Deep learning methods for visual content creation and understanding

Othman Sbai

École des Ponts ParisTech

THÈSE DE DOCTORAT
de l'École des Ponts ParisTech

Les méthodes d'apprentissage profond pour la création et la compréhension du contenu visuel

École doctorale MSTIC N° 532

Signal, Image, Automatique

Thèse préparée au sein du LIGM-IMAGINE / École des Ponts ParisTech

Financement: CIFRE / Facebook AI Research Paris

Thèse soutenue le 04 Octobre 2021, par
Othman SBAI

Composition du jury :

Josef, SIVIC Directeur de recherche, CTU Prague / INRIA	*Président*
Frédéric, DUFAUX Directeur de recherche, Université Paris-Saclay, CNRS	*Rapporteur*
Adrien, BOUSSEAU Directeur de recherche, Inria Sophia-Antipolis	*Rapporteur*
Maria,VAKALOPOULOU Chargée de recherche, École Centrale Supélec	*Examinatrice*
Camille, COUPRIE Chargée de recherche, Facebook Paris	*Co-encadrante*
Mathieu, AUBRY Directeur de recherche, ENPC	*Directeur de thèse*
Renaud, MARLET Directeur de recherche, ENPC	*Directeur de thèse*

École des Ponts ParisTech
LIGM-IMAGINE
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77455 Marne-la-Vallée cedex 2
France

Université Paris-Est Marne-la-Vallée
École Doctorale Paris-Est MSTIC
Département Études Doctorales
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77455 Marne-la-Vallée cedex 2
France

Facebook AI Research Lab
6 Rue Ménars
75002 Paris
France

Ce manuscrit est dédié à ma mère Mariyem et à mon père Abdellah.

Abstract

The goal of this thesis is to develop algorithms to help visual artists create and manipulate images easily with deep learning and computer vision tools. AI advances, in particular generative models, have enabled new possibilities that can be leveraged in the artistic domain to simplify the manipulation of digital visual content, and assist artists in finding inspiring ideas. Progress in this domain could democratize the access to visual content manipulation software, which still requires time, money and expert skills.

The first contribution of this thesis is the introduction of two methods for generating novel and surprising images: one for generating new fashion designs and one for creating unexpected visual blends. First, we show how generative adversarial networks can be used as an inspirational tool for fashion designers to create realistic and novel designs. While most image generation models aim to generate realistic images that cannot be differentiated from the real ones, they tend to reproduce the training examples. We instead focus on designing models that encourage novelty and surprise in the generated images. Second, we develop a visual blending model that allows the generation of compositions by blending objects in uncommon contexts based on visual similarity. Using recent advances in image retrieval, completion and blending, our simple model provides realistic and surprising visual blends. We study how the selection of the foreground object influences its novelty and realism. In the rest of the thesis, we focus on improving the image generation methods presented by exploring how generative models can be extended to resolution independent image generation and by studying the quality of image features used in image retrieval from a training data perspective.

The second contribution is a new layered image decomposition and generation model aimed at representing images in a resolution independent and easily editable way. Generating higher resolution images is challenging from a training time and stability perspectives. To alleviate these difficulties, we design the first deep learning based image generation model using vector mask layers. We frame vector mask generation using a parametric function (multi-layer perceptron) applied on a regular coordinate grid to obtain mask values at input pixel positions. Our model reconstructs images by predicting vector masks and their corresponding colors then iteratively blends colored masks. We train our model to reconstruct natural images, from face images to more diverse ones, we show how our model captures interesting mask embeddings that can be used for image editing and vectorization. Furthermore, we show our model can also be trained in an adversarial fashion.

The third contribution is focused on image retrieval and few-shot classification. Indeed, a large part of the artistic work and effort when creating visual blends is searching for relevant images to use. To simplify this tedious step of image search, deep features can be used as similarity measures to retrieve images. While there has been consequent work on learning image representations for image classification, and particularly using self-supervised techniques, the impact of the training dataset on the quality of learned features has not been extensively explored. Thus, we study the impact of the base dataset composition on the quality of features from a few-shot classification perspective. We show that designing the base training dataset is crucial for improving the features for few-shot classification performance. For instance, a careful dataset relabeling allows to increase the performance considerably using a simple competitive baseline model.

Keywords: deep learning, image generation, image editing, visual blends, few-shot image classification, image retrieval.

Résumé

L'objectif de cette thèse et de développer des algorithmes capables d'aider les artistes visuels à créer et à manipuler facilement des images avec les outils de l'apprentissage profond et de la vision par ordinateur. Les avancées de l'IA, en particulier les modèles génératifs, ont permis de nouvelles possibilités qui peuvent être utilisées dans le domaine artistique afin de simplifier la manipulation des contenus visuels et d'assister les artistes à trouver des idées inspirantes.

La première contribution de cette thèse est l'introduction de deux méthodes pour générer des images nouvelles et surprenantes : une pour générer de nouveaux designs de mode et une pour créer des mélanges visuels. Premièrement, dans la génération d'images de mode, nous montrons en particulier comment les réseaux génératifs adversaires peuvent être utilisés comme un outil d'inspiration pour les créateurs de mode pour créer des designs réalistes et novateurs. Alors que la plupart des modèles de génération d'images visent à générer des images réalistes qui ne peuvent pas être différenciées des vraies, ces modèles ont tendance à reproduire les exemples d'apprentissage. Nous nous concentrons plutôt sur la conception de modèles qui encouragent la nouveauté et la surprise dans les images générées. Deuxièmement, nous développons un nouveau modèle de collage qui permet la génération de compositions en mélangeant des objets dans des contextes inhabituels basés sur la similarité visuelle. En utilisant les avancées récentes dans la récupération, la complétion et le mélange d'images, notre modèle simple fournit des mélanges visuels réalistes et surprenants. Nous étudions comment la sélection de l'objet de premier plan influence l'originalité et le réalisme des compositions obtenues.

Dans le reste de la thèse, nous nous concentrons sur l'amélioration des méthodes de génération proposées dans la première partie. Tout d'abord, nous explorons une extension des modèles génératifs à la génération d'images à résolution indéfinie. Ensuite, nous étudions la qualité des représentations d'images pour la recherche d'images par rapport à la base d'images d'entraînement.

La deuxième contribution est un nouveau modèle de décomposition et de génération d'images en couches visant à représenter les images d'une manière indépendante de la résolution. La génération d'images à plus haute résolution est un défi du point de vue du temps et de la stabilité de l'entraînement. Pour pallier ces difficultés, nous concevons le premier modèle de génération d'images basé sur l'apprentissage profond utilisant des couches de masques vectoriels.

Nous exprimons la génération de masques vectoriels avec une fonction paramétrique (perceptron multicouche) appliquée sur une grille de coordonnées régulière pour obtenir des valeurs de masque aux positions des pixels d'entrée. Notre modèle reconstruit les images en prédisant les masques vectoriels et leurs couleurs correspondantes puis mélange itérativement les masques colorés. Nous entraînons notre modèle à reconstruire des images naturelles, des images de visage à des images plus diverses, nous montrons comment notre modèle capture des représentations de masque intéressantes qui peuvent être utilisées pour l'édition et la vectorisation d'images. De plus, nous présentons le premier modèle de génération d'images vectorielles formé de manière adversaire.

La troisième contribution est centrée sur la recherche d'images et la classification à partir de peu d'exemples. En effet, une grande partie du travail et de l'effort artistique lors de la création de mélanges visuels consiste à rechercher des images pertinentes à utiliser.

Pour simplifier cette étape fastidieuse de recherche d'images, des représentations profondes d'images peuvent être utilisées comme mesures de similarité pour récupérer des images. Bien qu'il y ait eu des travaux conséquents sur l'apprentissage des représentations d'images pour la classification des images, et en particulier à l'aide de techniques autosupervisées, l'impact de l'ensemble de données d'apprentissage sur la qualité des caractéristiques apprises n'a pas été exploré de manière approfondie. Ainsi, nous étudions l'impact de la composition de l'ensemble de données de base sur la qualité des caractéristiques du point de vue de la classification à partir de peu d'exemples. Nous montrons que la conception de l'ensemble de données d'entraînement de base est cruciale pour améliorer les fonctionnalités des performances de classification à partir de peu d'exemples. Par exemple, un réétiquetage minutieux de l'ensemble de données permet d'augmenter considérablement les performances à l'aide d'un modèle de base concurrentiel simple.

Mots-clés: Apprentissage profond, génération d'images, classification d'images avec peu d'exemples, recherche d'images.

Acknowledgments

Je remercie chaleureusement les membres de mon jury ainsi que les rapporteurs de la thèse. Leurs pertinents conseils contribuent à cette thèse.

Tout d'abord je suis très reconnaissant envers mes deux encadrants; Mathieu et Camille. Mathieu, merci d'avoir été un directeur de thèse exceptionnel, merci pour ton encadrement académique et pour tes précieux conseils et idées. Merci pour ta disponibilité et ton dévouement. Tu as su créer une belle équipe de doctorants, et je suis très fier d'en avoir fait partie. Camille, merci d'avoir été une excellente encadrante au sein de Facebook. Tu as toujours été à l'écoute et très attentionnée, et tu as su me pousser à me dépasser. Merci pour ton aide et tes conseils tout au long de la thèse et de mon stage de fin d'étude.

Tout au long de cette thèse cifre, j'ai eu la chance de partager mon temps entre de deux environnements de travail; le laboratoire Imagine et FAIR (Facebook AI Research).

Je remercie les chercheurs du labo Imagine, Renaud, Pascal, David, Vincent et Guillaume. Merci d'avoir été une source de connaissances et de conseils précieux.

Merci à mes amis doctorants au sein de Imagine; Xi, Pierre Alain, Thibault, Tom, Yang, Simon, Théo, François, Michael, Abderrahmane, Spyros, Georgia, Shell, Praveer, Xuchong et tout les autres. Certes nous nous sommes pas tous retrouvé depuis longtemps à cause du covid, mais je garde de bons souvenirs de nos matchs de foot, de nos discussions chez Jeanine ainsi que notre voyage pendant CVPR 2019 à Long Beach.

Je remercie particulièrement Xi et Pierre Alain pour leur amitié de longue date et pour nos échanges précieux tout au long de la thèse. Merci à Pierre Alain pour ton soutien important à la fin de la thèse pendant les entretiens d'embauche et les négotiations avec les RH. Merci Xi pour ta disponibilité, ton écoute et ton aide infaillibles.

Je remercie les collègues du labo FAIR, Natalia, Mohamed, Francisco, Sergey, Ludovic, Olivier, Antoine, Yann, Patricia, etc.

Antoine merci d'avoir été un bon mentor depuis le début de mon stage chez Facebook, d'avoir cru en moi tout au long de mon stage et de ma thèse.

Francisco, merci pour ton aide précieuse concernant pytorch et pas que. Ça m'a souvent évité plusieurs heures de debugging.

Je remercie les amis doctorants au sein de FAIR; Mathilde, Rahma, Hubert, Alexandre, Pierre, Louis, Nicolas, Neil, Guillaume, Alexis, et tout les autres. J'ai beaucoup appris par la diversité de nos sujets de thèses et par notre interminable et passionnante discussion de groupe. Je garde de très beaux souvenirs de nos voyages pendant les FAIR offsites.

Merci à toute les personnes au sein de Facebook qui ont fait de mon passage là-bas une expérience inoubliable et un cadre de travail idéal.

