation of information loss, the model can still collapse into generating empty masks at the

Figure 5.4 – Generated samples (not cherry-picked, zoom in for better visibility). For each dataset, the columns are from left to right: 1) input images, 2) ground-truth masks, 3) masks inferred by the model for object one, 4-7) generation by redrawing object one, 8-11) generation by redrawing object two. As we keep the same z_i on any given column, the color and texture of the redrawn object are kept constant across rows.

early steps of the training. While the regularization does alleviate the problem, we suppose that the mask generator F can collapse even before the network δ learns anything relevant and can act as a stabilizer. As the failures happen early and are easy to detect, we automatically restart the training should the case arise.

5.4 Results

Segmentation scores are presented in Table 5.4 and show that ReDO achieves reasonable performance on the three real-world datasets. Comparison against the supervised baseline with the same architecture (Figure 5.6) shows that the unsupervised performances are in the range of the ones obtained with a supervised method trained with around 50 or 100 examples depending on the dataset. For instance, on the LFW Dataset, the unsupervised model obtains an accuracy of 0.917 and an IoU of 0.781. The supervised model needs 50-60 labeled examples to reach similar performance.

We provide random samples of extracted masks (Figure 5.4) and the corresponding generated images with a redrawn object or background. Note that our objective is not to generate appealing images but to learn an object segmentation function. Therefore, ReDO generates images that are less realistic than the ones generated by state-of-the-art GANs. Focus is, instead, put on the extracted masks,

Figure 5.5 – Results for the LFW + Flowers dataset, arranged as in Figure 5.4. As z is kept constant on a column across all rows, we can observe that z codes for different textures depending on the class of the image even though the generator is never given this information explicitly.

Dataset	Train Acc	Train IoU	Test Acc	Test IoU
LFW	-	-	0.917 ± 0.002	0.781 ± 0.005
CUB	0.840 ± 0.012	0.423 ± 0.023	0.845 ± 0.012	0.426 ± 0.025
Flowers*	0.886 ± 0.008	0.780 ± 0.012	0.879 ± 0.008	0.764 ± 0.012
Flowers+LFW	-	-	0.856	0.691

Table 5.4 – Performance of ReDO in terms of accuracy (Acc) and intersection over union (IoU) on retrieved masks. Means and standard deviations are based on five runs with fixed hyper-parameters. LWF train set scores are not available since we trained on unlabeled images. *Please note that segmentations provided along the original Flowers dataset have been obtained using an automated method. We display samples with top disagreement masks between ReDO and ground-truth in Figure 5.8. In those cases, we find ours to provide better masks.

and we can see the good quality of the obtained segmentation in many cases. Best and worst masks, as well as more random samples, are displayed in Figures 5.8-.

We also trained ReDO on the fused Flowers+LFW dataset without labels. We re-used directly the hyper-parameters we have used to fit the Flowers dataset without further tuning and obtained, as preliminary results, a reasonable accuracy of 0.856 and an IoU of 0.691. This shows that ReDO is able to infer class information from masks even in a fully unsupervised setup. Samples are displayed in Figure 5.5.

5.5 Perspectives

We presented a novel method called ReDO for unsupervised learning to segment images. Our proposal is based on the assumption that if a segmentation model is accurate, then one could edit any real image by replacing any segmented object in a scene by another one, randomly generated, and the result would still be a realistic image. This principle allows casting the unsupervised learning of

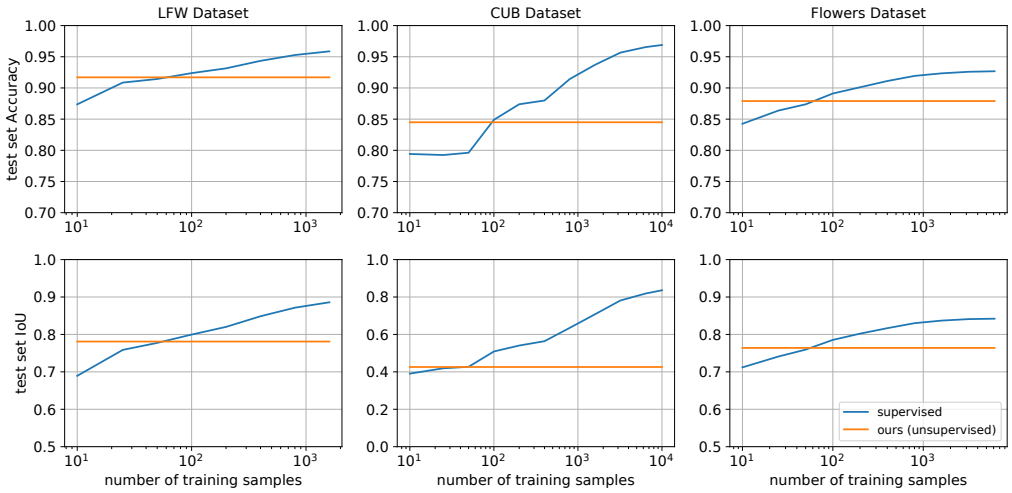


Figure 5.6 – Comparison against the supervised baseline as a function of the number of available training samples.

image segmentation as an adversarial learning problem. Our experimental results obtained on three datasets show that this principle works. In particular, our segmentation model is competitive with supervised approaches trained on a few hundred labeled examples.

This model is among the first proposed for data-driven unsupervised image segmentation that scaled to real natural images. Concurrently to our work, Bielski et al. 2019 propose to find object segmentation based on another assumption, that is, objects are regions of images that can be moved to a different location. Based on that assumption, they also use a constrained adversarially learned generative model to extract a segmentation module, which comforts us in this direction of research.

Current unsupervised models, however, have only been tested on relatively simple images as a first step. There is still a very large gap between those images and the complex ones that supervised methods can process. As mentioned in Section 5.1, our model could generalize to an arbitrary number of objects and objects of unknown classes via iterative design and/or class agnostic generators. Mostly, we are limited by our ability to effectively train GANs on those more complicated settings but rapid advances in image generation (Brock et al. 2019; Karras et al. 2019; Lucic et al. 2019) make it a reasonable goal to pursue in the near future. Meanwhile, it could be interesting to investigate the use of the model in a semi-supervised or weakly-supervised setup. Indeed, additional information would allow us to guide our model for harder datasets while requiring fewer labels than fully supervised approaches. Conversely, our model could act as a regularizer by providing a prior for any segmentation tasks.

Figure 5.7 – Masks obtained for images from the LFW test set. Each block of three rows depicts from top to bottom input image, ground-truth, and output of our model. Each block from top to bottom: 1) top masks according to accuracy 2) top masks according to IoU 3-4) randomly sampled masks 5) worst masks for accuracy 6) worst masks for IoU.

