Université de Montréal

Locality and compositionality in representation learning for complex visual tasks

par

Tristan Sylvain

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en Informatique

March 10, 2021

Université de Montréal

Faculté des arts et des sciences

Cette thèse intitulée

Locality and compositionality in representation learning for complex visual tasks

présentée par

Tristan Sylvain

a été évaluée par un jury composé des personnes suivantes :

Liam Paull
(président-rapporteur)

Yoshua Bengio
(directeur de recherche)

Devon Hjelm
(codirecteur)

Hugo Larochelle
(membre du jury)

Zeynep Akata
(examinateur externe)

Martin Arguin
(représentant du doyen de la FESP)

Résumé

L'utilisation d'architectures neuronales profondes associée à des innovations spécifiques telles que les méthodes adversarielles, l'entraînement préalable sur de grands ensembles de données et l'estimation de l'information mutuelle a permis, ces dernières années, de progresser rapidement dans de nombreuses tâches de vision par ordinateur complexes telles que la classification d'images de catégories préalablement inconnues (apprentissage zéro-coups), la génération de scènes ou la classification multimodale. Malgré ces progrès, il n'est pas certain que les méthodes actuelles d'apprentissage de représentations suffiront à atteindre une performance équivalente au niveau humain sur des tâches visuelles arbitraires et, de fait, cela pose des questions quant à la direction de la recherche future.

Dans cette thèse, nous nous concentrerons sur deux aspects des représentations qui semblent nécessaires pour atteindre de bonnes performances en aval pour l'apprentissage des représentations : la localité et la compositionalité. La localité peut être comprise comme la capacité d'une représentation à retenir des informations locales. Ceci sera pertinent dans de nombreux cas, et bénéficiera particulièrement à la vision informatique, domaine dans lequel les images naturelles comportent intrinsèquement des informations locales, par exemple des parties pertinentes d'une image, des objets multiples présents dans une scène... D'autre part, une représentation compositionnelle peut être comprise comme une représentation qui résulte d'une combinaison de parties plus simples. Les réseaux neuronaux convolutionnels sont intrinsèquement compositionnels, et de nombreuses images complexes peuvent être considérées comme la composition de sous-composantes pertinentes : les objets et attributs individuels dans une scène, les attributs sémantiques dans l'apprentissage zéro-coups en sont deux exemples. Nous pensons que ces deux propriétés détiennent la clé pour concevoir de meilleures méthodes d'apprentissage de représentations.

Dans cette thèse, nous présentons trois articles traitant de la localité et/ou de la compositionnalité, et de leur application à l'apprentissage de représentations pour des tâches visuelles complexes.

Dans le premier article, nous introduisons des méthodes de mesure de la localité et de la compositionnalité pour les représentations d'images, et nous démontrons que les représentations locales et compositionnelles sont plus performantes dans l'apprentissage zéro-coups. Nous utilisons également ces deux notions comme base pour concevoir un nouvel algorithme d'apprentissage des représentations qui atteint des performances de pointe dans notre cadre expérimental, une variante de l'apprentissage "zéro-coups" plus difficile où les informations externes, par exemple un pré-entraînement sur d'autres ensembles de données d'images, ne sont pas autorisées.

Dans le deuxième article, nous montrons qu'en encourageant un générateur à conserver des informations locales au niveau de l'objet, à l'aide d'un module dit de similarité de graphes de scène, nous pouvons améliorer les performances de génération de scènes. Ce modèle met également en évidence l'importance de la composition, car de nombreux composants fonctionnent individuellement sur chaque objet présent. Pour démontrer pleinement la portée de notre approche, nous effectuons une analyse détaillée et proposons un nouveau cadre pour évaluer les modèles de génération de scènes.

Enfin, dans le troisième article, nous montrons qu'en encourageant une forte information mutuelle entre les représentations multimodales locales et globales des images médicales en 2D et 3D, nous pouvons améliorer la classification et la segmentation des images. Ce cadre général peut être appliqué à une grande variété de contextes et démontre les avantages non seulement de la localité, mais aussi de la compositionnalité, car les représentations multimodales sont combinées pour obtenir une représentation plus générale.

Mots clef: apprentissage profond, localité, compositionnalité, apprentissage zéro-coups, modèles génératifs, données multi-modales.

Abstract

The use of deep neural architectures coupled with specific innovations such as adversarial methods, pre-training on large datasets and mutual information estimation has in recent years allowed rapid progress in many complex vision tasks such as zero-shot learning, scene generation, or multi-modal classification. Despite such progress, it is still not clear if current representation learning methods will be enough to attain human-level performance on arbitrary visual tasks, and if not, what direction should future research take.

In this thesis, we will focus on two aspects of representations that seem necessary to achieve good downstream performance for representation learning: locality and compositionality. Locality can be understood as a representation's ability to retain local information. This will be relevant in many cases, and will specifically benefit computer vision where natural images inherently feature local information, i.e. relevant patches of an image, multiple objects present in a scene... On the other hand, a compositional representation can be understood as one that arises from a combination of simpler parts. Convolutional neural networks are inherently compositional, and many complex images can be seen as composition of relevant sub-components: individual objects and attributes in a scene, semantic attributes in zero-shot learning are two examples. We believe both properties hold the key to designing better representation learning methods.

In this thesis, we present 3 articles dealing with locality and/or compositionality, and their application to representation learning for complex visual tasks.

In the first article, we introduce ways of measuring locality and compositionality for image representations, and demonstrate that local and compositional representations perform better at zero-shot learning. We also use these two notions as the basis for designing class-matching deep info-max, a novel representation learning algorithm that achieves state-of-the-art performance on our proposed "Zero-shot from scratch" setting, a harder zero-shot setting where external information, e.g. pre-training on other image datasets is not allowed.

In the second article, we show that by encouraging a generator to retain local object-level information, using a scene-graph similarity module, we can improve scene generation performance. This model also showcases the importance of compositionality as many components operate individually on each object present. To fully demonstrate the reach of our approach, we perform detailed analysis, and propose a new framework to evaluate scene generation models.

Finally, in the third article, we show that encouraging high mutual information between local and global multi-modal representations of 2D and 3D medical images can lead to improvements in image classification and segmentation. This general framework can be applied to a wide variety of settings, and demonstrates the benefits of not only locality, but also of compositionality as multi-modal representations are combined to obtain a more general one.

Keywords: Deep learning, Locality, Compositionality, Zero-shot learning, Generative Modeling, Multi-modal data.

Contents

Résumé 5

Abstract 7

List of tables 15

List of figures 17

List Of Acronyms And Abbreviations 19

Acknowledgements 23

Chapter 1. Introduction 25

1.1. Thesis structure 27

Chapter 2. Background 29

2.1. Mathematical concepts 29
2.1.1. Kullbach-Leibler divergence and entropy 29
2.1.2. Maximum-likelihood estimation 31
2.1.3. Mutual information 31

2.2. Specific data types 32
2.2.1. Uni- and multi-modal data 32
2.2.2. Images 32
2.2.3. Layouts 32
2.2.4. Scene graphs 32
2.2.5. Text 33
2.2.6. MRI sequences 33

2.3. Machine learning 33
2.3.1. Parameters and hyper-parameters 33
2.3.2. Learning objective, evaluation metric 34
2.3.3. Dataset splits and training phases 35

2.3.4. Hyper-parameter selection 35
2.3.5. Supervised learning 36
2.3.6. Unsupervised learning 36

2.4. Zero-shot learning 37
2.4.1. Representing classes 37
2.4.2. Datasets 38

2.5. Deep learning 38
2.5.1. Neural networks 38
2.5.2. Mathematical formalism 39
2.5.2.1. Forward propagation 39
2.5.3. Training neural networks using back-propagation 40
2.5.3.1. Definitions 40
2.5.3.2. Back-propagation 41
2.5.4. Common types of neural networks and layers 42
2.5.4.1. Multi-Layer Perceptron 42
2.5.4.2. Convolutional Neural Networks 42
2.5.4.3. Residual networks 44
2.5.5. Specific types of CNNs 45
2.5.5.1. Alexnet and VGG 45
2.5.5.2. Resnet variants 46
2.5.6. Auto-encoders 46

2.6. Generative Models 46
2.6.1. Introduction 46
2.6.2. Variational autoencoders 48
2.6.2.1. Components of a variational autoencoder 48
2.6.2.2. Variational inference 48
2.6.2.3. Training a VAE 49
2.6.3. Generative adversarial networks 49
2.6.3.1. Mathematical definition and notations 50
2.6.4. Conditional GANs 50
2.6.4.1. Conditional generation 50
2.6.4.2. Mathematical formalism 51

Chapter 3. Prologue to first article 53

3.1. Context ... 53

3.2. Contributions ... 53

3.3. Aftermath ... 54

Chapter 4. Locality and Compositionality in Zero-Shot Learning ... 55

4.1. Introduction ... 55

4.2. Principles that lead to good ZSL performance ... 56
4.2.1. Compositionality ... 57
4.2.2. Locality ... 57
4.2.3. Compositionality and locality with image data ... 58

4.3. Zero-Shot Learning from scratch ... 59

4.4. Methods ... 60
4.4.1. General approach ... 60
4.4.2. Parts classification for CUB evaluation ... 61
4.4.3. Measuring mutual information between local features of different images ... 61
4.4.4. TRE evaluation ... 62
4.4.5. Experimental Setup ... 62

4.5. Results and Discussion ... 63
4.5.1. Does locality help ZSL, and can local representations be learned? ... 63
4.5.2. How do different models encode information locally? ... 65
4.5.3. Do compositional representations perform better for ZSL? ... 67

4.6. Conclusion and future work ... 68

Chapter 5. Prologue to the second paper ... 69

5.1. Article details ... 69

5.2. Context ... 69

5.3. Contributions ... 69

Chapter 6. Object-Centric Image Generation from Layouts ... 71

6.1. Introduction ... 71

6.2. Related Work ... 73
6.2.0.1. Conditional scene generation ... 73

6.2.0.2. Layout-based synthesis 73
6.2.0.3. Scene-graphs and image matching 74

6.3. Proposed Method 75
6.3.1. Scene-Graph Similarity Module 75
6.3.2. Instance-Aware Conditioning 76
6.3.3. Architecture 77
6.3.3.1. Generator 78
6.3.3.2. Discriminator 78
6.3.4. Loss Functions 78

6.4. Experiments 79
6.4.1. Datasets 79
6.4.2. Implementation and Training Details 79
6.4.3. Baselines 80
6.4.4. Evaluation 80
6.4.4.1. Our proposed metric: SceneFID 80
6.4.5. Quantitative Results 81
6.4.6. Qualitative Results 83

6.5. Conclusion 84

Chapter 7. Prologue to the third paper 87

7.1. Article details 87

7.2. Context 87

7.3. Contributions 87

7.4. Aftermath 88

Chapter 8. Cross-Modal Information Maximization for Medical Imaging: CMIM 89

8.1. Introduction 89

8.2. Related Work 90
8.2.1. Cross-modality training 90
8.2.2. Mutual information maximization 90

8.3. Proposed method 91

8.3.1. Mutual information maximization ... 92
8.3.2. Cross-modality mutual information ... 92
8.3.3. Design choices ... 92

8.4. Experiments ... 93
8.4.1. Experimental setup ... 93
8.4.2. Classification task ... 94
8.4.3. Semantic segmentation task ... 94
8.4.4. Baselines ... 94
8.4.5. Implementation and Training Details ... 95
8.4.6. Results ... 95

8.5. Conclusion ... 95

Chapter 9. General conclusion ... 97

9.1. Improving our understanding of locality and compositionality ... 98

9.2. Towards better modeling of complex scenes ... 98

9.3. Final remarks ... 99

References ... 101

Appendix A. Appendix for the first paper ... 117

A.1. Extracting local and global features from a CNN encoder ... 117
A.1.1. Explaining local and global features ... 117
A.1.2. Explaining the role of local and global features for MI computation ... 117

A.2. Implementation details ... 118

A.3. MI heatmaps comparing CMDIM with different local losses ... 118

A.4. Parts ratio and relationship to different measures of image similarity. ... 119

A.5. How local do the representations need to be? ... 120

A.6. Bird parts location maps ... 120

A.7. Class Matching DIM ... 121

A.8. Definition of TRE ... 121

A.9. Architectures and datasets details ... 122

A.10. TRE Ratio Values 123
A.11. F1 part average scores 124
A.12. Full plots 126

Appendix B. Appendix for the second paper 129

B.1. Comparison with Semi-Parametric Methods 129
B.2. Dataset statistics 129
B.3. Spatial Relationships used for Generating the Scene-Graph 130
B.4. A Note on Evaluation 130
B.5. Complexity of scenes 131
B.6. Implementation and Training Details 131
B.7. Additional Qualitative Results 131

List of tables

6.1 Performance on 64, 128 and 256 dimension images 82
6.2 SceneFID scores for both datasets 83
6.3 User study results 84
6.4 Quantitative comparison of different ablated versions of our model on the COCO-Stuff dataset 84

8.1 Results on Open-i 93
8.2 Dice similarity coefficient (DSC) results on the BRATS test sets (%) in the "enhancing" setting introduced in Havaei et al. [2016] 94

A.1 Details of the datasets used 122
A.2 Basic 128x128 architecture details. 122
A.3 AlexNet 128x128 architecture details. 122
A.1 TRE ratio values for the CUB dataset 123
A.2 TRE ratio values for the AWA2 dataset 123
A.3 TRE ratio values for the SUN dataset 123
A.1 Part average F1 score, CUB dataset, basic encoder. 124
A.2 ZSL accuracy, comparing the different local losses. 125
A.3 ZSL accuracy, comparing the different local models. 126

B.1 Comparison of our method with the semi-parametric method PasteGAN Li et al. [2019a] 129
B.1 Statistics of the COCO-Stuff and Visual Genome datasets. 130

List of figures

2.1 Fake digits from the MNIST dataset, generated by a generative model conditioned on specific digit values . . . 30

2.2 Representation of grid search and random search. The black dots are hyper-parameter combinations investigated by the search algorithm. The curves in green plot model validation performance for different values of the important parameter. The plot in yellow represents the small effect the unimportant parameter has on model performance. Figure taken from [Bergstra and Bengio, 2012] . . . 36

2.3 Representation of a convolution operation on a image with a kernel of size 3×3 . 40

2.4 A fully-connected neural network with one hidden layer . . . 40

2.5 Representation of a convolution operation on a image with a kernel of size 3×3 . 44

2.6 Representation of the input (blue) and output (green) of a max-pooling operation. 44

2.7 Representation of a type of residual block . . . 45

2.8 Representation of an autoencoder consisting of 3 layers . . . 47

4.1 Dataset samples . . . 58

4.2 Parts F1 score for all models on CUB . . . 63

4.3 Relative improvement in terms of ZSL accuracy with respect to models trained without the auxiliary loss . . . 63

4.4 Mutual Information heatmaps . . . 65

4.5 Relationship between TRE ratio and ZSL accuracy . . . 67

4.6 Comparison between averaging representations and averaging predictions . . . 68

6.1 Donut experiment for our model and the baselines . . . 72

6.2 Examples of common failure modes of layout-to-image generation models . . . 73

6.3 The SGSM module . . . 75

6.4 Instance aware conditioning . . . 76

6.5 Overview of our OC-GAN model . . . 77

6.6 Our method on 128×128 COCO-Stuff test set images 81

8.1 Graphical representation of our method on multiple modalities 91

A.1 Global and local feature explanation 117
A.2 Global and local feature for MI estimation 117
A.3 Visualization of locality comparing encoders trained with CMDIM's loss and the proposed local losses 118
A.4 Relationship between the parts score and different measures of similarity 119
A.5 Comparing pre and post-pooling (respectively *Small* and *Big*) features in terms of ZSL accuracy 119
A.6 Comparison between averaging representations VS averaging scores for all models. 127
A.7 Comparison between small and big receptive field for all models. 128

B.1 Architecture diagrams for (a) Generator (b) Generator ResBlocks (c) Image Discriminator 132
B.2 Architecture diagrams for (a) Scene-Graph Encoder (b) Conditioning Module (c) Mask Net (d) Object Discriminator 133
B.3 128×128 COCO-Stuff test set images, taken from our method (OC-GAN) and multiple competitive baselines. 134
B.4 128×128 Visual Genome test set images, taken from our method (OC-GAN) and the LostGAN baseline. 135

List Of Acronyms And Abbreviations

GAN	Generative adversarial networks
DCGAN	A type of GAN using convolutional layers
CNN	Convolutional Neural Network
VGG19	A type of CNN (acronym for Vision and Geometry Group at Oxford)
GCN	Graph convolutional network, a variant of CNNs used on graphs
FC	Fully connected (refers to a type of neural network)
ReLU	Rectified linear unit, a type of activation function
MLP	Multi-layer perceptron, a feedforward fully connected neural network
ResBlocks	Residual blocks, a set of layers with a residual skip connection between the input and the result of the final convolution.

ResNet	Residual network, a specific convolutional architecture composed of ResBlocks
VAE	Variational auto-encoder
AAE	Adversarial auto-encoder
InfoMax	Mutual information maximization
DIM	Deep InfoMax, a representation learning technique
TRE	Tree Reconstruction Error, a measure of compositionality
AM-DIM	Augmented Multiscale Deep InfoMAx, a representation learning technique
CM-DIM	Class Matching Deep InfoMAx, a representation learning technique
AC / LC	Auxiliary classifier / Linear classifier, components used in CM-DIM
CMIM	Cross Modal information maximization, the architecture introduced in the third contribution.
OC-GAN	Object Centric Generative Adversarial Network, a model introduced in the second paper

SGSM	Scene Graph Similarity Module, a component introduced in the second paper
HeMIS	Hetero-modal image segmentation, a baseline considered in the third paper
ZSL	Zero-shot learning
CUB	The Caltech Birds dataset
AWA2	The Animals with Attributes dataset, second version
SUN	The Scene Understanding dataset
COCO(-Stuff)	The MSCOCO dataset (with the CocoStuff additional annotations)
VG	The Visual Genome dataset
BRATS	A dataset introduced in the Brain Tumor Segmentation challenge
MRI	Magnetic resonance imagery
FLAIR	Fluid-attenuated inversion recovery, a MRI sequence
T1W, T1C, T2	Other MRI sequences

Adam	Adaptive Momentum, a type of gradient descent algorithm
AUC	Area under the Curve, an evaluation metric
IS	Inception score, an evaluation metric for GANs
FID	Frechet Inception Distance, an evaluation metric for GANs
CA	Classification Accuracy, an evaluation metric for GANs

Acknowledgements

I consider myself extremely fortunate to have met and worked with many inspired and talented people on my journey towards my doctorate. I am very grateful to my advisor Yoshua Bengio for giving me the chance to do my PhD research in the extraordinary environment that is Mila. His encouragement, advice and support have been essential for me during the course of my PhD. I am likewise deeply thankful to my co-supervisor Devon Hjelm for advising me on numerous projects related to representation learning, and always contributing valuable advice that had a huge impact on the publications we were involved in. Many of the research projects I participated in were collaborations with industry. I would like to extend a warm thank you to Shikhar Sharma, Pengchuan Zhang, Eric Yuan, Saeed Amizadeh, Xiaotian Han and Jianwei Yang for making my internships at MSR extremely productive and enjoyable. I would also like to thank Lisa Di Jorio for making my time at Imagia a great learning experience.

In addition to the people who mentored me, I would like to thank the coauthors of research projects I was involved in and have not mentioned already: Linda Petrini, ByungIn Yoo, Margaux Luck, Andrea Lodi, Heloise Cardinal, Adriana Romero, Pierre-Luc Carrier, Akram Erraqabi, Tess Berthier, Francis Dutil.

I would further like to thank Dzmitry Bahdanau, Dimitry Serdyuk, César Laurent, Florian Bordes, Ahmed Touati, Benjamin Scellier, Damien Arnol, Michał Drożdżal, Ioannis Mitliagkas and Mohammad Havaei for valuable advice given during my PhD.

I am grateful to my friends for helping to make the winter less cold, and life in Montreal very enjoyable during my stay.

Last, but not least, I would like to thank my family. My father, Frank Sylvain, helped make me the scientist I am today by introducing me to mathematics and programming at a young age. My mother, Dominique Sylvain, taught me the love of reading and writing, and her natural warmth and good nature made me the person I am today. My brother Virgile has been a great source of comfort and friendship my whole life and even more so in recent months during Covid-related lockdown. My dog, Yuki, has helped me keep a positive mindset during the worst and best of times, and deserves every belly scratch I can give.

Chapter 1

Introduction

Achieving high-level visual understanding of images has been one of the major goals of computer vision since its inception. Initially, researchers were extremely optimistic about the deadlines: in 1966 Seymour Papert from the MIT AI lab proposed a plan to identify objects and other important elements of arbitrary scenes as part of a summer project [Papert, 1966]. With the advent of deep learning, the benefit of hindsight, and decades of progress in computer vision research, we have now reached some important milestones such as human-level image classification [He et al., 2015], or photo-realistic image synthesis [Zhang et al., 2016a, StackGAN]. Despite this, many challenges remain, mostly among higher-level visual reasoning tasks, such as zero-shot learning [Larochelle et al., 2008] or complex scene generation.

Computer vision pipelines for high-level tasks usually rely on progress made on low-level perceptual tasks. Commonly, they will borrow visual representations obtained from simpler perceptive tasks, and use them as the basic building block on which to construct more complex neural reasoning modules. As an example, zero-shot learning commonly relies on image representations extracted from convolutional neural networks [LeCun et al., 1989, CNNs] pre-trained on the Imagenet dataset [Russakovsky et al., 2015, Imagenet]. Similarly, recent visual question answering models [Chen et al., 2019] rely on pre-trained object detection models [Redmon et al., 2016, YOLO]. This is in large part possible because simple visual tasks, such as image classification or segmentation, have seen large performance gains in recent years, and are now mature enough for large scale deployment in practical applications. It is also desirable: methods relying on good transferred representations outperform others trained from scratch, as shown in our first presented work, or in the aforementioned case of visual question answering [Chen et al., 2019, Meta Module Networks]. As a result, for higher-level tasks, which make heavy use of these more basic perceptual modules, guaranteeing good representations i.e. visual representations that will result in good generalization for the downstream task, is essential.

In this dissertation, we will be focusing exclusively on representations parametrized by deep neural networks. Deep Learning is a machine learning paradigm that focuses on training deep neural networks, that is neural networks with multiple, and usually many layers. This is in contrast to the classical approach to many computer vision problems which used image transforms as a means of automatically extracting features [Lowe, 1999, SIFT]. In our specific context, deep learning has two main advantages. First and foremost, it has resulted in substantial performance gains in most areas of computer vision (among other fields) it has been applied to. As of the date this thesis is submitted, deep neural networks are essential components of state-of-the-art approaches to semantic segmentation [Zhang et al., 2020, Yuan et al., 2019], image classification [Pham et al., 2020, Foret et al., 2020, Yoo et al., 2020], object detection [Lehner et al., 2019] and image generation [Song et al., 2020, Parmar et al., 2018], among others. The second, more subtle reason, is adaptability. Deep neural networks can be generally applied with minor adaptations to multiple different computer vision fields. Models that perform well on image classification have per instance been applied to semantic segmentation and object-detection [Bui et al., 2016]. This point is one of the main reasons to be optimistic about one day creating a universal vision pipeline.

In keeping with the end goal of contributing to solving arbitrary tasks, this dissertation focuses on the following problem: what fundamental properties of visual representations lead to good performance, in particular when applying these representations to a complex downstream task? As discussed in the presented works, we will focus on two properties in particular that were found to be strongly linked to good downstream performance of representations: *locality* and *compositionality*.

In the context of this dissertation, a local representation is one that retains local information, or explicitly depends on relevant local cues. This can be measured in many ways, an example being to evaluate the performance of a machine learning model trained to take as input the representation and output the locations of relevant parts of the input (e.g. the beak and other biologically relevant parts for birds). Importantly, convolutional neural networks are local themselves, as higher-level neurons usually see a limited part of the input image each: the actual receptive field being quite small in practice. Locality has also been a common fixation of recent research, being the underlying motivation behind such advances as visual co-attention [Lu et al., 2016a], or self-attention [Zhang et al., 2019a]. In each of the proposed works we show, explicitly or implicitly through improved performance, that encouraging representations to retain local information leads to performance improvements and other benefits.

On the other hand, a compositional representation is one that arises from combinations of simpler parts. The cognitive science literature has focused extensively on compositional representations [Biederman, 1987, Hoffman and Richards, 1984] when studying the ability of intelligent agents to generalize to new concepts. In addition, they have been successfully applied to computational linguistics [Tian et al., 2016], generative models [Higgins et al., 2017a], meta learning [Alet et al., 2018, Tokmakov et al., 2018], and now zero-shot learning with the first presented work [Sylvain et al., 2020a]. As in the case of locality, convolutional neural networks are also inherently compositional[1]: the representation that arises from a CNN can be understood as a combination of representations of image patches extracted by lower-level neurons. While there is no inherent measure of compositionality, our first presented work makes a contribution in that direction. Our first work directly shows the benefits of compositionality, whereas the other two propose approaches that allow models to better handle the compositional nature of their data.

The research presented in this thesis can be seen as a series of steps towards solving our proposed problem. Each article has in common a focus on detailing the advantages and disadvantages of a specific representation learning algorithm, applied to a challenging visual task, such as zero-shot learning or image synthesis. Our first contribution focuses on showing that measures of locality and compositionality are strongly connected to zero-shot learning performance. We additionally propose a novel way of measuring compositionality, and a new representation learning algorithm, CM-DIM, that takes advantage of the insights previously gleaned. The second proposed work is more empirical: we improve scene generation from layouts by introducing a set of improvements and an novel scene-graph retrieval module, both relying on, and encouraging the locality of the representations involved. Many components of the architecture function independently on each object, highlighting the fact that a scene arises as a composition of different objects. Finally, the third proposed work uses advances in mutual information estimation between local and global features, to improve representation learning for 2D and 3D medical data.

1.1. Thesis structure

This thesis presents three articles that focus on different aspects of representation learning for complex visual tasks. Chapter 2 covers the shared background of the 3 articles. Chapters 3 through 8 contain the articles themselves, and a prologue for each of them. Lastly, chapter 9 presents the general conclusion of this dissertation.

[1]The compositionality is however of a somewhat different nature for CNNs. In computational linguistics or cognitive science, the compositionality is dynamic: we don't always compose the same functions. This is not the case for CNNs which can be understood as compositions of the same underlying convolutional operation.

Chapter 2

Background

2.1. Mathematical concepts

Later sections rely on a small number of mathematical concepts and notations that we will introduce here.

With X a random variable, and $\mathbb{P}_X$ a probability distribution over X, the expectation of a function f according to p will be written as:

$$\mathbb{E}_{x \sim \mathbb{P}_X}[f(x)]. \tag{2.1.1}$$

One of the most important probability distributions in our case is the Normal distribution. In the following chapters, we will denote the Normal distribution of mean μ and covariance matrix Σ by $\mathcal{N}(\mu, \Sigma)$.

Given two random variables X and Y, the conditional probability of X given Y is denoted:

$$\mathbb{P}(X|Y). \tag{2.1.2}$$

It corresponds to the probability of X occurring, knowing that Y has occurred, or its probability density if X is continuous-valued. Later in the thesis we will use the same notation as well for probability or probability density, with the interpretation depending on the type of the random variable.

In later chapters we will consider distributions conditioned on vectors (e.g. of attributes in the first article). Figure 2.1 gives an example of samples drawn from a distribution conditioned on specific values of digits. Each row corresponds to a set of samples drawn from the distribution conditioned on a specific digit value.

2.1.1. Kullbach-Leibler divergence and entropy

We will often need to measure how two probability distributions p and q defined over a set $\mathcal{X}$ differ from one another. The need for an estimation of the proximity of two probability distributions will be highlighted per instance in the section on generative models. A common

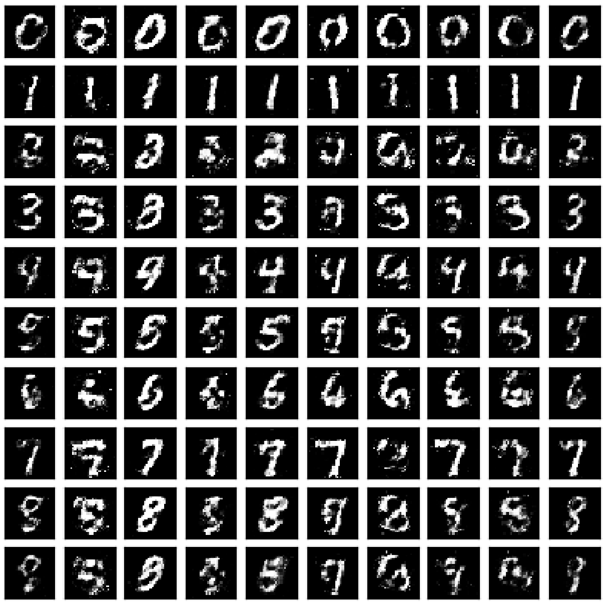


Fig. 2.1. Fake digits from the MNIST dataset, generated by a generative model conditioned on specific digit values

means of doing so is the Kullbach-Leibler (KL) divergence, defined as:

$$\mathcal{D}_{KL}(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{x\sim\mathbb{P}}\Big[\log\frac{\mathbb{P}(x)}{\mathbb{Q}(x)}\Big]. \tag{2.1.3}$$

It is always non-negative. In some of the presented articles, we will make use of another useful definition of the KL divergence called the Donsker Varadhan formulation [Donsker and Varadhan, 1975]. For two probability distributions $\mathbb{P}$ and $\mathbb{Q}$, we have that:

$$\mathcal{D}_{KL}(\mathbb{P}, \mathbb{Q}) = \sup_{T} \mathbb{E}_{\mathbb{P}}[T] - \log \mathbb{E}_{\mathbb{Q}}[e^T], \tag{2.1.4}$$

where the supremum is taken over the set of functions T for which the above two expectations are finite. A direct consequence of this result is that for a family $\mathcal{F}$ of functions (satisfying the condition that the expectations are finite), we have that:

$$\mathcal{D}_{KL}(p, q) \geq \sup_{T\in\mathcal{F}} \mathbb{E}_{\mathbb{P}}[T] - \log \mathbb{E}_{\mathbb{Q}}[e^T]. \tag{2.1.5}$$

On a separate note, we can define the *entropy* of $\mathbb{P}$ as:

$$\mathcal{H}(\mathbb{P}) = -\mathbb{E}_{x\sim\mathbb{P}} \log \mathbb{P}(x), \tag{2.1.6}$$

and the *cross-entropy* of $\mathbb{P}$ and $\mathbb{Q}$ as:

$$\mathcal{H}(\mathbb{P}, \mathbb{Q}) = -\mathbb{E}_{x\sim\mathbb{P}} \log \mathbb{Q}(x). \tag{2.1.7}$$

2.1.2. Maximum-likelihood estimation

Log-likelihood estimation is a means of estimating a model's parameters which is very commonly used in the machine learning community. As its name indicates, the log-likelihood is the logarithm of the likelihood. The likelihood is a function of the parameters of the model, given data. It is defined as the probability of the observed data given the model's parameters. Intuitively, a high log-likelihood corresponds to a model that assigns a large probability to the actually observed outcome.
For a probability distribution ρ over a model output, given θ the parameters of the model, and the set of observed values $(x_1, \cdots, x_n)$, we can define the log-likelihood as:

$$\log \rho(x_1, \cdots, x_n; \theta). \tag{2.1.8}$$

If the observations are independent and identically distributed, as is common in most cases in practice, we can further write this:

$$\sum_{i=1}^{n} \log \rho(x_i; \theta). \tag{2.1.9}$$

The log-likelihood estimator $\hat{\theta}_{\text{ML}}$ can be then expressed as:

$$\hat{\theta}_{\text{ML}} = \arg\max_{\theta} \sum_{i} \log \rho(x_i; \theta). \tag{2.1.10}$$

While other choices are possible, the maximum likelihood estimator is commonly used when estimating model parameters from training data.