Je remercie l'École des Ponts, de m'avoir accueilli ces huits dernières années, de m'avoir formé et de m'avoir donné tant d'opportunités.

Je remercie aussi mes amis Yousif, Nisrine, Nils, Amira, Rémi, Émilie, Marie-Alix, Maxence pour les bons moments passés à l'internat de Marius pendant la rédaction de ma thèse.

Je remercie Souli, Hamza, Ghizlane, Ismail, Yasmina, Meryem, Oussama et Imène ainsi que toute ma famille qui a toujours été ma ressource de bons moments familiaux.

Je remercie Ilham, ma chérie. Merci pour ton amour et ton soutien qui m'ont donné la force de braver toutes les difficultés.

Enfin, je remercie Maman et Papa, merci d'avoir été mes piliers infaillibles.

Contents

Chapter 1

Introduction

1.1 Goals

Can an AI inspire us to push the boundaries of creativity ? The goal of this thesis is to develop learning based image generation and manipulation tools with the perspective of assisting human artists. By leveraging the advances in visual representation learning and image generation, we build learning algorithms that can serve as an inspirational assistant by creating visually pleasing and novel images. Among the large number of artistic applications enabled by recent computer vision advances, we particularly focus on two: fashion image generation and visual blends creation. This leads us to tackle two fundamental challenges with impact beyond the considered creative applications. First, we aim to enable high resolution and easily interpretable image generation and second, we aim at understanding the impact of training data on supervised deep image features.

1.2 Motivations

Our main motivation for tackling creative image generation is to support digital artists with the next-generation tools for imagining, creating and manipulating the visual content in art, fashion and communication that we discuss hereafter.

Art The evolution of image generation algorithms is changing the way we think about art and the way we create. Early works in the computational creativity literature have used evolutionary methods with a human in the loop guiding the process, where the computer explores the creative space and the human plays the role of the observer whose feedback drives the process (Graf and Banzhaf, 1995). Recently, Creative Adversarial Networks (CAN) (Elgammal et al., 2017) introduced an image generation model trained on art paintings of different styles, that is able to generate realistic and aesthetically appealing images. Some of their best generations are shown in Fig. 1.1a. These image generation techniques transform the art domain; in 2018, an AI generated painting was valued for more than 400.000$ at an auction (Christies.com, 2018). The increased realism of the generations makes it difficult to distinguish real artwork from an automatically generated one. Many artists have embraced this new direction of painting by creating with code, and the perspective of having an algorithm able to generate realistic and meaning-

ful artworks is very exciting. Building on previous work such as CAN (Elgammal et al., 2017), we explore the potential of these generative models on fashion image generation.

Fashion There is a considerable interest in leveraging artificial intelligence for fashion design. Creating a garment is a complex process that requires imagination, creativity and continuous reinvention. Designers usually take inspiration from their environments and sources such as the internet. Having a virtual assistant able to inspire new designs based on a large set of images could allow them to explore new directions while maintaining high level controls over the influences of their designs. Nowadays, it is not difficult to imagine an AI-inspired collection of garments, an idea that could reinvent a large industry that is fashion. Fashion image generation differs from standard image generation because of the design elements specific to fashion that are shape and texture or fabric of the garment. Also, generating a garment needs to respect a set of wearability rules which makes it a challenging task for image generation models. Having algorithms able to generate garments with high levels controls could allow applications such as item personalization and enable designers and customers to quickly customize a garment with different styles and body poses as is shown in Fig. 1.1b.

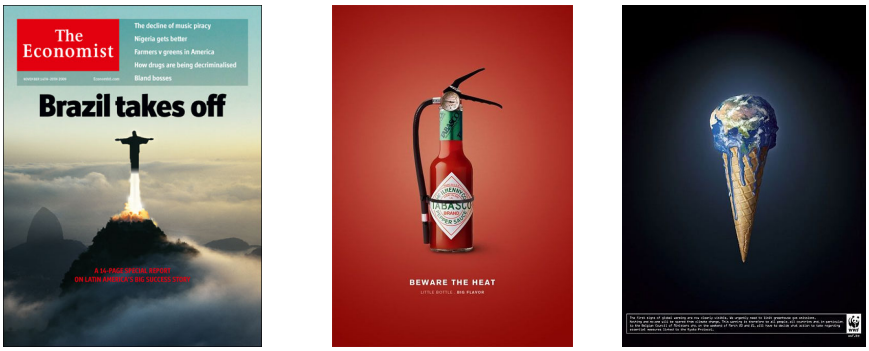
Advertising and communication Visual blends are a powerful expression tool commonly used in advertising, news and art. They consist in representing an analogy, whether phonetic, linguistic or pictorial using composite images. They have been extensively studied in multiple marketing and advertising papers showing their impact and understanding their properties (Gkiouzepas and Hogg, 2011; Phillips and McQuarrie, 2004; Jeong, 2008), and are also widely used by artists and shared by content creators on internet. Creating visual blends is a lengthy process that requires finding an analogy, searching for the right images to illustrate the concept and performing a visual blending step that can require professional skills. However, it can be simplified by leveraging automatic search using semantic, visual or phonetic similarities and automatic image blending techniques to the retrieved images. Fig. 1.1c shows visual blending examples that demonstrate the difficulty of the search and blending process to make a seamless and realistic visual blend. Another example of that is the Artbreeder (`https://www.artbreeder.com/`) tool which allows users to mix images and create new ones using automated tools. Other artists, such as @idriesk on Instagram, use visual metaphors to create striking and uncommon visual analogies that feature scenes, people or objects from different contexts and cultures. Inspired by these artists' works, we are interested in creating an algorithm to help artists find interesting visual analogies and compositions.

1.3 Context

Over the last decade, artificial intelligence has known impressive advances in a wide variety of domains. From ordinary tasks such as perception in computer vision (He et al., 2017, 2015a; Tan et al., 2020; Redmon and Farhadi, 2018) and speech processing (Schneider et al., 2019), natural language understanding (Collobert et al., 2011), generation (Brown et al., 2020; Roller et al., 2020) and translation (Lample and Conneau, 2019; Fan et al., 2020), common sense reasoning and planning (Silver et al., 2016) to new tasks

(a) Examples of images generated by Creative Adversarial Networks CAN (Elgammal et al., 2017), trained on an art paintings dataset.

(b) Using generative models to visualize new outfits by transferring the colors or the body pose Yildirim et al. (2019).



The Economist Nov. 2009 Tabasco Ad "Ice cream" by WWF

(c) Examples of visual blends. Top row: examples from art, fashion, advertising and communication.

Figure 1.1: Motivation

such as symbolic mathematics (Lample and Charton, 2020), medical diagnosis (McKinney et al., 2020; Muckley et al., 2020) and other scientific challenges (Senior et al., 2020; Chanussot et al., 2020; Li et al., 2020; Javaheri et al., 2020). A key direction for artificial intelligence research is artistic creativity where the goal is for machines to generate original items with realistic, aesthetic attributes. In fact, the evolution of generative modeling across different modalities such as speech, text and images allowed important advances

in generating realistic content and the exploration of the creative directions that resulted from them. For example, previous work has explored the use of deep learning for music generation (Briot et al., 2019), imitating the styles of great painters (Gatys et al., 2016; Dumoulin et al., 2017) or doodling sketches (Ha and Eck, 2018).

In this work, we are interested in image generation and its potential in artistic applications such as fashion item generation and visual blends creation. Using a deep learning based and a data oriented approach, we propose models that allow the generation and editing of novel content. Multiple factors have enabled this research direction; data, computing and software resources. First, the creation of large scale datasets dedicated to understanding the trends in art and fashion as well as to novel and realistic image generation (wikiart.org, 2010; Liu et al., 2016; Rostamzadeh et al., 2018). Second, the availability of powerful hardware such as Graphical Processing Units (GPUs) with large memory and number of processing cores that were adapted from their original purpose of graphical rendering to perform tensor operations for deep learning. GPU usage has been made accessible thanks to development of rich and easy-to-use deep learning frameworks such as PyTorch (Paszke et al., 2019) and Tensorflow (Abadi et al., 2015). Finally, reproducible code open-sourcing and open paper publishing have allowed this field to evolve rapidly.

1.4 Challenges and contributions

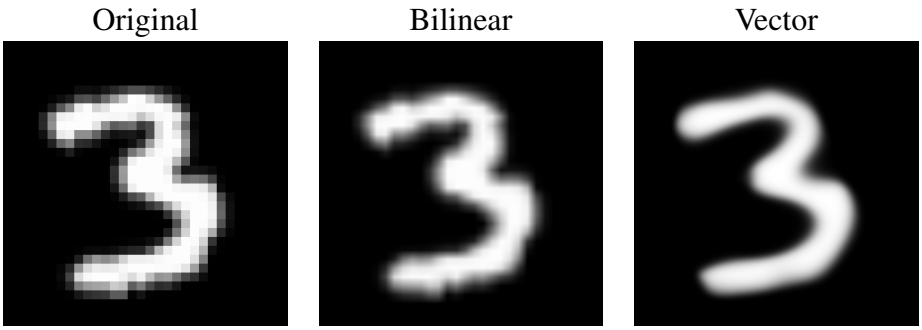
In the following, we present the challenges that we consider in this thesis and present the contribution that address them.

Realistic and novel image generation Generative models learn to reproduce the data distribution they are trained on, which leads to realistic samples very similar to the original ones. A major challenge of image generation is going beyond the creation of realistic images indistinguishable from the real ones, by enforcing novelty and diversity in the generations while maintaining realistic aspect of the generations.

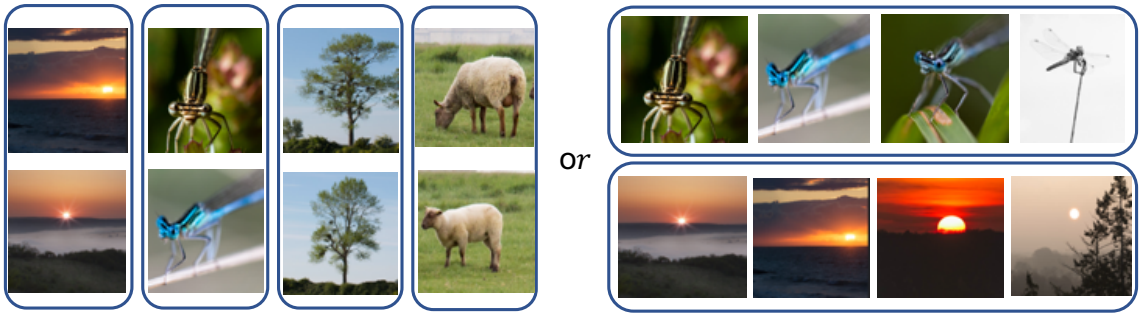
Resolution-independent and editable image generation Another challenging aspect of image generation is the resolution of generated images. While different works have introduced multiple techniques to stabilize training of image generation models, thus improving the possible resolution from 32×32 Goodfellow et al. (2014) to 1024×1024 Karras et al. (2017) and more, getting rid of resolution constraints by considering new paradigms of image generation is an open challenge. Instead of generating images pixel by pixel, can we generate images in a vector representation that can be then rendered at any desired resolution?

Learning strong image representations Having strong image features is crucial in image retrieval and understanding. Image features can be learned using classification on labelled training classes and a major characteristic of these image representations is their ability to generalize on new categories different from the labelled ones seen during the training. An interesting challenge is understanding the impact of the training dataset labeling on the quality of learned features and finding ways to improve the available image annotations to obtain better features.