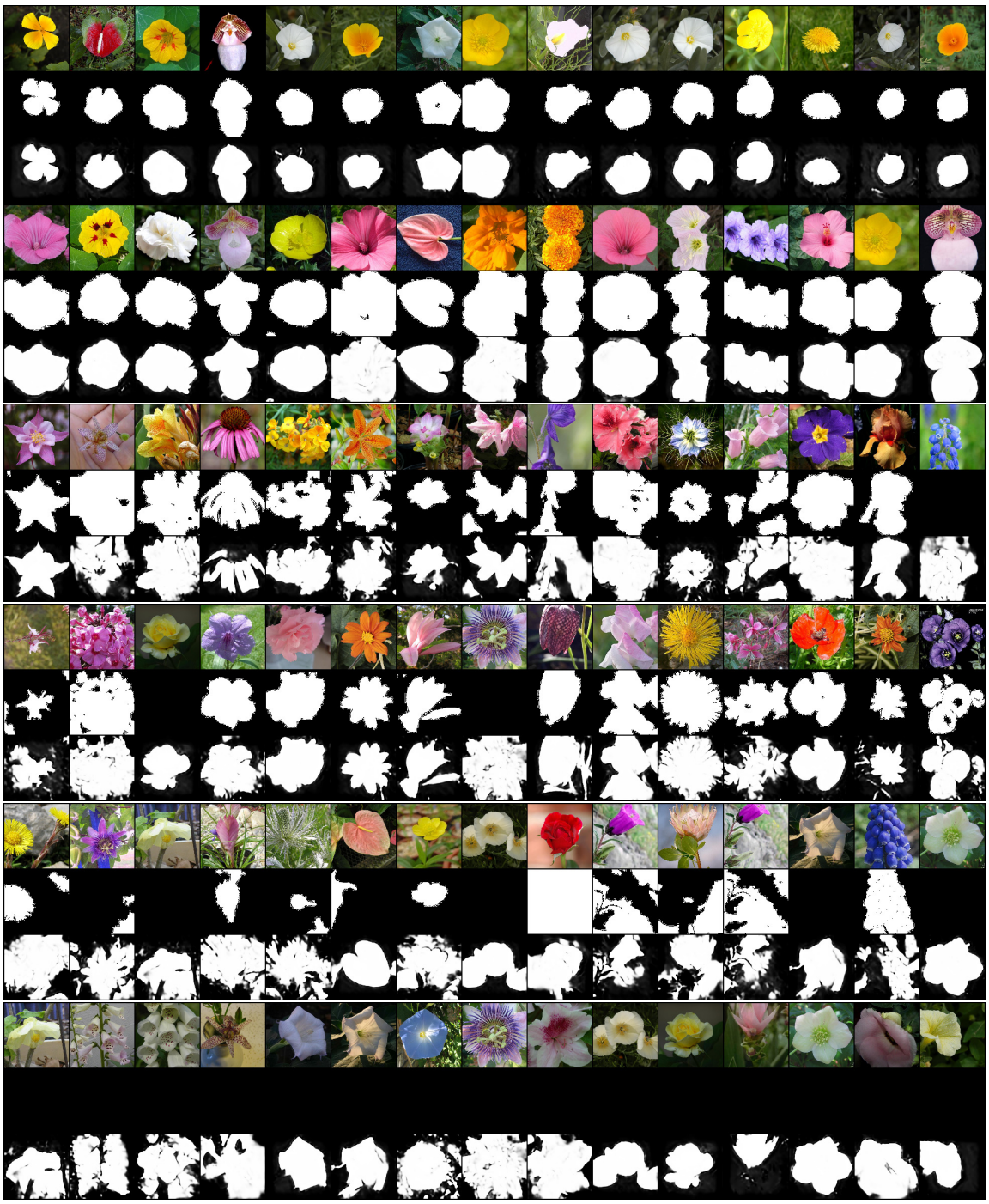


Figure 5.8 – Masks obtained for images from the Flowers test set, organized as in 5.7. Because the ground-truth masks for Flowers were obtained via an automated process, our model actually provides better predictions in worst agreement cases.

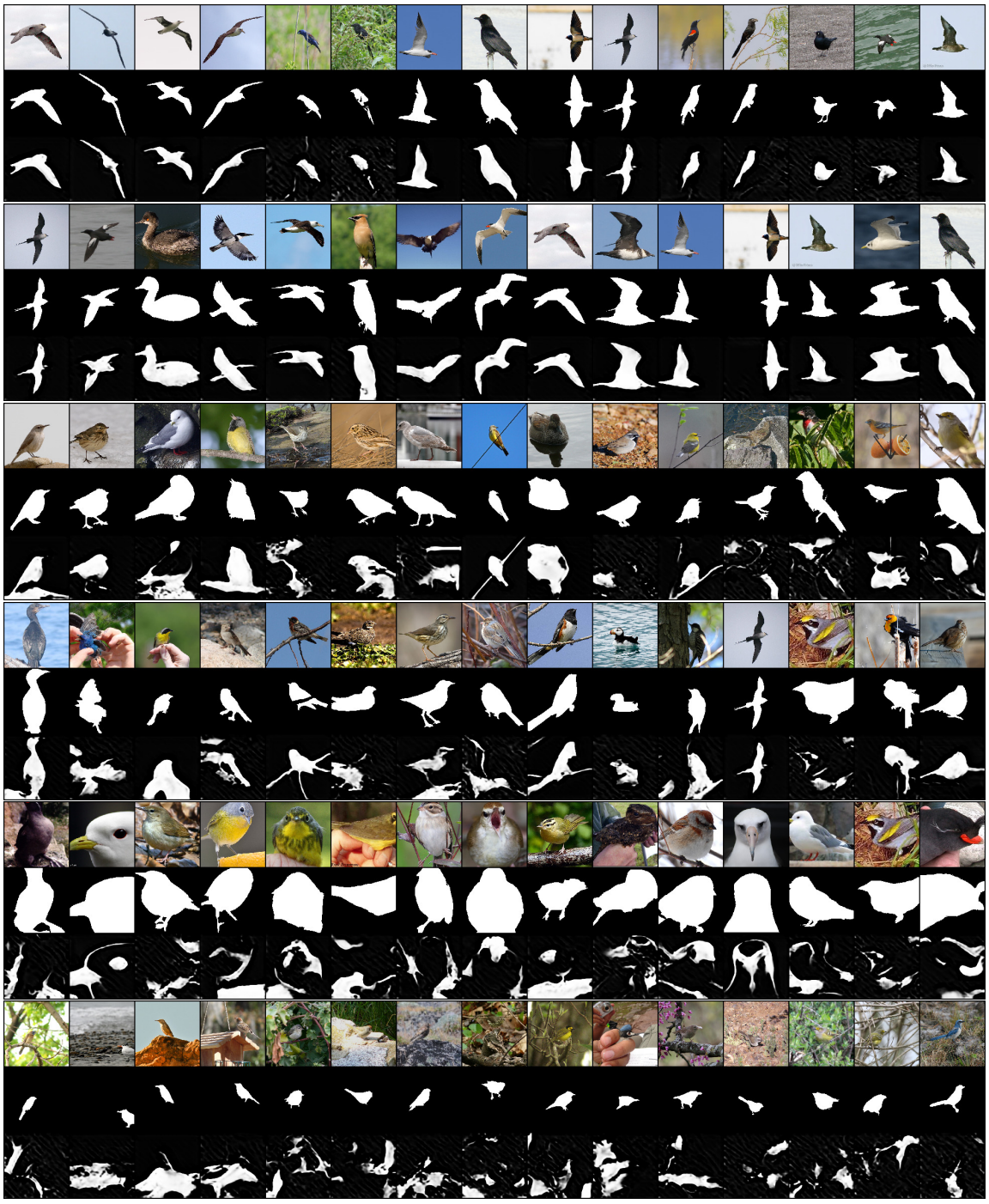


Figure 5.9 – Masks obtained for images from the CUB test set, organized as in 5.7
.