2.1.3. Mutual information

In general terms, mutual information is a measure of the dependence of two random variables. Given two random variables X and Y, we formally define their mutual information as:

$$I(X;Y) = \mathcal{D}_{KL}(\mathbb{P}_{(X,Y)}, \mathbb{P}_X \otimes \mathbb{P}_Y), \tag{2.1.11}$$

where $\mathbb{P}_{(X,Y)}$ is the joint distribution, and $\mathbb{P}_X \otimes \mathbb{P}_Y$ is the product of the marginal distributions. In the articles we present (specifically the first and third), we rely on estimating the mutual information between two representations. This multivariate problem is intractable in practice, therefore we rely on the differentiable mutual information estimator introduced in [MINE Belghazi et al., 2018a].

MINE (mutual information neural estimator) maximizes a lower bound on the mutual information between two probability distributions by replacing the right hand side of equation 2.1.11 by its lower bound in equation 2.1.5. The family of functions $\mathcal{F}$ mentioned previously is chosen to be the set of functions parametrized by a neural network, making the estimator fully differentiable.

Given this estimator, we are able, in the presented articles, to perform representation learning by maximizing mutual information between different representations or views of the data.

2.2. Specific data types

2.2.1. Uni- and multi-modal data

In general terms, a modality refers to the way in which something is experienced. In machine learning, a modality can broadly be defined as a specific type of data source (such as images, text) of a different nature than other potential data sources. Many datasets are uni-modal: they feature a single modality, such as the Imagenet dataset [Imagenet Russakovsky et al., 2015] which features images of different entities. Humans however experience perception as an inherently multi-modal phenomenon: we hear sounds, see images, taste, often simultaneously. Multi-modal data is therefore also highly relevant to ongoing research, and is the subject of the third article presented.

2.2.2. Images

Images are a common data type considered in machine learning problems. They commonly feature 1 or 3 color channels (black and white or RGB), along with two spatial dimensions, leading to an input with a shape of the form (number of channels, X dimension, Y dimension). Images will always be pre-processed by a convolutional network in the presented articles.

2.2.3. Layouts

The generative approach introduced in the second article takes scene layouts as inputs. In our specific case, a layout is a 2D image where a given pixel is annotated with a class corresponding to either background (no object) or the object at the given position.

2.2.4. Scene graphs

We will be considering scene graphs in the second article. A scene graph is a graph where the nodes are the objects in the scene, the edges are the relations (e.g. "looking at", "riding", ...) between two objects. Self-relations correspond to attributes (e.g. "standing", "smiling"). The objects, relations and attributes are represented by embeddings. Such an input is processed with a graph convolutional network [GCN, Goller and Kuchler, 1996].

2.2.5. Text

Text is a very common modality. In the present dissertation, only the third article refers to text inputs. In this case, each word is individually mapped to its Glove vector [Pennington et al., 2014], leading the input to our model to take the form of a sequence of fixed-size float vectors.

2.2.6. MRI sequences

The model presented in chapter 8 is trained on Magnetic Resonance Imaging (MRI) data. This imaging technique is used in the medical field to obtain images of different body parts, tissues... As its name indicates, imaging is done in large part via the application of strong magnetic fields. Under the effect of such a magnetic field, the electrons of hydrogen atoms inside the body are displaced. Once the field strength is reduced, the electrons return to their resting state, releasing energy in the process which is detected by the scanner. Different MR pulse sequences can be used, leading to different types of images, with varying performance in different tasks. In the third article, we consider T1 and T2 (and their variants), corresponding to the spin echo approach, and FLAIR (Fluid-attenuated inversion recovery), which allows the suppression of certain fluids. More details are outside the scope of this dissertation. The data is generally of dimension: (number of channels, sequence length, X dimension, Y dimension). The sequence length is variable. Such inputs are processed in our case with a 3D convolutional network.

2.3. Machine learning

This thesis presents three articles each describing a series of algorithms that learn to perform tasks from data. Therefore, it can be broadly said that it belongs to the field of *Machine Learning*. In the most general sense, machine learning is the study of algorithms that learn from data and/or experience. It is a subset of AI that itself includes the study of deep neural networks, Deep Learning, which will be the main focus of this dissertation. Machine learning can be divided into different experimental setups that require different approaches, datasets, annotations and algorithms. In this chapter, we will detail *supervised learning* and *unsupervised learning* (more specifically generative modeling), as they are the two setups that occur in the articles presented in this dissertation.

2.3.1. Parameters and hyper-parameters

A machine learning algorithm can be generally understood as consisting of three main components:

- a mapping from one set of variables to another (e.g. from a set of input features to a set of possible output values).
- a learning objective, a set of evaluation metrics.
- a learning algorithm.

The mapping is usually parametrized by a set of variables, the *parameters*, at least some of which are learnt during a training phase (referred to as trainable parameters). Usually, the parameters θ are in $\mathbb{R}^n$. We can denote the mapping as a function f, and make the dependence on parameters explicit, by using the notation f_θ.

Many machine learning algorithms also have a set of parameters that are not directly learned but rather chosen at some point in the training process, called hyper-parameters. This can be for numerous reasons, one of the most common being that the parameter can not be varied easily during the training procedure (notably if the parameter has an impact on the model architecture, e.g number of different clusters in a clustering algorithm), or because it has a direct impact on the training procedure itself (e.g. the learning rate in a gradient-based training algorithm). Common examples of hyper-parameters in machine learning include:

- Learning rate
- Number of layers of a deep neural network, width of these layers.
- Interpolation coefficient between different training losses (numerical objectives dependent on the model's parameters that are minimized during training by modifying the model's parameters).

2.3.2. Learning objective, evaluation metric

As seen previously, a machine learning algorithm commonly comprises a learning objective and an evaluation metric. Both take the form of a function mapping the model's outputs, usually to a scalar value. The evaluation metric is a score on which we will make the final evaluation of the model, e.g. accuracy, the percentage of model outputs that are correct. Often, the evaluation metric is not a suitable learning objective in itself, usually because it cannot be differentiated. The learning objective is then a proxy for this metric: the aim is that by improving the values taken by the learning objective, the values taken by the evaluation metric will also improve. In what follows, we will consider that we want to minimize the learning objective, which we will also call *loss*.

An example of a suitable pair of learning objective and evaluation metric is as follows. We consider a dataset of images x_i of digits. The digits are a discrete set: the numbers from 0 to 9, constituting the classes we want to predict (x_i has class y_i). We consider a model f_θ that maps an input image x to its probability of belonging to each class.

For a set of outputs $f_\theta(x_1), \cdots, f_\theta(x_n)$, the prediction for each output would be the mode: the class of highest predicted probability, and the *accuracy* would be the proportion correct predictions. This objective is non-differentiable wrt. the set of outputs as the changes of modes form discontinuities. As a result, we commonly replace it with a differentiable learning objective called the categorical cross-entropy, which is the cross-entropy between the distribution parametrized by f and the discrete distribution corresponding to the correct labels (i.e. all the probability mass for the correct labels).

2.3.3. Dataset splits and training phases

Machine learning generally relies on training data to train a model. Data is often separated into *splits* which have distinct uses during training. Commonly, the training process follows the following steps:

- Training is performed on a training set, with fixed hyper-parameters, resulting in a set of weights (learned weights) that parametrize the trained function.
- Performance is then evaluated on a validation set with a set of metrics. Validation performance is used as a signal to select good hyper-parameters. The first step is often repeated multiple times at this point.
- Final performance (with the best hyper-parameters found) is then reported on a held out test set. The test set should not have been seen during training.

2.3.4. Hyper-parameter selection

The second step, hyperparameter selection, involves exploring the set of possible hyperparameter combinations in an efficient fashion (as each trial involves training a model, which is often costly in terms of computational resources). There are multiple ways to explore the space of possible hyper-parameter combinations, including manual search, grid search, and random search (all of which have been used in the articles presented as part of this dissertation). There exist other methods, such as Bayesian optimization [Snoek et al., 2012], but those are outside the scope of this document.

Manual search consists in manually setting different parameter values, and observing the result on the validation set. The best-performing combination is then evaluated on the test set.

Grid search considers a finite set of possible values for each hyper-parameter. The algorithm then explores the grid of possible values sequentially, or in parallel (across different compute nodes). If we want to optimize a learning rate taking values in the set $\{0.01, 0.001, 0.0001\}$ and number of layers in the set $\{2, 3, 4, 5\}$ this would result in a grid of size 12.

Random search [Bergstra and Bengio, 2012], where random hyper-parameter values are sampled and evaluated until a certain stopping criteria is met, is another hyper-parameter selection method. Figure 2.2 compares grid and random search when two hyper-parameters are considered, an important and an unimportant one (in terms of effect on the model's

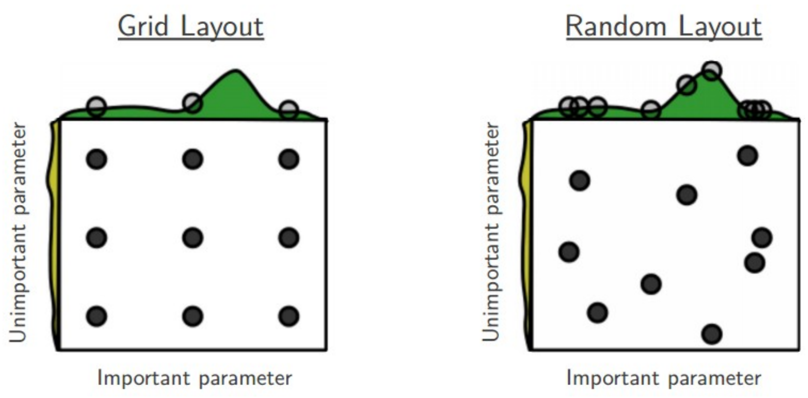


Fig. 2.2. Representation of grid search and random search. The black dots are hyperparameter combinations investigated by the search algorithm. The curves in green plot model validation performance for different values of the important parameter. The plot in yellow represents the small effect the unimportant parameter has on model performance. Figure taken from [Bergstra and Bengio, 2012].

performance). It can serve as an illustration of the superiority of random search for that specific problem.

2.3.5. Supervised learning

Supervised learning aims to learn a mapping from the set of possible inputs to the set of possible outputs. During *training*, (input, output) pairs are presented. Commonly, during *evaluation*, inputs are given, and the predicted output is compared to the actual output so as to compute different *evaluation metrics* that attempt to quantify the model's performance. Depending on the set of acceptable output values, the task will be further called *classification* (if the output values form a finite discrete set) or e.g. *regression* (if the output values belong to a continuous set, often a subset of $\mathbb{R}$). These two divisions are not the only ones, but will be the two supervised tasks we consider in the presented articles. In the MNIST dataset [LeCun et al., 1998] per instance, we are given 28 by 28 images of hand-drawn digits. The classification problem is to determine which of the ten classes these images correspond to. The ten classes are the digits from 0 to 9. In practice, the input data could be images, videos, text, ...

2.3.6. Unsupervised learning

In unsupervised learning, we are given examples (data points taken from a dataset) without targets. A single data point can be represented as a singleton X, where X is a set of features representing an example, such as a raw image (as in the MNIST example above) or text. In this case, the range of approaches varies strongly, as not all models aim to solve the same task. Generative modeling, which will be expanded upon in section 2.6, involves learning the distribution $\mathbb{P}(X)$, whereas other methods might introduce a proxy objective proxy(X) or auxiliary task and return to the supervised learning framework by training a model to predict the proxy or more generally solve the auxiliary task given X. This is usually a means of learning good representations of X which can then be applied to downstream tasks, and was in particular the basis of the DIM family of models we consider in the first and third presented articles.

2.4. Zero-shot learning

Zero-shot learning [ZSL Larochelle et al., 2008] is a machine learning problem that involves generalizing to a task from zero examples of that task, but presumably with data from other, related, tasks. We will be considering only zero-shot classification in this thesis, which is a classification problem defined by the fact that at least some of the examples in the test set belong to classes that are not present in the data available for training. As an example, the CUB dataset [Wah et al., 2011a] contains approximately 11,000 birds belonging to 200 distinct classes. All the examples corresponding to 150 classes will usually be used to train a model. At test time, all the examples corresponding to the 50 remaining classes will usually be used to evaluate the trained model. We are in this case training a model to classify images of birds into categories it has not seen at train time. Note that zero-shot learning is not limited to zero-shot classification: zero-shot semantic segmentation [Bucher et al., 2019] and zero-shot object detection [Bansal et al., 2018] have also been extensively studied, among other applications.

2.4.1. Representing classes

As a result of the nature of the zero-shot classification problem, most common supervised learning techniques would not work. To address this, Larochelle et al. [2008] considered attributes that enable transfer learning from the seen (training) classes to the unseen (test) classes. This approach is still the by far the most common today and we will focus exclusively on attribute-based zero-shot learning in this thesis. Actual examples of attributes for the Animals with Attributes 2 dataset [Xian et al., 2017] for the class zebra are:

- black: 1
- white: 1
- brown: 0
- stripes: 1

This means that a zebra is white and black, not brown, and has stripes. Using our knowledge of which attributes the class zebra has, and the examples of zebras in the training data, we hope to be able to recognize other, previously unseen, classes. Other works outside the scope of the presented articles have alternatively used class representations that are derived from text descriptions of the classes [Qiao et al., 2016], or from word vectors [Wang et al., 2018a].

2.4.2. Datasets

ZSL datasets commonly feature images and semantic information specific to classes. In the first article, we focus on the three most commonly used, namely Animals with Attributes 2 [**AwA2**, Xian et al., 2018], Caltech-UCSD-Birds-200-2011 [**CUB**, Wah et al., 2011b], SUN Attribute [**SUN**, Patterson and Hays, 2012]. Zero-shot learning has been applied to a wide variety of other datasets, such as Imagenet.

2.5. Deep learning

Deep learning is a field of machine learning that focuses on training artificial neural architectures with multiple layers, often with the use of gradient descent. The multiple layers apply successive non-linear transformations. Such architectures are able to learn multiple levels of representation of the input in order to solve a given task. They first were, and still are, inspired by biological neural networks. Current deep learning models are usually highly complex, and require specific training algorithms to achieve peak performance.

This section will start by defining a specific type of feedforward neural networks commonly used in research, and presenting the backpropagation algorithm used to train such networks. It will then introduce the following important neural network variants: multi-layer perceptrons, convolutional neural networks, residual networks and auto-encoders.

2.5.1. Neural networks

An *artificial neural network* is a learning algorithm originally inspired by biological neural networks present in the brains of animals. Neural networks are a collection of nodes or artificial neurons which are connected together. There are many different types of neural networks, making a precise and universal definition quite hard to formulate in general. Therefore we will focus essentially on neural networks as they occur in the works highlighted in this thesis.

A feed-forward neural network (in the sense that matters in this thesis) can be represented as a directed acyclic graph. The nodes of the graph are arranged into different layers. These layers consist in:

- An input layer. The number of neurons in the input layer is referred to as the *input dimensionality* of the network.
- An output layer. The number of neurons in the output layer is referred to as the *output dimensionality* of the network.
- Usually one or more hidden layers.

In the feedforward case, the signal can only propagate *forward*, from the input to the output. That means there are no feedback connections per instance where a layer would be connected to itself. A feed-forward network is said to be *fully connected* if every neuron (node) has a forward-directed connection to all other nodes in the next forward layer. If this is not the case the network is *partially connected.* The value taken by the output layer is the prediction of the network. Across different tasks the prediction can take different forms, such as a scalar, vector, image or variable length text per instance.

2.5.2. Mathematical formalism

2.5.2.1. Forward propagation. To compute the output of the feed-forward network, we apply an algorithm called forward propagation. The name originates from the fact that the computations are done forward, from the input layer to the output layer. We iterate the following process, starting with the input layer:

(1) For each neuron k, we compute a weighted sum of the inputs, v_k. We can write this as: $v_k = b_k + \sum_i w_{ki} x_i$, where the x_i are the values outputted by predecessor neurons, w_{ki} are the corresponding weights, and b_k the biases. All updates are done simultaneously on a given layer.

(2) The value outputted by each neuron k is $\phi(v_k)$ where ϕ is a function called the *activation function.* This function applies a non-linearity to the operation, which is essential if we are to be able to model non-linear input-to-output mappings: the forward propagation would otherwise be simply a sequence of matrix multiplications, i.e. a linear operation. If the current layer is an intermediate layer in the network, the output value will then be fed as input to the neurons of the next layer.

Examples of common activation functions include:

- Hyperbolic tangent ($\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$)
- Sigmoid ($\sigma(x) = \frac{1}{1+e^{-x}}$)
- Rectified linear (ReLU) ($x \to \max(0, x)$)

They are shown on Figure 2.3

Figure 2.4 represents a fully-connected neural network. It possesses one hidden layer consisting of 5 neurons (in blue). The connections between neurons are represented by arrows. Each connection will have its own associated weight.

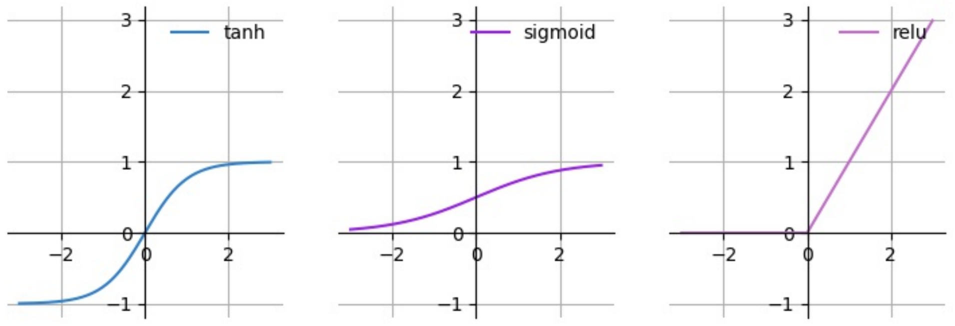


Fig. 2.3. Plots of 3 common activation functions: tanh, sigmoid and ReLU.

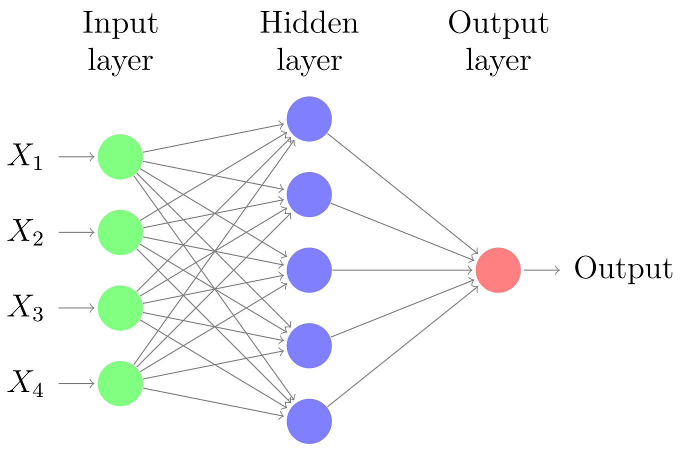


Fig. 2.4. A fully-connected neural network with one hidden layer. The input has 4 features, there are 5 hidden neurons, and a single output neuron.

2.5.3. Training neural networks using back-propagation

2.5.3.1. Definitions. The previous section mentioned a neural network's weights. In our general case, *weights* are any parameter that modulates the intensity of the connection between two neurons. We can optionally add a bias neuron to a layer. The bias neuron always outputs a constant value, independently of the input the layer receives. The values of the connections from the bias neuron to units of the next layer can be learned, and are referred to as the *biases*. Let θ be the parameters (the weights and biases as introduced in the previous section) of a neural network defining a function f. We suppose we have n training examples, indexed by i. The optimization problem can be written as:

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=0}^{n-1} \mathcal{L}(f(x_i, \theta), y_i), \tag{2.5.1}$$

with y_i the target value for the point x_i, and $\mathcal{L}$ is the loss function, a measure of the difference between the desired (y_i) and actual ($f(x_i;\theta)$) model output. Empirical risk is the average loss on a data set. We will denote $\mathcal{L}(f(x_i,\theta), y_i)$ as $\mathcal{L}_i$ in what follows for conciseness.

In practice we usually minimize the risk via *gradient descent*:

$$\theta \leftarrow \theta - \frac{\lambda}{n}\sum_{i=0}^{n-1}\frac{\partial \mathcal{L}_i}{\partial \theta}, \tag{2.5.2}$$

with λ the learning rate. The learning rate is a measure of the amplitude of gradient descent steps. A higher learning rate will mean that each gradient descent step has a larger effect on the parameters. There are different ways to do gradient descent. We could perform descent on the whole set (gradient descent), on a single example or on a mini-batch (a small subset of the training set).

The parameters are usually initialized according to a given probability distribution. For instance, we could initialize all the parameters by drawing from a normal distribution or by using the so-called Glorot initialization [Glorot and Bengio, 2010]. The choice of initialization method has an impact on downstream performance.

We usually train on a mini-batch, mainly because this results in faster convergence (as we can parallelize operations on graphical processor units) and better generalization [Bottou, 2010, Bottou et al., 2016, Keskar et al., 2016].

2.5.3.2. Back-propagation. As seen above, training via gradient descent requires computing the gradient of the loss function with respect to the model's parameters. This gradient can be computed using the chain rule. Let $\mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{R}^m$ be two vectors, and $g : \mathbb{R}^n \to \mathbb{R}^m, f : \mathbb{R}^m \to \mathbb{R}$ be two differentiable functions. If we can write $\mathbf{y} = g(\mathbf{x})$ and $z = f(\mathbf{y})$ then the chain rule states we have:

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j}\frac{\partial y_j}{\partial x_i}. \tag{2.5.3}$$

A neural network can be expressed as a composition of different differentiable operations, allowing us to apply the chain rule recursively over the computation graph, which defines the back-propagation algorithm.

We will now illustrate this procedure using an example: a simple neural network example where the computation graph has a chain structure. This is not always the case, as we will see for the more complex architectures presented later, such as resnets. The network can be written as a composition of operations:

$$f_N \circ f_{N-1} \circ \cdots \circ f_0 \tag{2.5.4}$$

where each f_i corresponds to the operation done by a single layer of the network, and is parametrized by weights θ_i. We can therefore apply the chain rule to obtain:

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = \frac{\partial \mathcal{L}}{\partial f_N}\frac{\partial f_N}{\partial f_{N-1}} \dots \frac{\partial f_{i+1}}{\partial f_i}\frac{\partial f_i}{\partial \theta_i}. \tag{2.5.5}$$

For the case of a feed-forward neural network, to obtain the weight updates at a given layer, we start from the top-most layer, and back-propagate through the graph using the following update equations:

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = \frac{\partial \mathcal{L}}{\partial f_i}\frac{\partial f_i}{\partial \theta_i} \tag{2.5.6}$$

$$\frac{\partial \mathcal{L}}{\partial f_{i-1}} = \frac{\partial \mathcal{L}}{\partial f_i}\frac{\partial f_i}{\partial f_{i-1}}. \tag{2.5.7}$$

This recursion is initialized by the gradient for the topmost layer, which is simply:

$$\frac{\partial \mathcal{L}}{\partial \theta_N} = \frac{\partial \mathcal{L}}{\partial f_N}\frac{\partial f_N}{\partial \theta_N} \tag{2.5.8}$$

and can be directly computed from the loss function.

In practice, other types of computation graphs are possible (e.g. with skip-connections), the main constraint being that the graph is directed acyclic. The back-propagation algorithm detailed above varies somewhat when the network is not feed-forward, and when multiple-loss terms are introduced. In all cases, we apply the back-propagation algorithm on the computation graph.

2.5.4. Common types of neural networks and layers

2.5.4.1. Multi-Layer Perceptron. The multi-layer perceptron (MLP) is a fully connected feed-forward network, commonly found in the deep learning literature. Figure 2.4 represents a small MLP.

2.5.4.2. Convolutional Neural Networks. A Convolutional Neural Network (CNN) [LeCun et al., 1989] is a neural network used for specific tasks such as image processing or signal analysis. CNNs take their name from the fact that they use a specific type of mathematical operation called a *convolution*. In addition, most CNNs also use pooling (explained below) and fully connected layers. Some of the explanations that follow draw from Goodfellow et al. [2016].

The layers of a CNN performing a convolution operation are called *convolutional layers*. Given an input $x(t)$ and a weighting function $w(a)$, we can define the convolution operation $*$ as:

$$(x * w)(t) = \int w(a)x(t-a)\mathrm{d}a. \tag{2.5.9}$$

In practice, convolutions used in image processing are discrete convolutions. The weight matrix is referred to as the *kernel*. Convolutional layers, as the name indicates, apply convolutions. These convolutions can be applied to multi-dimensional data (1D signals such as ECGs, 2D signals such as images, and 3D signals such as MRI volumes per instance). The convolutions are discrete as opposed to continuous (the weight matrix and input are only defined for integer coordinates), taking the following form:

$$(x * w)(t) = \sum_{a=-\infty}^{\infty} w(a)x(t-a) \tag{2.5.10}$$

or for images:

$$(X * W)(t,u) = \sum_{a=-\infty}^{\infty} \sum_{b=-\infty}^{\infty} W(a,b)X(t-a,u-b). \tag{2.5.11}$$

For practical reasons, the weight matrix is 0 except for a (usually small) set of points. In addition, the input is of finite size, and therefore the sum does not actually consist in an infinite number of elements. As mentioned in Goodfellow et al. [2016], the main advantages of such operations are that they introduce *parameter sharing*, *sparse interactions* and *equivariant representations*, shown to be beneficial in many problems, notably image processing. We will consider the example of an image in what follows, but the concepts generalize to other contexts. Parameter sharing is achieved by applying the same kernel to all positions of the image. Sparse interactions are the result of the kernel being 0 except for a small set of positions. The nature of the convolution operation ensures that the output is translation equivariant. In many real datasets this is an advantage as translating an object should not affect the class it belongs to: a dog translated in an image is still a dog, for example. Figure 2.5, taken from Dumoulin and Visin [2016], represents a convolution on an image.

A *pooling* operation outputs for each location a summary statistic of the inputs. Common types include max-pooling (where the output for a given location is the max of a small region surrounding the location), and mean pooling (replaces the max operation by the average).

Figure 2.6, taken from Dumoulin and Visin [2016], represents the max-pooling operation. Pooling operations are useful for different reasons:

- They force the network to become invariant to permutations inside the pool, e.g. small translations. By combining this operation with convolution layers, the network can learn a set of transformations to become invariant to.
- They are in practice essential when dealing with inputs of different dimensions (such as images of different sizes). Adapting the size of the pooling window to obtain the same output size for different input sizes makes possible training on datasets where input images have different sizes.
- When it doesn't destroy too much information, pooling effectively reduces the size of feature maps the network has to process, yielding better computational efficiency.

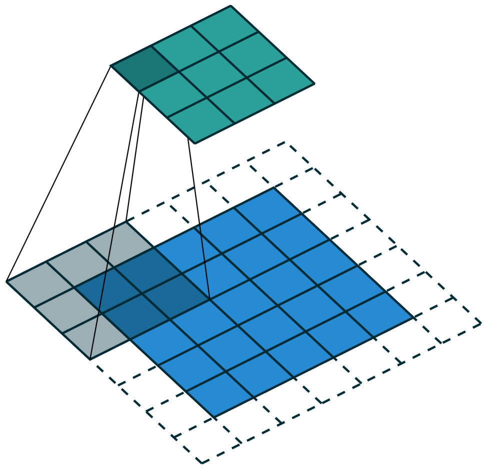


Fig. 2.5. Representation of a convolution operation on a image with a kernel of size 3×3. The input is in blue, the output for a specific location in green. Padding is applied, represented in white (1 pixel of padding on all sides in this example). The shaded area in the input (on the upper left) is convolved with the kernel, yielding 1 output value, the dark green square represented in the figure. The same operation is applied on all relevant parts of the image, resulting in weight-sharing.

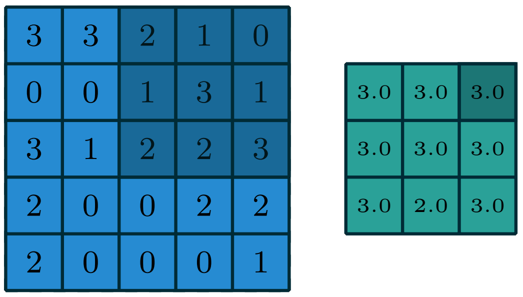


Fig. 2.6. Representation of the input (blue) and output (green) of a max-pooling operation.

2.5.4.3. Residual networks. Residual networks [He et al., 2016] are a type of neural network that incorporate skip connections that create shortcuts between certain layers. Convolutional residual networks and their variants are often close to state of the art on different image processing tasks. In future chapters we will refer to convolutional residual networks with k layers as "ResNet-k". While residual networks have been used extensively in the context of computer vision, they can benefit other fields as well, such as time-series modeling [Kachuee et al., 2018] or survival analysis [Luck et al., 2017, 2018]

A *skip connection* is a connection between two different layers of a neural network that bypasses at least one intermediate layer. Skip connection are useful in many areas of deep

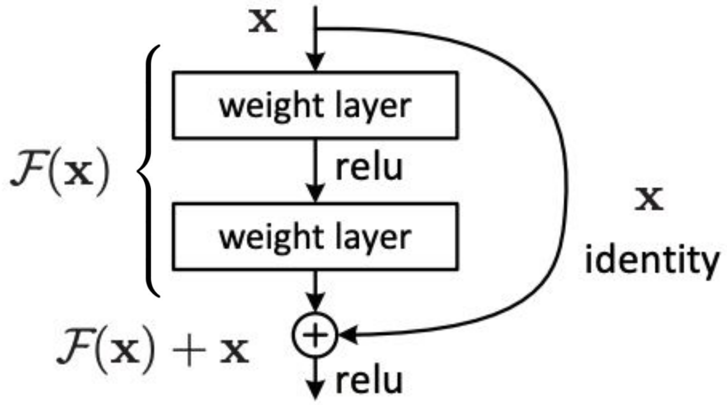

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 2.7. Representation of a type of residual block. A weight layer is a convolution, followed in practice by a batch normalization layer. A transformation $\mathcal{F}$, consisting in a convolution followed by a ReLU and another convolution, is applied to the input, yielding $\mathcal{F}(x)$. The final output is $\mathcal{F}(x) + x$.

learning by allowing better gradient flow. An example is U-Net [Ronneberger et al., 2015] which uses multiple skip connections to improve semantic segmentation performance for medical images.

Figure 2.7, taken from He et al. [2016], represents one of multiple possible residual blocks used in the construction of convolutional residual networks. Two convolutions with an intermediate ReLU non-linearity are applied to the input X. The result of this operation is added to the initial input X (the skip connection). A final ReLU is applied to this value before outputting the result. In practice, batch normalization [Ioffe and Szegedy, 2015] is applied after each convolution.

2.5.5. Specific types of CNNs

In the preceding subsection, we have introduced the notion of CNNs and an important variant, residual convolutional networks. As some of the presented articles refer to specific architectures that are well known in the computer vision community but not as well outside of it, we will introduce them here.