(a) By introducing new training objectives for generative models and building a search and composition model, we propose models that output original and realistic images.



(b) We propose a novel paradigm for resolution independent image representation and generation. A simple example on generating MNIST in 1024×1024 numbers from a learned vector representation. From left to right: original image, a bilinear upsampling and a high resolution using the vector representation.



(c) Should we annotate more classes or more examples per class for example? We study how the dataset annotation and design affects the quality of learned features.

Figure 1.2: Illustration of contributions

To tackle these challenges, we present three contributions:

- We build models that enable creative applications by suggesting novel images, generations or compositions with a trade-off between originality and realism. Fig. 1.2a
- We propose an image generation paradigm for vector and layered image generation allowing simple image editing using learned masks and resolution-independent image representation. Fig. 1.2b
- We study the importance of datasets design for learning image features. Fig. 1.2c

1.5 Thesis outline

This thesis is organized as follows:

- Chapter 2 gives an overview of the main concepts and works that are relevant for this dissertation.
- In chapter 3, we present two methods for novel image generation in the contexts fashion image generation and visual blends creation.
- Chapter 4, we present a new paradigm for image generation in vector layers that allows resolution-independent image representation and simple editing.
- In chapter 5, we present a study on the importance of dataset design in the context of feature learning for few-shot image classification.

1.6 Publication list

The work presented in this thesis has been published as described below.

- **Othman Sbai**, Mohamed Elhoseiny, Antoine Bordes, Yann LeCun and Camille Couprie (2018). DesIGN: Design inspiration from generative networks. In the first workshop on Computer Vision for Fashion, Art and Design at the European Conference on Computer Vision (ECCVW 2018).
- **Othman Sbai**, Camille Couprie and Mathieu Aubry. (2020) Unsupervised Image Decomposition in Vector Layers. In the International Conference on Image Processing (ICIP 2020).
- **Othman Sbai**, Camille Couprie and Mathieu Aubry. Impact of base dataset design on few-shot image classification. (2020) In European Conference on Computer Vision (ECCV 2020).
- **Othman Sbai**, Camille Couprie and Mathieu Aubry. (2021) Surprising image compositions. In the International Conference on Computational Creativity and in the fourth Workshop on Computer Vision for Fashion, Art, and Design at the Conference on Computer Vision and Pattern recognition (ICCC2021 and CVPRW 2021).

We open-sourced the code corresponding to the papers, and created web pages for each project with additional visualizations of the results available at `https://www.sbaiothman.com/`. I received the best paper award for the DesIGN paper at ECCV Workshop 2018 in Munich.

Chapter 2

Related work

In this chapter, we give an overview of the concepts and works that are the most relevant for this dissertation.

First, we start with an introduction of the machine learning framework, the important building blocks of deep learning and how it can be used to learn powerful image representations from data. Second, we present the generative adversarial learning framework, its major progress and how it is leveraged for creative image editing applications. Third, we present works on vector and layered image generation that allow image generation in a resolution-independent manner. Finally, we discuss important works that marked the research in machine learning for creative applications especially in fashion image generation and visual conceptual blending.

2.1 Learning visual representations

2.1.1 Supervised learning from data

The goal of machine learning is to build algorithms that endow machines with the ability to perform given tasks without explicitly hardcoding their solution. Machine learning algorithms can vary in their level of supervision from fully supervised to weakly and self-supervised or even unsupervised learning. They also can be either parametric or non-parametric depending on the assumptions on the form of the function that maps input variables to output ones.

In parametric machine learning, instead of specifying the algorithm for solving the task, a mathematical model with learnable parameters w is trained on a *training dataset* by minimizing a loss function that describes how well the model performs the task.

In the supervised setting, a target label y_i is known for each data sample x_i, and the goal of the model is to predict an output $\hat{y}_i$ as close as possible to y_i given x_i. The total loss function $\mathcal{L}_w$ is the average of individual cost ℓ_w on each data sample $(x_i, y_i), i \in [1, .., N]$ available for training:

$$\mathcal{L}_w = \frac{1}{N} \sum_{i=1}^{N} \ell_w(y_i, \hat{y}_i) \tag{2.1}$$

For example, in the case of classification, the loss function ℓ_w can be the negative log likelihood of the observed model's output and the expected target labels. Given C

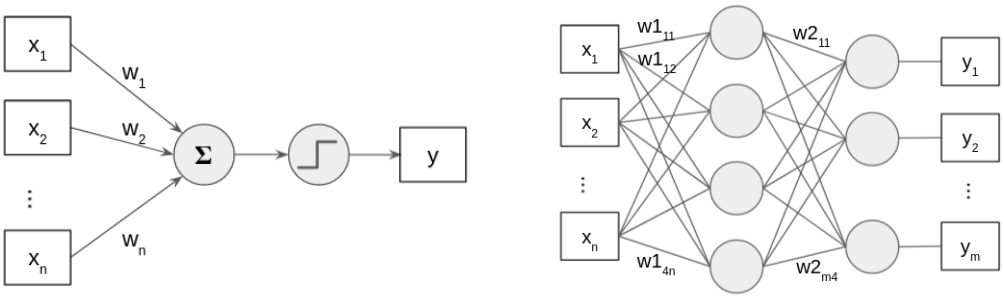


Figure 2.1: Visualization of the basic neural network architectures: the perceptron (left) and the multi-layer perceptron (right).

the number of classes, and y_{ij} are the one-hot encoded class targets y_i, the negative log-likelihood loss is:

$$\mathcal{L}_w = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}) \tag{2.2}$$

Similaly, in regression, the loss to optimize can be a least squares loss between the model's output and the expected target labels:

$$\mathcal{L}_w = -\frac{1}{N}\sum_{i=1}^{N} \|y_i - \hat{y}_i\|_2^2 \tag{2.3}$$

At each training iteration, the model's parameters are updated so that the loss function is minimized using the gradient of the loss on random samples with respect to the model's parameters.

2.1.2 Artificial neural networks

Deep learning is a specific branch of machine learning where the learning model is an artificial neural network. Neural networks are a family of parametric models constructed using differentiable modules that process data and forward it to connected modules. The parameters of the models are adjusted during training using the backpropagation algorithm that we explain below, which allows us to compute the gradient for each parameter to find a local optimum of the loss function. The simplest, and historically the first example of neural networks is the perceptron.

The perceptron Presented in (Rosenblatt, 1958), it is a single layer neural network that inputs a vector of numbers $x \in \mathbb{R}^d$ and outputs a binary class prediction $f_w(x) \in \{0, 1\}$, where w is a vector of real-valued weights. It performs of a dot product between inputs x and weights followed by a non-linear activation function $\psi(x)$, namely the Heaviside step function, defined by $\psi(x) = 1$ if $x > 0$ and 0 otherwise.

$$\hat{y} = f_w(x) = \psi(w.x)$$

To perform the classification task effectively, the parameters w are updated iteratively to find the set of parameters that best explains a given dataset using the parameter update rule for every training pair $(x_i, y_i) \in \mathcal{D}$ following:

$$w \leftarrow w + (y_i - f_w(x_i))x_i$$

as presented in the original paper, however, a similar network with different non linearity can be optimized using backpropagation.

In Fig. 2.1 we show the perceptron's simple architecture.

Multi-layer Perceptron. While the perceptron has a limited representational power, the multi-layer perceptron is a neural network architecture where the input vector is processed using multiple layers of perceptrons and activation functions sequentially as shown in Figure 2.1. Each layer is an affine transformation that can be implemented as a matrix multiplication between the layer weights W and the input X and then adding a bias, followed by a non-linearity σ (Sigmoid, ReLU, etc.):

$$Y = \sigma(W.X + b) \tag{2.4}$$

Multi-layer perceptrons are powerful models for learning complex functions. The universal approximation theorem (Cybenko, 1989) states that any continuous function can be approximated by a neural network with one hidden layer and a large enough number of neurons.

Convolutional neural networks. For high dimensional inputs such as images, it can be impractical to connect all neurons to all neurons in the previous volume. Convolutional neural networks take advantage of the structure of images and use convolutions to process them in a linear and translation covariant way. They are widely used in computer vision to process images efficiently thus reducing the amount of required parameters LeCun et al. (1998); Krizhevsky et al. (2012).

Backpropagation. To train artificial neural networks, backpropagation computes the gradient of the loss function with respect to the parameters of the network. Starting from the last layer and iterating backwards, backpropagation computes the gradient for each layer and combines them using the chain rule:

$$y = f(u), u = g(x) \implies \frac{dy}{dx} = \frac{dy}{du}.\frac{du}{dx}, \text{ where f and g are differentiable functions} \tag{2.5}$$

thus being more efficient than a naive gradient computation for each parameter.

2.1.3 Optimization and learning

Neural network optimization is challenging because of the non-convexity of the functions that we optimize. Multiple optimization techniques have been proposed to improve neural network optimization in order to avoid local minima, and to achieve faster convergence. An important optimization algorithm is batch Stochastic Gradient Descent (batch-SGD).

It is an iterative optimization technique that uses mini-batches of data to estimate an expectation of the gradient of the parameters, rather than the full gradient of the parameters using all available data. It represents a compromise between computing the gradient on the full dataset and on a single sample, by considering a mini-batch at each step. This algorithm can be improved in different ways. For instance, Adam (Kingma and Ba, 2014) is an adaptive learning rate optimization algorithm that utilises both momentum and scaling based on first and second moments of the gradients, usually improving the training speed and accuracy. Using momentum consists of remembering the past update at each iteration and defining the next update as a linear combination between the computed gradient and the previous update, while scaling refers to modulating the learning rate depending on moving averages of the gradients. Multiple techniques are also used to obtain faster convergence, avoid overfitting and local minima such as adding weight decay as an L2 regularization of the model's weights to reduce model overfitting, or using dropout layers (Srivastava et al., 2014). Furthermore, other architectural changes can also have an impact on the optimization landscape such as normalization layers (e.g. BatchNorm (Ioffe and Szegedy, 2015), LayerNorm (Ba et al., 2016)), or using skip connections (He et al., 2015b) to optimize the residual mappings rather than the original one.

2.1.4 Learning deep visual representations

Learning strong image representations has been a longstanding challenge in computer vision research. Images in their rasterised form do not provide a relevant descriptor for evaluating image similarity and for describing image content. Classical image descriptors were proposed to describe the image content using local features associated with distinct image keypoints. SIFT, (Lowe, 2004) propose a method of identifying scale invariant local features in images by locating pixel amplitude extrema. These handcrafted features provide an image descriptor that allows searching and locating objects in images. The evolution of deep learning has allowed the development of deep image features that are learned from data by training a model on a given task. For example, on the ImageNet classification benchmark (Deng et al., 2009b), the impressive performance improvement on classifying a thousand different categories of natural images (Krizhevsky et al., 2012; He et al., 2015b; Simonyan and Zisserman, 2014; Tan and Le, 2019) has provided good image features that are used in multiple other visual tasks. Using pretrained deep image features is crucial for multiple computer vision applications such as evaluating image similarity using cosine distance Zheng et al. (2016) and transfer learning by reusing mid level image representations (Oquab et al., 2014).