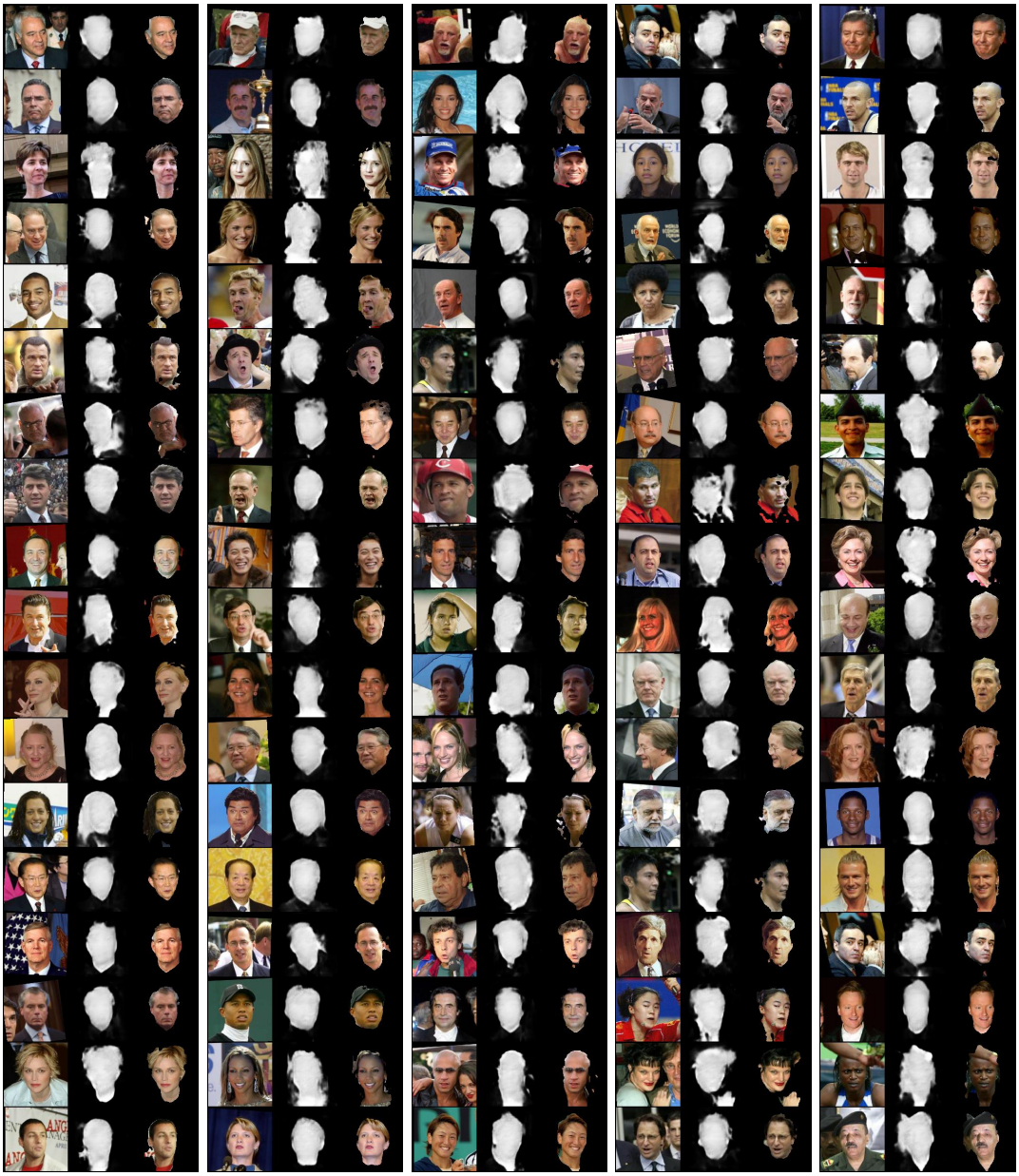


Figure 5.10 – More randomly sampled output masks for the LFW dataset.

Figure 5.11 – More randomly sampled output masks for the Flower dataset.