2.5.5.1. Alexnet and VGG. One of the first convolutional architectures used in practice was LeNet [LeCun et al., 1998], which used 5 layers, quite few by modern standards. This work was the inspiration behind Alexnet [Krizhevsky et al., 2012], which used more layers and resulted in groundbreaking performance increases on the Imagenet dataset [Russakovsky et al., 2015]. VGG19 (the acronym stems from the number of layers and the Visual Geometry Group at the University of Oxford) introduced a yet again deeper architecture while also reducing the size of the convolutional filters (from e.g. 11×11 in previous works) to mostly

3×3, making the training of deeper networks feasible. Note that Alexnet and VGG19 are famous and standard architectures, which explains our use of them in the first and second papers respectively. We use Alexnet-type architectures as a relatively low-footprint feature extractor, and VGG19 as the basis for computing a perceptual loss.

2.5.5.2. Resnet variants. In the presented articles, we refer in particular to Resnet-101 and Resnet-50, which as mentioned previously (subsubsection 2.5.4.3) are convolutional residual networks comprising 101 and 50 layers respectively.

2.5.6. Auto-encoders

An autoencoder is a neural network that aims to learn a representation (code) of an input. We present auto-encoders primarily for their relation to a class of generative models presented in the next chapter, variational auto-encoders.

An autoencoder can be seen as consisting of two components, an encoder and a decoder. Let $\mathcal{X}$ be the input space, and $\mathcal{Z}$ be the *latent space* over which the codes are designed.

The encoder is a mapping: $f_\theta : \mathcal{X} \to \mathcal{Z}$, and the decoder is a mapping: $g_\phi : \mathcal{Z} \to \mathcal{X}$. Here, θ (resp. ϕ) are the parameters of the encoder (resp. decoder). We also introduce a reconstruction loss $\mathcal{L}(\cdot, \cdot)$ that measures how well the autoencoder is able to reconstruct its input. This reconstruction loss can per instance be the L2 norm between input and reconstruction. The training objective (for a single example x) is:

$$\arg\min_{\theta,\phi} \mathcal{L}(x, g_\phi \circ f_\theta(x)) \tag{2.5.12}$$

that is, the training procedure aims to find the encoder and decoder that minimize what is referred to as the reconstruction error. These concepts will be useful when variational autoencoders are defined.

Figure 2.8[1] represents an autoencoder. 3 layers are shown. The leftmost layer is the input layer. The rightmost layer is the output layer. As we are reconstructing the input it has the same number of units as the input layer (with the exception of the added bias represented). The intermediate layer is the representation of the input.

2.6. Generative Models

2.6.1. Introduction

As outlined previously, a generative model learns to draw samples from a distribution approaching a target distribution. In the cases considered in this thesis, the data distribution we want to approach is the empirical data distribution $\mathbb{P}_{\text{data}}$. The distribution parametrized by the model is required to be close to the target distribution. The notion of proximity

[1]Taken from `http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/`

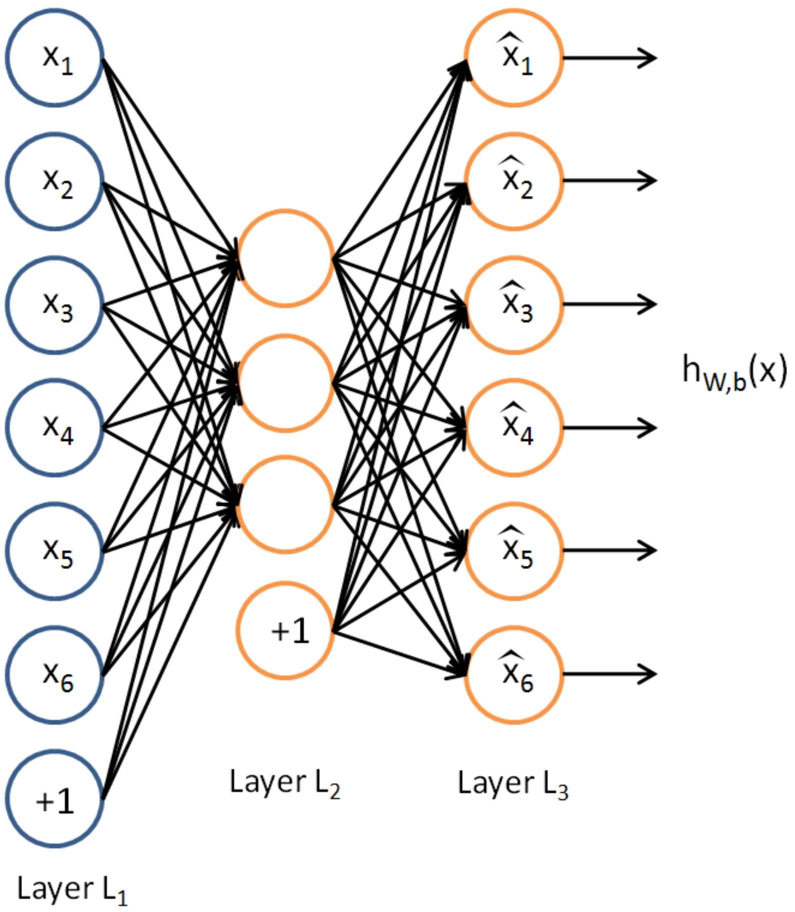


Fig. 2.8. Representation of an autoencoder consisting of 3 layers. The middle layer corresponds to the encoding of the input.

is often measured through a distance or divergence. In our use-cases we will use the KL divergence, but other mesures have been used, such as the Wasserstein distance [Gulrajani et al., 2017a].
Generative models are useful for many applications, including semi-supervised learning as in Dumoulin et al. [2016], robotics [Bousmalis et al., 2017] or image inpainting [Pathak et al., 2016]. In the presented articles, we propose both an improvement on classical generative models from layouts (second paper), and use generative models as baselines for a set of experiments (first article). We will be presenting two approaches, variational autoencoders (VAEs) and Generative Adversarial Networks (GANs). VAEs can be seen as the generative counterpart of auto-encoders. GANs are a framework involving a generator and a discriminator network.

2.6.2. Variational autoencoders

2.6.2.1. Components of a variational autoencoder. Variational autoencoders (VAE) [Kingma and Welling, 2013, Rezende et al., 2014] are a class of generative models that learn to perform approximate inference. They are trainable with gradient methods.
A VAE consists in two essential components, an encoder and a decoder, as was the case for the vanilla autoencoder. The encoder and decoder are usually neural networks in most practical applications. The encoder learns to map true samples to a latent space, as was the case for the encoder in an auto-encoder.

The encoder takes as input x sampled from the true distribution, and outputs parameters for a distribution (e.g. Gaussian) over the latent encoding z. The encoder defines the following distribution:

$$\mathbb{Q}_\theta(z|x) \tag{2.6.1}$$

with θ the parameters of the encoder.

The decoder takes as input z with domain in the latent space, and outputs a distribution over the same space as the true samples x. We can denote the distribution as:

$$\mathbb{P}_\phi(x|z) \tag{2.6.2}$$

with ϕ the parameters of the decoder. Finally, we consider a prior distribution over the latent vector z, $\mathbb{P}_z(z)$ which is in practice taken to be $\mathcal{N}(0, I)$, I being the identity matrix.

2.6.2.2. Variational inference. Variational inference is a family of techniques that allow the approximation of intractable integrals. As we mention in what follows, the main motivation for applying this method is that it will allow us to approximate an intractable integral in a VAE expression, making the full model learnable end-to-end. We suppose that we have a dataset consisting of the data points x_i, and x is an observable variable. As seen previously, we can write the log-likelihood as (under the iid. assumption):

$$\sum_i \log \mathbb{P}(x_i) \tag{2.6.3}$$

Computing $\log \mathbb{P}(x_i)$ would require marginalizing over z by $\mathbb{P}(x_i) = \int \mathbb{P}(x_i|z)\mathbb{P}_z(z)dz$, which is infeasible in practice. Using conditional probabilities we can write:

$$\log \mathbb{P}(x) = \mathbb{E}_{z\sim\mathbb{P}(z|x)} \log \frac{\mathbb{P}(x, z)}{\mathbb{P}(z|x)}. \tag{2.6.4}$$

$\mathbb{P}(z|x)$ is still intractable, so we replace it with a distribution $\mathbb{Q}(z|x)$ that approximates it. We want to minimize the KL-divergence between the two distributions. It can be written as:

$$D_{\text{KL}}(\mathbb{Q}(z|x)|\mathbb{P}(z|x)) = \int \mathbb{Q}(z|x) \log \frac{\mathbb{Q}(z|x)\mathbb{P}_x(x)}{\mathbb{P}(z,x)} dx \tag{2.6.5}$$

$$= \mathbb{E}_{\mathbb{Q}(z|x)}[\log \mathbb{Q}(z|x) - \log \mathbb{P}(z,x)] + \log(\mathbb{P}(x)). \tag{2.6.6}$$

As the KL divergence is always non-negative, we have that:

$$\log \mathbb{P}(x) \geq \mathbb{E}_{\mathbb{Q}(z|x)}[-\log \mathbb{Q}(z|x) + \log \mathbb{P}(z,x)]. \tag{2.6.7}$$

We will *maximize* this lower bound, which can be re-written as:

$$\mathbb{E}_{\mathbb{Q}(z|x)}[\log \mathbb{P}(z,x) - \log \mathbb{Q}(z|x)] = \mathbb{E}_{\mathbb{Q}(z|x)}[\log \mathbb{P}(x|z) + \log \mathbb{P}(z) - \log \mathbb{Q}(z|x)] \tag{2.6.8}$$

$$= \mathbb{E}_{\mathbb{Q}(z|x)} \log \mathbb{P}_\phi(x|z) - D_{\text{KL}}(\mathbb{Q}_\theta(z|x)||\mathbb{P}(z)). \tag{2.6.9}$$

It is possible to optimize this lower bound using the reparametrization trick [Kingma and Welling, 2013].

2.6.2.3. Training a VAE. As seen in the previous sub-section, the loss (to be *minimized*) for a single datapoint x_i is denoted as:

$$-\mathbb{E}_{z\sim\mathbb{Q}_\theta(z|x_i)} \log \mathbb{P}_\phi(x_i|z) + D_{\text{KL}}(\mathbb{Q}_\theta(z|x_i)||\mathbb{P}(z)). \tag{2.6.10}$$

The first term is a reconstruction loss (in the same sense as seen in the section on autoencoders), which can be seen as encouraging the VAE to reconstruct the input x_i. The second term is the Kullback-Leibler divergence between the two distributions $\mathbb{Q}_\theta(z|x_i)$ and $\mathbb{P}(z)$. It encourages the codes z to be diverse (close to the normal distribution) for a given input. This helps prevent a collapse whereby the VAE would simply behave as an autoencoder, able to reconstruct its inputs, but failing to generate realistic samples.

Historically, VAEs had the advantage of learning a useful latent representation (which resulted in their variants commonly performing well in disentanglement tasks [Higgins et al., 2017b], but resulted in often lower-quality samples compared to GANs (as evaluated by humans). This second point has to a large extent been corrected by more recent iterations [VQ-VAE Razavi et al., 2019].

2.6.3. Generative adversarial networks

Generative adversarial networks [GANs, Goodfellow et al., 2014a] are an unsupervised learning framework for training a generator. GANs rely on two models: a generator and a discriminator, the latter of which is optimized to estimate a *difference measure* between two distributions, the empirical target distribution and the one induced by the generator.

The training procedure is to alternatively train the discriminator to distinguish between real and generated samples, and update the generator to better fool the discriminator. The main strength of GANs lies in their ability to train the discriminator as a completely continuous function using only empirical samples from the target dataset and the generator.

2.6.3.1. Mathematical definition and notations. We will denote the generator network as G_θ, and the discriminator network as D_ϕ. θ and ϕ are the parameters of the two networks. In what follows, z is a variable sampled from the distribution $\mathbb{P}_z(z) = \mathcal{N}(0, I)$. $\mathbb{P}_{\text{data}}$ will denote the true distribution of the data.
The generator maps a latent noise vector z to the space of images (in our case), $\mathcal{X}$. We denote $\mathcal{Z}$ the latent space over which z is sampled. The discriminator takes as input a real or generated image, and outputs the estimated probability that this image is indeed real. We can write the two networks as:

$$G_\theta : \mathcal{Z} \to \mathcal{X} \tag{2.6.11}$$

$$D_\phi : \mathcal{X} \to [0, 1]. \tag{2.6.12}$$

We define the following value function:

$$V(D_\phi, G_\theta) = \mathbb{E}_{x\sim\mathbb{P}_{\text{data}}} \log D_\phi(x) + \mathbb{E}_{x\sim\mathbb{P}_z(z)} \log(1 - D_\phi(G_\theta(z)). \tag{2.6.13}$$

GAN optimization can be formulated as the following minimax problem:

$$\min_G \max_D V(D_\phi, G_\theta) \tag{2.6.14}$$

Theoretically, the generator would be trained to minimize:

$$\mathbb{E}_{x\sim\mathbb{P}_z(z)} \log(1 - D_\phi(G_\theta(z)). \tag{2.6.15}$$

However, as it is well-known that this loss often yields *vanishing gradients* w.r.t. θ [Goodfellow et al., 2014a, Arjovsky and Bottou, 2017], it is generally recommended to train the generator on a proxy:

$$\hat{\theta} = \arg\max_\theta \mathbb{E}_{z\sim\mathbb{P}_z(z)}[\log D_\phi(G_\theta(z))]. \tag{2.6.16}$$

2.6.4. Conditional GANs

2.6.4.1. Conditional generation. Let X be a random variable, and P a probability distribution over that variable. It is possible to condition a generative model such as a GAN on a variable c such that instead of generating:

$$X \sim \mathbb{P}(X), \tag{2.6.17}$$

it learns:

$$X \sim \mathbb{P}(X|c). \tag{2.6.18}$$

The most common way to do so is (in the case of a GAN but it also applies to VAEs) to concatenate c to z during generation, and to also input c to the discriminator. In what follows we will focus on GANs, but all of the relevant ideas are easily transferable to define conditional VAEs.
Conditional generation was used in the first article (GAN conditioned on attributes) and the second (GAN conditioned on local pixel-level class information provided by the input layout). There is little difference in practice, apart from the fact that the conditioning variable is 1D in the first case, and 2D in the second, requiring some architecture changes.

We will denote $\mathcal{Z}$ the space over which z is sampled, and $\mathcal{C}$ the space over which the conditioning variables are defined.

The generator and discriminator will also take as input the conditioning variables, and can now be written as:

$$G_\theta : \mathcal{Z} \times \mathcal{C} \to \mathcal{X} \tag{2.6.19}$$

$$D_\phi : \mathcal{X} \times \mathcal{C} \to [0, 1]. \tag{2.6.20}$$

In what follows, x is a dataset sample (image, or feature), c is a vector or tensor of conditioning variables.

We introduce the following distributions:

- $\mathbb{P}(x, c)$ is the joint distribution over image and conditioning variables.
- $\mathbb{P}_z(z)$ is as before the prior distribution over the noise vector z
- $\mathbb{P}_c(c)$ is the true distribution over conditioning variables.
- G_θ, $\mathbb{P}_z$ and $\mathbb{P}(c)$ define a joint distribution $\mathbb{Q}_{s,\theta}(x, c)$ over the images and conditioning variables, as follows: $\mathbb{Q}_{s,\theta}(x, c) = \mathbb{Q}_{s,\theta}(x|c)\mathbb{P}_c(c)$, where we have marginalized over the noise z on the right and left.

2.6.4.2. Mathematical formalism. Our goal for the conditional GAN is to find the parameters, θ, such that the joint distribution induced by G_θ, $\mathbb{P}_s(c)$ and $\mathbb{P}_z(z)$, $\mathbb{Q}_{s,\theta}(x, c) = \mathbb{Q}_{s,\theta}(x \mid c)\mathbb{P}_s(c)$, matches $\mathbb{P}_s(x, c)$.

D_ϕ estimates a difference measure between $\mathbb{P}_s(x, c)$ and $\mathbb{Q}_{s,\theta}(x, c)$. In the original formulation, the discriminator is trained to minimize the mis-classification error by maximizing the negative cross-entropy:

$$\begin{aligned} \hat{\phi} &= \arg\max_{\phi} \mathbb{E}_{\mathbb{P}_s(x,c)}[\log D_\phi(x, c)] + \mathbb{E}_{\mathbb{Q}_{s,\theta}(x|c)\mathbb{P}(c)}[\log(1 - D_\phi(x, c))] \\ &= \arg\max_{\phi} \mathbb{E}_{\mathbb{P}_s(x,c)}[\log D_\phi(x, c)] + \mathbb{E}_{\mathbb{P}_z(z)\mathbb{P}_s(c)}[\log(1 - D_\phi(G_\theta(z, c), c))]. \end{aligned} \tag{2.6.21}$$

As in the case for non-conditional GANs, it is generally recommended to train the generator on a proxy, which now takes the following form:

$$\hat{\theta} = \arg\max_{\theta} \mathbb{E}_{\mathbb{P}_z(z)\mathbb{P}_s(c)}[\log D_{\phi}(G(z,c),c)]. \tag{2.6.22}$$

Chapter 3

Prologue to first article

Locality and compositionality in zero-shot learning. Tristan Sylvain*, Linda Petrini*, Devon Hjelm (* denotes equal contribution). International Conference in Learning Representations 2020. Presented is a minimally edited version of this article.
Personal contributions I contributed a majority of the code base, in particular the part on the TRE score and general training of representations. We used the cortex framework [Devon, 2018, Cortex] as a basis for most of the experiments, and Linda also contributed heavily to the code. Most of the writing and experiments was done equally by myself and the shared-first author Linda. The extensive edits and advice given by Devon greatly increased the paper's final quality.

3.1. Context

This article was published as the result of long-term work on zero-shot learning with my co-authors. I had initially focused on improving zero-shot learning performance using generative models with the help of Devon Hjelm. This preliminary work had not resulted in a publication, but the large gap I had found between representations taken from generated and real images had given me the idea to start questioning what we really understood about image representations for zero-shot learning. Further discussions set us on the path towards explaining zero-shot learning from first principles.

3.2. Contributions

This paper can be seen as a first attempt to explain zero-shot learning by first principles. Commonly, zero-shot learning articles tend to focus on empirical performance along with tweaks to previous approaches. We chose instead to investigate how *locality* and *compositionality* of an image representation help explain its downstream performance in zero-shot learning tasks.

3.3. Aftermath

There is now a growing body of literature on the link between locality and compositionality, such as [Xu et al., 2020], which has to a certain extent been inspired by this paper.

Chapter 4

Locality and Compositionality in Zero-Shot Learning

4.1. Introduction

A crucial property of a useful model is to *generalize*, that is to perform well on test settings given learning on a training setting. While what is most commonly meant by generalization is being robust to having a limited number of training examples in distributionally-matched settings [Zhang et al., 2016b], many tasks are designed to address variations in the data between when a model is trained and when it is evaluated. For instance, some classification tasks address distributional changes in the input: from lacking guarantees of distributional match between train and test [e.g., covariate shift, Shimodaira, 2000] to having fundamental domain differences [e.g., domain adaptation, Crammer et al., 2007, Ben-David et al., 2007]. A number of tasks have also been designed specifically to understand models in terms of their ability to generalize to test situations that are poorly represented during training [e.g., Few-Show learning, Li et al., 2006], or even consist of a diverse and entirely novel set of sub-tasks [Zamir et al., 2018]. For supervised classification, Zero-Shot Learning [ZSL, Larochelle et al., 2008] is among the most difficult of these tasks, as it requires the model to make useful inferences about (e.g., correctly label) unseen concepts, given parameters learned only from seen training concepts and additional high-level semantic information.

The fundamental question we wish to address in this work is: *What are the principles that contribute to learning good representations for ZSL?* While the most successful ZSL models [Atzmon and Chechik, 2019, Wang et al., 2019] use pretrained features from Imagenet [Krizhevsky et al., 2012, Russakovsky et al., 2015], we wish to understand how these features can emerge given only the data provided from the ZSL task. Specifically, we explore the role of *compositionality* and *locality* [Tokmakov et al., 2018, Stone et al., 2017] as two principles that lead to good generalization. Our study focuses on image representations, so we explore various means of learning representations that are local and compositional for convolutional neural networks (CNNs). We also leverage the structure of CNNs and available annotations from ZSL datasets as a means of interpreting various models in terms of these factors. Overall, our results support the hypothesis that compositionality and locality are crucial principles for training models that generalize well.

Finally, in order to provide a cleaner framework for understanding the relationship between the above principles and generalization, we re-introduce *Zero-Shot Learning from scratch* (ZFS). In this setting, the model is *not allowed to be pretrained on another dataset*, such as Imagenet, and is evaluated on its ability to perform classification using auxiliary attributes and labels trained *only* using the data available from the training split of the target dataset. We believe that ZFS will provide researchers with a better experimental framework to understand which principles are important for Zero-Shot generalization.

The contributions of our work are as follows:

- We introduce Zero-Shot Learning from scratch (ZFS), an extension to ZSL, which we believe will be an important benchmark for understanding which learning principles lead to better generalization.
- We evaluate several supervised and unsupervised methods on their ability to learn features that generalize in the ZFS setting by training a prototypical network on top of those features [in a similar way to what was done in Snell et al., 2017, with Imagenet features]. We then relate this generalization performance with different proxies for locality and compositionality of the given representations, and show that both concepts contribute heavily.
- We introduce a novel version of Deep InfoMax [DIM, Hjelm et al., 2018] which draws local patch representations from other images with the same label as positive samples.
- We introduce a novel visualization technique based on Mutual Information, that allows to investigate local properties of the learned representations.

4.2. Principles that lead to good ZSL performance

Zero-Shot Learning [ZSL, Larochelle et al., 2008] is an important learning framework for understanding a model's capacity to be used in real world scenarios where many relevant test cases (e.g., classes) are not known or are infeasible to sample at training time. An important component of ZSL, particularly in Deep Learning, is learning features directly from raw data (e.g., pixels), that generalize to these test cases. While there are a number of commonly-used strategies for learning generalizable features for various tasks in Deep Learning [Neyshabur et al., 2017], we believe that ZSL in particular requires thinking beyond normal classification by incorporating principles such as compositionality [Boole, 1854] and locality [Fukushima, 1980].

How we formulate, exploit, and analyze these principles to learn models that solve image ZSL tasks will depend on our use of convolutional neural networks (CNNs) as the network architecture for encoding images as well as properties of the data, such as input statistics and available annotations. We will broadly define compositionality and locality, then relate these principles to the tools we have at our disposal from the network architecture and data.

4.2.1. Compositionality

Compositional representations have been a focus in the cognitive science literature [Biederman, 1987, Hoffman and Richards, 1984] with regards to the ability of intelligent agents to generalize to new concepts. Applications are found in computational linguistics [Tian et al., 2016], generative models [Higgins et al., 2017a], and Meta Learning [Alet et al., 2018, Tokmakov et al., 2018], to name a few, with approaches to encourage compositionality varying from introducing penalties [Tokmakov et al., 2018] to using modular architectures [Alet et al., 2018, Goyal et al., 2019, Goyal and Bengio, 2020].

In what follows, we will consider a representation to be compositional if it can be expressed as a combination of simpler parts [Andreas, 2019, Cresswell, 1976]. Let $\mathcal{P}$ denote the set of possible parts, $\mathcal{R}$ the representation space and $\mathcal{X}$ the input space. For each $x \in \mathcal{X}$, we assume the existence of a function D mapping x to $\mathcal{P}' \subseteq \mathcal{P}$, the set of its parts. These parts could be local image features (e.g., wings or beaks), or other generative factors (e.g., size, color, etc). Let $g : \mathcal{P} \to \mathcal{R}$ be a function that maps the parts to representations. Formally, $f(x) \in \mathcal{R}$ is compositional if it can be expressed as a combination of the elements of $\{g(p) | p \in D(x)\}$. The combination operator used is commonly a weighted sum [Brendel and Bethge, 2019a], although some works learn more complex combinations [Higgins et al., 2017a]. As we consider representations that are implicitly compositional, the above formalism might be approximately true which motivates our later use of the TRE metric.

4.2.2. Locality

Local features have been used extensively in representation learning. CNNs exploit local information by design, and locally-aware architectures have been shown to be useful for non-image dataset, such as graphs [Kipf and Welling, 2016] and natural language processing [Yu et al., 2018]. For supervised image classification, a bag of local features processed independently can do surprisingly well compared to processing the local features together [Brendel and Bethge, 2019b]. Attention over local features is commonly used in image captioning [Li et al., 2017], visual question answering [Kim et al., 2018] and fine-grained classification [Sun et al., 2018]. Self-attention over local features resulted in large improvements in generative models [Zhang et al., 2018a]. Self-supervised methods often exploit local information to learn useful representations: Doersch et al. [2015] proposes to learn representations by predicting the relative location of image patches, Noroozi and Favaro [2016] solves jigsaw puzzles of local patches, and Deep InfoMax [DIM, Hjelm et al., 2018] maximizes the mutual information between local and global features.

For our purposes with image data, we loosely define a local representation as one that has information that is specific to a patch. This helps motivate choices in architecture and learning principles that encourage locality. In this work, we take the most straightforward approach, and use features of CNNs which have receptive fields that are small compared to the size of the full image. This is similar to the motivations in Bachman et al. [2019], where the architecture is carefully designed to ensure that the receptive fields do not grow too large. However, this choice in architecture does not guarantee locality, as CNN representations might hypothetically contain only "global" information, such as the class or color of the object, despite having a limited receptive field. Therefore, we will evaluate a number of different models on their ability to encode only information specific to those locations. We will discuss relevant evaluation in Sections 4.4.2 and 4.4.3.

Note also that compositionality as discussed above and locality are not necessarily independent concepts, nor are they necessarily the same. The set of compositional factors could include local factors, such as parts of an object, but also be more "global" factors, such as general properties of a class (e.g., size, color, shape, etc).

4.2.3. Compositionality and locality with image data

We focus on three common ZSL datasets that allow us to explore compositionality and locality, namely Animals with Attributes 2 [**AwA2**, Xian et al., 2018], Caltech-UCSD-Birds-200-2011 [**CUB**, Wah et al., 2011b], SUN Attribute [**SUN**, Patterson and Hays, 2012]. Typical images from these datasets are shown in Fig. 4.1: CUB is a fine-grained dataset, where the object of interest is small relative to the total image. This is in contrast to AwA2, where subjects have variable size in relation to the total image. SUN is a scenes dataset, meaning that the object of interest is often the whole image.

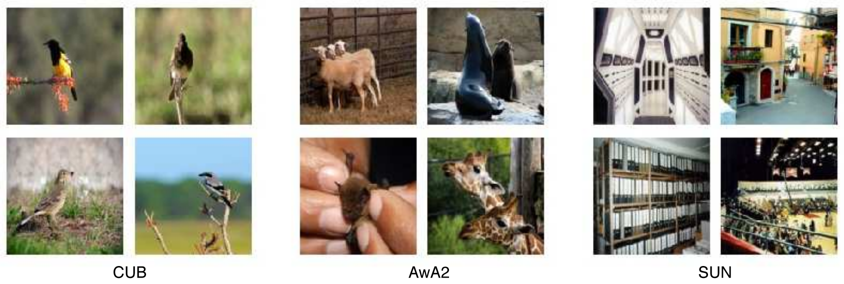


Fig. 4.1. Typical samples show how compositionality and locality are expressed differently in the datasets we consider in this study.

In our evaluation of compositionality, we can leverage different annotations provided by the datasets. All of these datasets provide attributes, which roughly correspond to high-level semantic information composed of a set of underlying factors. For CUB, these attributes describe visual characteristics such as wing colour or shape. For AwA2, these describe both visual and behavioral characteristics, such as the animal's ability to swim or its habitat. For SUN, the attributes are more diverse, ranging from describing the function of the scene, such as playing, to the spatial envelope, for instance man-made. As in Xian et al. [2017], we used ℓ_2 normalized versions of these attributes as semantic class representations. In addition to attributes, CUB comes with bounding boxes for the whole subject and parts locations for 15 different parts, e.g. beak, tail, belly. These can also be used to assess both locality and compositionality, with the compositional factors being the same as the local ones. The details of each dataset are in the Appendix (Table A.1).

4.3. Zero-Shot Learning from scratch

While the original ZSL setting introduced in Larochelle et al. [2008] was agnostic to the exact methodology, more recent image ZSL approaches almost uniformly use features from very large "backbone" neural networks such as InceptionV2 [Szegedy et al., 2016] and ResNet101 [He et al., 2016] pretrained on the Imagenet dataset. In terms of absolute performance, this approach appears to be well-justified, as state-of-the-art results on various ZSL [Yosinski et al., 2014, Sun et al., 2017, Huh et al., 2016, Azizpour et al., 2015] and non-ZSL benchmarks [Li et al., 2019b, Zhang et al., 2018b, He and Peng, 2017] all learn on top of similar pretrained backbones.

However, we have many concerns with this approach towards our goal of understanding the principles that contribute to good generalization. First, relative success in transfer learning has been shown to be highly dependent on the precise instantiation of the pretrained backbone encoder [Xian et al., 2018] or the pre-training dataset [Cui et al., 2018]. Next, while Imagenet features have been shown to work well for ZSL tasks with similar image datasets, there are no guarantees that a suitable pre-training framework would be available in general ZSL settings. Conversely, it can be hard in practice to meaningfully evaluate a Zero-Shot learner, as performance on specific classes is impacted by their presence in the pre-training dataset [Xian et al., 2017].

Finally, we believe this approach misses the point, in such a way that makes understanding the learning principles that contribute to good generalization difficult. We believe that ZSL should first and foremost be used as a framework for training, understanding, and evaluating models on their ability to reason about new, unseen concepts. Despite the *absolute performance* gains of the methods above that use Imagenet features, the use of backbones hyper-optimized for supervised performance on Imagenet and the Imagenet dataset itself represent *nuisance variables* in a larger effort to understand how to learn generalizable concepts from scratch.

In addition to the ZSL task framework outlined in Larochelle et al. [2008], ZFS simply adds one additional additional requirement: *No model parameters can contain information about (e.g., can be learned from) data outside that from the training split of the target dataset.*

4.4. Methods

Given our hypothesis on the importance of locality and compositionality, we consider a wide range of representation learning methods trained using the ZFS setting described in Section 4.3. To showcase the role of these principles, we will introduce a set of proxies for compositionality and locality below. We will also consider auxiliary losses that emphasize locality in the learned representation. Finally, we will introduce a visualization tool that can help identify which local representations are assigned higher importance by each method.

4.4.1. General approach

In this work, we train convolutional image encoders (CNN) using either supervised or unsupervised learning, then use prototypical networks to perform ZSL transfer on these fixed representations. Prototypical networks are chosen as they require minimal parameters or hyper-parameters tuning, are well-studied [Huang et al., 2019, Finn et al., 2017], and performance is very close to the state of the art for Imagenet-pretrained benchmarks. Our setup is representative of the current state of ZSL models, most of which [Akata et al., 2015, Changpinyo et al., 2016, Kodirov et al., 2017, Zhang et al., 2017a, Sung et al., 2018] rely on metric learning by applying two steps: (1) learning a suitable embedding function that maps data samples and class attributes to a common subspace, (2) performing nearest neighbor classification at test-time with respect to the embedded class attributes.

For our study, we compare pre-training the image encoder with a diverse, yet representative set of models:

- *Fully supervised*: Fully supervised label classifier (**FC**)
- *Unsupervised / reconstruction based / generative*: Variational auto-encoders[**VAE**, Kingma and Welling, 2013], β-**VAE** [Higgins et al., 2017b], Adversarial auto-encoders [**AAE**, Makhzani et al., 2015],
- *Local self-supervision and variants*: Augmented Multiscale Deep InfoMax [**AMDIM**, Bachman et al., 2019] and Class Matching DIM (**CMDIM**).