The quality of image features can be evaluated from different perspectives. In deep metric learning, the goal is to learn image features that are suited for image retrieval by embedding images of similar objects closer than those of different concepts. The metric learning literature provides efficient methods for retrieving similar images (Teh et al., 2020; Johnson et al., 2017; Jegou et al., 2010). In addition, in few-shot learning, the goal is to obtain features that are able to generalize to new tasks (new classes in case of few-shot classification). Classification based features provide a good baseline for few-shot classification using a nearest neighbor classifier (Wang et al., 2019; Gidaris and Komodakis, 2018; Zhai and Wu, 2018) that sometimes surpass complex meta-learning methods (Ravi and Larochelle, 2017; Chen et al., 2019). Beyond the learning algorithm

complexity, data also plays an important role on the quality of learned features that has been tackled from different perspectives (Triantafillou et al., 2020; Huh et al., 2016).

2.2 Image generation powered creative tools

2.2.1 Generative adversarial learning

Generative adversarial learning is a framework introduced by Goodfellow et al. (2014) for learning to generate samples from data distributions using an adversarial zero-sum game where two networks are trained simultaneously with opposite objectives. A first discriminator network is trained to recognize real from fake or generated data, while a second generator network G is trained to generate samples from a latent code that fools the discriminator network D. This adversarial formulation is particularly useful when it is difficult to formulate a loss as it is the case for realistic image generation. To avoid crafting a loss that evaluates the realism of images, we use a discriminator network to learn this task during training while the generator learns the data distribution. Since the discriminator is differentiable, we can obtain gradients to train the generator.

This is formulated as a two-player minmax game :

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

where p_{data} and p_z are respectively the real data distribution and the noise distribution we sample from.

2.2.2 Adversarial image generation and editing

Since their introduction by (Goodfellow et al., 2014), generative adversarial networks (GANs) have allowed a plethora of derivative works and applications related to realistic image generation. The original GAN paper proposes a framework for learning generative models via an adversarial process, and demonstrates modest but promising performance on image generation on low resolution images.

This adversarial image generation framework happens to be particularly challenging to train especially for generating high resolution realistic images due to the instability of the adversarial game. Among the many possible failure cases, mode collapse represents the setup where the generated samples do not cover the entire data distribution. In the following we present few main directions that shaped the research in the GAN literature.

Initial GAN architectures. Stabilizing the training process of training generative adversarial networks and scaling up the generations has been a constant goal since the introduction of the GAN framework. Multiple papers propose novel GAN architectures, losses and techniques to stabilize the training procedure and produce more realistic samples. Figure 2.2 shows the evolution of the best trainable resolution since the original GAN paper in 2014. We present some of the notable contributions that influenced this domain. (Denton et al., 2015) combines a conditional GAN with a laplacian pyramid approach by upscaling generated images in an additive manner. (Zhang et al., 2017) decomposes the problem into a two-stage sketch-refinement process, by first generating low

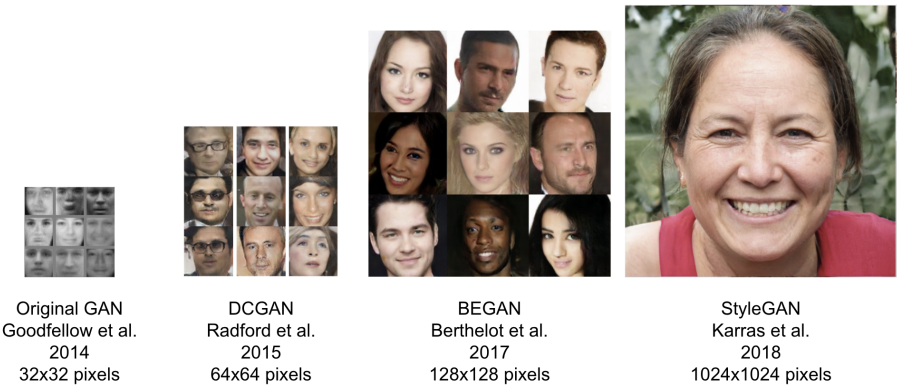


Figure 2.2: Evolution of generated images resolution since the introduction of the generative adversarial networks framework.

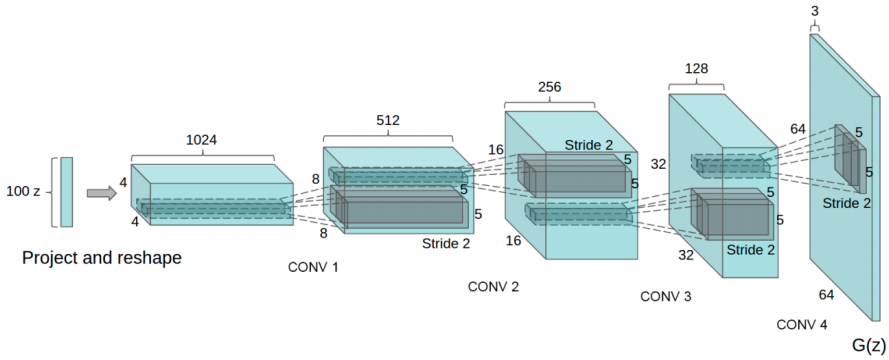


Figure 2.3: Fully convolutional generator architecture from (Radford et al., 2016) for adversarial image generation.

resolution images in a first stage and then adding compelling details in a second one. (Radford et al., 2016) proposed a deep convolutional architecture for the generator and the discriminator able to generate samples at 64×64 pixels resolution, the generator architecture is shown in Fig. 2.3. (Odena et al., 2017) proposes a variant with an auxiliary classifier GAN for the discriminator to predict the class label in addition to the probability of the input being real. While not being very different, this simple architecture modification and data supervision produces better results and appears to slightly stabilize the training. (Wang and Gupta, 2016) introduces a two-component approach factoring image generation: structure representing the underlying 3D model and style being the texture mapped onto structure. They demonstrate its potential in learning RGB D representations in an unsupervised manner.

Techniques for stable training of GANs. In addition, another line of work has focused on proposing more or less principled techniques for training generative adversarial networks with better convergence guarantees. (Salimans et al., 2016) present numerous architectural features and training procedures to train generative adversarial networks. For example, feature matching as a new objective for the generator to match activations on an intermediate layer of the discriminator, and historical averaging of the parameters. Some papers have explored the use of different loss metrics to replace the Jensen-Shannon di-

vergence used in original GAN formulation. (Arjovsky et al., 2017; Arjovsky and Bottou, 2017) propose WGAN an alternative training method of GANs using the Earth-Mover distance or Wasserstein-1 distance between the distributions of real and generated samples, which was later improved with a gradient penalty to penalize the norm of the gradient of the discriminator with respect to its input to avoid unstable behavior. (Heusel et al., 2017) introduces a two time-scale update rule to further stabilize the training and outperform the quality of the generations. In addition to the Wasserstein GAN training loss, (Mao et al., 2017) propose the use of L2 distance to avoid the vanishing gradients that can come from using the sigmoid cross entropy loss in the original GAN formulation. (Berthelot et al., 2017) propose to use an autoencoder as a discriminator and optimize a lower bound of the Wasserstein distance between auto-encoder loss distributions on real and fake data. They introduce an additional hyperparameter to control the equilibrium between the generator and discriminator. (Mescheder et al., 2018) propose to penalize the gradient of the discriminator on the real data alone and prove global convergence for simple settings where the true data distribution is a single point or a single Gaussian distribution which was then generalized by (Sun et al., 2020). (Miyato et al., 2018) introduces spectral normalization, a lightweight and very efficient technique that helps control the Lipschitz constant of the discriminator by literally constraining the spectral norm of each layer. Finally, (Kurach et al., 2019) and (Lucic et al., 2018) provide large scale studies comparing these different losses, architectures and normalization techniques in a thorough empirical analysis.

High resolution image generation. The presented techniques allowed the training of more stable image generation models able to train at high resolution images. The first paper able to generate realistic 1024×1024 images was (Karras et al., 2017) by introducing a progressive training scheme that starts training at 4×4 then doubles the resolution until the final 1024×1024 pixels. However, this requires many training tricks such as keeping a moving average of the generator's weight used for inference, using an equalized learning rate among other techniques. This architecture was further simplified and improved in (Karras et al., 2019) with the use of an adaptive instance normalization module to incorporate style vectors at different levels of the generation, leading to a generator with controllable coarse and fine features. Finally, (Brock et al., 2019) train generators that output 1024×1024 samples by adopting orthogonal regularization in a class-conditional image synthesis setup using a self-attention based architecture (Zhang et al., 2019).

Evaluating generated images Evaluation of image generation models is a long standing challenge. Several automatic metrics have been proposed to alleviate the need of human annotation and to have a more quantitative objective comparison.

(Salimans et al., 2016) propose the Inception score, an image quality assessment score that uses a classification Inception v3 network pre-trained on ImageNet to assess the entropy of the distribution of generated samples taking into account both the variety of generated images and the entropy of their classification as shown in Eq. 2.6. This score has been further improved to account for the distribution of real images in (Zhou et al., 2017).

$$\text{IS}(G) = \exp\left(\mathbb{E}_{x \sim p_g} \, D_{KL}(\, p(y|x) \parallel p(y) \,) \right), \tag{2.6}$$

where $x \sim p_g$ indicates that x is an image sampled from p_g, $D_{KL}(p\|q)$ is the KL-divergence between the distributions p and q, $p(y|x)$ is the conditional class distribution, and $p(y) = \int_x p(y|x)p_g(x)$ is the marginal class distribution. The exp in the expression is there to make the values easier to compare.

(Heusel et al., 2017) introduce the Fréchet Inception Distance FID which captures the similarity of generated images to real ones better than the inception score, by using the first two moments (mean $\boldsymbol{m}$ and covariance $\boldsymbol{C}$) of the activations of a pre-trained Inception v3 network, $(\boldsymbol{m}_g, \boldsymbol{C}_g)$ and $(\boldsymbol{m}_r, \boldsymbol{C}_r)$ for generated and real images respectively, as shown in Eq 2.7.

$$d^2((\boldsymbol{m}_g, \boldsymbol{C}_g), (\boldsymbol{m}_r, \boldsymbol{C}_r)) = \|\boldsymbol{m}_g - \boldsymbol{m}_r\|_2^2 + \mathrm{Tr}\big(\boldsymbol{C}_g + \boldsymbol{C}_r - 2\big(\boldsymbol{C}_g\boldsymbol{C}_r\big)^{1/2}\big) \tag{2.7}$$

It is shown to better correlate with generation quality.

2.2.3 Creative image editing with generative models

Image generation models allow high-level image editing beyond the pixel level which is time-consuming and requires professional and artistic skills. There are multiple approaches to image editing with generative models. First, using image to image translation models that maps an image from its original domain to a new one. Second, using style transfer, to apply a style from an given image. Third, by manipulating images in a latent space of a generative model. Fourth, using attribute based image editing. We discuss a few notable papers and applications in each image editing approach.

Image to image translation. Image to image translation relies on using a neural network that inputs a 2D image and outputs another one with the desired edits such as U-Nets (Ronneberger et al., 2015). The network thus learns a mapping from a source domain to another such as in (Isola et al., 2017) with applications to colorization (Zhang et al., 2016), semantic segmentation (Long et al., 2015), generating images from semantic maps (Park et al., 2019b; Wang et al., 2018a), image inpainting (Yu et al., 2019), super-resolution (Ledig et al., 2017), etc. This mapping can be learned using only domain level supervision without paired data samples (Zhu et al., 2017a; Liu et al., 2017; Kim et al., 2017) and across more than multiple domains (Choi et al., 2018). These works use mainly a combination of the adversarial loss to make the generated image indistinguishable from real ones and a cycle consistency loss through reconstruction of the original image given the back-translated one. These unsupervised image to image translation methods have been successfully applied to artistic applications by mapping real photographs to a domain of Monet paintings for example or inversely to visualize how paintings would look in real life (Zhu et al., 2017a). Similarly, in Fig. 2.4 an artist uses image to image translation to map a 2D drawing into a geometric 3D form.