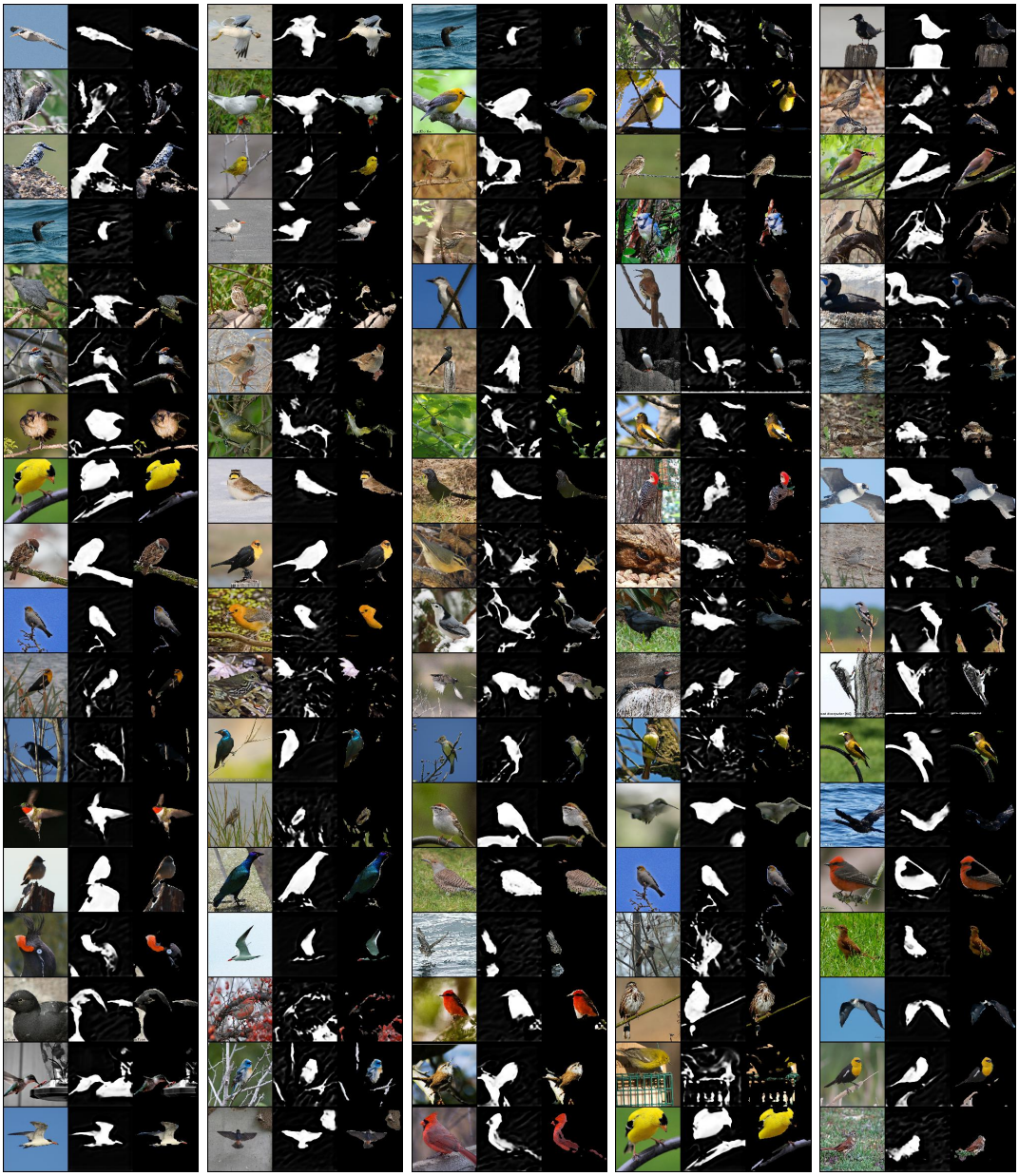


Figure 5.12 – More randomly sampled output masks for the CUB dataset.

With this final contribution, we provided an example of how a sufficiently constrained mixing process can provide a signal to learn a function in a case where supervision is not available. The next chapter will summarize the contributions we proposed and discuss future directions of research.

CHAPTER 6

CONCLUSION

Contents

In this thesis, we tackled challenging problems when lacking full supervision, using deep generative models and adversarial learning. Indeed, while supervised learning has been at the forefront of advances in deep learning in the past decade, supervision is not always available in practical cases. Therefore, it is crucial to develop a variety of tools that can exploit different types of supervision. Since their introduction in recent years, GANs and VAEs have been showing promising results in that regard. We provided an overview of such successes in Chapter 2. In this context, our contribution consisted in exploring and designing novel approaches, leveraging Deep Generative Models and Adversarial Learning. We summarise those contributions in this Section 6.1. We then propose perspectives and future work in Section 6.2.

6.1 Summary of Contributions

Our contributions have been organized following three points we restate in this section.

6.1.1 Performing Stochastic Predictions

In Chapter 3, we focused on designing models that could perform stochastic predictions. The goal of such models is to capture the full array of possible predictions and to propose different plausible scenarios for a given input. This contrasts with classical methods that can only produce relatively simple outputs. The task is well-suited to deep generative models as they can find regularities and

variabilities even in datasets in which each input is only associated with a single realization of the possible outputs.

We first presented MV-BiGAN (Chen et al. 2017), a model for stochastic prediction in the context of multi-view data. The model had to be able to propose different possible outcomes for a given input. As a multi-view model, it also had to handle cases when some of the views are missing in the input. To build our stochastic prediction model, we encoded inputs into stochastic latent variables on which predictions were performed. We based our implementation on Bidirectional-GANs to handle the stochastic mappings, and we used a KLD constraint, reminiscent of VAE-based models, on the latent variables to maintain consistency of predictions when additional views were provided. The resulting model behaved as expected on MNIST-based benchmarks, but our implementation quickly showed its limits in slightly more complex cases. Those limitations can probably be attributed to training instability of our model, hampering performances, and to the poor capacity of Bidirectional-GANs in terms of reconstruction power.

Later, building on advances in both generative models and dynamic models, we presented SRVP (Franceschi et al. 2020), a model for stochastic video prediction. SRVP features a fully latent dynamic model based on residual updates, augmented with stochastic variables to handle uncertainty. Contrary to other approaches, it isn't frame-autoregressive and doesn't use previously predicted frames as input for the next step. This effectively removes the dependency of the dynamic module on the generation module. The model further complements this separation by splitting representations of static content and dynamics, as in other works on the same topic. This approach allows us to obtain state-of-the-art performances on multiple benchmarks and highlights the importance of controlling information flow in the latent embeddings.

6.1.2 Adversarial Methods for Learning Factorized Representations

In Chapter 4, we focused on adversarial methods to factorize information in those latent embeddings. Controlling the properties and contents of learned latent representations is an important topic in deep learning and research in this direction could potentially allow for better models in terms of performances or interpretability.

We first explored adversarial methods for censoring a latent vector and separating style and action in the context of Motion Capture sequence modeling (Wang et al. 2018b; Wang et al. 2018a). The idea is that by learning an auto-encoder that factorizes the latent representation into action and style, it would be possible, at test time, to perform any chosen action in any chosen style. We validated that the strategy is effective, and compared different models. In particular, we

observed that models that use a single label as code for the action underperformed compared to models that extract the action from an example and encode it into a vector of small size.

In a second work on this topic, we used a GAN-based method in the context of multi-view image generation. Assuming content and style are independent, we built GMV (Chen et al. 2018b), a generative model that produces images by mixing two independent random variables. We supervised our training with pairs of data of the same content so that we don't have to rely on categorical class labels. It resulted in a controllable generative model (CGMV) that could alter the style of an image while preserving its content. The model also discovered style embeddings that are transferable across different contents without using style labels.

6.1.3 Unsupervised Object Segmentation

In Chapter 5, we tackled the task of object segmentation in images. Image segmentation is a prominent task in Computer Vision that have strong impacts in many applications such as vehicle automation and medical imaging. It is also a challenging task for which the best current methods rely on large scale datasets with complex annotations. Reducing the need for precise annotations would directly impact the usability of those models in many situations.

We presented Redrawing Of Objects (ReDO) (Chen et al. 2019a), a novel model for unsupervised object segmentation that relied on an assumption of independence between the contents of the meaningful regions in an image. Using a GAN-based framework similar to that of our work on multi-view generation, we build a system that generated images in which the texture of each region is controlled by an independent latent variable. The model was constrained so that the segmentation function we wished to learn had to find such regions in the input images. The work demonstrated a novel use of generative models for segmentation tasks and was among the first to propose an unsupervised data-driven object segmentation method that worked convincingly on real, albeit relatively simple, images.

6.2 Perspectives and Future Work

This thesis started in October 2016. Earlier in the same year, Radford et al. 2016 had demonstrated that GANs could be used to generate convincing realistic images in a publication that largely contributed to the rise of GANs and VAEs as prominent topics of interest in the fields of machine learning and computer vision. As such, much of the work done during this thesis was about exploring the new possibilities offered by those tools. By the exploratory nature of our work, a lot of potential future work would consist in finding ways to scale-up different and more complex datasets. Part of the problems comes with current limitations

of generative models and, considering the rate at which they are improved, we expect some of those to be alleviated in the near future. We provide in this section some further directions of research.

For stochastic video prediction, SRVP is already able to generate realistic human motions on a difficult dataset (Human3.6M). Still, the images are blurry and of low quality. It could help to combine our VAE-based approach with an adversary that would control the quality of the generated images, provided we are careful to not hinder the diversity of the generated sequences and the good properties of the learned latent space. We also observe on the BAIR dataset that it is difficult to capture the movement of small objects, especially since most of them are static in most examples. Combining our approach with a multi-object modelization, such as one proposed by Burgess et al. 2019, might be necessary to achieve more complex scenarios.