We pick variants of DIM [Hjelm et al., 2018] as opposed to other self-supervision methods [Doersch and Zisserman, 2017, Noroozi and Favaro, 2016] because extensions have achieved state-of-the-art on numerous related tasks [Veličković et al., 2018, Bachman et al., 2019].

We introduce Class-Matching DIM (CMDIM), a novel version of DIM that draws positive samples from other images from the same class. The goal is to learn representations that focus less on the information content of a single input, while extracting information that

is discriminative between classes. The hyperparameter p determines the probability of performing intra-class matching, we experiment with $p \in \{1, 0.5, 0.1\}$. A more detailed description is provided in Appendix A.7.
Local classification and attribute auxiliary loss. We encourage the image encoder to extract semantically relevant features at earlier stages of the network by introducing an auxiliary local loss to the local feature vectors of a convolutional layer (whose receptive field covers a small patch of the image). When used, this auxiliary loss is either from an attribute-based classifier (AC) or a label-based classifier (LC) using the attributes or the labels as supervised signal, respectively. A schematic explanation can been seen in Fig. A.1 in the appendix.

4.4.2. Parts classification for CUB evaluation

For each of the 15 parts labelled in the CUB dataset, we use the MTurk worker annotations to construct 15 boolean map for each local feature. We project these boolean maps through the CNN to generate ground truth variables that indicate whether the given part is present and *visible* at a location specific to the CNN encoder features. We then train a linear probe for each part, without back-propagating through the encoder, and measure the average F1 score across all locations and parts. This gives us a measurement on how well the encoder represents the parts of the image *at the correct locations*. For more details, please refer to Section A.6 in the Appendix.

4.4.3. Measuring mutual information between local features of different images

As a tool for local interpretability, we propose estimating the mutual information (MI) between global features given from one image and local features from a second image. A schematic explanation is provided in Fig. A.2 in the Appendix. In order to do this, we rely on MINE [Belghazi et al., 2018b] which uses a *statistics network*, T_ϕ, with parameters ϕ to formulate a lower bound to MI, which is effective for high dimensional, continuous random variables. In our case, the statistics network takes two inputs: a global and local feature vector either sampled from the joint, where each comes from the same image, or from the product of marginals, where the global and local features are sampled independently from each other. The statistics network optimizes a lower bound to the MI:

$$\widehat{I}(G_\theta(X); L_\theta(X)) \geq \mathbb{E}_{p(X)}[T_\phi((G_\theta(X), L_\theta(X))] - \log \mathbb{E}_{p(X)\otimes p(X')}[e^{T_\phi((G_\theta(X), L_\theta(X'))}]. \quad (4.4.1)$$

Where $p(X) = p(X')$ is the data distribution, and L and G random variables corresponding to the local and global feature vectors of the encoder. At optimum, the output of the statistics network, T_ϕ provides an estimate for the Pointwise Mutual Information (PMI), defined as $\log \frac{p(G,L)}{p(G)p(L)} = \log \frac{p(L|G)}{p(L)}$, which roughly gives us a measure of how similar the global and local representations are in terms of information content. Normally, we could try to estimate the marginal term $p(L)$ to get an estimate for the conditional density, but we will normalize our score across the local patches of the target image to get a *relative* score of the relatedness of each local feature to a given global feature. This analysis is similar to that done in Bachman et al. [2019], which looks at different augmentations of the same image using the same MI estimator used to train the encoder.

4.4.4. TRE evaluation

We focus on a metric of compositionality introduced in Andreas [2019], the Tree Reconstruction Error (TRE), as a proxy for compositionality. We will write $\mathrm{TRE}(\mathcal{X}, \mathbf{a})$ for the TRE computed on a dataset $\mathcal{X}$ over the set of primitives $\mathbf{a}$. For details on its definition, see Section A.8 in the Appendix.

As we mostly care about the compositionality with respect to attributes in the context of zero-shot learning and some representations are inherently more decomposable than others (such as VAEs due to the gaussian prior), we consider instead the ratio of the TRE computed with respect to attributes and the TRE computed with respect to uninformative variables (random assignment). We define the TRE ratio as: $\frac{\mathrm{TRE}(\mathcal{X},\hat{\mathbf{a}})}{\mathrm{TRE}(\mathcal{X},\tilde{\mathbf{a}})}$, where $\tilde{\mathbf{a}}$ is a random binary matrix (random cluster assignment), and $\hat{\mathbf{a}}$ are the actual visual primitives, attributes in our case.

4.4.5. Experimental Setup

Image encoders. We considered both an encoder derived from the DCGAN architecture [Radford et al., 2015] and similar in capacity to those used in early Few Shot Learning models such as MAML [Finn et al., 2017]. We also consider a larger AlexNet [Krizhevsky et al., 2012] based architecture to gain insight on the impact of the encoder backbone. It is important to note that overall the encoders we use are significantly smaller than the “standard" backbones common in state-of-the-art Imagenet-pretrained ZSL methods (note that similarly to most recent ZSL methods, the encoder is fixed after pre-training, and used as a feature extractor). We believe restricting the encoder’s capacity decreases the performance, but does not hinder our ability to extract understanding of what methods work from our experiments. A detailed description of the architectures can be found in the Appendix (Tables A.2 and A.3).
Evaluation Protocol. We used the ZSL splits constructed in Xian et al. [2017], as they are the most commonly used in the literature. All models are evaluated on Top-1 accuracy. We pretrain the encoder using each of the previously mentioned methods (strictly on the considered dataset, as per the ZSF requirement). We then train a Prototypical Network on top of the (fixed) learned representation. All the implementation details are available in the Appendix, in Section A.2.

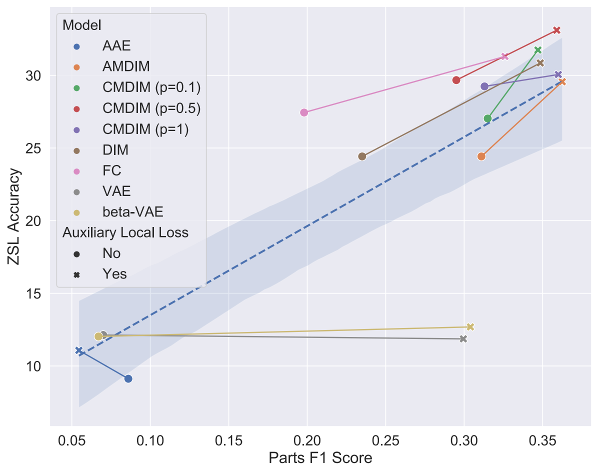


Fig. 4.2. Parts F1 score for all models on CUB with a DCGAN-based encoder plotted against ZSL accuracy. There is a clear relationship between the two: encoders that have a good understanding of local information (as measured by the parts F1 score) perform better in zero-shot learning. The addition of a local loss increases parts F1 score for all models. This improves generalization for all models except those trained with a reconstruction objective.

4.5. Results and Discussion

In this section, we describe in more detail our experiments and analyze the results. The full numerical results and plots for all considered models can be found in the Appendix.

4.5.1. Does locality help ZSL, and can local representations be learned?

Representations that predict parts at the correct location tend to perform better at ZSL. We hypothesize that if the encoder represents information that is descriptive of parts, it should also be able to generalize better. To test this, we compare ZSL performance to the part classification F1 score described in 4.4.2. In Fig. 4.2, the average F1 scores across the 15 classifiers is plotted against the ZSL accuracy for each model. The two measures are clearly correlated (Person's correlation of 0.73). This relationship doesn't hold for reconstruction-based methods such as VAEs, which could be due to these models needing to represent information related to all pixels, including the background, in order to reconstruct well.

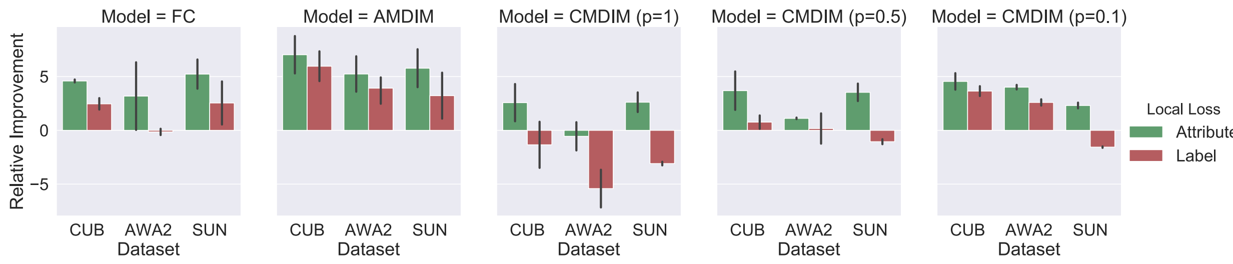


Fig. 4.3. Relative improvement in terms of ZSL accuracy with respect to models trained without the auxiliary loss. Attribute information results in a bigger improvement. Surprisingly, for certain models label information results in a decrease in generalization performance.

Encouraging local representations to contain similar information improves ZSL performance For variants of Deep InfoMax [DIM, Hjelm et al., 2018], AMDIM and CMDIM, the local representations are encouraged to be similar to a global representation through the mutual information maximization objective. While locally specified, this is somewhat contrary to our definition of locality in 4.2.2. Among the models we tested, these variants generally perform very well, with CMDIM performing the best overall.

This indicates that, while important, locality by itself is not sufficient to learn good ZSL representations. The local representations must also share information, e.g., through a global representation or the class. We hypothesize that such constraints help the encoder learn information that is present locally, but relevant to discriminating the class or important high-level semantic meaning. The above observation also holds for the local losses (AC and LC introduced in 4.4.1). These losses both encourage the model to rely on local information (local features must capture important semantic information), and for these representations to share information (by having high mutual information with either the attributes or labels).

Adding local losses helps supervised and self-supervised models. We investigate in more detail the effect of encouraging the model to take into account different types of local information. As can be seen in Fig. 4.2, the addition of a local loss improves both ZSL and parts score for all models except the generative ones (VAE, AAE). Interestingly, for these models the parts score also increases, indicating more *locality*, but this does not translate to better ZSL performance.

To better investigate why local losses improve generalization for supervised and self-supervised models, in Fig. 4.3 we show the relative improvement of each type of local classifier over the performance of the encoder only trained with its global loss. We can see how for the supervised model, the attribute based auxiliary loss has a much bigger impact, which indicates that label information is already exploited by the model, while attributes actually provide more information. For AMDIM, both losses seem to have a consistent positive effect, possibly because the model is unsupervised and hence any label-related information is useful. For CMDIM, the LC auxiliary loss actually hurts performance. This is likely due to the fact that both CMDIM and the LC loss focus on discriminating classes at the local level, and that the LC objective is inherently less effective than the CMDIM formulation for this task (in terms of downstream ZSL performance). As a result, forcing the model to account for both terms lowers downstream performance. DIM and AMDIM discriminate instances and not classes so adding class and attribute information in the form of the AC and LC losses helps performance. CMDIM is already exposed to class information, so only gains from being exposed to the (more informative) attribute information in the form of AC.

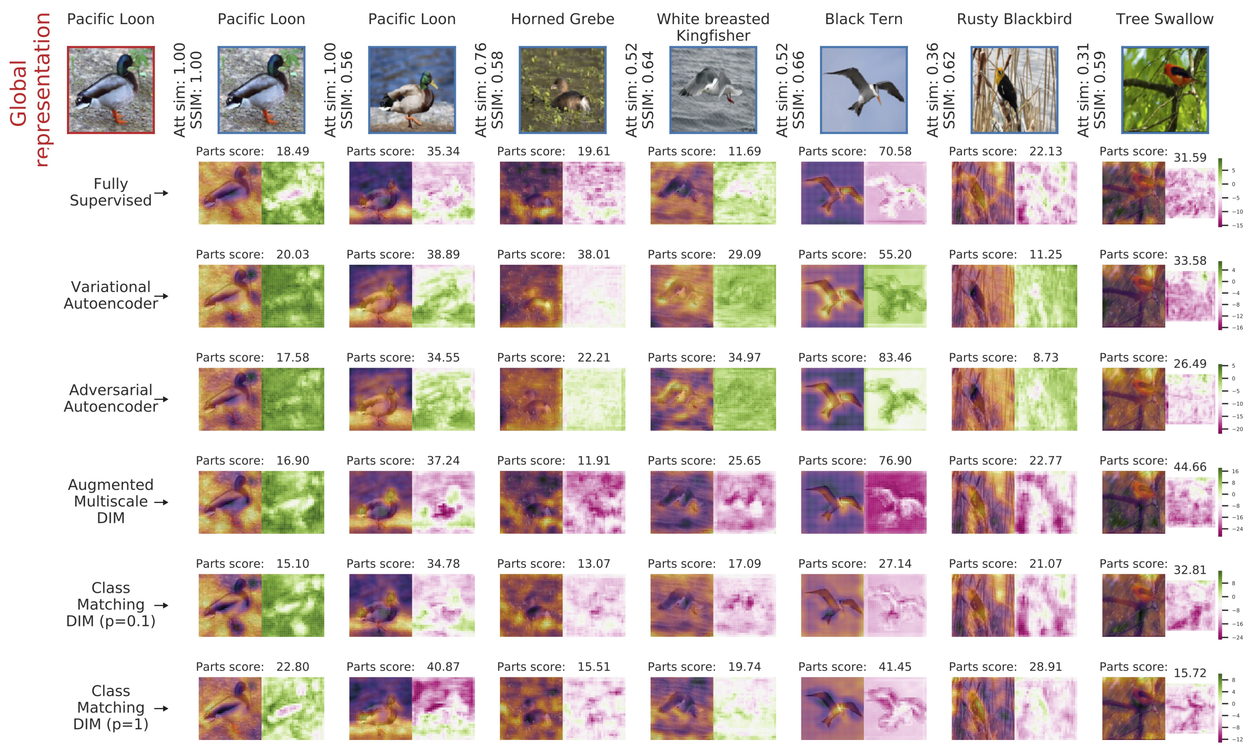


Fig. 4.4. Mutual Information heatmaps allow to understand which local patches contributed the most to the final representation. For each heatmap we plot the absolute values in the rightmost plot and the superposition of the heatmap and the original image on the left, to increase interpretability. Yellow corresponds to higher scores.

4.5.2. How do different models encode information locally?

In Fig. 4.4 we apply the PMI-based visualization technique introduced in Section 4.4.3 to pairs of images from the CUB dataset. In this case, we are examining the global representation extracted by each model for the top left image (the Pacific Loon) and comparing it with local features from images of various classes. By noticing which patches each model pays attention to (have higher mutual information), we can infer how information is coded locally. The main take-aways are the following:

- **Supervised models.** The fully supervised model in the first row seems to be able to focus on relevant semantic details, such as the tail of the Horned Grebe and the head of the Back Tern.
- **Unsupervised models.** In the second and third row we can see how models based on a reconstruction loss seem to fail at highlighting semantic information: for images with patterns and colours similar to the Pacific Loon, such as the other Pacific Loon or the White breasted Kingfisher, PMI is high across all local features, while scores are very low for the Tree Swallow with the uniform green background. This could possibly

indicate that these models focus more on pixel statistics necessary for reconstruction, and are unable to extract as much semantic information.

- **Self-Supervised models.** AMDIM manages to recover some semantic structure, e.g. for the other Pacific Loon or for the Rusty Blackbird, but fails in some other cases. CMDIM on the other hand, especially for high matching probability p, produces heatmaps that are very similar to those of the supervised models, hence managing to recover what is discriminative between classes.

Models that encode semantic information well perform well on ZSL generalization. Models that reconstruct pixels well perform worse. To confirm our intuitions on how some families of models focus more on semantic information, while others are more sensitive to pixel statistics, we rely again on the parts binary maps described in 4.4.2. For each pair of images, we compute the ratio between the score assigned to patches containing *any* part (i.e., a logical OR computed across all binary maps) and the overall score of all features. We refer to this measure as the *Parts ratio*. This ratio is sensitive to both how highly the model scores the relevant parts and to whether the rest of the features are assigned a lower score. We then compute two different types of similarity between the considered images: a semantic one, defined as the cosine distance between the attributes associated to the images' classes, and an pixel-wise one, by measuring the Structural Similarity index (SSIM) between the images. We then measure correlation between the Parts ratio and these measures of similarity. Our interpretation is the following:

- **Positive correlation with attribute similarity:** If two images are semantically similar, the part ratio should be higher if the model manages to extract the common semantically relevant patches (and we know that they are the parts for CUB).
- **Negative correlation with SSIM score:** If a model is too sensitive to pixel similarity, the parts score will be higher for images that are very different, where the only thing in common (pixel-wise) is to depict a bird, while for very similar images the model will just assign a high score to all local features.

We find that for VAEs, the ratio and the attribute similarity are not correlated, but the ratio and the SSIM scores correlate negatively. The effect is reversed for Supervised and Self-Supervised models. This confirms our intuition that VAEs and reconstruction models are not well suited to learn representations that generalize in our context. More details about this experiment and the correlation coefficients are reported in the Appendix, in Fig. A.4.

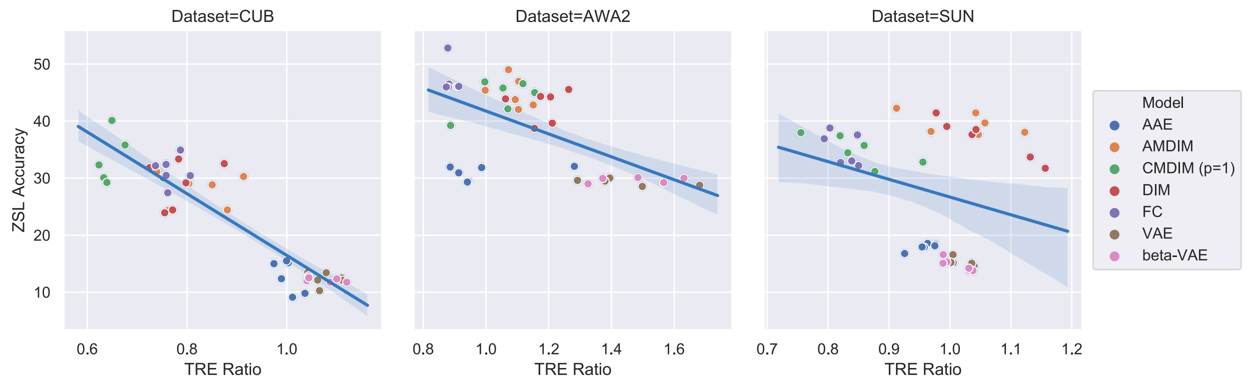


Fig. 4.5. Relationship between TRE ratio and ZSL accuracy for each dataset (lower TRE ratio is better).

4.5.3. Do compositional representations perform better for ZSL?

Intuitively, we expect compositionality to be an advantage: if a model has a good understanding of how parts map to representations, it can learn to combine known concepts to describe new classes. The experiments show:

- **Measures of implicit compositionality correlate strongly with ZSL performance.** Fig. 4.5 shows the relationship between the TRE ratio introduced in 4.4.4, and ZSL accuracy. The Pearson correlation coefficients between the TRE Ratio and ZSL Accuracy are the following: -0.90 for CUB, -0.60 for AwA2 and -0.30 for SUN.
- **The relation is strongest when the attributes are strongly relevant to the image.** For the AWA2 and CUB, datasets for which the attributes are semantically very meaningful, we observe that there is a direct relationship between TRE ratio and ZSL performance. This relationship degrades for SUN, for which the attributes are per-image, and averaged over classes, meaning that they are less likely to map to information actually present in a given image.

We also consider the effect of combining local representations directly (instead of relying on the global output of the model). Given local representations, there are several ways to employ them to perform classification: one option is to create a final representation by averaging the local ones, another option is to classify each patch separately and then average this predictions.

An explicitely compostional model based on local features helps ZSL. The results for this comparison are shown in Fig. 4.6. Averaging representations can be seen as directly enforcing a notion of compositionality: the representation of the whole input is directly built as a weighted sum of the patch representations (that we can imagine being more similar across different data points) and where the weights are uniform. For CUB, and to a lesser extent AwA2, where only few patches encode important information such as beaks, tails, the effect is quite pronounced. There is less of a difference for SUN, where the object is usually the whole scene, meaning all patches are expected to contribute.

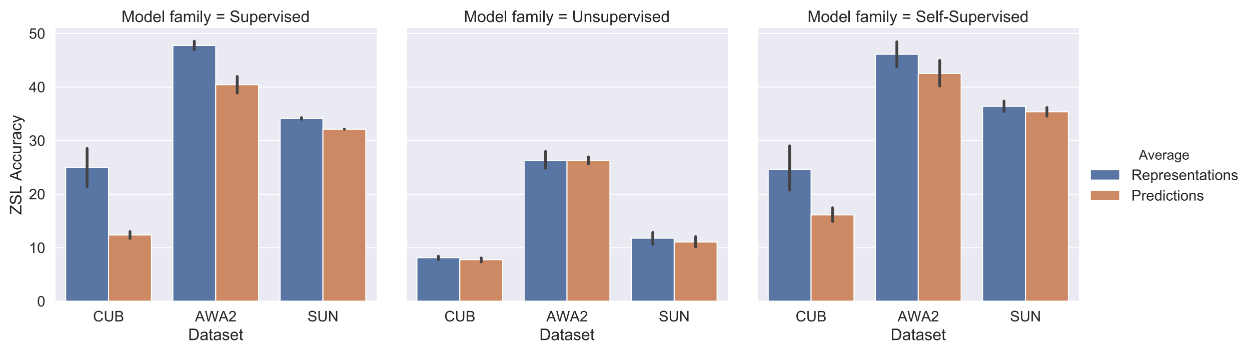


Fig. 4.6. Comparison between averaging representations and averaging predictions. We can see how, for the more local model families, this notion of compositionality is most useful for CUB, where the object of interest is likely only present in few patches.

4.6. Conclusion and future work

Motivated by the need for more realistic evaluation settings for Zero-Shot Learning methods, we proposed a new evaluation framework where training is strictly performed only on the benchmark data, with no pre-training on additional datasets. In the proposed setting, we hypothesize that locality and compositionality are fundamental ingredients for successful zero-shot generalization. We perform a series of tests of the relationship between these two aspects and zero-shot performance of a diverse set of representations. We find that models that encourage both these aspects, either explicitly (through a penalty per instance) or implicitly by construction, tend to perform better at zero-shot learning. We also find that models that focus on reconstruction tasks fail at capturing the semantic information necessary for good generalization, calling into question their applicability as general representation learning methods.

Chapter 5

Prologue to the second paper

5.1. Article details

Object-Centric Image Generation from Layouts [Sylvain et al., 2020b, OC-GAN]. Tristan Sylvain, Pengchuan Zhang, Yoshua Bengio, R Devon Hjelm, Shikhar Sharma. Association For The Advancement Of Artificial Intelligence 2021.
Personal contributions As lead author, I wrote most of the paper, ran the large majority of the experiments, and wrote a large majority of the code. Shikhar helped a lot at all stages of the project and was truly instrumental in making this paper what it is today. All my other collaborators were very helpful at all stages.

5.2. Context

The paper is the result of my first internship at Microsoft Research Montreal in 2019. The whole team had a strong interest in image generation, with in particular Pengchuan having co-authored a major paper in the field [Xu et al., 2018a]. After many research discussions, we started to gain a better understanding of common failure modes of layout-to-image models. The paper grew from our efforts to address those, by introducing new components and through a proposed new metric.

5.3. Contributions

Our paper addresses the issues commonly found in layout-to-image models: spurious objects, poor layout-fidelity (i.e. generated images do not correctly match their layout), and issues with the evaluation setting (for more details, please refer to the paper). As a result, we introduce the scene graph similarity module (SGSM), which alleviates some of these issues, and discuss the introduction of SceneFID, our new metric, as a replacement for the commonly considered FID. Finally, we perform extensive evaluation of our proposed approach to showcase its value.

Chapter 6

Object-Centric Image Generation from Layouts

6.1. Introduction

Generative Adversarial Networks (GANs) [Goodfellow et al., 2014b] have been at the helm of significant recent advances in image generation [Goodfellow et al., 2014b, Radford et al., 2016, Gulrajani et al., 2017b, Miyato and Koyama, 2018, Brock et al., 2019]. Apart from unsupervised image generation, GAN-based image generation approaches have done well at conditional image generation from labels [Radford et al., 2016, Zhang et al., 2019b, Brock et al., 2019], captions [Reed et al., 2016a, Zhang et al., 2017b, Xu et al., 2018a, Li et al., 2019c, Yin et al., 2019], conversations [Sharma et al., 2018, El-Nouby et al., 2019, Li et al., 2019d], scene graphs [Johnson et al., 2018, Mittal et al., 2019, Ashual and Wolf, 2019], layouts [Zhao et al., 2019a, Sun and Wu, 2019], segmentation masks [Park et al., 2019], *etc.* While the success in single-domain or single-object image generation has been remarkable, generating complex scenes with multiple objects is still challenging.

Generating realistic multi-object scenes is a difficult task because they have many constituent objects [e.g., the Visual Genome dataset, Krishna et al., 2017, can contain as many as 30 different objects in an image]. Past methods focus on different input types, including scene graphs [Johnson et al., 2018, Ashual and Wolf, 2019], pixel-level semantic segmentation [Li et al., 2019c], and bounding box-level segmentation [Zhao et al., 2019a, Sun and Wu, 2019]. In addition, some methods also consider multi-modal data, such as instance segmentation alongside pixel-wise semantic segmentation masks [Park et al., 2019, Wang et al., 2018b]. Orthogonal to input-related considerations, methods tend to rely on additional components to help with the complexity of scene generation, such as attention mechanisms [Xu et al., 2018a, Li et al., 2019c] and explicit disentanglement of objects from the background [Singh et al., 2019].

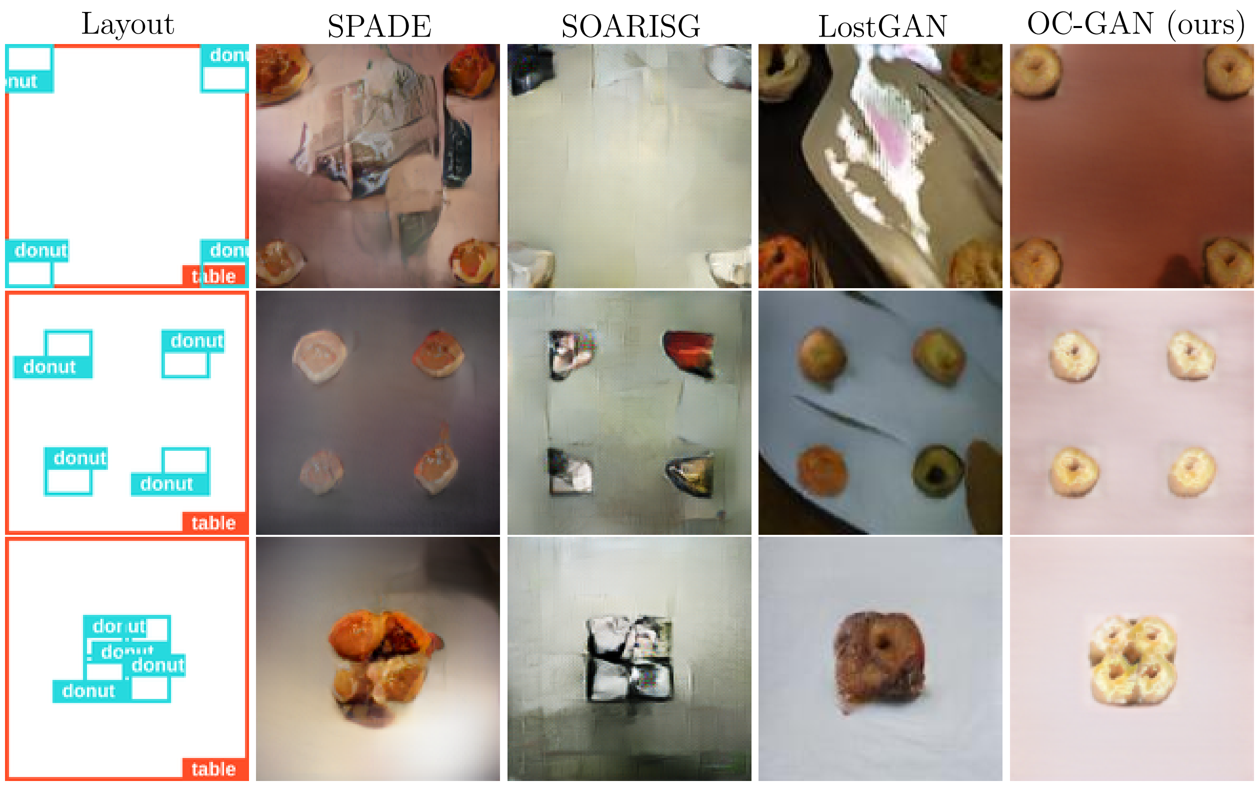


Fig. 6.1. Each row depicts a layout and the corresponding images generated by various models. Along each column, the donuts converge to the centre. In addition to more clearly defined objects, our method is the only one that maintains distinct objects for the final layout, for which bounding boxes slightly overlap.

Despite these advances, models still struggle in creating realistic scenes. As shown in Figs. 6.1 and 6.2, even simple layouts can result in merged objects, spurious objects, and images that do not match the given layout (low layout-fidelity). To counter this, we propose Object-Centric GAN (OC-GAN), an architecture to generate realistic images with *high layout-fidelity* and *sharp objects*. Our primary contributions are:

- We introduce a set of novel components that are well-motivated and improve performance for complex scene generation. Our proposed scene-graph-based retrieval module (SGSM) improves layout-fidelity. We also introduce other improvements, such as conditioning on instance boundaries, that help generating sharp objects and realistic scenes.
- Our model improves significantly on the previous state of the art in terms of a set of classical metrics. In addition to standard metrics, we also perform a detailed ablation study to highlight the effect of each component, and a human evaluation study to further validate our findings.

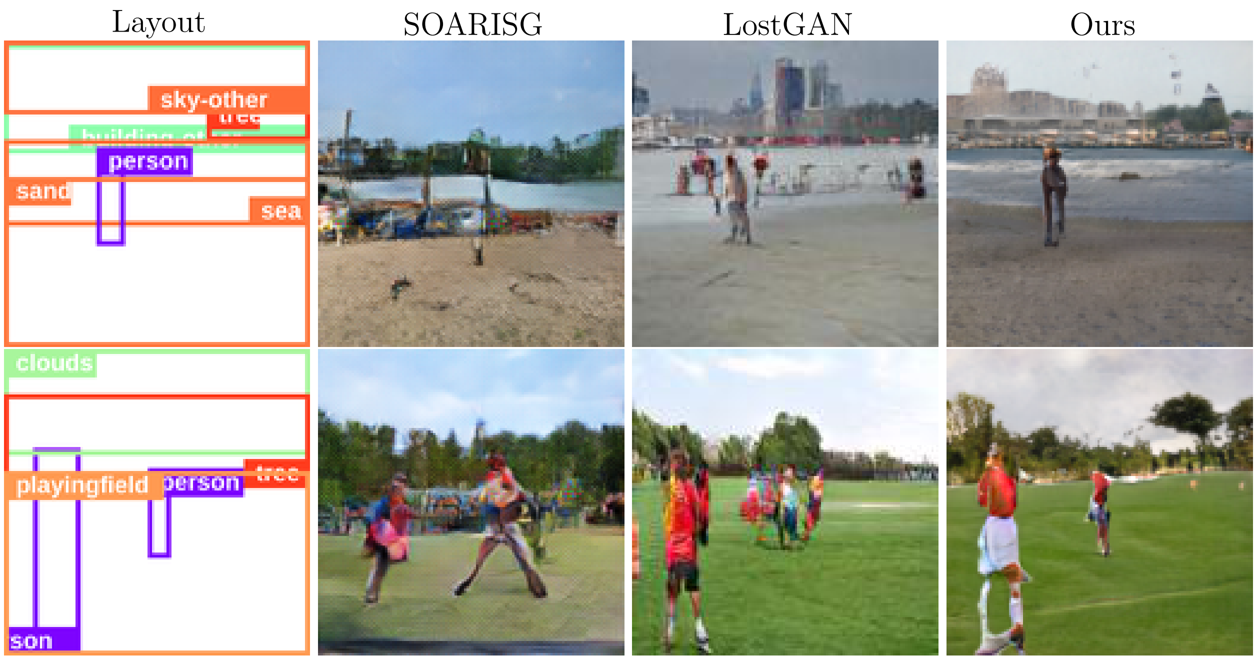


Fig. 6.2. Existing models introduce spurious objects not specified in the layout, a failure mode over which our model improves significantly.