Style transfer. A similar application is style transfer, where the goal is to change the style of an image given an input style from another one, for example, applying the style from a Van Gogh painting to a real world image. There have been multiple works on style transfer (Gatys et al., 2016; Ulyanov et al., 2016; Huang and Belongie, 2017). These

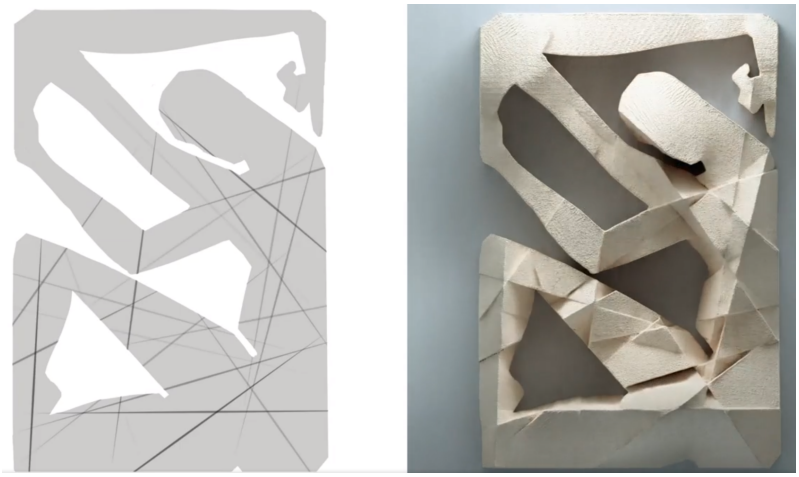


Figure 2.4: Using an image to image translation network (Wang et al., 2018a), an artist explores abstract, geometric 3D form through a simple drawing interface (Eaton, 2019).

methods rely on an image representation of content and style obtained using deep neural network's outputs at different layers as introduced in (Gatys et al., 2016) and shown in Fig. 2.5. The content representations are the output layer activations themselves, while the style representations are obtained as correlations between different features in different layers.

Image manipulation in a latent space Generative models are not only interesting for their ability to generate samples from a learned distribution similar to the training one, but they also provide a continuous manifold of images suited for image editing. One commonly demonstrated property of generative models is the interpolation between latent image representations, leading to a smooth transition sequence of generated images that lie on the natural images manifold. Several works propose methods to use generative models for image editing. For example, Zhu et al. (2016) propose a framework for image editing that allows coloring, sketching and warping while constraining the output to lie on the learned manifold of a generative neural network via optimization. However, most of these methods require a way to project real images on the latent space of a trained GAN, also known as inverting a generative network or latent space embedding. There are two existing approaches to embed instances from the image space into the latent space. First, is by learning an encoder network (Kingma and Welling, 2014; Donahue et al., 2016). Second, is using optimization from a random initial latent code (Zhu et al., 2016; Creswell and Bharath, 2018; Abdal et al., 2019). While the first method is fast by only requiring a forward pass through the encoder network, the optimization based approach leads to a more general and stable solution (Abdal et al., 2019).

In addition to latent space embedding of real images, other works explore different methods to find meaningful directions for applying edits using trained generative networks. Bau et al. (2019) present a method for analyzing GANs by identifying groups of interpretable units that are related to object concepts. Härkönen et al. (2020) identify important latent directions based on Principal Component Analysis (PCA) leading to a simple but powerful way to manipulate images with existing GANs as shown in Figure 2.6. Similarly, Collins et al. (2020) use clustering techniques on the activations of

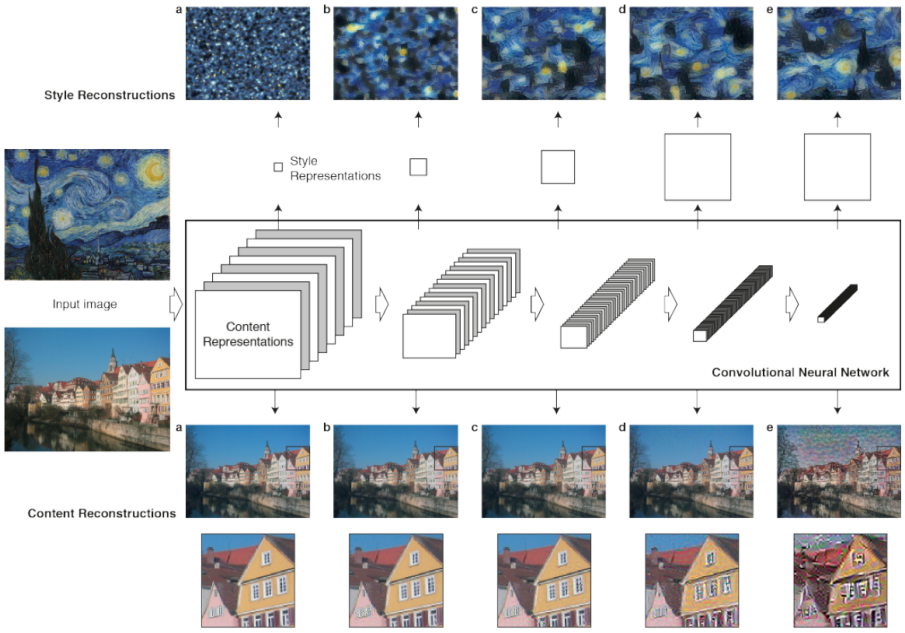


Figure 2.5: Using intermediate representations from a pretrained VGG network (Simonyan and Zisserman, 2014) to extract content and style information at different layers. These content and style features are used in style transfer and to define the perceptual loss (Johnson et al., 2016).

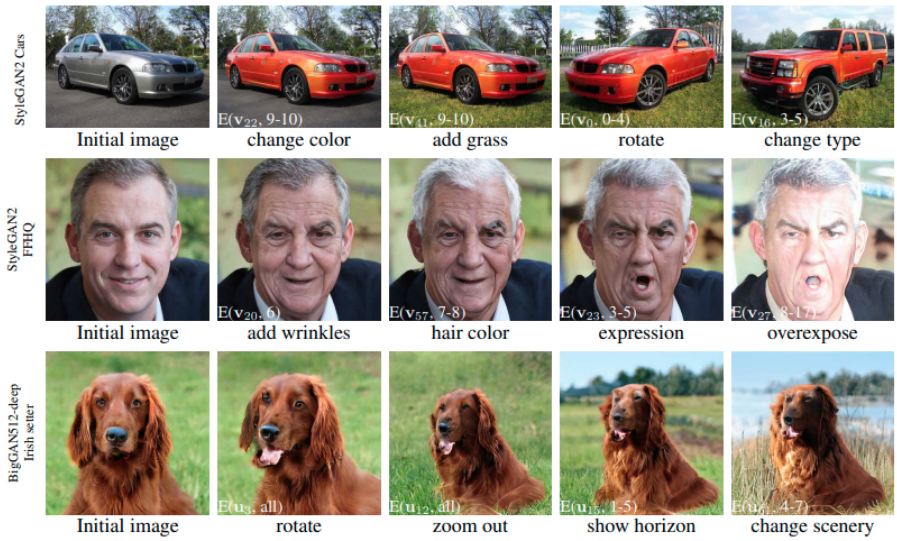


Figure 2.6: Sequence of image edits obtained with discovered controls using a pre-trained generative model. (Härkönen et al., 2020)

generative models to identify clusters corresponding to semantic objects and parts and use them to perform local edits. Finally, other methods are used to identify interpretable directions in GAN latent space (Voynov and Babenko, 2020; Cherepkov et al., 2020).

Attribute based image editing One way to enforce the learning of a particular attribute editing is by using additional inputs to condition the generative model. While it would be simple to train in a supervised case if images with each attribute were available, many methods aim to disentangle image content from controllable attributes as in (Perarnau et al., 2016; Lample et al., 2017; He et al., 2019b). For example, (Lample et al., 2017) learns an attribute invariant latent representation using an encoder-decoder architecture and adversarial training. Thus, changing the attribute manually effectively translates into the corresponding semantic edit.

2.3 Vector and layered image generation

Most approaches to image generation consider generating raster images through the use of convolutional and upsampling layers, thus leading to fixed resolution output. One way to overcome the resolution challenge of generative models is to create vector images that are resolution independent. While the image vectorization literature is very important, only few works consider generating vector images using deep learning techniques. In the following we present notable image vectorization methods and sequential stroke based image generation methods. Finally, we introduce recent works on generating and representing vector images with deep neural networks using implicit functions.

2.3.1 Image vectorization algorithms and representations

Many vector representations have been proposed to represent an image into a vector way using simple shapes with uniform color or linear gradient or more complex and mathematical representation using triangular patches with Bézier curves (Xia et al., 2009) or rectangular patches with Ferguson patches (Sun et al., 2007). Multiple vectorization algorithms have been proposed to convert a raster image into a vector one (Richardt et al., 2014; Favreau et al., 2017; Liao et al., 2012; Orzan et al., 2008). While most region partitioning approaches consider images as one layer with hard boundaries between regions, others use a multi-layer approach to decompose the image into soft segments (Aksoy et al., 2017; Tai et al., 2007). Scalable Vector Graphics (SVG) is a widely used format for encoding vector images as XML files of basic shapes or paths, with colors, transforms and other information. It is based on simple shapes such as circles, rectangles, lines etc. or more complex Bézier curves that allow the creation of more complex paths. Generating images from their SVG representation has been framed recently in (Carlier et al., 2020) using a Transformer architecture for sequential generation to generate the SVG drawing commands. Their approach provides a new deep learning based approach for vector image generation that allows interpolation between vector images for example. Lopes et al. (2019) also present a learned representation for scalable vector graphics through a sequential generative model. Using an inverse graphics approach, their model consists of a variational autoencoder (VAE) (Kingma and Welling, 2014) and an autoregressive SVG

Figure 2.7: Sample images generated by (Mellor et al., 2019). showing discovered visual that resemble those made by children and novice illustrators.

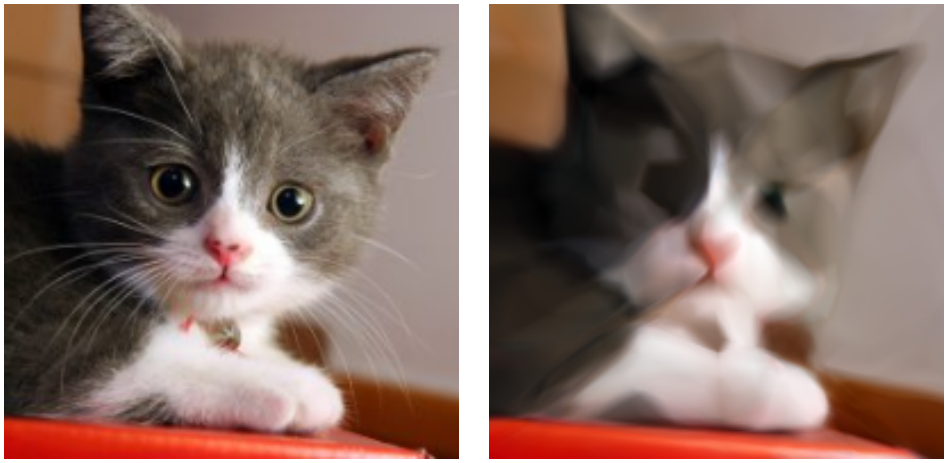
decoder. They also exploit the learned latent space for style propagation on a dataset of fonts.