Our work on multi-view generation split the representation into two parts, using only information on one of them. The model could also be extended to separate into more factors to handle more complex cases, either via more supervision or by combining it with unsupervised methods. The latter could also provide more insights, as it is unclear how current unsupervised disentangling methods interact in more complex systems with multiple objectives. For practical application, as a controllable generation method, it might be interesting to further explore if data generated by CGMV would be useful to counteract class-imbalance in datasets.

As for image segmentation, unsupervised methods still need a lot of improvement to reach the capabilities of their supervised counterparts. Augmenting ReDO using reccurent networks for the segmentation function and scene composition function could be a first step to handle more complex images. In its current state, our work shows that independence between the content of the different regions is a useful assumption, and it would be interesting to see if it can improve performances when incorporated into semi-supervised or weakly-supervised approaches.

From a broader perspective, the common strategy behind our models was about organizing latent variables into generative explanatory factors. This approach can also apply to questions related to the interpretability of models, causal intervention and, fairness. Links between controllable generative models and interventions in causal graphs have been explored from different directions by Kocaoglu et al. 2018 and Suter et al. 2019. Meanwhile, applications of unsupervised disentanglement to fairness have been proposed by Locatello et al. 2019a and Sarhan et al. 2020. We believe our approach could be relevant in those contexts.

BIBLIOGRAPHY

Almahairi, Amjad, Sai Rajeswar, Alessandro Sordoni, Philip Bachman, and Aaron C. Courville (2018). "Augmented CycleGAN: Learning Many-to-Many Mappings from Unpaired Data". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 195–204.

Antoniou, Antreas, Amos J. Storkey, and Harrison Edwards (2018). "Augmenting Image Classifiers Using Data Augmentation Generative Adversarial Networks". In: *Artificial Neural Networks and Machine Learning - ICANN 2018 - 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III*, pp. 594–603.

Arbeláez, Pablo Andrés, Jordi Pont-Tuset, Jonathan T. Barron, Ferran Marqués, and Jitendra Malik (2014). "Multiscale Combinatorial Grouping". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pp. 328–335.

Arjovsky, Martín, Soumith Chintala, and Léon Bottou (2017). "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 214–223.

Artetxe, Mikel, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho (2018). "Unsupervised Neural Machine Translation". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Atrey, Pradeep K., M. Anwar Hossain, Abdulmotaleb El-Saddik, and Mohan S. Kankanhalli (2010). "Multimodal fusion for multimedia analysis: a survey". In: *Multimedia Syst.* 16.6, pp. 345–379.

Aubry, Mathieu, Daniel Maturana, Alexei A. Efros, Bryan C. Russell, and Josef Sivic (2014). "Seeing 3D Chairs: Exemplar Part-Based 2D-3D Alignment Using a Large Dataset of CAD Models". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pp. 3762–3769.

Babaeizadeh, Mohammad, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine (2018). "Stochastic Variational Video Prediction". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Bachman, Philip, R. Devon Hjelm, and William Buchwalter (2019). "Learning Representations by Maximizing Mutual Information Across Views". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Infor-*

mation Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, pp. 15509–15519.

Bayer, Justin and Christian Osendorfer (2014). "Learning Stochastic Recurrent Networks". In: *CoRR* abs/1411.7610. URL: http://arxiv.org/abs/1411.7610.

Behrmann, Jens, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jörn-Henrik Jacobsen (2019). "Invertible Residual Networks". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 573–582.

Ben-younes, Hedi, Rémi Cadène, Nicolas Thome, and Matthieu Cord (2019). "BLOCK: Bilinear Superdiagonal Fusion for Visual Question Answering and Visual Relationship Detection". In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 8102–8109.

Bengio, Yoshua, Aaron C. Courville, and Pascal Vincent (2013a). "Representation Learning: A Review and New Perspectives". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.8, pp. 1798–1828.

Bengio, Yoshua, Eric Laufer, Guillaume Alain, and Jason Yosinski (2014). "Deep Generative Stochastic Networks Trainable by Backprop". In: *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 226–234.

Bengio, Yoshua, Li Yao, Guillaume Alain, and Pascal Vincent (2013b). "Generalized Denoising Auto-Encoders as Generative Models". In: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.* Pp. 899–907.

Bézenac, Emmanuel de, Ibrahim Ayed, and Patrick Gallinari (2019). "Optimal Unsupervised Domain Translation". In: *CoRR* abs/1906.01292.

Bielski, Adam and Paolo Favaro (2019). "Emergence of Object Segmentation in Perturbed Generative Models". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 7254–7264.

Bora, Ashish, Ajil Jalal, Eric Price, and Alexandros G. Dimakis (2017). "Compressed Sensing using Generative Models". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 537–546.

Bora, Ashish, Eric Price, and Alexandros G. Dimakis (2018). "AmbientGAN: Generative models from lossy measurements". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Bordes, Patrick, Eloi Zablocki, Laure Soulier, Benjamin Piwowarski, and Patrick Gallinari (2019). "Incorporating Visual Semantics into Sentence Representations within a Grounded Space". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pp. 696–707.

Bowman, Samuel R., Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio (2016). "Generating Sentences from a Continuous Space". In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pp. 10–21.

Brock, Andrew, Jeff Donahue, and Karen Simonyan (2019). "Large Scale GAN Training for High Fidelity Natural Image Synthesis". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Burgess, Christopher P., Loïc Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew Botvinick, and Alexander Lerchner (2019). "MONet: Unsupervised Scene Decomposition and Representation". In: *CoRR* abs/1901.11390.

Cesa-Bianchi, Nicolò, David R. Hardoon, and Gayle Leen (2010). "Guest Editorial: Learning from multiple sources". In: *Machine Learning* 79.1-2, pp. 1–3.

Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam (2018a). "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, pp. 833–851.

Chen, Mickaël, Thierry Artières, and Ludovic Denoyer (2019a). "Unsupervised Object Segmentation by Redrawing". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 12705–12716.

Chen, Mickaël and Ludovic Denoyer (2017). "Multi-view Generative Adversarial Networks". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part II*, pp. 175–188.

Chen, Mickaël, Ludovic Denoyer, and Thierry Artières (2018b). "Multi-View Data Generation Without View Supervision". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Chen, Tian Qi, Xuechen Li, Roger B. Grosse, and David Duvenaud (2018c). "Isolating Sources of Disentanglement in Variational Autoencoders". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.* Pp. 2615–2625.

Chen, Tian Qi, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud (2018d). "Neural Ordinary Differential Equations". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pp. 6572–6583.

Chen, Ting, Mario Lucic, Neil Houlsby, and Sylvain Gelly (2019b). "On Self Modulation for Generative Adversarial Networks". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Chen, Xi, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel (2016). "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 2172–2180.

Chung, Junyoung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio (2015). "A Recurrent Latent Variable Model for Sequential Data". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 2980–2988.

Damianou, Andreas C., Carl Henrik Ek, Michalis K. Titsias, and Neil D. Lawrence (2012). "Manifold Relevance Determination". In: *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*.

Denton, Emily L. and Vighnesh Birodkar (2017). "Unsupervised Learning of Disentangled Representations from Video". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 4414–4423.

Denton, Emily L., Soumith Chintala, Arthur Szlam, and Rob Fergus (2015). "Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 1486–1494.