- We discuss the validity of the metrics currently used to evaluate layout-to-image methods, and building on our findings, motivate the use of SceneFID, a new evaluation setting which is more adapted to multi-object datasets.

6.2. Related Work

6.2.0.1. Conditional scene generation. For some time, the image generation community has focused on scenes that contain multiple objects in the foreground [Reed et al., 2016a, Zhang et al., 2017b, Johnson et al., 2018]. Such scenes, which can contain large amount of objects of very different scales, are very complex relative to single-object images. Several conditional image generation tasks have been formulated using different subsets of annotations. Text-based image generation using captions [Reed et al., 2016a, Zhang et al., 2017b, Xu et al., 2018a, Li et al., 2019c, Yin et al., 2019] or even multi-turn conversations [Sharma et al., 2018, El-Nouby et al., 2019, Li et al., 2019d] have gained significant interest. However, with increasing numbers of objects and their relationships in the image, understanding long textual captions becomes difficult [Johnson et al., 2018, Sharma et al., 2018]. Text-based image generation approaches are also not immune to small perturbations in text leading to quite different images [Yin et al., 2019].

6.2.0.2. Layout-based synthesis. Generating images from a given layout makes the analysis more interpretable by decoupling the language understanding problem from the image generation task. Another advantage of generating from layouts is more controllable generation:

it is easy to design interfaces to manipulate layouts. In this work we will focus on coarse layouts, where the scene to be generated is specified by bounding-box-level annotations. Layout-based approaches fall into 2 broad categories. Some methods take scene-graphs as inputs, and learn to generate layouts as intermediate representations [Johnson et al., 2018, Ashual and Wolf, 2019]. In parallel, other approaches have focused on generating directly from coarse layouts [Sun and Wu, 2019, Zhao et al., 2019a]. Models that perform well on fine-grained pixel-level semantic maps also can be easily applied to this setting [Park et al., 2019, Isola et al., 2017, Wang et al., 2018b]. Almost all recent approaches have in common the use of *patch* and *object discriminators* (to ensure whole image and object quality). In addition to this, image quality has been improved by the addition of *perceptual losses* [Park et al., 2019, Ashual and Wolf, 2019, Wang et al., 2018b], *multi-scale patch-discriminators* [Park et al., 2019], which motivate some of our architecture choices. Finally, modulating the parameters of batch- or instance-normalization layers [Ioffe and Szegedy, 2015, Ulyanov et al., 2016] with a function of the input condition can provide significant gains, and this is done per-channel in Odena et al. [2017] or per pixel in Park et al. [2019], Sun and Wu [2019]. As bounding box layouts are coarse for this task, it is common to introduce *unsupervised mask generators* [Sun and Wu, 2019, Ma et al., 2018] to provide estimated shapes for this conditioning.

Finally, there is a growing body of literature involving semi-parametric [Qi et al., 2018, Li et al., 2019a] models that use ground-truth training images to aid generation. We consider the case of such models in the Appendix.

6.2.0.3. Scene-graphs and image matching. Scene graphs are an object-centric representation that can provide an additional useful learning signal when dealing with complex scenes. Scene-graphs are often used as intermediate representations in image captioning [Yang et al., 2019, Anderson et al., 2016], reconstruction [Gu et al., 2019] and retrieval [Johnson et al., 2015], as well as in sentence to scene graph [Schuster et al., 2015] and image to scene graph prediction [Lu et al., 2016b, Newell and Deng, 2017].

By virtue of being a simpler abstraction of the scene than a layout, they emphasize *instance awareness* more than layouts which focus on pixel-level class labels. Secondly, for scenarios that might require generating multiple diverse images, they provide more variability in reconstruction and matching tasks as the mapping from a scene graph to an image is one to many usually. These points explain their use in higher-level visual reasoning tasks such as visual question answering [Teney et al., 2017] and zero-shot learning [Sylvain et al., 2020a,c], and also motivate the use of scene graph-based retrieval in our model. In our work, we generate scene graphs depicting positional relationships (such as "to the left of", "above", "inside", *etc.*) from given spatial layouts and leverage them to learn the relationships between objects, which would be more difficult for a model to distill from pixel-level layouts.

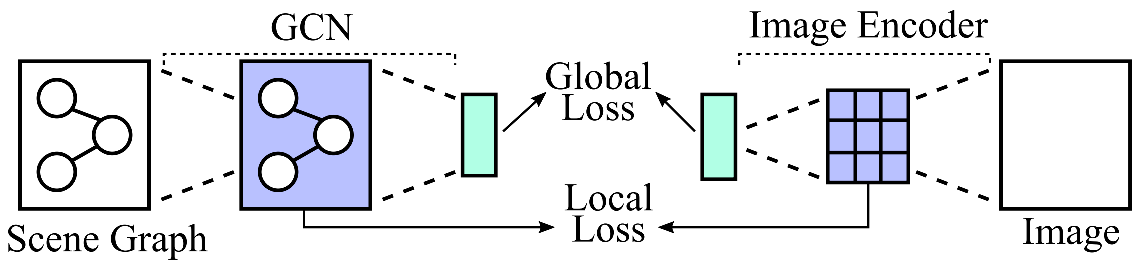


Fig. 6.3. The SGSM module. The SGSM module computes similarity between the scene-graph and the generated image, providing fine-grained matching-based supervision between the positional scene-graph and the generated image.

There has been strong interest in image and caption similarity modules for retrieval [Fang et al., 2015, Huang et al., 2013] and for text-to-image generation, most recently with the DAMSM model proposed in Xu et al. [2018a, Attngan]. Despite similar interest in scene graph to image retrieval [Johnson et al., 2015, Quinn et al., 2018], and the large improvements in text-to-image synthesis resulting from the DAMSM [Xu et al., 2018a, Li et al., 2019c], our approach is the first to use a scene graph to image retrieval module when training a generative model.

6.3. Proposed Method

6.3.1. Scene-Graph Similarity Module

We introduce the Scene Graph Similarity Module (SGSM) as a means of increasing the *layout-fidelity* of our generated images. This multi-modal module, described summarily in Fig. 6.3, takes as input an image and a scene-graph (nodes corresponding to objects, and edges corresponding to spatial relations). We extract *local visual features* $\boldsymbol{v}_i$ from the *mixed_6e* layer in an Inception-V3 network [Szegedy et al., 2016] pre-trained on the ImageNet dataset. We extract *global visual features* $\boldsymbol{v}^G$ from the final pooling layer. We encode the graph using a Graph Convolutional Network [GCN, Goller and Kuchler, 1996] to obtain *local graph features* $\boldsymbol{g}_j$ and apply a set of graph convolutions followed by a graph pooling operation to obtain *global graph features* $\boldsymbol{g}^G$. Note that each local and global feature is extracted and linearly projected to a common semantic space. In what follows, cos is the cosine similarity, and the γ_ks are normalization constants. We use L/G when the local and global terms are interchangeable. We use the modified dot-product attention mechanism of Xu et al. [2018a] to compute the *visually attended local graph embeddings* $\tilde{\boldsymbol{g}}_j$:

$$\boldsymbol{s}_{ij} = \gamma_1 \frac{\exp\left(\boldsymbol{g}_j{}^T \boldsymbol{v}_i\right)}{\sum_{i'} \exp\left(\boldsymbol{g}_j{}^T \boldsymbol{v}_{i'}\right)}, \qquad \tilde{\boldsymbol{g}}_j = \frac{\sum_i \exp(\boldsymbol{s}_{ij}) \boldsymbol{v}_i}{\sum_i \exp(\boldsymbol{s}_{ij})}. \tag{6.3.1}$$

Then we can define a *local similarity metric* between the source graph embedding $\boldsymbol{g}_j$ and the visually aware local embedding $\tilde{\boldsymbol{g}}_j$ similar to Xu et al. [2018a]. Intuitively, the similarity will be strong when the source graph embedding is close to the visually aware embedding. This local similarity will encourage different patches of the image to match the objects expected from the scene graph. The *global similarity metric* is classically the cosine distance between embeddings:

$$\begin{cases} \mathrm{Sim}^L(S, I') = \log\left(\sum_j \exp\left(\gamma_2 \cdot \cos(\tilde{\boldsymbol{g}}_j, \boldsymbol{g}_j)\right)\right)^{\frac{1}{\gamma_2}} & (6.3.2) \\ \mathrm{Sim}^G(S, I') = \cos\left(\boldsymbol{v}^G, \boldsymbol{g}^G\right) & (6.3.3) \end{cases}$$

Finally we can define a global and local probability model in a similar way to e.g. Huang et al. [2013]:

$$\mathbb{P}^{L/G}(S, I') \propto \exp\left(\gamma_3 \cdot \mathrm{Sim}^{L/G}(S, I')\right). \tag{6.3.4}$$

Normalizing over the images or scenes in the batch B (negative examples are selected by mis-matching the image and scene-graph pairs in the batch) leads to e.g.: $\mathbb{P}^{L/G}(S|I) = \frac{\mathbb{P}^{L/G}(S,I)}{\sum_{I' \in B} \mathbb{P}^{L/G}(S,I')}$. We define the loss terms as the log posterior probability of matching an image I and *the corresponding* scene graph (and vice-versa):

$$\begin{cases} \mathcal{L}_{L/G} = -\log \mathbb{P}_{L/G}(S|I) - \log \mathbb{P}_{L/G}(I|S) & (6.3.5) \\ \mathcal{L}_{\mathrm{SGSM}} = \mathcal{L}_L + \mathcal{L}_G & (6.3.6) \end{cases}$$

Empirically, the SGSM resulted in large gains in performance as shown in Table 6.4. Our hypothesis is that the scene graph, in a similar way to a caption, provides easier, simpler to distil relational information contained in the layout, which results in stronger performance compared to generation using just the layout. Architectural details of the SGSM and related data processing are described in the Appendix.

6.3.2. Instance-Aware Conditioning

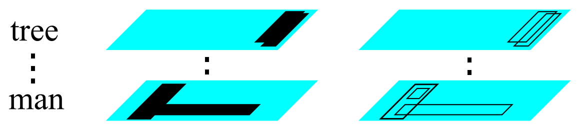


Fig. 6.4. Blue indicates 0 and black indicates 1. (Left) The per-class mask constructed from the layout by many previous methods makes it impossible to distinguish unique object instances in several cases. (Right) Our mask consists of instance boundaries making it easier for the model to distinguish unique object instances using no extra information than already contained in the layout.

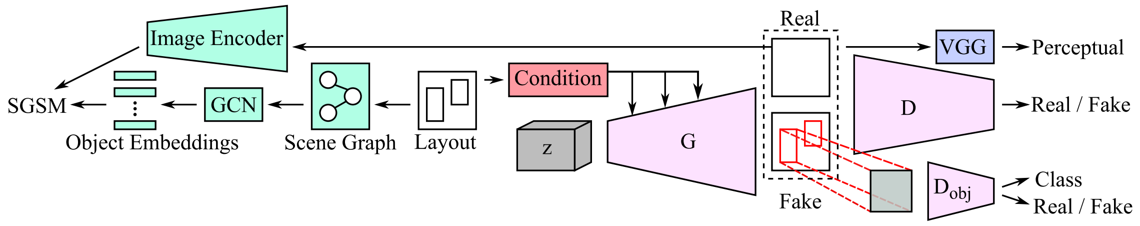


Fig. 6.5. Overview of our OC-GAN model. The GCN and Image Encoder modules are trained separately and then frozen. The condition for the Generator's normalization and the Scene Graph encoding the spatial relationships between objects are both derived from the input layout. The SGSM and the instance-aware normalization lead our model to generate images with higher layout-fidelity and sharper, distinct objects. The 'Condition' box corresponds to the three inputs listed in the subsection on the instance-aware conditioning.

As in Park et al. [2019], Sun and Wu [2019], the parameters γ, β of our batch-normalization layers are *conditional* and determined on a per-pixel level [as opposed to classical conditional batch-normalization, De Vries et al., 2017]. In our case, these parameters are determined by three concatenated inputs: *masked object embeddings, bounding-box layouts and bounding-box instance boundaries.* Masked object embeddings [Ma et al., 2018, Sun and Wu, 2019] and bounding-box layouts (using 1-hot embeddings) have been previously used in the layout to image setting. A shortcoming of these conditioning inputs is that they do not provide any way to distinguish between objects of the same class if their bounding boxes overlap. We use the layout's bounding-box boundaries, shown in Figure 6.4, as additional conditioning information. The addition of the bounding-box instance boundaries helps the model in mapping overlapping conditioning semantic masks to separate object instances, the absence of which led previous state-of-the-art methods to generate merged outputs as shown in the donut example in Fig. 6.1. Importantly, the instance boundaries do not add any additional information compared to the baselines: (1) they are bounding-box rather than fine-grained boundaries, and (2) instance information is already available to other models (Layout2Im and LostGAN have object-specific codes as an example). Rather, adding these boundaries acts like a prior encouraging our model to focus on generating distinct objects.

6.3.3. Architecture

Our OC-GAN model is based on the GAN framework. The generator module generates the images conditioned on the ground-truth layout. The discriminator predicts whether the input image is generated or real. The discriminator has an additional component which has to discriminate objects present in the input image patches corresponding to the ground-truth layout object bounding boxes. We present an overview of the model in Fig. 6.5 and describe the components below. Additional details are in the Appendix.

6.3.3.1. Generator. As a means of disentangling our model's performance from a specific choice of generator architecture, we used a classical residual [He et al., 2016] architecture consisting of 4 layers for 64×64 inputs, and 5 layers for 128×128 inputs, as used recently in Park et al. [2019], Sun and Wu [2019], Wang et al. [2018b]. The residual decoder G takes as input image-level noise. As described in the previous section, we further condition the generation by making the normalization parameters of the batch-norm layers of the decoder dependent on the layout and instance boundaries.

6.3.3.2. Discriminator. We use two different types of discriminators, an object discriminator, and a set of patch-wise discriminators. The object discriminator D_{obj} takes as input crops of the objects (as identified by their input bounding boxes) in real and fake images resized to size 32×32 and is trained using the Auxiliary-Classifier [AC, Odena et al., 2017] framework, resulting in a classification and an adversarial loss. Next, two *patch-wise discriminators* D_1^p, D_2^p output estimates of whether a given patch is consistent with the input layout. We apply them to the original image and the same image down-sampled by a factor of 2 (no weight sharing) in a similar fashion to Park et al. [2019], Wang et al. [2018b].

6.3.4. Loss Functions

In the following, x denotes a real image, l a layout, and z noise. We also denote objects with o and their labels y_o.
Perceptual loss. Adding a perceptual loss [Dosovitskiy and Brox, 2016, Gatys et al., 2016, Johnson et al., 2016] to our model improved results slightly. We extract features using a VGG19 network [Simonyan and Zisserman, 2015]. The loss has expression: $\mathcal{L}_P = \mathbb{E}_{x,l,z} \sum_{i=1}^{N} \frac{1}{D_i} ||F^{(i)}(x) - F^{(i)}(G(l,z))||_1$ where $F^{(i)}$ extracts the output at the i-th layer of the VGG and D_i is the dimension of the flattened output at the i-th layer.
Generator and Discriminator losses. We train the generator and patch discriminators using the adversarial hinge loss [Lim and Ye, 2017]:

$$\mathcal{L}_G^{\text{GAN}} = -\mathbb{E}_{l,z}\Big[D_1^p(G(l,z),l) + D_2^p(G(l,z),l)\Big] \tag{6.3.7}$$

$$\begin{aligned}\mathcal{L}_{D^p} = \sum_{i=1}^{2}\Big\{ &-\mathbb{E}_{x,l}\Big[\min(0, -1 + D_i^p(x,l))\Big] \\ &- \mathbb{E}_{l,z}\Big[\min(0, -1 - D_i^p(G(l,z),l)\Big]\Big\}.\end{aligned} \tag{6.3.8}$$

The object discriminator follows the AC-GAN framework, leading to $\mathcal{L}_G^{AC}$ and $\mathcal{L}_{D_{obj}}^{AC}$. The final expression is:

$$\mathcal{L}_G = \mathcal{L}_G^{\text{GAN}} + \lambda_P \mathcal{L}_{\text{P}} + \lambda_{\text{SGSM}} \mathcal{L}_{\text{SGSM}} + \lambda_{\text{AC}} \mathcal{L}_G^{AC} \tag{6.3.9}$$

$$\mathcal{L}_D = \mathcal{L}_{D^p} + \lambda_o \mathcal{L}_{D_{obj}}^{AC}. \tag{6.3.10}$$

We fix $\lambda_P = 2, \lambda_o = 1, \lambda_{\text{SGSM}} = 1, \lambda_{\text{AC}} = 1$ in our experiments.

6.4. Experiments

6.4.1. Datasets

We run experiments on the COCO-Stuff [Caesar et al., 2018] and Visual Genome (VG) [Krishna et al., 2017] datasets which have been the popular choice for layout- and scene-to-image tasks as they provide diverse and high-quality annotations. The former is an expansion of the Microsoft Common Objects in Context (MS-COCO) dataset [Lin et al., 2014]. We apply the same pre-processing and use the same splits as Johnson et al. [2018], Zhao et al. [2019a]. The summary statistics of the two datasets are presented in the appendix, Table B.1.

Our OC-GAN model takes three different inputs:

- The spatial layout *i.e.* object bounding boxes and object class annotations.
- Instance boundary maps computed directly from the layout. While they appear redundant once the bounding boxes are provided, they aid the model in better differentiating different objects especially different instances of the same object class.
- Scene-graphs. These are constructed from the objects and spatial relations inferred from the bounding box positions following the setup in Johnson et al. [2018]. While VG provides more complex scene graphs, we restricted ourselves to spatial relations only for compatibility between the two datasets.

6.4.2. Implementation and Training Details

Our code is written in PyTorch [Paszke et al., 2019]. We apply Spectral Normalization [Miyato et al., 2018] to all the layers in both the generator and discriminator networks. Each experiment ran on 4 V100 GPUs in parallel. We use synchronized BatchNorm (all summary statistics are shared across GPUs).

We used the Adam [Kingma and Ba, 2015] solver, with $\beta_1 = 0.5$, $\beta_2 = 0.999$. The global learning rate for both generator and discriminators is 0.0001. 128×128 models and above were trained for up to 300 000 iterations, 64×64 models were trained for up to 200 000 iterations (early stopping on a validation set). The SGSM module is trained separately for 200 epochs. It is then fixed, and the rest of the model is trained.

6.4.3. Baselines

We consider all recent methods that allow layout-to-image generation (Layout2Im [Zhao et al., 2019a], LostGAN [Sun and Wu, 2019], LostGAN-v2 [Sun and Wu, 2020]). We report results for scene-graph-to-image methods (SG2Im [Johnson et al., 2018], SOARISG [Ashual and Wolf, 2019]) evaluated with *ground-truth layouts* for a fair comparison. Finally, methods originally designed for generation from pixel-level semantic segmentation maps (SPADE [Park et al., 2019] and Pix2PixHD [Wang et al., 2018b]) are also considered as they can be readily adapted to this new context.

6.4.4. Evaluation

Evaluation of GANs is a complex issue, and the subject of a vast body of literature. In this paper, we focus on three existing evaluation metrics: Inception Score (IS) [Salimans et al., 2016], Fréchet Inception Distance (FID) [Heusel et al., 2017] and Classification Accuracy (CA). For the CA score, a ResNet-101 [He et al., 2016] network is trained on object crops obtained from the real images of the train set of the corresponding dataset, as suggested by Ashual and Wolf [2019]. The FID metric computes the 2-Wasserstein distance between the real and generated distributions, and therefore serves as an efficient proxy for the diversity and visual quality of the generated samples. While the FID metric focuses on the whole image, the CA metric allows us to demonstrate the ability of our model to generate realistic-looking objects within a scene. Finally, we include the Inception Score as a legacy metric.

6.4.4.1. Our proposed metric: SceneFID. We note that there exist many concerns in the literature regarding the use of metrics that are not designed or adapted to the task at hand. The Inception Score has been criticised [Barratt and Sharma, 2018], notably due to issues caused by the mismatch between the domain it was trained on (the ImageNet dataset comprising single objects of interest) and the domain of VG and COCO-Stuff images (comprising multiple objects in complex scenes), making it a potentially poor metric to evaluate generative ability of models in our setting. While the FID metric was introduced in response to Inception Score's criticisms, and was shown empirically to alleviate some of the concerns with it [Im et al., 2018, Xu et al., 2018b, Lucic et al., 2018], it still suffers from problems in the layout-to-image setting. In particular, the single manifold assumption behind FID was found in Liu et al. [2018] to be problematic in a multi-class setting. This is *a fortiori* the case in a multi-object setting as in VG and COCO. While Liu et al. [2018] introduce a class-aware version of FID, this is not applicable to our setting. We introduce the *SceneFID* metric, where we compute the FID on the crops of all objects, resized to same size (224×224), instead of on the whole image. Thus, the SceneFID metric measures FID in the single manifold assumption it was designed for and extends it to the multi-object setting.

In addition to the above quantitative metrics, we also perform qualitative assessment of the model, notably by considering the effect of modifying the input layout on the output image.

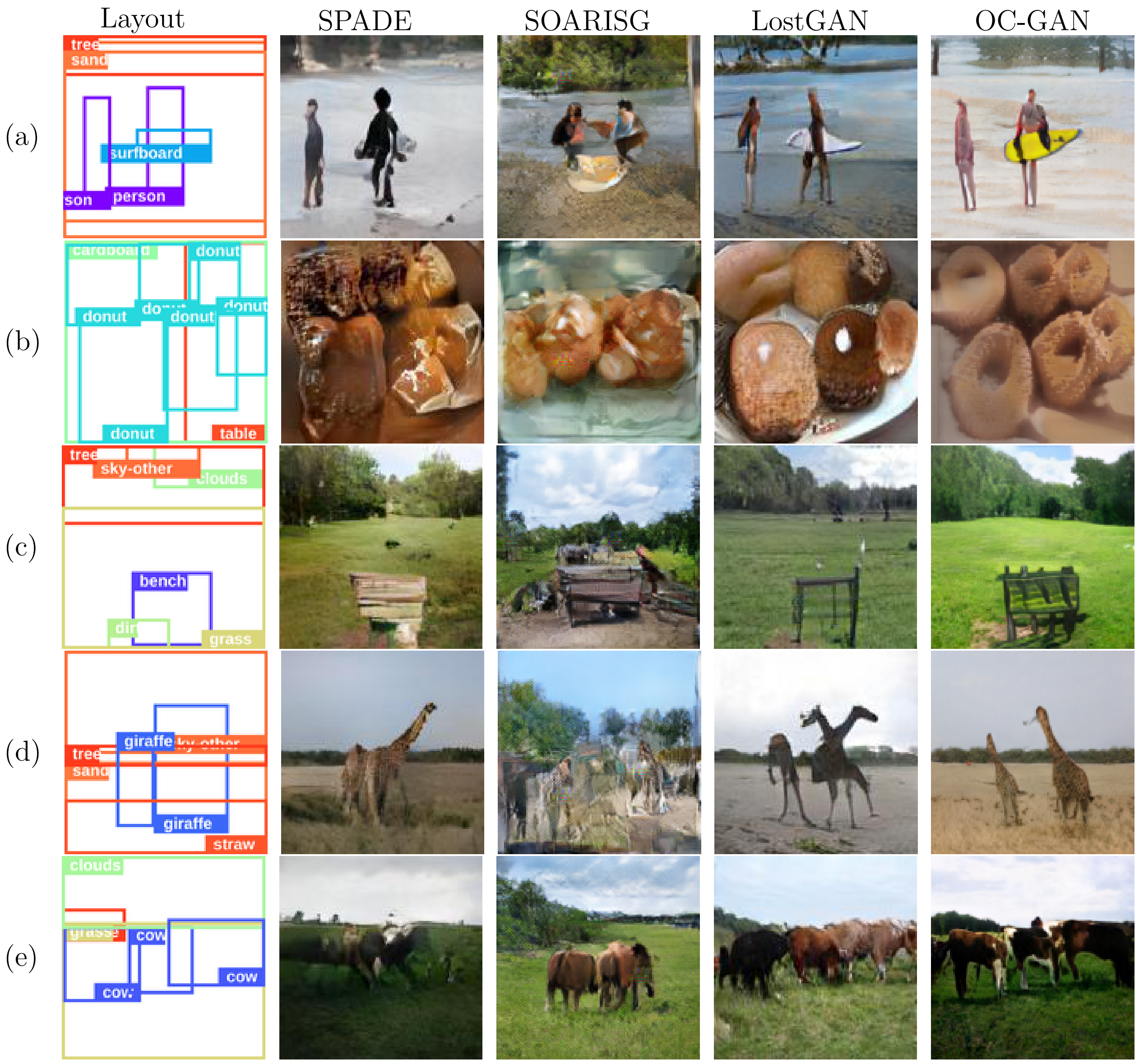


Fig. 6.6. 128 × 128 COCO-Stuff test set images, taken from our method (OC-GAN), and multiple competitive baselines. Note the overall improved visual quality of our samples. In addition, for (d, e) many baselines introduce spurious objects, and for (b, d, e) spatially close objects are poorly defined and sometimes fused for the baselines.

6.4.5. Quantitative Results

We report comparisons of our model's performance to the set of all recent state-of-the-art methods. Where applicable and possible, we use metric values reported by the authors of

	Methods	Inception Score ↑		FID ↓		CA ↑	
		COCO	VG	COCO	VG	COCO	VG
Real Images	64 × 64	16.3 ± 0.4	13.9 ± 0.5	0	0	54.48	49.57
	128 × 128	22.3 ± 0.5	20.5 ± 1.5	0	0	60.71	56.25
	256 × 256	28.10 ± 0.5	28.6 ± 1.2	0	0	63.04	60.40
64 × 64	SG2Im [Johnson et al., 2018]†	7.3 ± 0.1	6.3 ± 0.2	67.96	74.61	30.04	40.29
	Pix2PixHD [Wang et al., 2018b]	7.2 ± 0.2	6.6 ± 0.3	59.95	47.71	20.82	16.98
	SPADE [Park et al., 2019]	8.5 ± 0.3	7.3 ± 0.1	43.31	35.74	31.61	23.81
	Layout2Im [Zhao et al., 2019a]†	9.1 ± 0.1	8.1 ± 0.1	38.14	31.25	50.84	48.09
	SOARISG [Ashual and Wolf, 2019]∗ †	10.3 ± 0.1	N/A	48.7	N/A	46.1	N/A
	OC-GAN (ours)	**10.5 ± 0.3**	**8.9 ± 0.3**	**33.1**	**22.61**	**56.88**	**57.73**
64 × 64 with flips	LostGAN [Sun and Wu, 2019] (flips) †	9.8 ± 0.2	8.7 ± 0.4	34.31	34.75	37.15	27.1
	OC-GAN (ours)	**10.8 ± 0.5**	**9.3 ± 0.2**	**29.57**	**20.27**	**60.39**	**60.79**
128 × 128	Pix2PixHD [Wang et al., 2018b]	10.4 ± 0.3	9.8 ± 0.3	62	46.55	26.67	25.03
	SPADE [Park et al., 2019]	13.1 ± 0.5	11.3 ± 0.4	40.04	33.29	41.74	34.11
	Layout2Im [Zhao et al., 2019a] ◇	12.0 ± 0.4	10.1 ± 0.3	43.21	38.21	49.06	51.13
	SOARISG [Ashual and Wolf, 2019] †∗	12.5 ± 0.3	N/A	59.5	N/A	44.6	N/A
	OC-GAN (ours)	**14.0 ± 0.2**	**11.9 ± 0.5**	**36.04**	**28.91**	**60.32**	**58.03**
128 × 128 with flips	LostGAN [Sun and Wu, 2019] †	13.8 ± 0.4	11.1 ± 0.6	29.65	29.36	41.38	28.76
	LostGAN-V2 [Sun and Wu, 2020] †	14.2 ± 0.4	10.71 ± 0.27	**24.76**	29.00	43.27	35.17
	OC-GAN (ours)	**14.6 ± 0.4**	**12.3 ± 0.4**	36.31	**28.26**	**59.44**	**59.40**
256 × 256	SOARISG [Ashual and Wolf, 2019] †∗	15.2 ± 0.1	N/A	65.95	N/A	45.3	N/A
	OC-GAN (ours)	**17.0 ± 0.1**	14.4 ± 0.6	**45.96**	39.07	**53.47**	57.89
256 × 256 with flips	LostGAN-V2 [Sun and Wu, 2020] †	**18.0 ± 0.5**	14.1 ± 0.4	42.55	47.62	54.40	53.02
	OC-GAN (ours)	17.8 ± 0.2	**14.7 ± 0.2**	**41.65**	**40.85**	**57.16**	**53.28**

Table 6.1. Performance on 64, 128 and 256 dimension images. All models use ground-truth layouts. We use † to denote results taken from the original paper. ∗ denotes a model that uses pixel-level semantic segmentation during training. ◇ denotes models for which the openly available source code was not adapted to generation at a specific image size. We altered the code to allow this and ran a hyperparameter search on the new models.

the papers. SOARISG [Ashual and Wolf, 2019] depends on semantic segmentation maps being available, and therefore it was not feasible to include results on VG for this method. Some papers introduced additional data-augmentation, such as LostGAN [Sun and Wu, 2019] which introduced flips of the real images during training. Where applicable, we report results using the same experimental setup as the authors, and highlight it in the results table. For all models that do not report CA scores, we evaluate them using images generated with the pre-trained models provided by their authors.

Table 6.1 shows that our model consistently outperforms the baselines in terms of IS, FID and CAS, often significantly. We note that for some models, the CAS score is above that reported for ground-truth images. This is due to the fact that a sufficiently capable generator will start to generate objects that are both realistic, and of the same distribution as the training distribution, rather than the test one.

On the proposed SceneFID metric, Table 6.2 shows that our method outperforms the others significantly. Thus, our model is significantly better at generating realistic objects compared to the baselines. Note that the LostGAN model obtains better FID compared to our

	SceneFID ↓	
Methods	COCO	VG
Pix2PixHD [Wang et al., 2018b]	42.92	42.98
SPADE [Park et al., 2019]	23.44	16.72
Layout2Im [Zhao et al., 2019a]	22.76	12.56
SOARISG [Ashual and Wolf, 2019]$*$	33.46	N/A
LostGAN [Sun and Wu, 2019] (flips)	20.03	13.17
OC-GAN (ours w/ flips)	**16.76**	**9.63**

Table 6.2. SceneFID scores on object crops resized to size 224 × 224, extracted from the 128 × 128 outputs of the different models, for both datasets. All models use ground-truth layouts. $*$ denotes a model that uses pixel-level semantic segmentation during training. SOARISG cannot be trained on VG due to the absence of pixel-level semantic segmentations.

model exceptionally on 128 × 128 COCO-Stuff images but our OC-GAN model outperforms it on the SceneFID metric which is more appropriate in this multi-class setting.