2.3.2 Sequential stroke based image generation

Image generation can be framed as generating a sequence of commands that build the image from an empty canvas. In fact, it is the most similar way to the human way of drawing and creating vector images.

(Ha and Eck, 2017) have been the first ones to use a recurrent neural network to construct stroke-based drawings of common objects. Reinforcement learning based approaches have been developed to learn image generation as a sequence of drawing commands. (Xie et al., 2013) present an RL approach for oriental ink painting by interacting with a simulated environment of smooth and natural strokes. They first learn the reconstruction of simple brush strokes before generalizing it to global image reconstruction. (Ganin et al., 2018) also represent images as high level programs and train an RL agent to generate programs that are executed by an external drawing engine. This adversarial reinforced learning allows the use of any external drawing engine, however leads to a slower training. It has been further improved in (Mellor et al., 2019) showing that generative agents can learn to produce images with a degree of visual abstraction as shown in Figure 2.7.

(Zheng et al., 2018) contributes a model-based method to approximate the external non-differentiable environment with neural networks, leading to a faster convergence on several datasets. Similarly, (Huang et al., 2019) builds a differentiable neural renderer to render strokes, allowing the use of model-based deep reinforcement learning algorithms. Inspired by (Ha and Schmidhuber, 2018), (Nakano, 2019) learn a generative model for a given painting environment to simulate the same outputs given similar action inputs, leading to a fully differentiable architecture that can be trained with regular adversarial methods.



Original image Reconstructed image

Figure 2.8: Overfitting a Multi-Layer Perceptron on a single image taking as input only pixel coordinates. Using the demo from (Karpathy, 2015)

2.3.3 Implicit functions for resolution independent image generation

There have been significant efforts for finding suitable data representations for modeling images and 3D information. Implicit functions have recently gained a lot of attention as functions of the spatial dimensions represented by a multi-layer perceptron MLP. In the image domain, using implicit functions to represent an image dates back to (Stanley, 2007) with the introduction of CPPN (Compositional Pattern Producing Networks). These networks were in fact a mapping from the pixel coordinates to the pixel RGB colors. Several works also have explored overfitting a single image with an MLP and representing images in a resolution independent manner (Ha, 2016a,b; Karpathy, 2015) as shown in the Figure 2.8.

In fact, in 3D recent works have demonstrated the powerful capacity of MLPs to represent complex 3D scenes and reconstruct 3D objects. Groueix et al. (2018) use an MLP to deform points on a 2D surface into a 3D one to reconstruct parts of objects. (Park et al., 2019a) represent a 3D scene as an implicit signed distance function modeled by an MLP, recovering the surface of the objects as the iso-surfaces of value zero. More recently, (Mildenhall et al., 2020) propose a powerful 3D scene representation for generating multiple scene views given camera position and orientation. (Tancik et al., 2020) propose a Fourier transform of the low dimensional spatial coordinates to allow these MLP to learn high frequency details. Similarly, (Sitzmann et al., 2020) makes use of periodic activation functions as a way to increase the representation power of the MLP representing complex natural signals from spatial dimensions. Finally, multiple works have used MLP type architectures for learning resolution independent image generators (Anokhin et al., 2020). (Lin et al., 2021) proposes a model for two-stage image generation with infinite resolution; a first structure synthesizer based on an implicit function, and a texture synthesizer based on a fully convolutional StyleGAN2 generator that renders local details.

2.4 Machine learning and creativity

While great strides have been made into the machine's capability of simulating human intelligence aspects (perception, understanding, planning, reasoning, etc.), creativity remains probably the most difficult one to tackle. There have been multiple efforts for defining creativity. According to (Boden et al., 2004), creativity is the "ability to come up with ideas or artifacts that are new, surprising, and valuable".

There are many works in the computational creativity domain, fueled by the evolution of generative models. Using different modalities, machine learning researchers explore diverse creative applications. In the musical domain, multiple efforts are made on music generation, artistic style transfer of melodies from one instrument or domain to another, etc (Hadjeres and Pachet, 2017; Briot et al., 2017). In the linguistic domain, many works leverage the advances of natural language processing for automated story generation and automated pun generation (Ritchie, 2005; Yu et al., 2018). In the visual domain, creativity focused works have tackled visual conceptual blending (Cunha et al., 2020), generating novel artwork (Elgammal et al., 2017), neural style transfer (Gatys et al., 2016), drawing and sketching (Parikh and Zitnick, 2020) etc. An important aspect of computational creativity is the collaboration and interaction between human artists and the machine, in what is usually refered to as co-creativity (Kantosalo and Takala, 2020).

The developments of generative models have shown the capacity of machines to generate realistic and aesthetic content. The evaluation of these generative models remains a challenge to pick the most interesting generations.

In the next subsections, we discuss two main creative applications of generative models; namely fashion image generation and visual metaphor creation.

2.4.1 Creative fashion generation

Can artificial intelligence algorithms, like fashion designers, create new and creative fashion collections? Most efforts of the machine learning and computer vision research community on fashion have focused on understanding the clothing trends from a large collection of images (Yamaguchi et al., 2012; Liu et al., 2016; Al-Halah et al., 2017).

Recent works have tackled the problem of fashion image generation from a perspective of enabling virtual try-on, by generating images of human models in controllable outfits (Zhu et al., 2017c; Lassner et al., 2017). Rostamzadeh et al. (2018) proposes a new fashion dataset with large resolution images and textual annotations to enable text-to-image image generation.

While these models focus on generating realistic rendering of the person wearing the garment, they focus most of the model's capacity on learning semantic segmentation of the piece of clothing on the person's body, and do not contribute to generating creative garments. Date et al. (2017) instead uses neural style transfer algorithm (Gatys et al., 2016) to synthesize new custom clothes based on user preferences.

2.4.2 Visual conceptual blending

Visual metaphors or conceptual blends are composite images obtained by juxtaposing objects from different contexts that share a given analogy. They are a powerful way

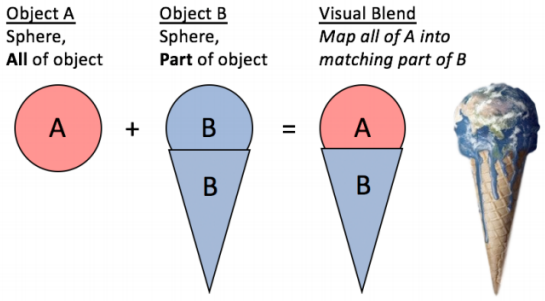


Figure 2.9: An illustration of a simple design pattern for constructing visual blends from (Chilton et al., 2019)

of communication in marketing, art and public service announcements, and have been studied thoroughly in the marketing literature (Gkiouzepas and Hogg, 2011; Phillips and McQuarrie, 2004; Jeong, 2008). For example, Chilton et al. (2019) highlight that advertisements with visual metaphors are much more persuasive than those with plain images or text alone when conveying a message. Metaphors have been first and foremost studied in the linguistic research community with a focus on detecting figurative use of words (Mohler et al., 2013; Hovy et al., 2013; Tsvetkov et al., 2014). Forceville (1994) proposes a theory of the mechanics underlying visual metaphors. When viewers encounter a visual metaphor, they recognize an object, but also quickly identify something odd about it that leads them to construct a figurative interpretation of the composition. However, some visual compositions can be hard to decode which sets a tradeoff between the complexity of the underlying message to convey and the clarity of the composition.

The creation of these composite images is a time consuming process that includes three main phases; a brainstorming phase to find the idea of the analogy, an image search phase to find the best suitable image to illustrate the idea and the last phase of performing the image blending. Chilton et al. (2019) describe a flexible and collaborative workflow for brainstorming and synthesizing visual blends. It is part of a research direction for leveraging collaborative efforts for ideation and brainstorming (Siangliulue et al., 2015, 2016)

When interpreting visual blends, the human visual system uses many different visual features at different stages to recognize an object including its 3D shape, silhouette, depth, color and details (Birney and Sternberg, 2011). Fig. 2.9 describes a simple shape mapping design pattern for obtaining a visual blend.

Xiao et al. (2015) propose Vismantic, a semi-automatic system for generating composite images from pairs of a subject word and a message word. It searches for candidate images, which are filtered manually before being combined using juxtaposition, fusion or replacement implemented using a combination of object extraction, inpainting and texture transfer. Karimi et al. (2018) proposes a system for creative ideation through the exploration of conceptual shifts using sketch similarity to find similar sketches from different categories. More recently, Cunha et al. (2020) provides a roadmap for generating visual blends. They highlight important steps for the conceptualisation of the generated composite by grounding it using perceptual, naming/homophones or affordance attributes.

Chapter 3

Original image generation

3.1 Abstract

Can an algorithm create original and compelling images to serve as an inspirational assistant ? To help answer this question, we design and investigate two different image generation models associated with creativity in fashion image generation and visual conceptual blending domains. We particularly explore how to adapt image generation methods to create original, surprising and realistic images by crafting novel loss functions for generative models as in Fig. 3.1 (top row) or by searching for interesting object compositions using visual similarity and semantics, and performing blending as in Fig. 3.1 (bottom row). A key challenge of this study is the tradeoff between novelty and realism of the generated images and the evaluation of their quality, hence we put together an evaluation protocol associating automatic metrics and human experimental studies to assess the novelty, the likability and realism of generated images.

Keywords: artistic image generation, generative adversarial networks, image blending.

3.2 Introduction

Artificial Intelligence (AI) research has been making huge progress in the machine's capability of human level understanding across the spectrum of perception, reasoning and planning (He et al., 2017; Andreas et al., 2016; Silver et al., 2016). Another key yet still relatively understudied direction is creativity where the goal is for machines to generate original items with realistic, aesthetic and/or thoughtful attributes, usually in artistic contexts. We can indeed imagine AI to serve as inspiration for humans in the creative process and also to act as a sort of creative assistant able to help with more mundane tasks, especially in the digital domain. Previous work has explored writing pop songs (Briot et al., 2017), imitating the styles of great painters (Gatys et al., 2016; Dumoulin et al., 2017) or doodling sketches (Ha and Eck, 2018) for instance. However, it is not clear how *creative* such attempts can be considered since most of them mainly tend to mimic training samples without expressing much originality. Creativity is a subjective notion that is hard to define and evaluate, and even harder for an artificial system to optimize for.