Denton, Emily and Rob Fergus (2018). "Stochastic Video Generation with a Learned Prior". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 1182–1191.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186.

Dinh, Laurent, David Krueger, and Yoshua Bengio (2015). "NICE: Non-linear Independent Components Estimation". In: *3rd International Conference on Learning*

Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings.

Donahue, Chris, Zachary C. Lipton, Akshay Balsubramani, and Julian J. McAuley (2018). "Semantically Decomposing the Latent Spaces of Generative Adversarial Networks". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Donahue, Jeff, Philipp Krähenbühl, and Trevor Darrell (2017). "Adversarial Feature Learning". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Donahue, Jeff and Karen Simonyan (2019). "Large Scale Adversarial Representation Learning". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 10541–10551.

Dumoulin, Vincent, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville (2017a). "Adversarially Learned Inference". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Dumoulin, Vincent, Jonathon Shlens, and Manjunath Kudlur (2017b). "A Learned Representation For Artistic Style". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Durand, Thibaut, Taylor Mordan, Nicolas Thome, and Matthieu Cord (2017). "WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5957–5966.

Ebert, Frederik, Chelsea Finn, Alex X. Lee, and Sergey Levine (2017). "Self-Supervised Visual Planning with Temporal Skip Connections". In: *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, pp. 344–356.

Edwards, Harrison and Amos J. Storkey (2016). "Censoring Representations with an Adversary". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Eslami, S. M. Ali, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E. Hinton (2016). "Attend, Infer, Repeat: Fast Scene Understanding with Generative Models". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 3225–3233.

Finn, Chelsea, Ian J. Goodfellow, and Sergey Levine (2016). "Unsupervised Learning for Physical Interaction through Video Prediction". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 64–72.

Fourati, Nesrine and Catherine Pelachaud (2014). "Emilya: Emotional body expression in daily actions database". In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, pp. 3486–3493.

Fragkiadaki, Katerina, Sergey Levine, Panna Felsen, and Jitendra Malik (2015). "Recurrent Network Models for Human Dynamics". In: *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 4346–4354.

Franceschi, Jean-Yves, Edouard Delasalles, Mickaël Chen, Sylvain Lamprier, and Patrick Gallinari (2020). "Stochastic Latent Residual Video Prediction". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, forthcoming*.

Ganin, Yaroslav and Victor S. Lempitsky (2015). "Unsupervised Domain Adaptation by Backpropagation". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1180–1189.

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, pp. 249–256.

Gong, Mingming, Yanwu Xu, Chunyuan Li, Kun Zhang, and Kayhan Batmanghelich (2019a). "Twin Auxiliary Classifiers GAN". In: pp. 12705–12716.

Gong, Rui, Wen Li, Yuhua Chen, and Luc Van Gool (2019b). "DLOW: Domain Flow for Adaptation and Generalization". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 2477–2486.

Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2672–2680.

Greff, Klaus, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner (2019). "Multi-Object Representation Learning with Iterative Variational Inference". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 2424–2433.

Gregor, Karol, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra (2015). "DRAW: A Recurrent Neural Network For Image Generation". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1462–1471.

Gregor, Karol, George Papamakarios, Frederic Besse, Lars Buesing, and Theophane Weber (2019). "Temporal Difference Variational Auto-Encoder". In: *7th*

International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019.

Gretton, Arthur, Dougal Sutherland, and Wittawat Jitkrittum (2019). *Interpretable comparison of distributions and models*. NeurIPS 2019 Tutorial. URL: http://www.gatsby.ucl.ac.uk/~gretton/papers/neurips19_1.pdf.

Gulrajani, Ishaan, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville (2017). "Improved Training of Wasserstein GANs". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 5767–5777.

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick (2019). "Momentum Contrast for Unsupervised Visual Representation Learning". In: *CoRR* abs/1911.05722.

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick (2017). "Mask R-CNN". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 2980–2988.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification". In: *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 1026–1034.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778.

Hénaff, Olivier J., Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, S. M. Ali Eslami, and Aäron van den Oord (2019). "Data-Efficient Image Recognition with Contrastive Predictive Coding". In: *CoRR* abs/1905.09272.

Higgins, Irina, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner (2017). "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Hsu, Kuang-Jui, Yen-Yu Lin, and Yung-Yu Chuang (2018). "Co-attention CNNs for Unsupervised Object Co-segmentation". In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. Pp. 748–756.

Hsu, Kuang-Jui, Yen-Yu Lin, and Yung-Yu Chuang (2019). "DeepCO3: Deep Instance Co-Segmentation by Co-Peak Search and Co-Saliency Detection". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 8846–8855.

Huang, Gary B., Vidit Jain, and Erik G. Learned-Miller (2007a). "Unsupervised Joint Alignment of Complex Images". In: *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*, pp. 1–8.

Huang, Gary B., Manu Ramesh, Tamara Berg, and Erik Learned-Miller (Oct. 2007b). *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. Tech. rep. 07-49. University of Massachusetts, Amherst.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 448–456.

Ionescu, Catalin, Fuxin Li, and Cristian Sminchisescu (2011). "Latent structured models for human pose estimation". In: *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pp. 2220–2227.

Ionescu, Catalin, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu (2014). "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 36.7, pp. 1325–1339.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros (2017). "Image-to-Image Translation with Conditional Adversarial Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5967–5976.

Ji, Xu, João F Henriques, and Andrea Vedaldi (2019). "Invariant information distillation for unsupervised image segmentation and clustering". In: *International Conference on Computer Vision (ICCV)*.

Kae, Andrew, Kihyuk Sohn, Honglak Lee, and Erik G. Learned-Miller (2013). "Augmenting CRFs with Boltzmann Machine Shape Priors for Image Labeling". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pp. 2019–2026.

Kalchbrenner, Nal, Aäron van den Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu (2017). "Video Pixel Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 1771–1779.

Kanezaki, Asako (2018). "Unsupervised Image Segmentation by Backpropagation". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pp. 1543–1547.

Karl, Maximilian, Maximilian Sölch, Justin Bayer, and Patrick van der Smagt (2017). "Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Karras, Tero, Samuli Laine, and Timo Aila (2019). "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 4401–4410.

Kim, Hyunjik and Andriy Mnih (2018). "Disentangling by Factorising". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 2654–2663.

Kim, Taeksoo, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim (2017a). "Learning to Discover Cross-Domain Relations with Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 1857–1865.

Kim, Taeksoo, Byoungjip Kim, Moonsu Cha, and Jiwon Kim (2017b). "Unsupervised Visual Attribute Transfer with Reconfigurable Generative Adversarial Networks". In: *CoRR* abs/1707.09798.

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kingma, Diederik P. and Max Welling (2014). "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Klami, Arto, Seppo Virtanen, and Samuel Kaski (2013). "Bayesian Canonical correlation analysis". In: *J. Mach. Learn. Res.* 14.1, pp. 965–1003.

Kocaoglu, Murat, Christopher Snyder, Alexandros G. Dimakis, and Sriram Vishwanath (2018). "CausalGAN: Learning Causal Implicit Generative Models with Adversarial Training". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Krishnan, Rahul G., Uri Shalit, and David A. Sontag (2017). "Structured Inference Networks for Nonlinear State Space Models". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pp. 2101–2109.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2017). "ImageNet classification with deep convolutional neural networks". In: *Commun. ACM* 60.6, pp. 84–90.