6.4.6. Qualitative Results

We compare and analyse image samples generated by our method and competitive baselines in Fig. 6.6. In addition to generating higher quality images, our OC-GAN model does not introduce spurious objects (objects not specified in the layout but present in the generated image). This can be attributed to the SGSM module which, by virtue of the retrieval task and the scene-graph being a higher-level abstraction than pixels, aids the model in learning a better mapping from the spatial layout to the generated image. Our model also keeps object instances identifiable even when bounding boxes of objects of the same class overlap slightly or are in close proximity.

To further validate the previous observations, in Fig. 6.1, we consider the effect of generating from artificial layouts of gradually converging donuts, to tease out the model's ability to correctly generate separable object instances. Our model generates distinct donuts even when occluded, whereas the other models generate realistic donuts when the bounding boxes are far apart, but fail to do so when they overlap.

We also conducted a user study to evaluate the model's layout-fidelity. 10 users were shown 100 layouts from the test sets of both datasets, with the corresponding images generated by our OC-GAN, LostGAN, and for COCO-Stuff, SOARISG, shuffled in a random order. For each layout, users were asked to select the model which generates the best corresponding image. The results from this study are in Table 6.3 and demonstrate that our model has higher layout-fidelity than previous SOTA methods.

In Table 6.4, we present an ablation study performed by removing certain components of our model. The effect of adding another patch discriminator is measurable, both in terms of FID and CA. Removing the patch discriminator significantly lowers FID (the model has no

Dataset	SOARISG	LostGAN	Ours
COCO-Stuff	16.8%	36.8%	**46.4%**
VG	N/R	31.4%	**68.6%**

Table 6.3. User study results. 10 computer-science professionals were shown 100 COCO-Stuff and 100 VG test set layouts and corresponding images generated by various models, shuffled randomly. Users were asked to select the highest layout-fidelity image for each layout at 128×128 resolution. SOARISG is marked marked non-rated (N/R), as it cannot be trained on VG.

	FID ↓	CA ↑
Full	**29.57**	60.27
Single patchD	30.54	59.86
No patchD	33.85	**62.48**
No objectD	31.62	48.03
No bounding-box instance boundaries	30.12	59.54
No SGSM	34.32	52.57
No objectD, no SGSM	33.15	41.50
No perceptual loss	31.14	57.22
No perceptual loss, no SGSM	36.54	47.94

Table 6.4. Quantitative comparison of different ablated versions of our model on the COCO-Stuff dataset (64×64 images). These results highlight the importance of the SGSM (and its positive interaction with the perceptual loss) in the bottom row block, as well as the impact of removing some of the discriminators (middle row block).

more supervision in terms of matching the distribution of the real full images. This actually improves the CA, as the generator will use more capacity to focus on generating realistic objects.

We also find that removing either the object discriminator or the SGSM results in a significant drop in performance. This does not however prevent the model from generating realistic objects (the CA score remains above some of the baselines), meaning that the roles of the two components are to some extent complementary. As soon as both are removed, the CA score drops sharply.

Removing the perceptual loss has little effect in itself, but it greatly helps the SGSM when present. Removing the SGSM altogether strongly impairs results, highlighting its importance. Finally, removing the bounding-box instance boundaries has a modest impact on both metrics, but a large qualitative impact with more clearly defined objects.

6.5. Conclusion

We observed that current state-of-the-art layout-to-image generation methods exhibit low layout-fidelity and tend to generate low quality objects especially in cases of occlusion. We proposed a novel Scene-Graph Similarity Module that mitigated the layout-fidelity issues aided by an improved understanding of spatial relationships derived from the layout. We also proposed to condition the generator's normalization layers on instance boundaries which led to sharper, more distinct objects compared to other approaches. The addition of the proposed components to the image generation pipeline led to our model outperforming previous state-of-the-art approaches on a variety of quantitative metrics. A comprehensive ablation study was performed to analyse the contribution of the proposed and existing components of the model. Human users also rated our approach higher on generating better-suited images for the layout over existing methods.

Evaluation metrics for GAN popularized in the single-object-class setting have been criticized as inappropriate in the multi-class setting in literature. Our proposed SceneFID metric addresses those concerns and presents a useful metric for the image generation community which will increasingly deal with multi-class settings in the future. Our proposed OC-GAN model also showed a large improvement over existing approaches on the SceneFID evaluation criteria which further highlights the impact of our contributions.

Chapter 7

Prologue to the third paper

7.1. Article details

Cross-Modal Information Maximization for Medical Imaging: CMIM [Sylvain et al., 2020d]. Tristan Sylvain, Francis Dutil, Tess Berthier, Lisa Di Jorio, Margaux Luck, Devon Hjelm, Yoshua Bengio. ICASSP 2021[1].
Personal contributions This paper is the result of my internship at Imagia Cybernetics, a Montreal-based company focusing on medical imaging. As lead author, I developed most of the ideas, conducted most of the experiments, and wrote most of the code and papers. Francis and Tess helped implement some of the baselines. Lisa, Margaux, Devon and Yoshua contributed valuable advice throughout the life of the project and helped with the writing.

7.2. Context

At the time I started to come up with some of the ideas behind this paper, mutual-information based representation learning was starting to enter the research mainstream [DIM Hjelm et al., 2018]. As I had contributed to this body of literature myself (cf. the first article), I started thinking about whether such techniques could help improve multi-modal representation learning as well. In the case of multi-modal data, the different views of the input data are more explicit (compared to DIM and other related works) as they directly correspond to different data modalities. However, the domain gap between views also increases significantly. As a result, specific adaptations are necessary, as outlined in the main paper.

7.3. Contributions

The contributions of this paper are two-fold. First, we introduce a means of using recent advances in mutual-information estimation [Belghazi et al., 2018b] and local representations of the data to improve multi-modal representation learning. Second, we evaluate these

[1]© 2021 IEEE. Reprinted, with permission, from [Sylvain et al., 2020d]

components on a diverse set of tasks, and lay the ground for their application in other settings.

7.4. Aftermath

This paper has been accepted to ICASSP 2021. Work on this paper was also a source of inspiration for other related papers I contributed to. Recently, Fedorov et al. [2020a] was accepted to the 2021 IEEE International Symposium on Biomedical Imaging. Another paper, Fedorov et al. [2020b] is still under submission. I hope that this paper and the other works will help showcase an interesting research direction.

Chapter 8

Cross-Modal Information Maximization for Medical Imaging: CMIM

8.1. Introduction

The practice of keeping hospital patient data inside information silos restricts the range of possible data analyses that could improve patient care. The richness of hospital databases that manifests itself in their increasing volumes and modalities/sources could offer unique opportunities for data analysis improvement through the acquisition and use of multiple views of the same patient coming from, for example, different medical imaging exams (CT scans, MRI, PET, Ultrasound) and associated radiology reports.

However, patient data may contain a large variety of modalities, many of which may be missing for a specific patient due to differing clinical procedures between specialists and hospitals. This occurs when we are considering a specific modality in medical data sets that tends to only be present in a few data points (for instance due to cost or rarity of the medical condition requiring it). In addition, for some modalities such as radiology reports and imagery, hospital-specific guidelines can lead to non-standardized annotations, image acquisition artifacts, etc. This leads to medical data sets that are often too sample-poor to fully take advantage of deep learning techniques. A solution to this problem would be to build a deep learning model that will take advantage of the multiple modalities available at training time by learning single-modal representations that minimize the information loss when compared to multi-modal representations of the same input. This would encourage robustness to modality dropping (i.e., the model must be able to perform well in the absence of one or more modalities) at testing time. A way to do that is to apply recent advances in mutual information maximization [Hjelm et al., 2018, Belghazi et al., 2018a].

In this paper, we idealize this problem setting by considering extreme modality dropping at testing time (i.e., multiple modalities at train time, one at test time) to improve classification of chest x-rays using the open-source Open-I data set and the segmentation of different

MRI modalities using the publicly available BRATS-2015 data set. Our contributions are as follows:

- We reformulate cross-modal training as a mutual information maximization problem, and propose an innovative framework harnessing recent advances in mutual-information estimation to address it.
- By design, we are able to exploit learned representations for every modality and exploit them at test time even when one modality is missing.
- Our proposed approach outperforms state-of-the-art baselines on two challenging tasks, image classification and semantic segmentation.

8.2. Related Work

8.2.1. Cross-modality training

Multi-modal data has been exploited in numerous medical tasks including: caption generation [Wang et al., 2018c] (text and images), lesion detection [Hadad et al., 2017] (mammogram and MRI), image classification [Zhang et al., 2019c] (image and knowledge graphs) and few-shot semantic segmentation [Zhao et al., 2019b]. While such systems yield performance improvements, there are few works on creating systems that while benefiting from additional training modalities are *robust* to modality dropping at test-time.

Solutions generally fall into three broad categories. In the first case, missing modalities are inferred at test-time via e.g. retraining a model with the missing modalities [Hofmann et al., 2008], synthesizing missing-modalities [van Tulder and de Bruijne, 2015], or bootstrapping from a classifier trained on the full set of features [Hor and Moradi, 2015]. The second approach maps modalities to a common subspace via e.g. an abstraction layer focusing on first-order statistics [Havaei et al., 2016] or adversarial methods [Saito et al., 2016]. The third, to which our method belongs, optimizes some similarity metric between different views/modalities of the data, by e.g. canonical correlation analysis [Hotelling, 1992, Andrew et al., 2013] or attention combined with shared tasks such as MDNet [Zhang et al., 2017c] and TieNet [Wang et al., 2018c].

8.2.2. Mutual information maximization

Mutual information (MI), despite being a useful quality to evaluate, is hard to estimate in practice for non-discrete representations. Mutual Information Neural Estimation [Belghazi et al., 2018a] introduces an estimator of mutual information via an auxiliary network. Deep InfoMax [Hjelm et al., 2018] and more recently AM-DIM [Bachman et al., 2019] apply this framework to representation tasks by maximizing mutual information between local and global representations of an input. ST-DIM [Anand et al., 2019] and CM-DIM [Sylvain et al.,

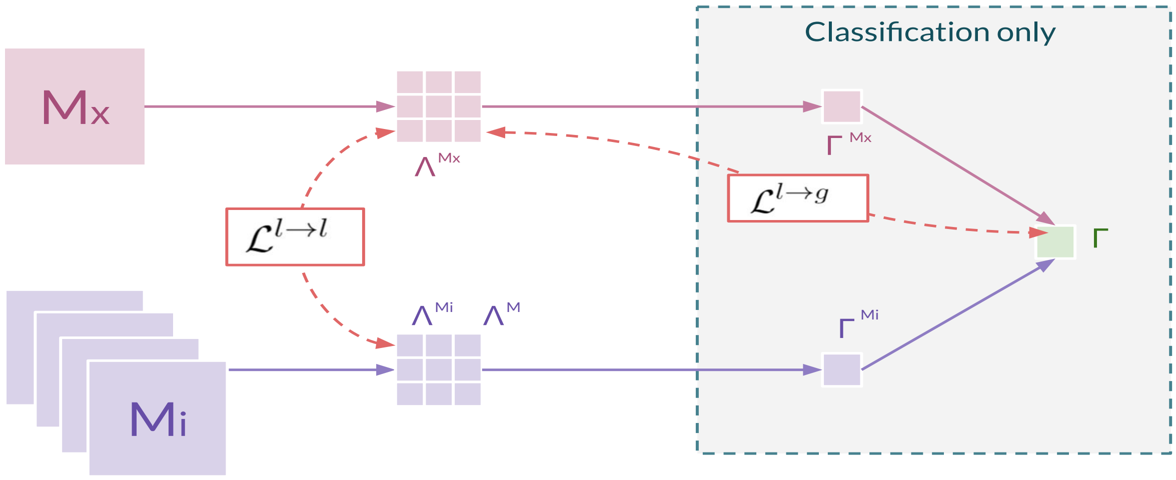


Fig. 8.1. Graphical representation of our method on multiple modalities. We train on a set $\mathbf{M} = \left\{\mathbf{M_X}, \mathbf{M_1}, \cdots, \mathbf{M_n}\right\}$ modalities at train time. At test-time, only $\mathbf{M_X}$ is provided. We map the input to both per-modality local features $\mathbf{\Lambda^{M_i}}$, per-modality global features $\mathbf{\Gamma^{M_i}}$ and a multi-modal global embedding $\mathbf{\Gamma}$ shared across modalities. The local-local and local-global losses correspond to the mutual information terms introduced in the proposed method section. We omit the global embeddings for the segmentation task. In addition to the losses shown, we also train with the task specific segmentation loss $\mathcal{L}^{\text{seg}}$ (pixel-wise categorical cross-entropy) and classification loss $\mathcal{L}^{\text{classif}}$ (categorical cross entropy) not shown in this figure.

2020a,c] apply this in turn to reinforcement learning and zero-shot learning respectively. Our work is the first to consider maximizing mutual information between representations of different modalities of a same input.

Previous works on applying mutual information to cross-modal learning usually constrain the architecture, such as the shared weights approach of Rastegar et al. [2016] or introduce other constraints whereas our approach is more general.

8.3. Proposed method

Our approach, represented in Figure 8.1, aims to improve supervised downstream performance in the setting where a subset of modalities present at train time are not present at test time[1]. We do this by maximizing mutual information between representations of different modalities of a given input. This will encourage each modality to retain as much discriminative information as possible.

[1]In practice we consider only one modality present at test-time

8.3.1. Mutual information maximization

Our work applies the mutual information neural estimator (MINE) introduced in Belghazi et al. [2018a]. Formally, the mutual information between two random variables X and Y is defined as the KL-divergence between the joint distribution and the product of the marginals, i.e., $\mathcal{D}_{KL}(\mathbb{P}_{XY}||\mathbb{P}_X \otimes \mathbb{P}_Y)$. MINE maximizes a lower bound on that quantity derived from the Donsker Varadhan formulation. In our case, we found, similarly to Hjelm et al. [2018], that performance was improved by considering instead the Jensen-Shanon estimator, leading to:

$$\hat{I}_\theta(X,Y) = \mathbb{E}_{\mathbb{P}_{XY}}[-\text{sp}(T_\theta(x,y))] - \mathbb{E}_{\mathbb{P}_X \otimes \mathbb{P}_Y}[\text{sp}(T_\theta(x,y)], \tag{8.3.1}$$

where $sp(z) = \log(1+e^z)$ and T_θ is a neural network with parameters θ.

8.3.2. Cross-modality mutual information

In this work, we are concerned with optimizing mutual information between representations of different modalities of a given input. We train on a set $\mathbf{M} = \left\{\mathbf{M_X}, \mathbf{M_1}, \cdots, \mathbf{M_n}\right\}$ of modalities at train time. At test-time, only $\mathbf{M_X}$ is provided. Each modality $\mathbf{M_i}$ can be mapped to local features $\mathbf{\Lambda}^{\mathbf{M_i}}$ (2D and 1D pre-pooling convolution maps for images, and text respectively), and global features (pooled convolution maps) $\mathbf{\Gamma}^{\mathbf{M_i}}$. Similarly, we can obtain representations $\mathbf{\Lambda}^{\mathbf{M}}$ and $\mathbf{\Gamma}^{\mathbf{M}}$ for all the input modalities. For more details, see the section on design choices. We can then define cross-modal local-local, local-global and global-global losses as respectively:

$$\mathcal{L}^{l\to l} = \frac{1}{N^2}\sum_{n,m}^{N} \hat{I}(\mathbf{\Lambda}^{\mathbf{M_i}}{}_n, \mathbf{\Lambda}^{\mathbf{M}}{}_m) \tag{8.3.2}$$

$$\mathcal{L}^{l\to g} = \frac{1}{N}\sum_{n} \hat{I}(\mathbf{\Lambda}^{\mathbf{M_i}}{}_n, \mathbf{\Gamma}^{\mathbf{M}}) \tag{8.3.3}$$

$$\mathcal{L}^{g\to g} = \hat{I}(\mathbf{\Gamma}^{\mathbf{M_i}}, \mathbf{\Gamma}^{\mathbf{M}}). \tag{8.3.4}$$

8.3.3. Design choices

We have presented a global framework that can tackle different cases. In what follows, we will apply it to *semantic segmentation* and *image classification*. We only optimize the local-local mutual information loss in the first case. For classification, we optimize two losses: local-local and local-global. This choice is motivated by empirical performance, and the argument that semantic segmentation tasks benefit less from global information.

Each model is in addition to the mutual information losses optimized with its task-specific loss, $\mathcal{L}^{\text{seg}}$ (pixel-wise categorical cross-entropy) and $\mathcal{L}^{\text{classif}}$ (categorical cross-entropy).

Method	Training		Testing		AUC
	I	T	I	T*	
ResNet	●	○	●	○	0.785
TieNet	●	●	●	●	0.741
CMIM	●	●	●	○	**0.793**

Table 8.1. Results on Open-i. The *Train phase* and *Test phase* columns indicates which modality were used, among the image **I**, the text **T**, and the generated text **T*** obtained from a captioning model (● denotes presence, ○ absence). Note that the true text modality **T** is never present at test time. As we can see, our model outperforms the baselines, and contrary to TieNet, is actually able to leverage the second modality during training.

The final training classification and segmentation losses are respectively:

$$\mathcal{L}^C = \lambda_{l\to g}\mathcal{L}^{l\to l} + \lambda_{l\to l}\mathcal{L}^{l\to l} + \lambda_C\mathcal{L}^{\text{classif}} \tag{8.3.5}$$

$$\mathcal{L}^S = \lambda_{l\to l}\mathcal{L}^{l\to l} + \lambda_S\mathcal{L}^{\text{seg}}, \tag{8.3.6}$$

$$\tag{8.3.7}$$

where the λ are hyper-parameters regulating the importance of the different losses during training.

We use different architectures for the two downstream classification and segmentation tasks. For the segmentation task, we consider 4 MR *modalities* (FLAIR, T1W, T1C, T2), which are encoded using a U-Net [Ronneberger et al., 2015]-type model, due to its use in past literature, and overall good performance in medical segmentation. It takes as input either one or multiple MRI sequences as distinct channels to compute the representations.

For the classification setting, two *modalities* are present: text and image. Text was encoded using 300-dimension Glove vectors [Pennington et al., 2014] trained on Wikipedia. We did not perform fine-tuning of the embedding as this negatively impacted performance. Image representations are obtained using a ResNet50 [He et al., 2016] encoder, and text representations using a residual CNN variant of [Zhang et al., 2017d]. The global embedding is a bilinear embedding of the two previous representations.

For T_θ, we used architectures similar to the "concat-and-convolve" architecture found in Hjelm et al. [2018] (see Figure 5 in Hjelm et al. [2018]).

8.4. Experiments

8.4.1. Experimental setup

For each task, we train using the full set of available modalities, and evaluate using a single modality. Such as setting occurs frequently in practice as per instance there might be a small overlap between the MRI modalities a model has been trained on and the set of

Test-time modalities F T_1 T_1c T_2	CMIM	*HeMIS	*Mean (baseline)	*MLP (baseline)
● ○ ○ ○	**23.37**	5.57	6.25	15.90
○ ● ○ ○	**14.15**	4.67	6.25	10.78
○ ○ ● ○	49.00	**49.93**	30.02	32.92
○ ○ ○ ●	**29.56**	20.31	6.25	18.62

Table 8.2. Dice similarity coefficient (DSC) results on the BRATS test sets (%) in the "enhancing" setting introduced in Havaei et al. [2016]. We consider the case where only one of the 4 modalities is present at test-time (● denotes presence, ○ absence). All 4 modalities are used at train-time. Note that both these conditions create a very challenging setting, explaining the overall low dice scores reported. * denotes results taken from Havaei et al. [2016]. Our approach outperforms HEMIS and the other baselines on this setting with the exception of the T_1c modality where our model is a close second. In particular, strong gains are observed for "weaker" modalities such as F and T_1

acceptable test-time modalities (due to the absence of some, or domain shifts due to device calibration making some modalities unusable)

8.4.2. Classification task

Open-I [Demner-Fushman et al., 2015] is a publicly available radiography dataset collected by Indiana University. It contains 7470 chest x-rays with 3955 radiology reports. We prepared the data using the same methodology as Wang et al. [2018c], i.e. keeping 14 categories of findings as the classes for the classification problem, and only considering frontal images with associated reports. As the orientations of the X-ray images are not specified, and in order to keep only the frontal views, we performed manual analysis of all images, also removing some that were heavily distorted. We re-balanced the dataset as the raw data had heavy class imbalance. We report Area under the Curve (AUC) for all methods.

8.4.3. Semantic segmentation task

BRATS-2015 [Menze et al., 2015, Bakas et al., 2017] is a brain MRI dataset containing 220 subjects with high grade tumors, and 54 subjects with low grade tumors. There are 4 MR *modalities* present (FLAIR, T1W, T1C, T2), alongside a voxel-level segmentation ground truth of 5 labels: *health*, *necrosis*, *edema*, *non-enhancing tumor* and *enhancing tumor*. As in the *enhancing* setting in Havaei et al. [2016], the target is a binary map corresponding to a 1-versus-rest segmentation on the *enhancing tumor* class.

8.4.4. Baselines

For the classification task, we compare our results to TieNet [Wang et al., 2018c], a state of the art method for multi-modal X-ray classification. We also benchmark against a ResNet50 [He et al., 2016] supervised on the image modality only. For the segmentation task, we compare with Hemis [Havaei et al., 2016], a state-of-the art approach on this dataset. We also considered the same baselines that Hemis suggested: missing modality completion by mean (Mean) and a multi-layer perceptron (MLP). To ensure conformity with their experimental setup, we used the same splits and code for data preparation.

8.4.5. Implementation and Training Details

Our code is written in PyTorch. Each experiment ran on V100 GPUs, using the Adam [Kingma and Ba, 2015] solver with a global learning rate of 0.0001. Models were trained up to convergence (early stopping on a validation set).

8.4.6. Results

When applying our model to the two experimental tasks, we had to make small adaptations. As local information tends to be more important in segmentation, we empirically found that local-global and global-global did not improve performance. This was not the case for classification, where we also used the local-global loss.

For the classification setting, as we can see in table 8.1, our method outperformed the other baselines. The discrepancy between TieNet's result and the other methods can be explained by the low number of training examples for the captioning model. Indeed, compared to TieNet's original paper where 100 000 reports are available, only a few thousands are present in Open-I. This causes the model to quickly disregard the image modality and to only focus on a few keywords to make its decision. However, CMIM alleviates this problem by forcing the representation of both modalities to have high mutual-information. This in turn encourages discriminative information to be present in both representations at inference time.

The results for the semantic segmentation task can be seen in table 8.2. Our model outperforms the other methods overall. Interestingly, our approach seems to perform better for the "weaker" modalities (F and T1 are known to perform poorly for enhanced tumor detection [Havaei et al., 2016]), where less information is present at test-time. This validates our hypothesis that CMIM is able to enhance discriminative features, even when the modality contains a low amount of signal.

8.5. Conclusion

In this paper, we introduced a method based on mutual information for cross-modal training. These kind of approaches can be particularly useful when some modalities are missing, as is often the case with real world data, in particular medical data. We validated our approach in two different tasks, each one implying different type of modality: text and image for a classification task, and different MRI modalities for a segmentation task. In both cases, results are promising. Interestingly, for MRI segmentation, our approach yields the best results when the modality present at test time conveys less discriminative information.

For future work, we plan on adapting the current model to be able to use multiple modalities at test time. Furthermore, we hope that our setup will pave the way for zero-shot learning approaches, where we would present the model with unseen modalities at test time.

Chapter 9

General conclusion

Despite many advances in computer vision in recent years, including attaining human-level performance on simple visual perception tasks such as classification and semantic segmentation, many more complex visual tasks remain elusive. These tasks are of a higher order, and often involve visual reasoning. Many of the algorithms developed to solve these complex tasks depend on good representations of images. In the articles presented in this thesis, we attempted to explain and improve the performance of representation learning algorithms by returning to first principles, in particular locality and compositionality, two principles with a proven and important effect on the quality of representation learning.

In chapter 4, we analyzed the extent to which different image representations were local and compositional, and found a strong relationship between these two quantities and downstream ZSL generalization performance. We use these results to propose CM-DIM, a novel representation learning algorithm based on preserving important local information, that outperforms other methods on our experimental setup. Additionally, the zero-shot learning setup we introduce, zero-shot learning from scratch [Sylvain et al., 2020c], is shown to be a good means of disentangling zero-shot concept learning from simpler transfer learning of previously trained visual representations. This in turn could be very useful for future research in the field.

In chapter 6, we showed that by introducing a scene-graph similarity module (that encouraged a model to retain local information, and also to understand a scene as a composition of different objects and their relations), we could improve layout-to-image performance as measured by a large set of metrics. This is both a validation of the importance of locality and compositionality on a different experimental setting, but also a demonstration that these principles can be used from the bottom up to design new deep learning models.

Finally, in chapter 8, we showed that using a mutual-information-based cross-modal loss term improved the generalization of learning algorithms in a series of experimental setups. This work was the basis for two follow up articles that analyze in more detail the benefits of cross-modal mutual information in a medical setting. Future work will now focus on actually deploying these innovations in a clinical context.

9.1. Improving our understanding of locality and compositionality

In the first presented article, our focus was on zero-shot image classification. However, the methods presented are much more general. Current work is ongoing on extending these insights to zero-shot semantic segmentation. In this context, the analysis will be geared towards incorporating more precise location information (given that we have access to semantic segmentation maps as opposed to images and their global labels). It would be also interesting to extend this analysis to a completely different domain of adaptation: natural language processing (NLP). There is extensive work on zero-shot NLP tasks, but no work currently exploiting the inherently strong compositional nature of natural text.

We can also consider a different research direction. Recently, there has been impressive progress on applying transformers [Vaswani et al., 2017] to vision problems, either with the perspective of joint text and image representations [Li et al., 2020, Oscar], or on image patches [Dosovitskiy et al., 2021, ViT], in both cases with impressive results. Rather than stop at the direct conclusion that this is simply yet another proof of the importance of both locality and compositionality, it would be essential to perform a related analysis to the one performed in our first presented work. The main question that remains to be answered is to what extent we can link different measures of both concepts to downstream performance. Can we encourage such models to be even more local, and does it improve performance?

9.2. Towards better modeling of complex scenes

The second work showcases different learning techniques and architectural components being applied to improve visual representation and generation of complex scenes. Scene representation is often also essential for the related problem of visual question answering. Currently, I am working on a project aiming to measure the effect of improving visual representations of complex scenes on the downstream performance of a question answering model. This approach is designed so as to disentangle the reasoning part from the analysis: we rely on a black-box visual question answering model [Amizadeh et al., 2020] that we adapted to only take as input predictions of the presence of objects, relations and attributes. As a result, I am able to measure changes to downstream performance by changing the quality of the visual representations and resulting predicted probabilities. This analysis should answer many questions related to generalization in visual question answering, the most notable being: is a good visual representation all you really need to answer questions? This ties in to one of the major points of this thesis first mentioned in the introduction: visual representations are essential for generalization in many complex tasks. The end goal of this body of work is therefore to contribute to building a universal model of arbitrarily complex images, that could then be simply and efficiently applied to many visual tasks with minimal adaptation.

9.3. Final remarks

Overall, we have demonstrated the importance of locality and compositionality in representation learning, and in particular its strong impact on downstream performance for a large number of tasks. Additionally, we have highlighted a series of future directions in which these insights could be beneficial to the computer vision community in general.

References

Mohammad Havaei, Nicolas Guizard, Nicolas Chapados, and Yoshua Bengio. Hemis: Hetero-modal image segmentation. In *MICCAI*, pages 469–477. Springer, 2016.

Yikang Li, Tao Ma, Yeqi Bai, Nan Duan, Sining Wei, and Xiaogang Wang. Pastegan: A semi-parametric method to generate image from scene graph. In *NeurIPS*, pages 3948–3958. Curran Associates, Inc., 2019a.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

Seymour A Papert. The summer vision project, 1966.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016a.

Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. Zero-data learning of new tasks. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, AAAI'08, pages 646–651. AAAI Press, 2008. ISBN 978-1-57735-368-3. URL `http://dl.acm.org/citation.cfm?id=1620163.1620172`.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Wenhu Chen, Zhe Gan, Linjie Li, Yu Cheng, William Wang, and Jingjing Liu. Meta module network for compositional visual reasoning. *arXiv preprint arXiv:1910.03230*, 2019.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer* *vision and pattern recognition*, pages 779–788, 2016.

David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi Zhang, Haibin Lin, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020.

Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. *arXiv preprint arXiv:1909.11065*, 2019.

Hieu Pham, Qizhe Xie, Zihang Dai, and Quoc V Le. Meta pseudo labels. *arXiv preprint arXiv:2003.10580*, 2020.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Byungin Yoo, Tristan Sylvain, Yoshua Bengio, and Junmo Kim. Joint learning of generative translator and classifier for visually similar classes. *IEEE Access*, 8:219160–219173, 2020.

Johannes Lehner, Andreas Mitterecker, Thomas Adler, Markus Hofmarcher, Bernhard Nessler, and Sepp Hochreiter. Patch refinement–localized 3d object detection. *arXiv preprint arXiv:1910.04093*, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. *arXiv preprint arXiv:1802.05751*, 2018.

Hieu Minh Bui, Margaret Lech, Eva Cheng, Katrina Neville, and Ian S Burnett. Object recognition using deep convolutional features transformed by a recursive network structure. *IEEE Access*, 4:10059–10066, 2016.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. *Advances in neural information processing systems*, 29:289–297, 2016a.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019a.

Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 94(2):115, 1987.

Donald D Hoffman and Whitman A Richards. Parts of recognition. *Cognition*, 18(1-3):65–96, 1984.

Ran Tian, Naoaki Okazaki, and Kentaro Inui. Learning semantically and additively compositional distributional representations. *arXiv preprint arXiv:1606.02461*, 2016.

Irina Higgins, Nicolas Sonnerat, Loic Matthey, Arka Pal, Christopher P Burgess, Matko Bosnjak, Murray Shanahan, Matthew Botvinick, Demis Hassabis, and Alexander Lerchner. Scan: Learning hierarchical compositional visual concepts. *arXiv preprint arXiv:1707.03389*, 2017a.

Ferran Alet, Tomás Lozano-Pérez, and Leslie P Kaelbling. Modular meta-learning. *arXiv preprint arXiv:1806.10166*, 2018.

Pavel Tokmakov, Yu-Xiong Wang, and Martial Hebert. Learning compositional representations for few-shot recognition. *arXiv preprint arXiv:1812.09213*, 2018.

Tristan Sylvain, Linda Petrini, and Devon Hjelm. Locality and compositionality in zero-shot learning. In *ICLR*, 2020a.

Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *ICML*, 2018a.

Christoph Goller and Andreas Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 1, pages 347–352, 1996.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2951–2959. Curran Associates, Inc., 2012. URL `http://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf`.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011a.

Maxime Bucher, Tuan-Hung Vu, Matthieu Cord, and Patrick Pérez. Zero-shot semantic segmentation. *arXiv preprint arXiv:1906.00817*, 2019.

Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-shot object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 384–400, 2018.

Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning - A comprehensive evaluation of the good, the bad and the ugly. *CoRR*, abs/1707.00600, 2017. URL `http://arxiv.org/abs/1707.00600`.

Ruizhi Qiao, Lingqiao Liu, Chunhua Shen, and Anton Van Den Hengel. Less is more: zero-shot learning from online textual documents with noise suppression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2249–2257, 2016.