In this work, we tackle two different creative applications, namely, fashion image

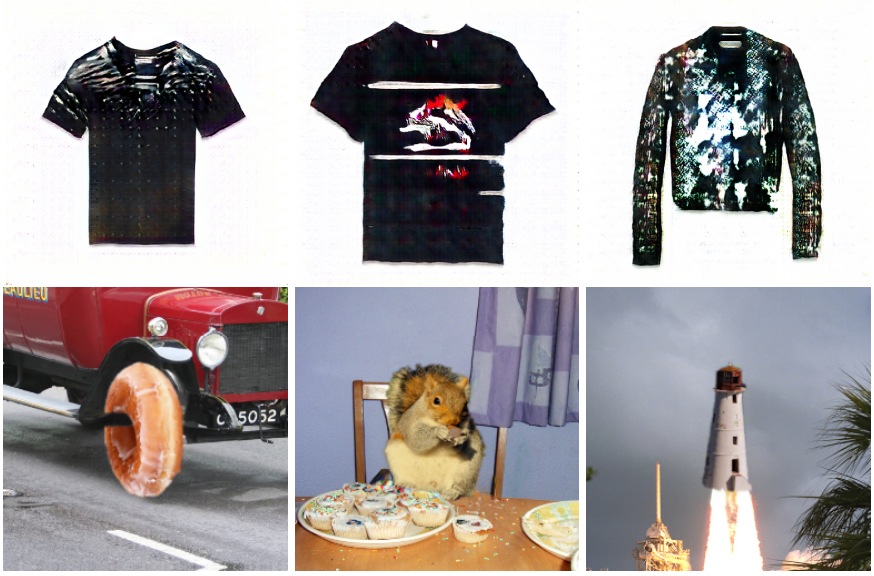


Figure 3.1: Top row: Image samples from our trained generative model with novelty losses leading to realistic and creative 512×512 fashion images. Bottom row: Examples of obtained visual analogies using our search and composition method. From left to right: Bagel/Wheel, Squirrel/Boy, Lighthouse/Rocket.

generation and visual blends creation using generative networks and image blending. We particularly explore how computer vision algorithms can help augment artists workflow by suggesting interesting and novel images. Using recent advances in image generation and understanding, algorithms have the potential to create inspiring new images in a co-creative approach guided by the artist's intent.

First, we study how AI can generate creative samples for fashion. Based on a collaboration with a fashion industry that provides a collection of high quality images of garments, we are interested in exploring the potential of the advances of generative models into inspiring designers. Fashion image generation opens the door for breaking creativity into design elements (shape and texture in our case), which is a novel aspect of our work. In contrast to most generative models works, the creativity angle we introduce makes us go beyond replicating images seen during training. More specifically, this work explores various architectures and losses that encourage GANs to deviate from existing fashion styles covered in the training dataset, while still generating realistic pieces of clothing without needing any image as input at test time. To the best of our knowledge, this work is the first attempt at incorporating creative fashion generation by explicitly relating it to its design elements.

Second, we propose an algorithm for suggesting surprising image compositions based on image similarity with deep features and object semantics. Visual blends and metaphors are very commonly used in advertising, news and art (Gkiouzepas and Hogg, 2011; Phillips and McQuarrie, 2004; Forceville, 1994). In fact, Jeong (2008) shows that visual advertisements are much more persuasive when based on visual metaphors. They are usually used to challenge and trigger thoughts or simply entertain. While they are sometimes hard to decode (Petridis and Chilton, 2019), it is even more challenging to obtain image compositions that have a strong conceptual grounding. The collage creation pro-

cess can be tedious as it not only requires the artist to find a new interesting analogy idea but it also involves a lengthy process of image search and image blending. In this work, we leverage recent visual object retrieval and image composition advances to improve the collage creation experience by suggesting varied combinations given a selected input object. Our approach grounds the composition using perceptual features and semantic ones. Our contributions are two-fold:

- We propose a novelty loss on image generation of fashion items with a specific conditioning of texture and shape, learning a deviation from existing ones. Our best models manage to generate realistic images with high resolution 512×512 (Fig. 3.1, top row) using a relatively small dataset (about 4000 images). More than 60% of our generated designs are judged as being created by a human designer while also being considered original, showing that an AI could offer benefits serving as an efficient and inspirational assistant.

- We design a foreground image search and blending method adapted to the real-time setting that suggests interesting foreground combinations based on the local features similarity with the query foreground object (Fig. 3.1, bottom row). Our simple copy pasting model performs geometric and color adaptations to the foreground object in addition to image inpainting. Our composition network is easier to train than competing methods, relying solely on supervised training on synthetic images, but proves to be robust and effective. In particular, we experimentally study the trade-off between the quality of the composite image and the surprising aspect of the composition.

3.3 Original fashion image generation

3.3.1 Related work

There has been a growing interest in generating images using convolutional neural networks and adversarial training, given their ability to generate appealing images unconditionally, or conditionally like from text, class labels, and for paired and unpaired image translations (Zhu et al., 2017a). GANs (Goodfellow et al., 2014) allow image generation from random numbers using two networks trained simultaneously: a generator is trained to fool an adversarial network by generating images of increasing realism. The initial resolution of generated images was 32×32. From this seminal work, progress in generating higher resolution images has been achieved, using a cascade of convolutional networks (Denton et al., 2015) and deeper network architectures (Radford et al., 2016). The introduction of auxiliary classifier GANs (Odena et al., 2017) then consisted of adding a label input in addition to the noise and training the discriminator to classify the synthesized 128×128 images. The addition of text inputs (Reed et al., 2016; Zhang et al., 2017) allowed the generative network to focus on the area of semantic interest and generate photo-realistic 256×256 images. Recently, impressive 1024×1024 results were obtained using a progressive growth of the generator and discriminator networks (Karras et al., 2017), by training models during several days.

Neural style transfer methods (Gatys et al., 2016; Johnson et al., 2016) opened the door to the application of existing styles on clothes (Date et al., 2017), the difference

with generative models being the constraint to start from an existing image in input. Isola et al. (2017) relaxes this constraint partly by starting from a binary image of edges, and presents some generations of handbags images. Another way to control the appearance of the result is to enforce some similarity between the input texture patch and the resulting image (Xian et al., 2018). Using semantic segmentation and large datasets of people wearing clothes, Zhu et al. (2017c); Lassner et al. (2017) generate full bodies images and are conditioning their outputs either on text descriptions, color or pose information. In this work, we are interested in exploring the creativity of generative models and focus on presenting results using only random or shape masks as inputs to leave freedom for a full exploration of GANs creative power.

3.3.2 Novelty losses

Architecture	Novelty Loss	Design Elements
DCGAN (Radford et al., 2016)	CAN (Elgammal et al., 2017)	Texture (ours)
StyleGAN (ours)	MCE (ours)	Shape (ours)
StackGAN		Shape & Texture (ours)

Table 3.1: Dimensions of our study. We propose fashion image generation models that differ in their architecture and their novelty loss that encourages the generations to deviate from existing shapes, textures, or both.

Table 3.1 summarizes our models and losses exploration. Let us consider a dataset $\mathcal{D}$ of N images. Let x_i be a real image sample and z_i a vector of n of real numbers sampled from a normal distribution. In practice $n = 100$.

GANs. As in Goodfellow et al. (2014); Radford et al. (2016), the generator parameters θ_G are learned to compute examples classified as real by D:

$$\min_{\theta_G} \mathcal{L}_{G \text{ real/fake}} = \min_{\theta_G} \sum_{z_i \in \mathbb{R}^n} \log(1 - D(G(z_i))).$$

The discriminator D, with parameters θ_D, is trained to classify the true samples as 1 and the generated ones as 0:

$$\min_{\theta_D} \mathcal{L}_{D \text{ real/fake}} = \min_{\theta_D} \sum_{x_i \in \mathcal{D},\ z_i \in \mathbb{R}^n} -\log D(x_i) - \log(1 - D(G(z_i))).$$

GANs with auxiliary classification loss. Following Odena et al. (2017), we use shape and texture labels to learn additional shape and texture classifiers in the discriminator. Adding these labels improves over the plain model and stabilizes the training for larger resolution. Let us define the texture and shape integer labels of an image sample x by $\hat{t}$ and $\hat{s}$ respectively. We are adding to the discriminator network either one branch for texture D_t or shape D_s classification or two branches for both shape and texture classification $D_{\{t,s\}}$. In the following section, for genericity, we employ the notation $D_{b,k}$, designating

the output of the classification branch b for class $k \in \{1, ..., K\}$, where K is the number of different possible classes of the considered branch (shape or texture). We add to the discriminator loss the following classification loss:

$$\mathcal{L}_D = \lambda_{D_r}\mathcal{L}_{D \text{ real/fake}} + \lambda_{D_b}\mathcal{L}_{D\text{classif}} \text{ with } \mathcal{L}_{D \text{ classif}} = -\sum_{x_i \in \mathcal{D}} \log\left(\frac{e^{D_{b,\hat{c}_i}(x_i)}}{\sum_{k=1}^{K} e^{D_{b,k}(x_i)}}\right),$$

where $\hat{c}_i$ is the label of the image x_i for branch b.

Losses encouraging original generations. Because GANs learn to generate images very similar to the training images, we explore ways to make them deviate from this replication by studying the impact of an additional loss for the generator. The final loss of the generator that is optimized jointly with $\mathcal{L}_D$ is:

$$\mathcal{L}_G = \lambda_{G_r}\mathcal{L}_{G \text{ real/fake}} + \lambda_{G_e}\mathcal{L}_{G \text{ novelty}}$$

We explored different losses for novelty that we detail in this section. First, we employ binary cross entropies over the adversarial network outputs as in CANs. Second, we suggest employing the multi-class cross entropy (MCE) as a natural way to normalize the penalization across all classes.

Binary cross entropy loss (CAN (Elgammal et al., 2017)). Given the adversarial network's branch D_c trained to classify different textures or shapes, we can use the CAN loss $\mathcal{L}_{\text{CAN}}$ as $\mathcal{L}_{G \text{ novelty}}$ to create a new style that confuses D_b:

$$\mathcal{L}_{\text{CAN}} = -\sum_i \sum_{k=1}^{K} \frac{1}{K}\log(\sigma(D_{b,k}(G(z_i)))) + \frac{K-1}{K}\log(1 - \sigma(D_{b,k}(G(z_i)))), \quad (3.1)$$

where σ denotes the sigmoid function.

Multi-class Cross Entropy loss. We propose to use $\mathcal{L}_{G \text{ novelty}}$ the Multi-class Cross Entropy (MCE) loss between the class prediction of the discriminator and the uniform distribution. The goal is for the generator to make the generations hard to classify by the discriminator.

$$\mathcal{L}_{\text{MCE}} = -\sum_i \sum_{k=1}^{K} \frac{1}{K}\log\left(\frac{e^{D_{b,k}(G(z_i))}}{\sum_{q=1}^{K} e^{D_{b,q}(G(z_i))}}\right) = -\sum_i \sum_{k=1}^{K} \frac{1}{K}\log(\hat{D}_i), \quad (3.2)$$

where $\hat{D}_i$ is the softmax of $D_b(G(z_i))$. In contrast to the CAN loss that treats every classification independently, the MCE loss should better exploit the class information in a global way.

3.3.3 Generation architectures

We experiment using three architectures : modified versions of the DCGAN model (Radford et al., 2016), StackGANs (Zhang et al., 2017) with no text conditioning, and our

proposed styleGAN which decomposes image generation into a 2-step shape and texture generation.

DCGAN. The DCGAN generator's architecture was only modified with more layers to output 256×256 or 512×512 images. The discriminator architecture also includes these modifications and contains additional classification branches depending on the employed loss function.

Unconditional StackGAN. Conditional StackGAN (Zhang et al., 2017) has been proposed to generate 256×256 images conditioned on captions. The method first generates a low resolution 64×64 image conditioned on text. Then, the generated image and the text are tiled on a $16 \times 16 \times 512$ feature map extracted from the 64×64 generated image to compute the final 256×256 image. We adapted the architecture by removing the conditional units (i.e. the text) but realized that it did not perform well for our application. The upsampling in Zhang et al. (2017) was based on nearest neighbors which we found ineffective in our setting. Instead, we first generate a low resolution image from normal noise using a DCGAN architecture (Radford et al., 2016), then conditioning on it, we build a higher resolution image of 256×256 with a generator inspired from the pix2pix architecture with 4 residual blocks (Isola et al., 2017). The upsampling was performed using transposed convolutions.