Lample, Guillaume, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato (2018a). "Unsupervised Machine Translation Using Monolingual Corpora Only". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Lample, Guillaume, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou (2018b). "Word translation without parallel data". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Lample, Guillaume, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, and Marc'Aurelio Ranzato (2017). "Fader Networks: Manipulating Images by Sliding Attributes". In: *Advances in Neural Information Processing*

Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, pp. 5967–5976.

Learned-Miller, Gary B. Huang Erik (May 2014). *Labeled Faces in the Wild: Updates and New Reporting Procedures*. Tech. rep. UM-CS-2014-003. University of Massachusetts, Amherst.

LeCun, Yann, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Lee, Alex X., Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine (2020). "Stochastic Adversarial Video Prediction". In:

Li, Annan, Shiguang Shan, Xilin Chen, Bingpeng Ma, Shuicheng Yan, and Wen Gao (2015). "Cross-pose Face Recognition by Canonical Correlation Analysis". In: *CoRR* abs/1507.08076.

Li, Yingzhen and Stephan Mandt (2018). "Disentangled Sequential Autoencoder". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 5656–5665.

Lim, Jae Hyun and Jong Chul Ye (2017). "Geometric GAN". In: *CoRR* abs/1705.02894.

Lin, Tsung-Yi, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick (2014). "Microsoft COCO: Common Objects in Context". In: *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, pp. 740–755.

Liu, Kuan-Hsien, Ting-Yen Chen, and Chu-Song Chen (2016). "MVC: A Dataset for View-Invariant Clothing Retrieval and Attribute Prediction". In: *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, ICMR 2016, New York, New York, USA, June 6-9, 2016*, pp. 313–316.

Liu, Ming-Yu, Thomas Breuel, and Jan Kautz (2017). "Unsupervised Image-to-Image Translation Networks". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 700–708.

Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (2015). "Deep Learning Face Attributes in the Wild". In: *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 3730–3738.

Locatello, Francesco, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem (2019a). "On the Fairness of Disentangled Representations". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 14584–14597.

Locatello, Francesco, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem (2019b). "Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations".

In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 4114–4124.

Lucic, Mario, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet (2018). "Are GANs Created Equal? A Large-Scale Study". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pp. 698–707.

Lucic, Mario, Michael Tschannen, Marvin Ritter, Xiaohua Zhai, Olivier Bachem, and Sylvain Gelly (2019). "High-Fidelity Image Generation With Fewer Labels". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 4183–4192.

Makhzani, Alireza, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow (2015). "Adversarial Autoencoders". In: *CoRR* abs/1511.05644.

Mathieu, Michaël, Junbo Jake Zhao, Pablo Sprechmann, Aditya Ramesh, and Yann LeCun (2016). "Disentangling factors of variation in deep representation using adversarial training". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 5041–5049.

Minderer, Matthias, Chen Sun, Ruben Villegas, Forrester Cole, Kevin P. Murphy, and Honglak Lee (2019). "Unsupervised learning of object structure and dynamics from videos". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 92–102.

Mino, Ajkel and Gerasimos Spanakis (2018). "LoGAN: Generating Logos with a Generative Adversarial Neural Network Conditioned on Color". In: *17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018, Orlando, FL, USA, December 17-20, 2018*, pp. 965–970.

Mirza, Mehdi and Simon Osindero (2014). "Conditional Generative Adversarial Nets". In: *CoRR* abs/1411.1784.

Misra, Ishan and Laurens van der Maaten (2019). "Self-Supervised Learning of Pretext-Invariant Representations". In: *CoRR* abs/1912.01991.

Miyato, Takeru, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida (2018a). "Spectral Normalization for Generative Adversarial Networks". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Miyato, Takeru and Masanori Koyama (2018b). "cGANs with Projection Discriminator". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Nair, Vinod and Geoffrey E. Hinton (2010). "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pp. 807–814.

Ngiam, Jiquan, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng (2011). "Multimodal Deep Learning". In: *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pp. 689–696.

Nilsback, Maria-Elena and Andrew Zisserman (2007). "Delving into the Whorl of Flower Segmentation". In: *Proceedings of the British Machine Vision Conference 2007, University of Warwick, UK, September 10-13, 2007*, pp. 1–10.

Nilsback, Maria-Elena and Andrew Zisserman (2008). "Automated Flower Classification over a Large Number of Classes". In: *Sixth Indian Conference on Computer Vision, Graphics & Image Processing, ICVGIP 2008, Bhubaneswar, India, 16-19 December 2008*, pp. 722–729.

Nowozin, Sebastian, Botond Cseke, and Ryota Tomioka (2016). "f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 271–279.

Odena, Augustus, Christopher Olah, and Jonathon Shlens (2017). "Conditional Image Synthesis with Auxiliary Classifier GANs". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 2642–2651.

Oord, Aäron van den, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves (2016). "Conditional Image Generation with PixelCNN Decoders". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 4790–4798.

Ostyakov, Pavel, Roman Suvorov, Elizaveta Logacheva, Oleg Khomenko, and Sergey I. Nikolenko (2018). "SEIGAN: Towards Compositional Image Generation by Simultaneously Learning to Segment, Enhance, and Inpaint". In: *CoRR* abs/1811.07630.

Pajot, Arthur, Emmanuel de Bézenac, and Patrick Gallinari (2019). "Unsupervised Adversarial Image Reconstruction". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Parkhi, Omkar M., Andrea Vedaldi, and Andrew Zisserman (2015). "Deep Face Recognition". In: *Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7-10, 2015*, pp. 41.1–41.12.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems*

32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, pp. 8024–8035.

Perez, Ethan, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville (2018). "FiLM: Visual Reasoning with a General Conditioning Layer". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 3942–3951.

Pham, Trung T., Thanh-Toan Do, Niko Sünderhauf, and Ian D. Reid (2018). "SceneCut: Joint Geometric and Object Segmentation for Indoor Scenes". In: *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pp. 1–9.

Radford, Alec, Luke Metz, and Soumith Chintala (2016). "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Razavi, Ali, Aäron van den Oord, and Oriol Vinyals (2019). "Generating Diverse High-Fidelity Images with VQ-VAE-2". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 14837–14847.

Remez, Tal, Jonathan Huang, and Matthew Brown (2018). "Learning to Segment via Cut-and-Paste". In: *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, pp. 39–54.

Rezende, Danilo Jimenez and Shakir Mohamed (2015). "Variational Inference with Normalizing Flows". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1530–1538.

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 1278–1286.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, pp. 234–241.

Rother, Carsten, Thomas P. Minka, Andrew Blake, and Vladimir Kolmogorov (2006). "Cosegmentation of Image Pairs by Histogram Matching - Incorporating a Global Constraint into MRFs". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pp. 993–1000.

Rousseau, François, Lucas Drumetz, and Ronan Fablet (2019). "Residual networks as flows of diffeomorphisms". In: *Journal of Mathematical Imaging and Vision*, pp. 1–11.