Xiaolong Wang, Yufei Ye, and Abhinav Gupta. Zero-shot recognition via semantic embeddings and knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6857–6866, 2018a.

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset, 2011b.

Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *Proceeding of the 25th Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *arXiv preprint arXiv:1606.04838*, 2016.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *ArXiv e-prints*, mar 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

M. Kachuee, S. Fazeli, and M. Sarrafzadeh. Ecg heartbeat classification: A deep transferable representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 443–444, 2018. doi: 10.1109/ICHI.2018.00092.

Margaux Luck, Tristan Sylvain, Héloïse Cardinal, Andrea Lodi, and Yoshua Bengio. Deep Learning for Patient-Specific Kidney Graft Survival Analysis. *arXiv:1705.10245*, 2017.

URL `http://arxiv.org/abs/1705.10245`.

Margaux Luck, Tristan Sylvain, Joseph Paul Cohen, Heloise Cardinal, Andrea Lodi, and Yoshua Bengio. Learning to rank for censored survival data. *arXiv preprint arXiv:1806.01984*, 2018.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017a.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017.

Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017b.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. In *Advances in Neural Information Processing Systems*, pages 14866–14876, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in*

Neural Information Processing Systems, pages 2672–2680, 2014a.

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

R. Devon. Cortex. `https://github.com/rdevon/cortex`, 2018.

Wenjia Xu, Yongqin Xian, Jiuniu Wang, Bernt Schiele, and Zeynep Akata. Attribute prototype network for zero-shot learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016b.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

Koby Crammer, Michael Kearns, and Jennifer Wortman. Learning from multiple sources. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 321–328. MIT Press, 2007. URL `http://papers.nips.cc/paper/2972-learning-from-multiple-sources.pdf`.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*, pages 137–144, 2007.

Fei-Fei Li, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018.

Yuval Atzmon and Gal Chechik. Adaptive confidence smoothing for generalized zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11671–11680, 2019.

Xinsheng Wang, Shanmin Pang, Jihua Zhu, Zhongyu Li, Zhiqiang Tian, and Yaochen Li. Visual space optimization for zero-shot learning. *arXiv preprint arXiv:1907.00330*, 2019.

Austin Stone, Huayan Wang, Michael Stark, Yi Liu, D Scott Phoenix, and Dileep George. Teaching compositionality to cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5058–5067, 2017.

Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2018.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956, 2017.

George Boole. *An investigation of the laws of thought, on which are founded mathematical theories of logic and probabilities*. New York, Dover, 1 edition, 1854.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*, 2019.

Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *arXiv preprint arXiv:2011.15091*, 2020.

Jacob Andreas. Measuring compositionality in representation learning. *arXiv preprint arXiv:1902.07181*, 2019.

MJ Cresswell. Formal philosophy, selected papers of richard montague. *Philosophia*, 6(1): 193–207, 1976.

Wieland Brendel and Matthias Bethge. Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In *International Conference on Learning Representations*, 2019a. URL `https://openreview.net/forum?id=SkfMWhAqYQ`.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016. URL `http://arxiv.org/abs/1609.02907`.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*, 2018.

Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *CoRR*, abs/1904.00760, 2019b. URL `http://arxiv.org/abs/1904.00760`.

Linghui Li, Sheng Tang, Lixi Deng, Yongdong Zhang, and Qi Tian. Image caption with global-local attention. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. In *Advances in Neural Information Processing Systems*, pages 1564–1574, 2018.

Ming Sun, Yuchen Yuan, Feng Zhou, and Errui Ding. Multi-attention multi-class constraint for fine-grained image recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 805–821, 2018.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018a.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. *CoRR*, abs/1505.05192, 2015. URL `http://arxiv.org/abs/1505.05192`.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *CoRR*, abs/1603.09246, 2016. URL `http://arxiv.org/abs/1603.09246`.

Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. What makes imagenet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016.

Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. Factors of transferability for a generic convnet representation. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1790–1802, 2015.

Hengduo Li, Bharat Singh, Mahyar Najibi, Zuxuan Wu, and Larry S. Davis. An analysis of pre-training on object detection, 2019b.

Yabin Zhang, Hui Tang, and Kui Jia. Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 233–248, 2018b.

Xiangteng He and Yuxin Peng. Fine-grained image classification via combining vision and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5994–6002, 2017.

Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4109–4118, 2018.

Gabriel Huang, Hugo Larochelle, and Simon Lacoste-Julien. Centroid networks for few-shot clustering and unsupervised few-shot classification. *arXiv preprint arXiv:1902.08605*, 2019.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. Evaluation of output embeddings for fine-grained image classification. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015. doi: 10.1109/cvpr.2015.7298911. URL `http://dx.doi.org/10.1109/CVPR.2015.7298911`.

Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. Synthesized classifiers for zero-shot learning. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016. doi: 10.1109/cvpr.2016.575. URL `http://dx.doi.org/10.1109/CVPR.2016.575`.

Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017. doi: 10.1109/cvpr.2017.473. URL `http://dx.doi.org/10.1109/CVPR.2017.473`.

Li Zhang, Tao Xiang, and Shaogang Gong. Learning a deep embedding model for zero-shot learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017a. doi: 10.1109/cvpr.2017.321. URL `http://dx.doi.org/10.1109/CVPR.2017.321`.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. doi: 10.1109/cvpr.2018.00131. URL `http://dx.doi.org/10.1109/cvpr.2018.00131`.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv e-prints*, art. arXiv:1312.6114, Dec 2013.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015. URL `http://arxiv.org/abs/1511.05644`.

Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. *CoRR*, abs/1708.07860, 2017. URL `http://arxiv.org/abs/1708.07860`.

Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R. Devon Hjelm. Mutual information neural estimation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 531–540, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018b. PMLR. URL `http://proceedings.mlr.press/v80/belghazi18a.html`.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Tristan Sylvain, Pengchuan Zhang, Yoshua Bengio, R Devon Hjelm, and Shikhar Sharma. Object-centric image generation from layouts. *arXiv preprint arXiv:2003.07449*, 2020b.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *CVPR*, 2018a.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680. Curran Associates, Inc., 2014b.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NIPS*, pages 5767–5777. Curran Associates, Inc., 2017b.

Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. In *ICLR*, 2018.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 7354–7363, 2019b.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *ICML*, volume 48 of *Proceedings of Machine Learning Research*, pages 1060–1069, 2016a.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017b.

Wenbo Li, Pengchuan Zhang, Lei Zhang, Qiuyuan Huang, Xiaodong He, Siwei Lyu, and Jianfeng Gao. Object-driven text-to-image synthesis via adversarial training. In *CVPR*, 2019c.

Guojun Yin, Bin Liu, Lu Sheng, Nenghai Yu, Xiaogang Wang, and Jing Shao. Semantics disentangling for text-to-image generation. In *CVPR*, 2019.

Shikhar Sharma, Dendi Suhubdy, Vincent Michalski, Samira Ebrahimi Kahou, and Yoshua Bengio. ChatPainter: Improving text to image generation using dialogue. In *ICLR Workshop*, 2018.

Alaaeldin El-Nouby, Shikhar Sharma, Hannes Schulz, Devon Hjelm, Layla El Asri, Samira Ebrahimi Kahou, Yoshua Bengio, and Graham W. Taylor. Tell, draw, and repeat: Generating and modifying images based on continual linguistic instruction. In *ICCV*, 2019.

Yitong Li, Zhe Gan, Yelong Shen, Jingjing Liu, Yu Cheng, Yuexin Wu, Lawrence Carin, David Carlson, and Jianfeng Gao. Storygan: A sequential conditional gan for story visualization. In *CVPR*, 2019d.

Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *CVPR*, 2018.

Gaurav Mittal, Shubham Agrawal, Anuva Agarwal, Sushant Mehta, and Tanya Marwah. Interactive image generation using scene graphs. In *ICLR: Deep Generative Models for Highly Structured Data Workshop*, 2019.

Oron Ashual and Lior Wolf. Specifying object attributes and relations in interactive scene generation. In *ICCV*, 2019.

Bo Zhao, Lili Meng, Weidong Yin, and Leonid Sigal. Image generation from layout. In *CVPR*, 2019a.

Wei Sun and Tianfu Wu. Image synthesis from reconfigurable layout and style. In *ICCV*, 2019.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017.

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018b.

Krishna Kumar Singh, Utkarsh Ojha, and Yong Jae Lee. Finegan: Unsupervised hierarchical disentanglement for fine-grained object generation and discovery. In *CVPR*, 2019.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, pages 2642–2651, 2017.

Liqian Ma, Xu Jia, Stamatios Georgoulis, Tinne Tuytelaars, and Luc Van Gool. Exemplar guided unsupervised image-to-image translation with semantic consistency. *arXiv preprint arXiv:1805.11145*, 2018.

Xiaojuan Qi, Qifeng Chen, Jiaya Jia, and Vladlen Koltun. Semi-parametric image synthesis. In *CVPR*, pages 8808–8816, 2018.

Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning. In *CVPR*, pages 10685–10694, 2019.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *ECCV*, pages 382–398, 2016.

Jiuxiang Gu, Handong Zhao, Zhe Lin, Sheng Li, Jianfei Cai, and Mingyang Ling. Scene graph generation with external knowledge and image reconstruction. In *CVPR*, pages 1969–1978, 2019.

Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *CVPR*, pages 3668–3678, 2015.

Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D Manning. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *Proceedings of the fourth workshop on vision and language*, pages 70–80, 2015.

Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *ECCV*, pages 852–869. Springer, 2016b.

Alejandro Newell and Jia Deng. Pixels to graphs by associative embedding. In *NIPS*, pages 2171–2180, 2017.

Damien Teney, Lingqiao Liu, and Anton van Den Hengel. Graph-structured representations for visual question answering. In *CVPR*, pages 1–9, 2017.

Tristan Sylvain, Linda Petrini, and R Devon Hjelm. Zero-shot learning from scratch (zfs): leveraging local compositional representations. *arXiv preprint arXiv:2010.13320*, 2020c.

Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From captions to visual concepts and back. In *CVPR*, pages 1473–1482, 2015.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338, 2013.

Max H Quinn, Erik Conser, Jordan M Witte, and Melanie Mitchell. Semantic image retrieval via active grounding of visual situations. In *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, pages 172–179, 2018.

Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. Modulating early visual processing by language. In *NIPS*, pages 6594–6604, 2017.

Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. In *NIPS*, pages 658–666, 2016.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, pages 2414–2423, 2016.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, pages 694–711, 2016.

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

Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context". In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *ECCV*, pages 740–755, 2014. ISBN 978-3-319-10602-1.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NIPS*, pages 8024–8035. Curran Associates, Inc., 2019.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

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

Wei Sun and Tianfu Wu. Learning layout and style reconfigurable gans for controllable image synthesis. *arXiv preprint arXiv:2003.11571*, 2020.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, pages 6626–6637, 2017.

Shane Barratt and Rishi Sharma. A note on the inception score. *arXiv preprint arXiv:1801.01973*, 2018.

Daniel Jiwoong Im, Alllan He Ma, Graham W. Taylor, and Kristin Branson. Quantitatively evaluating GANs with divergences proposed for training. In *ICLR*, 2018.

Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv:1806.07755*, 2018b.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In *NeurIPS*, pages 700–709, 2018.

Shaohui Liu, Yi Wei, Jiwen Lu, and Jie Zhou. An improved evaluation framework for generative adversarial networks. *arXiv preprint arXiv:1803.07474*, 2018.

Tristan Sylvain, Francis Dutil, Tess Berthier, Lisa Di Jorio, Margaux Luck, Devon Hjelm, and Yoshua Bengio. Cross-modal information maximization for medical imaging: Cmim. *arXiv preprint arXiv:2010.10593*, 2020d.

Alex Fedorov, Lei Wu, Tristan Sylvain, Margaux Luck, Thomas P DeRamus, Dmitry Bleklov, Sergey M Plis, and Vince D Calhoun. On self-supervised multi-modal representation learning: An application to alzheimer's disease. *arXiv preprint arXiv:2012.13619*, 2020a.

Alex Fedorov, Tristan Sylvain, Margaux Luck, Lei Wu, Thomas P DeRamus, Alex Kirilin, Dmitry Bleklov, Sergey M Plis, and Vince D Calhoun. Taxonomy of multimodal self-supervised representation learning. *arXiv preprint arXiv:2012.13623*, 2020b.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, and Ronald M Summers. Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays. In *CVPR*, pages 9049–9058, 2018c.

Omer Hadad, Ran Bakalo, Rami Ben-Ari, Sharbell Hashoul, and Guy Amit. Classification of breast lesions using cross-modal deep learning. In *Biomedical Imaging (ISBI 2017)*, pages 109–112. IEEE, 2017.

Dehai Zhang, Menglong Cui, Yun Yang, Po Yang, Cheng Xie, Di Liu, Beibei Yu, and Zhibo Chen. Knowledge graph-based image classification refinement. *IEEE Access*, 7:57678–57690, 2019c.

Amy Zhao, Guha Balakrishnan, Fredo Durand, John V. Guttag, and Adrian V. Dalca. Data augmentation using learned transformations for one-shot medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019b.

Matthias Hofmann, Florian Steinke, Verena Scheel, Guillaume Charpiat, Jason Farquhar, Philip Aschoff, Michael Brady, Bernhard Schölkopf, and Bernd J Pichler. Mri-based attenuation correction for pet/mri: a novel approach combining pattern recognition and atlas registration. *Journal of nuclear medicine*, 49(11):1875–1883, 2008.

Gijs van Tulder and Marleen de Bruijne. Why does synthesized data improve multi-sequence classification? In *MICCAI*, pages 531–538. Springer, 2015.

Soheil Hor and Mehdi Moradi. Scandent tree: A random forest learning method for incomplete multimodal datasets. In *MICCAI*, pages 694–701. Springer, 2015.

Kuniaki Saito, Yusuke Mukuta, Yoshitaka Ushiku, and Tatsuya Harada. Demian: Deep modality invariant adversarial network. *arXiv preprint arXiv:1612.07976*, 2016.

Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer, 1992.

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *ICML*, pages 1247–1255. PMLR, 2013.

Zizhao Zhang, Yuanpu Xie, Fuyong Xing, Mason McGough, and Lin Yang. Mdnet: A semantically and visually interpretable medical image diagnosis network. In *CVPR*, pages 6428–6436, 2017c.

Ankesh Anand, Evan Racah, Sherjil Ozair, Yoshua Bengio, Marc-Alexandre Côté, and R Devon Hjelm. Unsupervised state representation learning in atari. In *NeurIPS*, pages 8769–8782, 2019.

Sarah Rastegar, Mahdieh Soleymani, Hamid R Rabiee, and Seyed Mohsen Shojaee. Mdl-cw: A multimodal deep learning framework with cross weights. In *CVPR*, pages 2601–2609, 2016.

Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. Deconvolutional paragraph representation learning. In *NeurIPS*, pages 4169–4179, 2017d.

Dina Demner-Fushman, Marc D Kohli, Marc B Rosenman, Sonya E Shooshan, Laritza Rodriguez, Sameer Antani, George R Thoma, and Clement J McDonald. Preparing a collection of radiology examinations for distribution and retrieval. *Journal of the American Medical Informatics Association*, 23(2):304–310, 2015.

Bjoern H Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, Yuliya Burren, Nicole Porz, Johannes Slotboom, Roland Wiest, et al. The multimodal brain tumor image segmentation benchmark (brats). *IEEE transactions on medical imaging*, 34(10):1993–2024, 2015.

Spyridon Bakas, Hamed Akbari, Aristeidis Sotiras, Michel Bilello, Martin Rozycki, Justin S Kirby, John B Freymann, Keyvan Farahani, and Christos Davatzikos. Advancing the cancer genome atlas glioma mri collections with expert segmentation labels and radiomic features. *Scientific data*, 4:170117, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

Saeed Amizadeh, Hamid Palangi, Oleksandr Polozov, Yichen Huang, and Kazuhito Koishida. Neuro-symbolic visual reasoning: Disentangling "visual" from "reasoning". In *Proceedings of the 37th International Conference on Machine Learning (ICML-2020)*, pages 10696–10707. ACM, 2020.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016b.

Appendix A

Appendix for the first paper

A.1. Extracting local and global features from a CNN encoder

A.1.1. Explaining local and global features

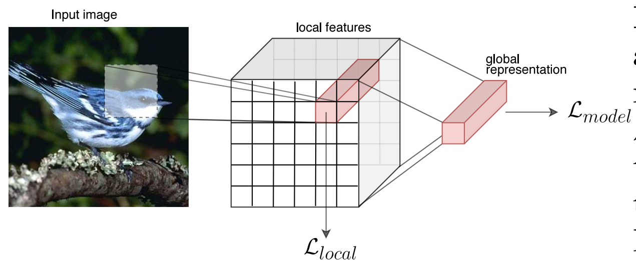


Fig. A.1. The convolutional encoder takes as input an image and outputs a global representation - used to compute the model loss $\mathcal{L}_{model}$. To encourage locality and compositionality, label or attribute based classification is performed on the activations from early layers($\mathcal{L}_{local}$).

A.1.2. Explaining the role of local and global features for MI computation

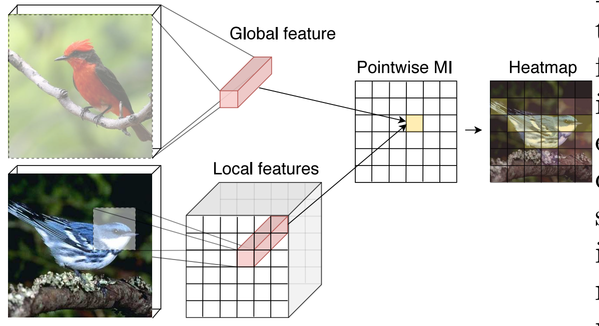


Fig. A.2. Local and global features are extracted from different images (where local features are activations from an early layer in the CNN encoder), then scored against each other. A high score means the two are considered likely to be extracted from the same image by the model, giving us insight in what information is encoded by the global representation. A heatmap of the scores is used to make the result interpretable.

A.2. Implementation details

All models used in this paper have been implemented in PyTorch, and code will be made publically available. All images were resized to size 128×128, random crops with aspect ratio 0.875 were used during training, and center crops with the same ratio were used during test. While most ZSL approaches do not use crops (due to the fact that they used pre-computed features), this experimental setup was shown to be efficient in the field of text to image synthesis [Reed et al., 2016b]. All models are optimized with Adam with a learning rate of 0.0001 with a batch size of 64. The final output of the encoder was chosen to be 1024 across all models. Local experiments were performed extracting features from the third layer of the network. These features have dimension $27 \times 27 \times 384$ for the AlexNet based encoder and $14 \times 14 \times 256$ for the DCGAN encoder.

A.3. MI heatmaps comparing CMDIM with different local losses

In Fig. A.3 we show how local losses affect the type of information CMDIM extracts. We can see how in some cases, e.g. the Nashville Warbler and Rusty Blackbird, the label-based local loss (LC) results in the encoder focusing more on the background and missing out on discriminative features. On the other hand, the label-based (LC) local loss helps the model focus on more localised distinctive patches, especially for the Rusty Blackbird and the Rufous Hummingbird.

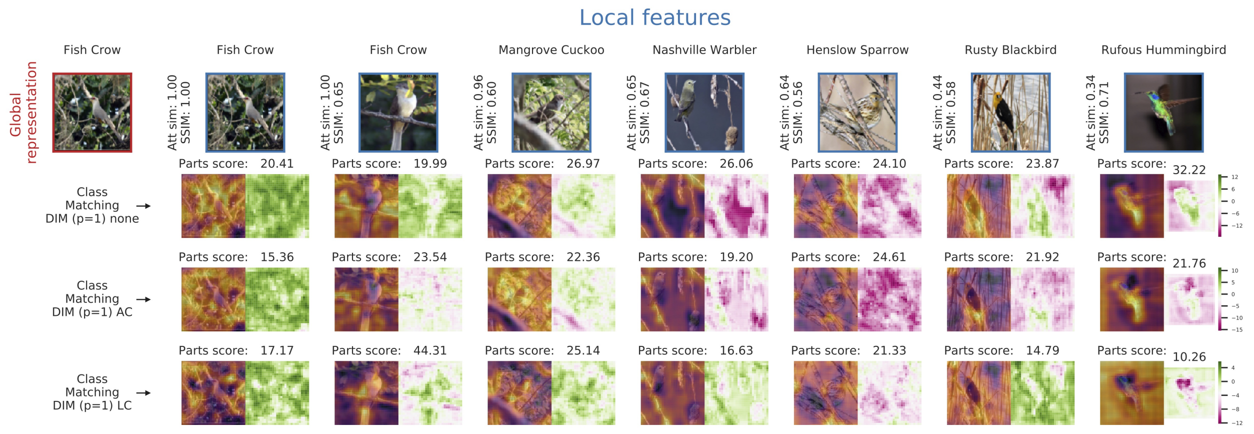
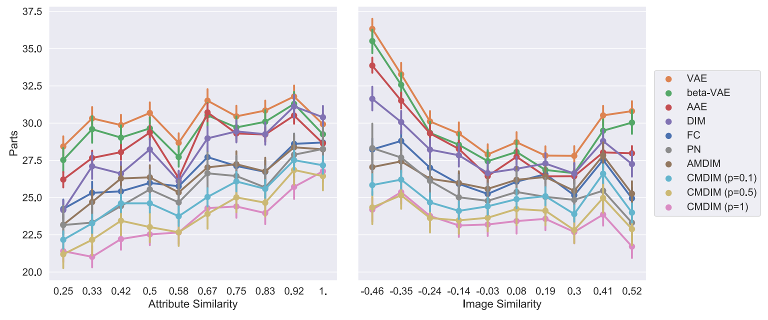


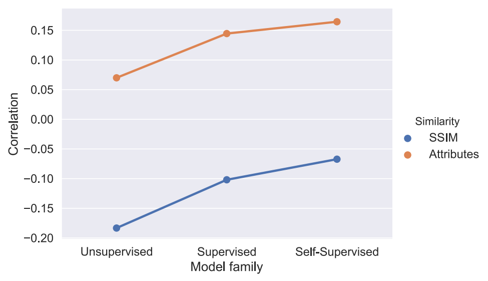
Fig. A.3. Visualization of locality comparing encoders trained with CMDIM's loss and the proposed local losses. The Figure highlights the impact of local losses on the content extracted by the encoders.

A.4. Parts ratio and relationship to different measures of image similarity.

In Fig. A.4, we plot the Parts ratio against the two different measure of image similarity considered in our experiments and we report the correlation between them across the considered families of models. The correlation was computed over 20.000 pairs of images for each family. While not being a strong correlation, our experiments show how it's a statistically significant one, with associated p-value (expressing the probability of a not-correlated sample resulting in the reported correlation coefficients) of less than 10^{-6}.



(a)



(b)

Fig. A.4. Relationship between the parts score and different measures of similarity. On the left, the parts scores is plotted against the two different measures of similarity. We can see there is a clear trend for all the models: the parts score increases for more semantically similar images, and decreases as the images become more similar pixel-wise. The figure on the right shows Pearson's correlation coefficient between the metrics for different models

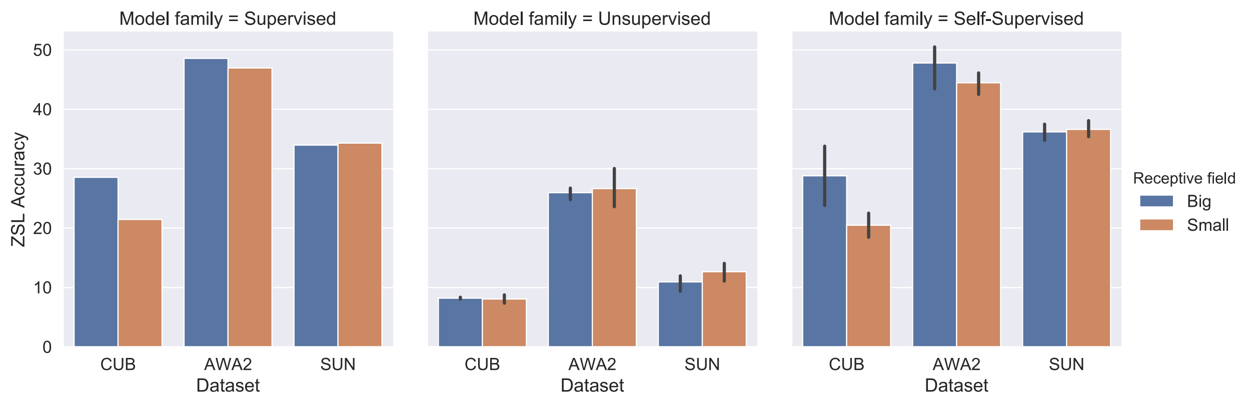


Fig. A.5. Comparing pre and post-pooling (respectively *Small* and *Big*) features in terms of ZSL accuracy. The effect of a varying receptive field size strongly depends on the model type and on the dataset, highlighting how locality is expressed differently in the datasets.

A.5. How local do the representations need to be?

A somewhat alternative way to test if local information is relevant for generalization is to explicitly only consider local information. To achieve this, we consider performing classification directly on features whose receptive field doesn't cover the entire image. In our experiments we consider the features extracted from the AlexNet based encoder right before the flattening and final linear layers. To see the effect of varying the receptive field, we perform the same experiment and pre and post last pooling layer, going from a receptive field of 65 pixels (referred to as *small* in the plots) to 85 pixels (*big*), out of 128 in the original input. The way these local prediction are combined is described in the following section. The results are summarized in Fig. A.5. We can see how the dataset seems to make quite a difference for class-matching DIM that benefits from pooling for the datasets where usually the object is in a small part of input, while for SUN, where the whole image tends to be relevant as the images depict scenes, pooling either does not affect or has a negative effect on performance. For reconstruction based models on the other side we see a different trend, where not performing pooling consistently results in better performance across all datasets.

A.6. Bird parts location maps

The parts annotations provided with the CUB dataset give us the ability to explicitly quantify whether the encoder is learning to extract meaningful local information. To evaluate this, we train a classifier for each part, that takes as input local features extracted from a specific layer of the CNN encoder and outputs the probability of that part being present within the receptive field of the local feature. To construct a ground truth for this evaluation, we pre-process the parts clicks annotations as follows: the datasets provides, for each input and part, a list of multiple parts location as perceived by multiple MTurk workers. Each annotation is provided as (x, y) coordinates of the center of the part and a boolean feature *visible* indicating whether the part is hidden in the considered input. The ground truth for the classifiers is obtained by converting each part annotation into a boolean semantic map, where a truth value is assigned to a square of side 10 pixels centered in all the locations provided by different MTurk users for each part when visible. This process is repeated separately for all the 15 parts. The obtained boolean masks are then processed through a CNN to project them to the size compatible with the extracted features, so that the classifier's loss can be computed. Importantly, this loss is never backpropagated through the encoder, as these classifiers are meant to only evaluate whether the considered local features are predictive of the parts.

A.7. Class Matching DIM

Deep InfoMax [Hjelm et al., 2018] is a self-supervised representation learning algorithm that is trained by maximizing Mutual Information (MI) between local and global features extracted from the network. More specifically, DIM's objective is a lower bound to MI based on the Donsker-Varadhan representation of the KL divergence that computes two expectations: one over the joint distribution of local and global features, and one over the product of the marginals. In the original DIM setting, samples from the joint distribution (positive samples) are defined as local-global pairs extracted from the same input, while for the product of marginals (negative samples) local and global features are extracted from different inputs.

Class Matching DIM performs a similar operation, but samples from the joint distribution are defined to be pairs of local-global features extracted from *different inputs* belonging to the *same class*, and negative samples are pairs where features are extracted from inputs of different classes. Moreover, we add a hyper-parameter p that allows to control the interplay between DIM and CMDIM, so that positive samples are extracted from inputs of the same class with probability p and from the same input otherwise. Intuitively, this would push the encoder to extract features relevant to a specific input while identifying what features are shared across a single class.

A.8. Definition of TRE

We introduce the following notations:

- $\mathcal{X}$ is a dataset, split into train $\mathcal{X}_{\text{tr}}$ and test $\mathcal{X}_{\text{te}}$ sets.
- For $x \in \mathcal{X}$ belonging to a class with binary attributes a_i (for continuous attributes we threshold them beforehand), we define $D(x)$ to be the set of 1-valued attributes (present attributes).
- Each attribute a_i is assigned a learnable vector representation $f_\eta(\mathbf{a}_i) = \eta_i$.
- $\delta(\cdot, \cdot)$ is a distance function, chosen to be cosine similarity as in Andreas [2019].

As in Andreas [2019] we combine individual attribute representations by summation:

$$f_\eta(D(x)) = \sum_{\mathbf{a}_i \in D(x)} f_\eta(\mathbf{a}_i)$$

We can now define:

$$\text{TRE}(x,\, \mathbf{a}\,;\, \eta) = \delta\Big(f_\eta(x), f_\eta(D(x))\Big)$$

$$\text{TRE}(\mathcal{X}, \mathbf{a}; \eta) = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \text{TRE}(x, \mathbf{a}; \eta)$$

We compute $\eta = \arg\min_{\eta'} \text{TRE}(\mathcal{X}_{\text{tr}}, \mathbf{a}; \eta')$ and omit it in what follows by abuse of notations.

A.9. Architectures and datasets details

Table A.1. Details of the datasets used

Dataset	#Images	#Attributes	#Classes	#Train classes	#Test classes
CUB	11,788	312	200	150	50
AWA	30,475	85	50	40	10
SUN	14,340	102	717	645	72

Table A.2. Basic 128x128 architecture details.

Layer	Layer type	Layer params	pooling	activation
0	conv	(64, 4, 2, 1), batch norm	-	ReLU
1	conv	(128, 4, 2, 1), batch norm	-	ReLU
2	conv	(256, 4, 2, 1), batch norm	-	ReLU
3	conv	(512, 4, 2, 1), batch norm	-	ReLU
4	conv	(1024, 4, 2, 1), batch norm	-	ReLU
5	flatten	-	-	-
6	linear	(1024), batch norm	-	ReLU

Table A.3. AlexNet 128x128 architecture details.

Layer	Layer type	Layer params	pooling	activation
0	conv	(96, 3, 1, 1), batch norm	(MaxPool2d, 3, 2)	ReLU
1	conv	(192, 3, 1, 1), batch norm	(MaxPool2d, 3, 2)	ReLU
2	conv	(384, 3, 1, 1), batch norm	-	ReLU
3	conv	(384, 3, 1, 1), batch norm	-	ReLU
4	conv	(192, 3, 1, 1), batch norm	(MaxPool2d, 3, 2)	ReLU
5	conv	(192, 3, 1, 1), batch norm	(MaxPool2d, 3, 2)	ReLU
6	flatten	-	-	-
7	linear	(4096,), batch norm	-	ReLU
8	linear	(4096,), batch norm	-	ReLU

A.10. TRE Ratio Values

Table A.1. TRE ratio values for the CUB dataset

	Normal	AC	LC	Normal_a	AC_a	LC_a
Model	Basic			Alex		
classifier_full	0.761	0.737	0.807	0.758	0.787	0.758
vae	1.062	1.042	1.066	1.111	1.079	1.11
vae_beta	1.04	1.044	1.044	1.12	1.087	1.1
aae	1.011	0.989	1.036	1.003	0.974	0.999
local_dim	0.771	0.875	0.798	0.755	0.783	0.725
local_twopass	0.881	0.913	0.85	0.762	0.804	0.739
local_twopass_class_matching	0.639	0.633	0.752	0.676	0.649	0.624

Table A.2. TRE ratio values for the AWA2 dataset

	Normal	AC	LC	Normal_a	AC_a	LC_a
Model	Basic			Alex		
classifier_full	0.88	0.873	0.913	0.882	0.878	0.909
vae	1.395	1.292	1.379	1.566	1.499	1.681
vae_beta	1.325	1.33	1.372	1.631	1.485	1.567
aae	0.886	0.941	0.986	1.282	0.945	0.913
local_dim	1.211	1.206	1.174	1.154	1.264	1.062
local_twopass	1.103	1.072	1.093	1.151	0.998	1.105
local_twopass_class_matching	0.996	1.155	0.887	1.054	1.118	1.07

Table A.3. TRE ratio values for the SUN dataset

	Normal	AC	LC	Normal_a	AC_a	LC_a
Model	Basic			Alex		
classifier_full	0.85	0.803	0.82	0.839	0.794	0.848
vae	1.005	1.005	1.002	1.04	1.036	1.035
vae_beta	0.989	0.995	0.989	1.038	1.031	1.031
aae	0.963	0.976	0.975	0.954	0.926	0.958
local_dim	1.132	0.977	1.043	1.156	0.995	1.036
local_twopass	1.047	0.912	0.969	1.123	1.042	1.057
local_twopass_class_matching	0.832	0.755	0.877	0.859	0.82	0.956

A.11. F1 part average scores

Table A.1. Part average F1 score, CUB dataset, basic encoder.

Loss	**Normal**	**AC**	**LC**
Model			
FC	0.198	0.368	0.284
VAE	0.07	0.334	0.265
beta-VAE	0.067	0.353	0.255
AAE	0.086	0.085	0.024
DIM	0.235	0.393	0.304
AMDIM	0.311	**0.406**	0.319
CMDIM (p=1)	0.313	**0.406**	0.314
CMDIM (p=0.5)	0.295	0.397	0.321
CMDIM (p=0.1)	0.315	0.382	0.312
PN	0.288		

Table A.2. ZSL accuracy, comparing the different local losses.

Encoder		**alex128x128**			**basic128x128**		
Loss		**Normal**	**AC**	**LC**	**Normal**	**AC**	**LC**
Model	**Dataset**						
FC	CUB	30.47	34.92	32.40	27.44	32.17	30.44
VAE	CUB	12.08	13.41	12.51	12.13	13.46	10.27
beta-VAE	CUB	11.75	11.77	12.33	12.03	12.85	12.52
AAE	CUB	15.16	15.00	15.49	9.12	12.36	9.80
DIM	CUB	23.93	33.35	31.84	24.42	32.54	29.17
AMDIM	CUB	24.34	29.05	31.12	24.42	30.29	28.83
CMDIM (p=1)	CUB	35.80	**40.11**	32.31	29.24	30.08	30.04
CMDIM (p=0.5)	CUB	35.12	37.02	35.27	29.67	**35.15**	31.06
CMDIM (p=0.1)	CUB	29.83	33.60	33.02	27.03	32.35	31.14
PN	CUB	37.59	-	-	26.29	-	-
FC	AWA2	46.48	**52.81**	46.04	45.94	45.98	46.09
VAE	AWA2	29.17	28.54	28.76	30.02	29.60	29.48
beta-VAE	AWA2	29.98	30.11	29.22	29.00	29.47	29.94
AAE	AWA2	32.07	29.46	30.93	31.94	29.31	31.85
DIM	AWA2	38.73	45.54	43.89	39.63	44.23	44.32
AMDIM	AWA2	42.84	45.41	46.95	42.04	49.01	43.77
CMDIM (p=1)	AWA2	45.80	46.56	42.14	46.87	45.00	39.70
CMDIM (p=0.5)	AWA2	46.87	48.06	48.45	46.87	47.92	45.63
CMDIM (p=0.1)	AWA2	47.29	51.51	50.17	45.71	**49.51**	48.01
PN	AWA2	46.53	-	-	45.23	-	-
FC	SUN	33.02	36.89	37.57	32.20	38.79	32.74
VAE	SUN	14.61	15.08	14.33	15.14	16.58	15.22
beta-VAE	SUN	13.80	14.20	13.79	15.08	15.29	16.58
AAE	SUN	17.93	16.78	17.86	18.55	18.41	18.13
DIM	SUN	31.73	39.06	37.64	33.69	41.44	38.52
AMDIM	SUN	38.04	41.44	39.67	37.64	42.26	38.19
CMDIM (p=1)	SUN	35.73	37.43	32.81	34.44	37.98	31.18
CMDIM (p=0.5)	SUN	37.43	40.15	36.62	35.39	39.74	34.10
CMDIM (p=0.1)	SUN	40.01	**42.05**	38.51	40.56	**43.13**	38.93
PN	SUN	32.00	-	-	29.82	-	-

Table A.3. ZSL accuracy, comparing the different local models.

		Average before		Average after	
Model	Dataset	pool	no pool	pool	no pool
FC	CUB	28.55	21.45	13.02	11.75
VAE		8.37	8.06	8.33	8.15
beta-VAE		8.04	8.77	7.91	8.02
AAE		8.20	7.37	7.28	6.84
DIM		18.48	14.39	15.07	11.81
AMDIM		21.45	17.68	16.67	14.04
CMDIM (p=1)		**34.11**	22.16	13.51	15.17
CMDIM (p=0.5)		33.48	22.88	**19.07**	16.35
CMDIM (p=0.1)		26.17	19.23	18.38	15.89
FC	AWA2	48.57	46.95	38.90	41.98
VAE		26.36	30.04	25.47	26.37
beta-VAE		24.78	26.32	27.63	27.11
AAE		26.74	23.59	26.08	25.23
DIM		35.37	34.77	33.54	33.99
AMDIM		41.37	41.34	40.57	37.12
CMDIM (p=1)		49.66	46.26	**48.16**	42.57
CMDIM (p=0.5)		49.17	45.99	45.90	43.28
CMDIM (p=0.1)		**50.95**	44.26	44.90	37.79
FC	SUN	33.97	34.31	32.13	32.15
VAE		11.48	12.84	10.73	10.14
beta-VAE		11.96	14.06	11.89	13.06
AAE		9.38	11.07	9.31	11.32
DIM		25.82	26.83	24.80	25.56
AMDIM		34.04	35.19	34.51	34.58
CMDIM (p=1)		37.02	36.75	33.70	36.53
CMDIM (p=0.5)		37.98	**38.93**	35.53	**37.29**
CMDIM (p=0.1)		35.73	35.60	34.58	36.39

A.12. Full plots

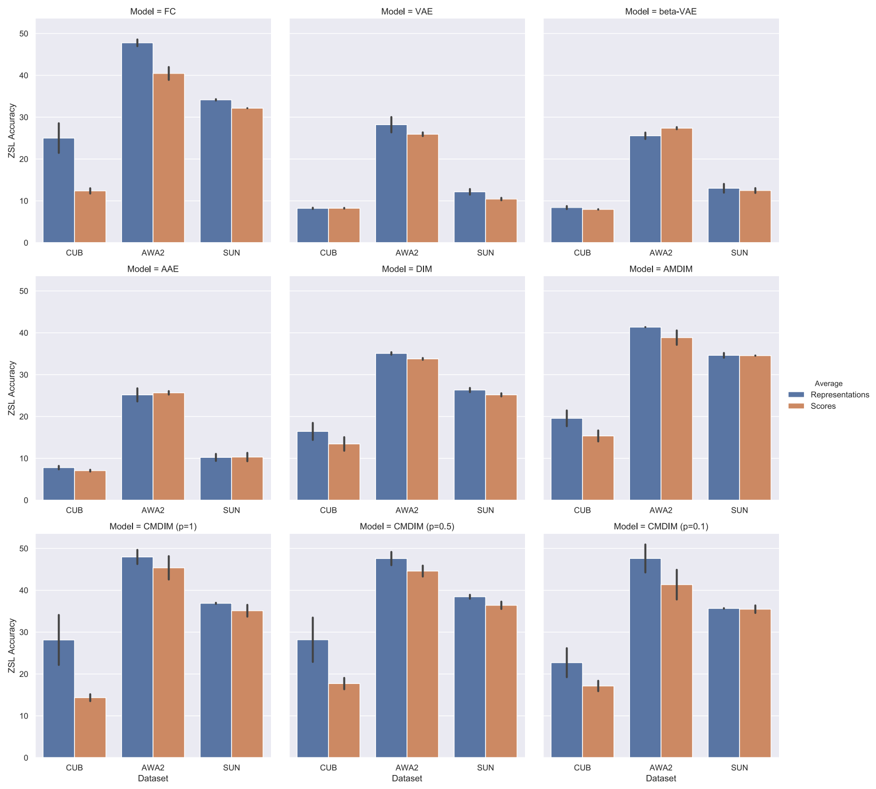


Fig. A.6. Comparison between averaging representations VS averaging scores for all models.

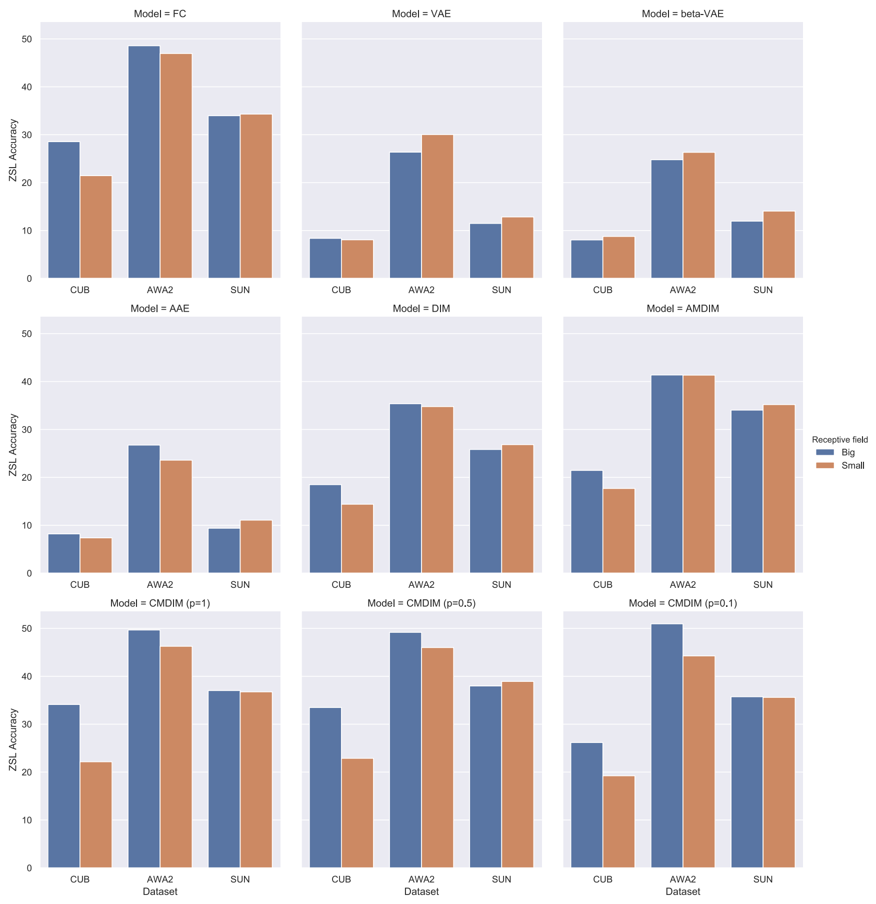


Fig. A.7. Comparison between small and big receptive field for all models.

Appendix B

Appendix for the second paper

B.1. Comparison with Semi-Parametric Methods

Recently, semi-parametric methods have been proposed in the field of layout-to-image generation Li et al. [2019a]. We excluded a comparison with these methods in the main paper due to the fact that (1) they are structurally different (they incorporate real images when generating images) leading to difficulties in making a fair comparison and (2) they function in diverse ways, not all of which can be applied to our setting Qi et al. [2018].

We include a comparison with the state-of-the art semi-parametric model, PasteGAN Li et al. [2019a] in Table B.1. This method outperforms most of the other baselines, but still performs worse than our method.

B.2. Dataset statistics

The dataset statistics are presented in Table B.1.

	Inception Score $\uparrow$	
Methods	COCO	VG
PasteGAN Li et al. [2019a] †	10.2 ± 0.2	8.2 ± 0.2
OC-GAN (ours)	$\mathbf{10.5 \pm 0.3}$	$\mathbf{8.9 \pm 0.3}$
	FID $\downarrow$	
Methods	COCO	VG
PasteGAN Li et al. [2019a] †	38.29	35.25
OC-GAN (ours)	**33.10**	**22.61**

Table B.1. Comparison of our method with the semi-parametric method PasteGAN Li et al. [2019a]. We use † to denote results taken from the original paper. The best results in each category are in bold. Our method outperforms this baseline across the evaluation metrics considered.

Dataset	COCO-Stuff	VG
# Train Images	24 972	62 565
# Valid Images	1 024	5 506
# Test Images	2 048	5 088
# Objects	171	178
# Objects in Image	$3 \sim 8$	$3 \sim 30$

Table B.1. Statistics of the COCO-Stuff and Visual Genome datasets.

B.3. Spatial Relationships used for Generating the Scene-Graph

We used 6 spatial relationships to generate the scene-graphs from layouts. All of the spatial relationships are derived from the bounding box coordinates specified in the layouts. If an edge in the scene-graph is represented as <subject, relationship, object>, then the possible relationships we consider are:

- "left of": subject's centre is to the left of object's centre
- "right of": subject's centre is to the right of object's centre
- "above": subject's centre is above object's centre
- "below": subject's centre is below object's centre
- "inside": subject contained inside object
- "surrounding": object contained inside subject

B.4. A Note on Evaluation

Inception Score and FID were computed using the official Tensorflow implementations [1,2] (the most commonly available PyTorch implementations give slightly different but close values), to ensure compliance with the literature. In the past, papers considering layout and scene graph to image generation have used different values for the number of splits when computing the Inception score, ranging usually from 3 to 5 (as shown in the different official implementations and via contacting some of the authors). Empirically, we found that lowering the split size results in better numerical values for the inception score, for all methods relevant to this work. Out of fairness considerations, we opted for splits of size 5 and note that in addition to this issue, the size of the evaluation set for Inception score computation is very low compared to recommended sizes. This impacts the relevance of this metric.

In addition to the above concerns, some models used different network architectures to compute the inception score (e.g. Zhao et al. [2019a] uses a VGG net as opposed to the standard

[1]`https://github.com/openai/improved-gan` for Inception Score
[2]`https://github.com/bioinf-jku/TTUR` for FID

Inception-V3 network as noted in their paper). We used the official Inception-V3-based evaluation on all models.

Some models introduce non-standard data-augmentation (e.g. Sun and Wu [2019] uses image flips during training). Out of fairness considerations, we compared our approach to the official reported values, and used the same data-augmentation as the compared methods, when applicable.

B.5. Complexity of scenes

We focus on generating images of complex scenes, which warrants a definition of what complex scenes are specifically. In this work, we use the following heuristic. Complex scenes are first and foremost defined with respect to single object datasets: for the most part, images in MS-COCO and VG contain multiple objects (up to 30 in our case). In addition to this, images in both datasets come annotated with relations and attributes (which we do not use in this work, in accordance with the literature). The underlying variability of the scene graphs is also a source of complexity.

B.6. Implementation and Training Details

Architecture diagrams for all the modules of our model OC-GAN are presented in Figs. B.1 and B.2. Some additional hyper-parameter details:

- In the SGSM module, images are resized to size 299×299 before being processed by the image encoder.
- In the SGSM module, the common semantic space for graph and image embeddings has a dimension of 256.

B.7. Additional Qualitative Results

We present additional qualitative 128×128 samples on the COCO-Stuff dataset in Fig. B.3 and on the Visual Genome dataset in Fig. B.4.

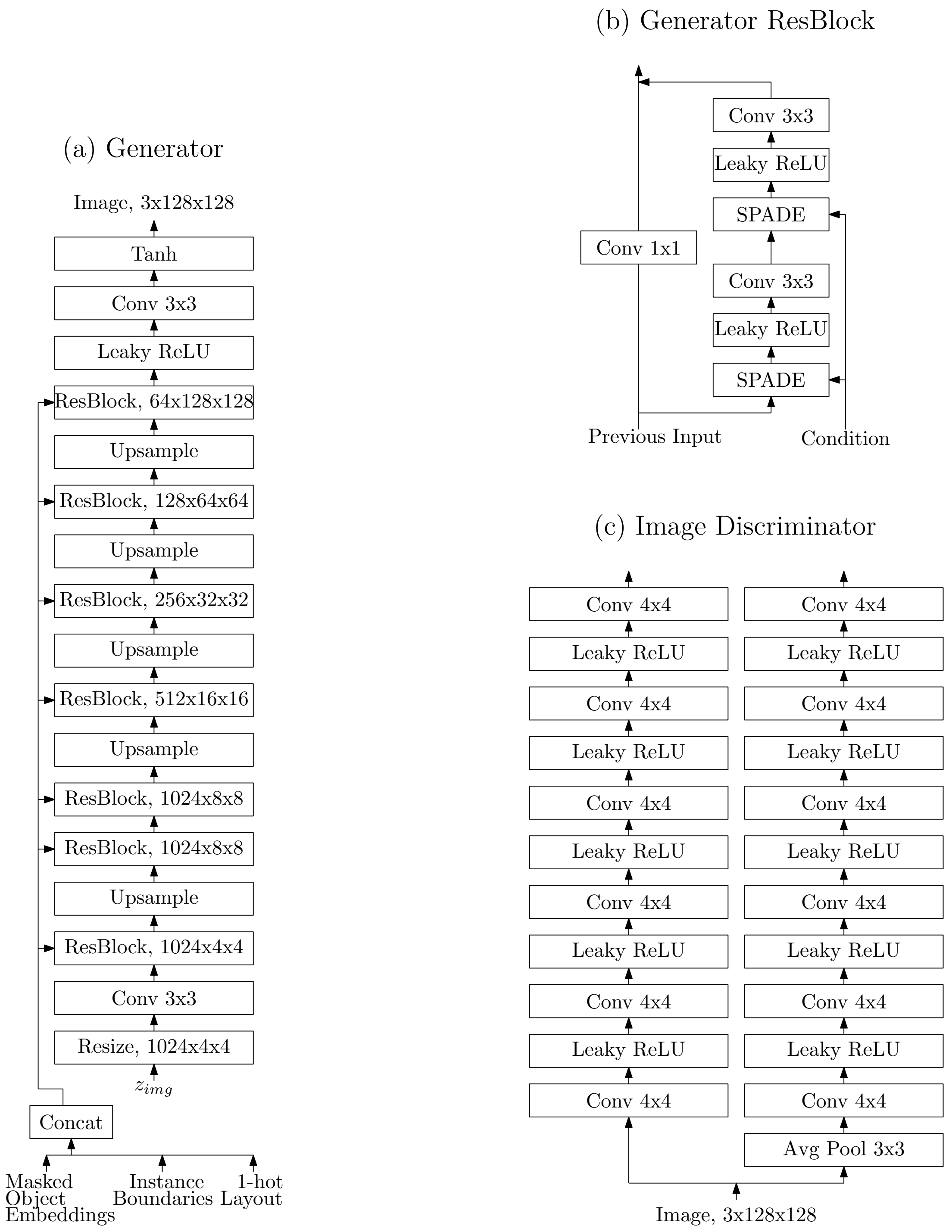


Fig. B.1. Architecture diagrams for (a) Generator (b) Generator ResBlocks (c) Image Discriminator. All generator inputs are derived from the layout. The Masked Object Embeddings are produced by the Conditioning Module. If input and output dimensions match for the Generator ResBlock, then the shortcut is a skip connection.

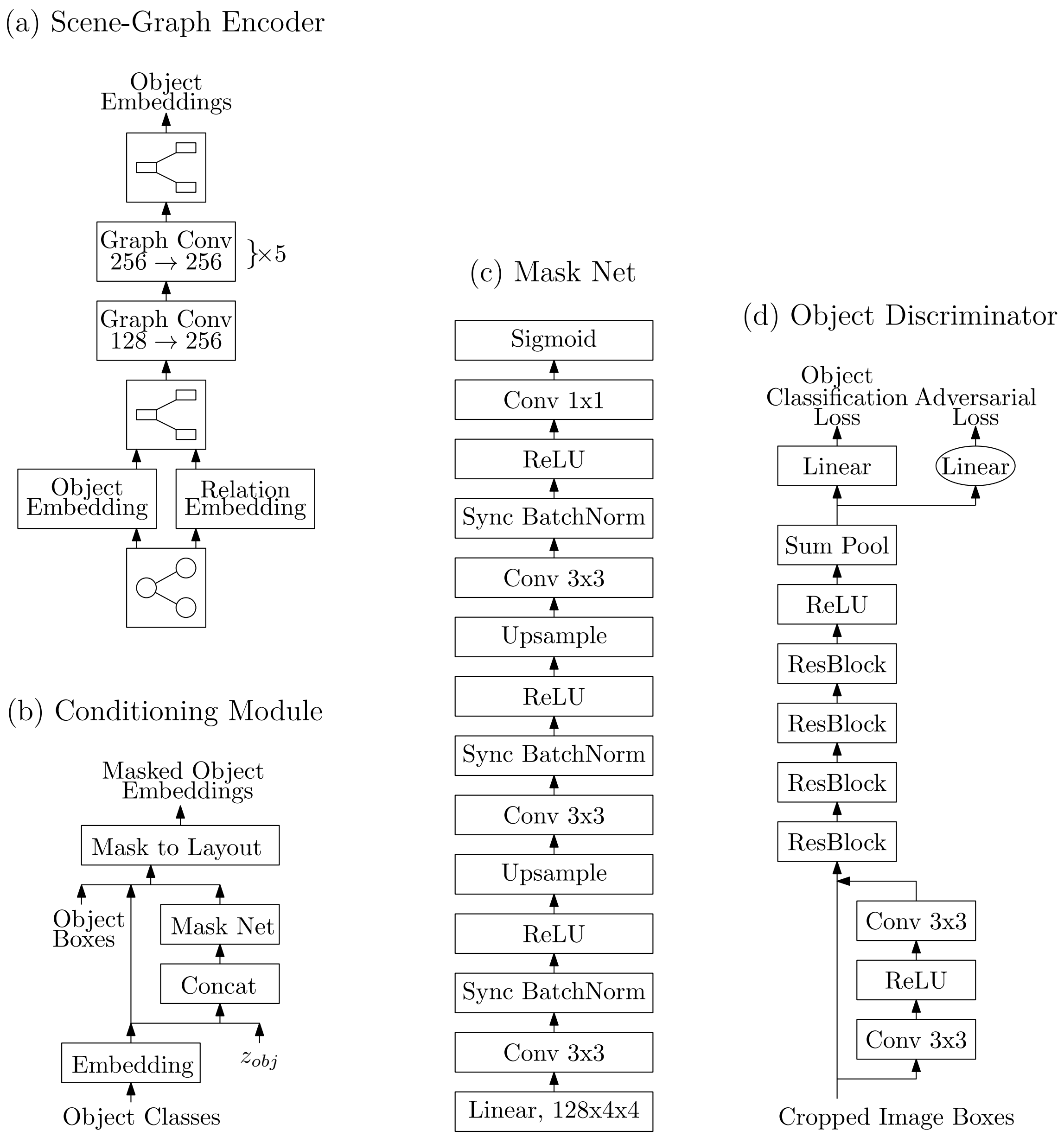


Fig. B.2. Architecture diagrams for (a) Scene-Graph Encoder (b) Conditioning Module (c) Mask Net (d) Object Discriminator. The Scene-Graph Encoder takes as input a scene-graph derived from the layout and processes it with a Graph Convolutional Network. The Conditioning Module generates the Masked Object Embeddings, which along with instance boundaries and 1-hot layout, are the conditioning information for the Generator. The Mask Net is a submodule of the Conditioning Module. The Object Discriminator operates on cropped image boxes in an AC-GAN framework, predicting whether the crop is real or generated as well as classifying the object inside the crop.

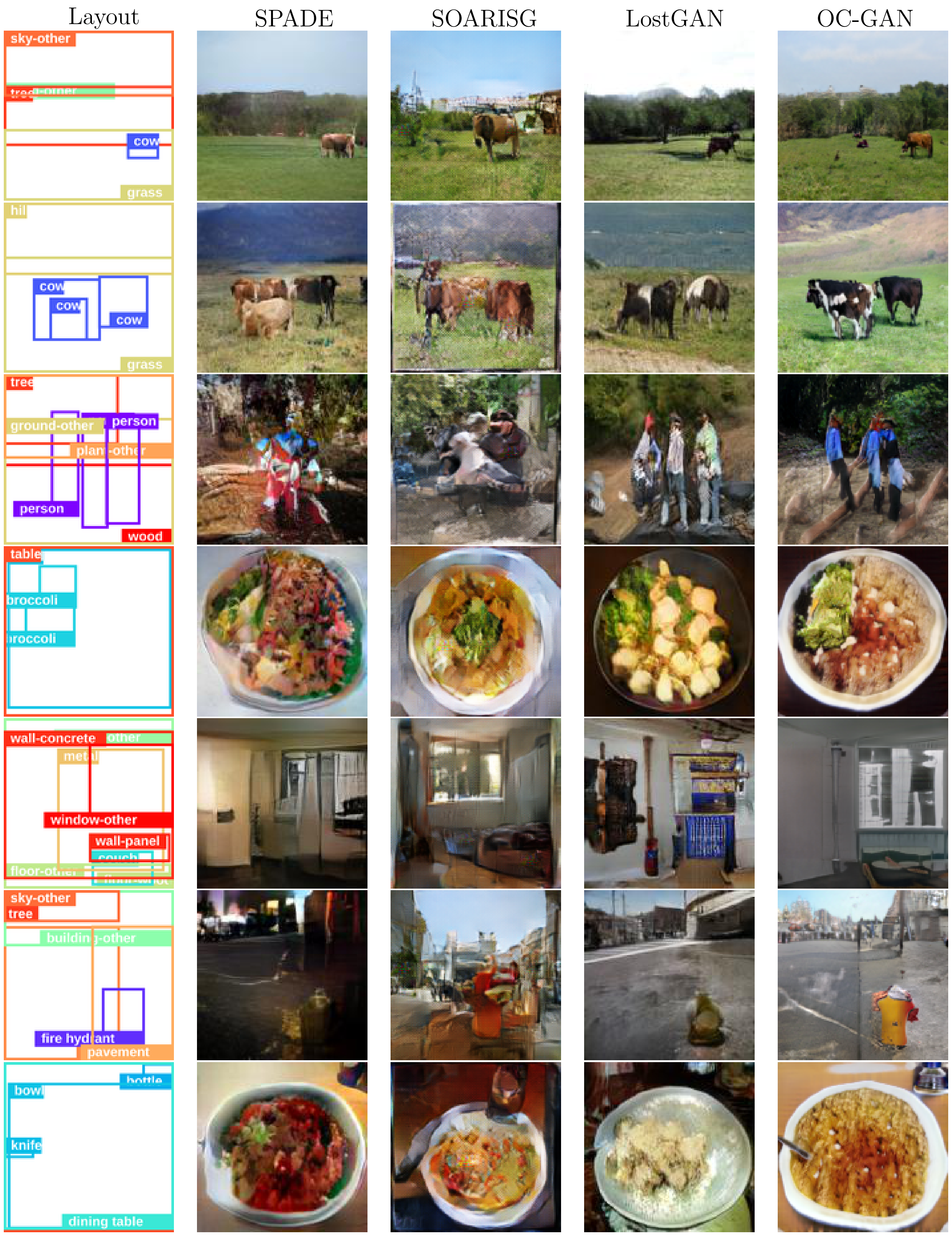


Fig. B.3. 128 × 128 COCO-Stuff test set images, taken from our method (OC-GAN) and multiple competitive baselines.

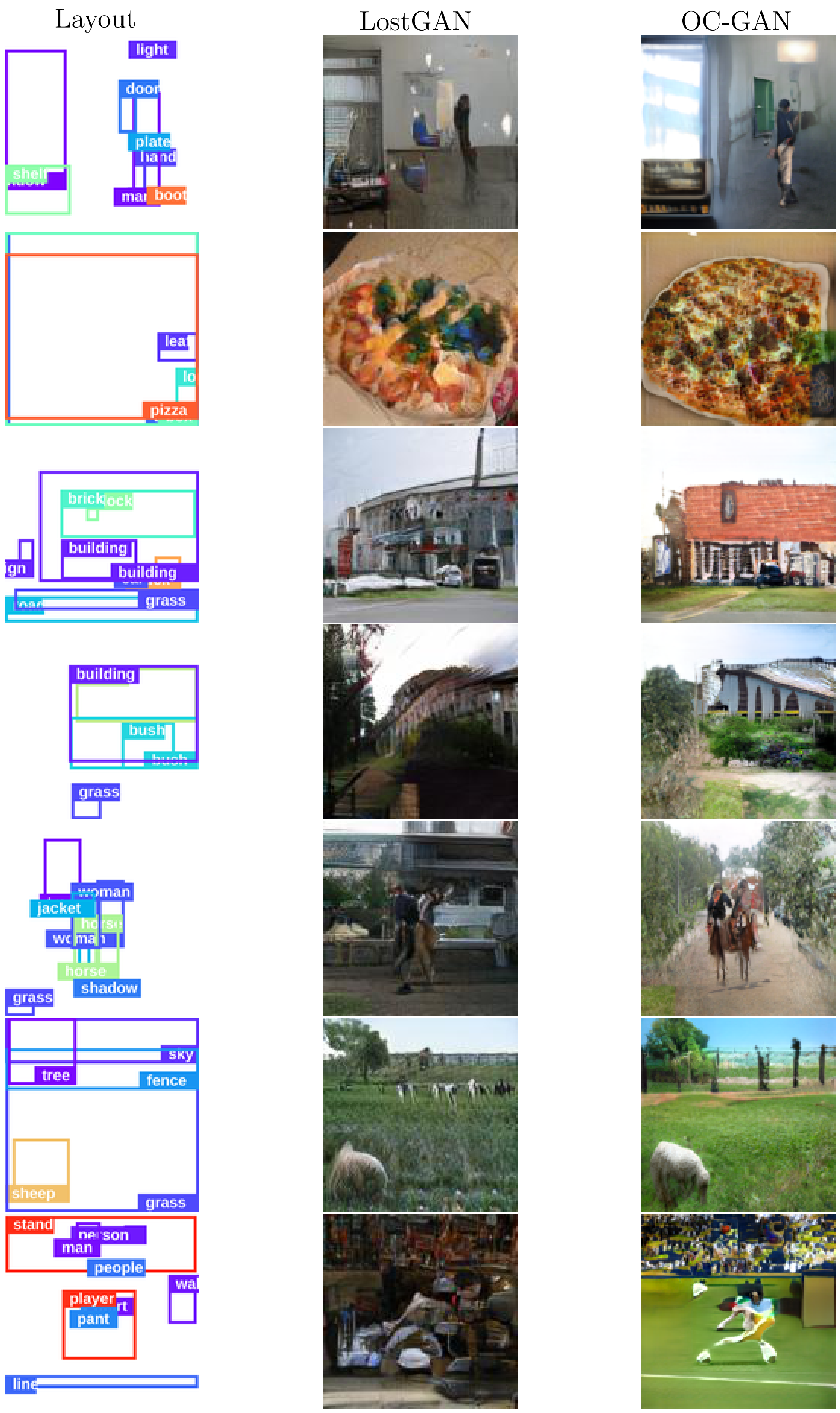


Fig. B.4. 128 × 128 Visual Genome test set images, taken from our method (OC-GAN) and the LostGAN baseline.