Rubanova, Yulia, Tian Qi Chen, and David Duvenaud (2019). "Latent Ordinary Differential Equations for Irregularly-Sampled Time Series". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 5321–5331.

Rubenstein, Paul K., Bernhard Schölkopf, and Ilya O. Tolstikhin (2018). "Learning Disentangled Representations with Wasserstein Auto-Encoders". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li (2015). "ImageNet Large Scale Visual Recognition Challenge". In: *Int. J. Comput. Vis.* 115.3, pp. 211–252.

Sabour, Sara, Nicholas Frosst, and Geoffrey E. Hinton (2017). "Dynamic Routing Between Capsules". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 3856–3866.

Sarhan, Mhd Hasan, Nassir Navab, Abouzar Eslami, and Shadi Albarqouni (2020). "Fairness by Learning Orthogonal Disentangled Representations". In: *CoRR* abs/2003.05707.

Saxe, Andrew M., James L. McClelland, and Surya Ganguli (2014). "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Sbai, Othman, Camille Couprie, and Mathieu Aubry (2018). "Vector Image Generation by Learning Parametric Layer Decomposition". In: *CoRR* abs/1812.05484.

Schüldt, Christian, Ivan Laptev, and Barbara Caputo (2004). "Recognizing Human Actions: A Local SVM Approach". In: *17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004*, pp. 32–36.

Sermanet, Pierre, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine (2018). "Time-Contrastive Networks: Self-Supervised Learning from Video". In: *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pp. 1134–1141.

Shen, Yikang, Shawn Tan, Alessandro Sordoni, and Aaron C. Courville (2019). "Ordered Neurons: Integrating Tree Structures into Recurrent Neural Networks". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Shi, Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo (2015). "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 802–810.

Silberman, Nathan, Derek Hoiem, Pushmeet Kohli, and Rob Fergus (2012). "Indoor Segmentation and Support Inference from RGBD Images". In: *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V*, pp. 746–760.

Simonyan, Karen and Andrew Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Sohn, Kihyuk, Honglak Lee, and Xinchen Yan (2015). "Learning Structured Output Representation using Deep Conditional Generative Models". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 3483–3491.

Srivastava, Nitish and Ruslan Salakhutdinov (2012). "Multimodal Learning with Deep Boltzmann Machines". In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.* Pp. 2231–2239.

Steenkiste, Sjoerd van, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem (2019). "Are Disentangled Representations Helpful for Abstract Visual Reasoning?" In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 14222–14235.

Sun, Liang, Wenjing Kang, Yuxuan Han, and Hongwei Ge (2018). "Multi-View Transformation via Mutual-Encoding InfoGenerative Adversarial Networks". In: *IEEE Access* 6, pp. 43315–43326.

Suter, Raphael, Đorđe Miladinovic, Bernhard Schölkopf, and Stefan Bauer (2019). "Robustly Disentangled Causal Mechanisms: Validating Deep Representations for Interventional Robustness". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 6056–6065.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 3104–3112.

Tang, Qingming, Weiran Wang, and Karen Livescu (2017). "Acoustic Feature Learning via Deep Variational Canonical Correlation Analysis". In: *Interspeech*

2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017, pp. 1656–1660.

Tolstikhin, Ilya O., Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf (2018). "Wasserstein Auto-Encoders". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Tran, Dustin, Rajesh Ranganath, and David M. Blei (2017). "Deep and Hierarchical Implicit Models". In: *CoRR* abs/1702.08896.

Tucker, Ledyard R (1958). "An inter-battery method of factor analysis". In: *Psychometrika* 23.2, pp. 111–136.

Ulyanov, Dmitry, Andrea Vedaldi, and Victor S. Lempitsky (2018). "Deep Image Prior". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 9446–9454.

Unterthiner, Thomas, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly (2018). "Towards Accurate Generative Models of Video: A New Metric & Challenges". In: *CoRR* abs/1812.01717.

Wah, C., S. Branson, P. Welinder, P. Perona, and S. Belongie (2011). *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology.

Wang, Qi, Thierry Artières, Mickaël Chen, and Ludovic Denoyer (2018a). "Adversarial learning for modeling human motion". In: *The Visual Computer*, pp. 1–20.

Wang, Qi, Mickaël Chen, Thierry Artières, and Ludovic Denoyer (2018b). "Transferring style in motion capture sequences with adversarial learning". In: *26th European Symposium on Artificial Neural Networks, ESANN 2018, Bruges, Belgium, April 25-27, 2018*.

Wang, Weiran, Raman Arora, Karen Livescu, and Jeff A. Bilmes (2015). "On Deep Multi-View Representation Learning". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1083–1092.

Wang, Xiaolong and Abhinav Gupta (2016). "Generative Image Modeling Using Style and Structure Adversarial Networks". In: *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, pp. 318–335.

Xia, Xide and Brian Kulis (2017). "W-Net: A Deep Model for Fully Unsupervised Image Segmentation". In: *CoRR* abs/1711.08506.

Xuan, Qi, Zhuangzhi Chen, Yi Liu, Huimin Huang, Guanjun Bao, and Dan Zhang (2018). "Multiview generative adversarial network and its application in pearl classification". In: *IEEE Transactions on Industrial Electronics* 66.10, pp. 8244–8252.

Xue, Tianfan, Jiajun Wu, Katherine L. Bouman, and Bill Freeman (2016). "Visual Dynamics: Probabilistic Future Frame Synthesis via Cross Convolutional

Networks". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 91–99.

Yang, Jianwei, Anitha Kannan, Dhruv Batra, and Devi Parikh (2017). "LR-GAN: Layered Recursive Generative Adversarial Networks for Image Generation". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Yi, Zili, Hao (Richard) Zhang, Ping Tan, and Minglun Gong (2017). "DualGAN: Unsupervised Dual Learning for Image-to-Image Translation". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 2868–2876.

Yu, Fisher, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao (2015). "LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop". In: *CoRR* abs/1506.03365.

Zhang, Han, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena (2019). "Self-Attention Generative Adversarial Networks". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 7354–7363.

Zhang, Meng, Yang Liu, Huanbo Luan, and Maosong Sun (2017a). "Adversarial Training for Unsupervised Bilingual Lexicon Induction". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pp. 1959–1970.

Zhang, Meng, Yang Liu, Huanbo Luan, and Maosong Sun (2017b). "Earth Mover's Distance Minimization for Unsupervised Bilingual Lexicon Induction". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pp. 1934–1945.

Zhang, Richard, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang (2018). "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 586–595.

Zhao, Bo, Xiao Wu, Zhi-Qi Cheng, Hao Liu, Zequn Jie, and Jiashi Feng (2018). "Multi-View Image Generation from a Single-View". In: *2018 ACM Multimedia Conference on Multimedia Conference, MM 2018, Seoul, Republic of Korea, October 22-26, 2018*, pp. 383–391.

Zhao, Hengshuang, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia (2017). "Pyramid Scene Parsing Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 6230–6239.

Zhou, Yanzhao, Yi Zhu, Qixiang Ye, Qiang Qiu, and Jianbin Jiao (2018). "Weakly Supervised Instance Segmentation Using Class Peak Response". In: *2018 IEEE*

Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pp. 3791–3800.

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros (2017). "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 2242–2251.