StyleGAN[1]: Conditioning on masks. To grant more control on the design of new items and get closer to standard fashion processes where shape and texture are handled by different specialists, we also introduce a model taking binary masks representing a desired shape in input. Since even for images on white background a simple threshold fails to extract accurate masks, we compute them using the graph based random walker algorithm (Grady, 2006).

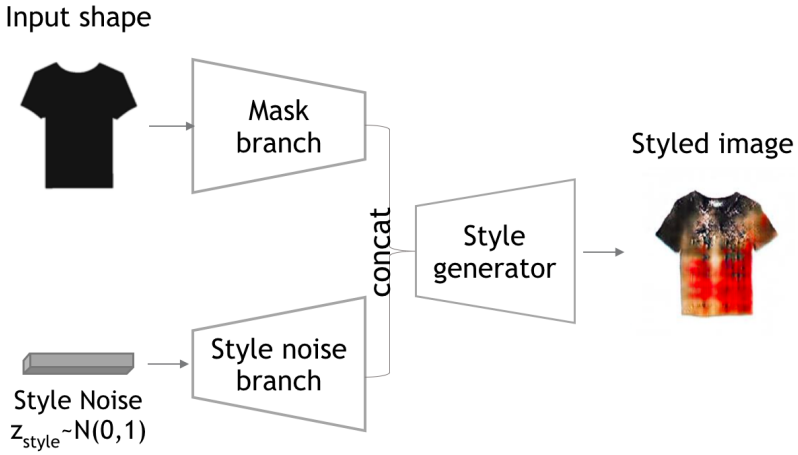


Figure 3.2: From the segmented mask of a fashion item and different random vector z, our StyleGAN model generates different styled images.

In the StyleGAN model, a generator is trained to compute realistic images from a mask input and noise representing style information (see Fig. 3.2), while a discriminator is trained to differentiate real from fake images. We use the same discriminator architecture

[1]this work was done prior to the publication of the similarly named Karras et al. (2019)

as in DCGAN with classifier branches that learn shape and texture classification on the real images on top of predicting real/fake discrimination.

Previous approaches of image to image translation such as pix2pix (Isola et al., 2017) and CycleGAN (Zhu et al., 2017a) create a deterministic mapping between an input image to a single corresponding one, i.e. edges to handbags for example or from one domain to another. This is due to the difficulty of training a generator with two different inputs, namely mask and style noise, and making sure that no input is being neglected. In order to allow sampling different textures for the same shape as a design need, we avoid this deterministic mapping by enforcing an additional ℓ_1 loss on the generator:

$$\mathcal{L}_{rec} = \sum_i \sum_{p \in \mathcal{P}} |G(m_{i,p}, z_i = 0) - m_{i,p}|, \tag{3.3}$$

where $m_{i,p}$ denotes the mask of a sample image x_i at pixel p, and $\mathcal{P}$ denotes the set of pixels of m_i. This loss encourages the reconstruction of the input mask in case of null input z (i.e. zeros) and hence ensures the impact of the mask in the generations.

3.3.4 Experiments and results

After presenting our datasets, we describe some automatic metrics we found useful to sort models in a first assessment, present quantitative results followed by our human experiments that allow us to identify the best models.

Dataset Unlike similar work focusing on fashion item generation (Lassner et al., 2017; Zhu et al., 2017c), we choose datasets which contain fashion items in uniform background allowing the trained models to learn features useful for creative generation without generating wearers face and the background. We augment each dataset five times by jittering images with random scaling and translations.

The RTW dataset. We have at our disposal a set of 4157 images of Ready To Wear (RTW) items of size 3000×3760. Each piece is displayed on a uniform white background. These images are classified into seven clothes categories: jackets, coats, shirts, tops, t-shirts, dresses and pullovers, and seven texture categories: uniform, tiled, striped, animal skin, dotted, print and graphical patterns.

Attribute discovery dataset. We extracted from the attribute discovery dataset (Berg et al., 2010) 5783 images of bags, keeping for our training only images with white background. There are seven different handbags categories: shoulder, tote, clutch, backpack, satchel, wristlet, hobo. We also classify these images by texture into the same seven texture classes as the RTW dataset.

Automatic evaluation metrics Training generative models on fashion datasets generates impressive designs mixed with less impressive ones, requiring some effort of visual cherry-picking. We propose some automated criteria to evaluate trained models and compare different architectures and loss setups. Later, we study how these automatic metrics correlate with the human evaluation of generated images.

Evaluating the diversity and quality of a set of images has been tackled by scores such as the inception score and variants like the AM score (Zhou et al., 2017). We adapt both of

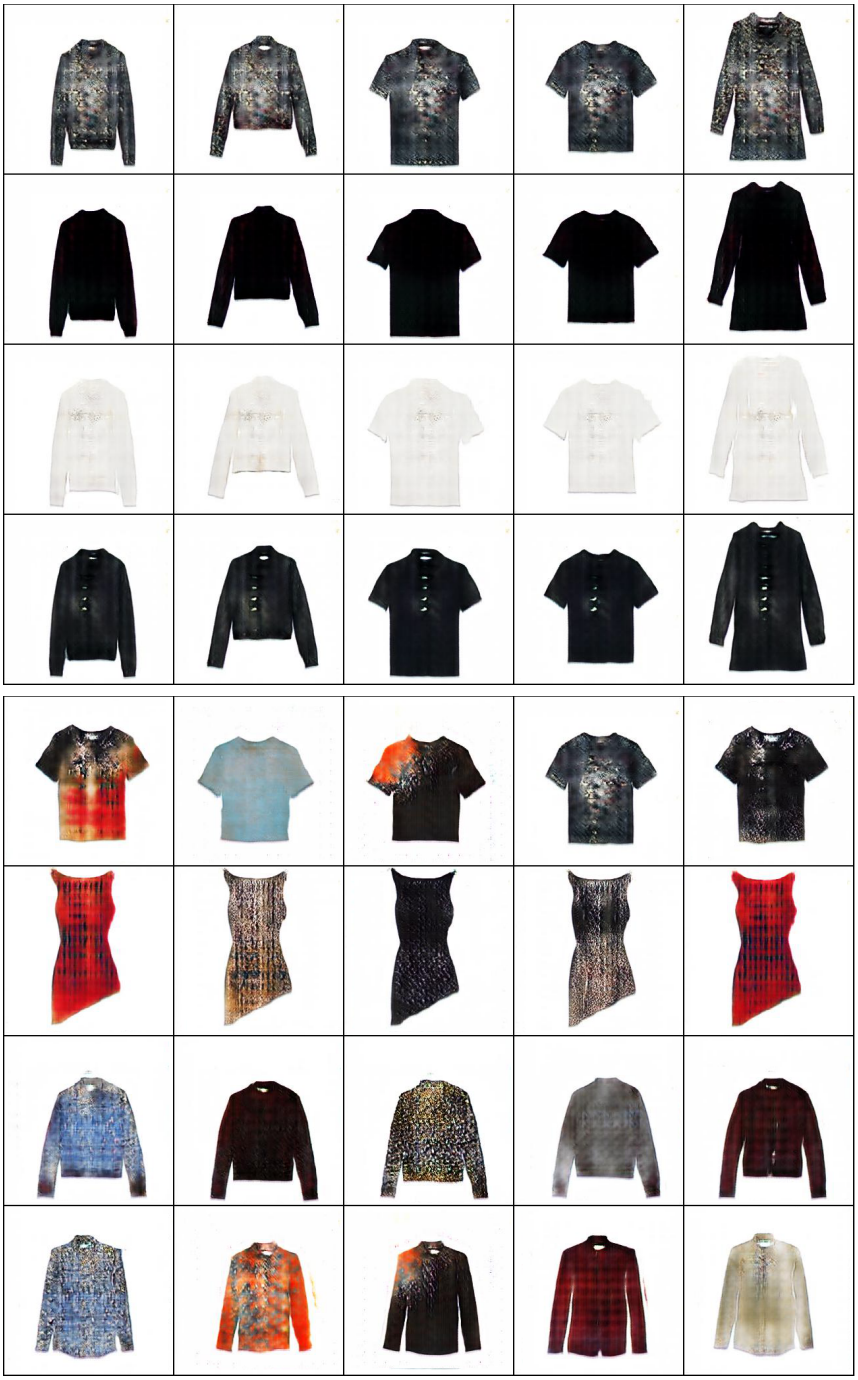


Figure 3.3: From the mask of a garment, our StyleGAN model generates different styled images for each style noise.

them for our two attributes specific to fashion design (shape and texture) and supplement them by a mean nearest neighbor distance. Our final set of automatic scores contains ten metrics:

- Shape score and texture score, each based on a Resnet-18 classifier of (shape or texture respectively);
- Shape AM score and texture AM score, based on the output of the same classifiers;

- Distance to nearest neighbors images from the training set;
- Texture and shape confusion of classifier;

Inception-like scores. The Inception score (Salimans et al., 2016; Warde-Farley and Bengio, 2017) was introduced as a metric to evaluate the diversity and quality of generations, with respect to the output of a considered classifier (Szegedy et al., 2017). For evaluating N samples $\{x\}_1^N$, it is computed as

$$I_{score}(\{x\}_1^N) = \exp(\mathbb{E}[KL(c(x)||\mathbb{E}[c(x)])]),$$

where $c(x)$ is the softmax output of the trained classifier c, originally the Inception network.

Intuitively, the score increases with the confidence in the classifier prediction (low entropy score) of each image and with the diversity of all images (high overall classification entropy). In this paper, we exploit the shape and texture class information from our two datasets to train two classifiers on top of Resnet-18 (He et al., 2015a) features, leading to the *shape score* and *texture score*.

AM scores. We also use the AM score proposed in Zhou et al. (2017). It improves over the inception score by taking into consideration the distribution of the training samples $\bar{x}$ as seen by the classifier c, which we denote $\bar{c}^{train} = \mathbb{E}[c(\bar{x})]$. The AM score is calculated as follows:

$$AM_{score}(\{x\}_1^N) = \mathbb{E}[KL(\bar{c}^{train}||c(x)] - KL(\bar{c}^{train}||\mathbb{E}[c(x)])]$$

The first term is maximized when each of the generated samples is far away from the overall training distribution, while the second term is minimized when the overall distribution of the generations is close to that of the training. In accordance with Zhou et al. (2017), we find that this score is more sensible as it accounts for the training class distribution.

Nearest neighbors distance. To be able to assess the creativity of the different models while making sure that they are not reproducing training samples, we compute the mean distance for each sample to its retrieved k-Nearest Neighbors (NN), with $k = 10$, as the Euclidean distance between the features extracted from a Resnet18 pre-trained on ImageNet (He et al., 2015a) by removing its last fully connected layer. These features are of size 512. This score gives an indicator of the similarity to the training data. A high NN distance may either mean that the generated images have some artifacts, in this case, it could be seen as an indicator of failure, or it could mean that the generation is novel and highly creative.

We experiment using weights λ_{G_e} of 1 and 5 for the MCE novelty loss. It appeared that the weight 1 worked better for the bags dataset, and 5 for the RTW dataset. We also tried different weights for the CAN and SM loss but they did not have a large influence on the results and was fixed to 1. All models were trained using the default learning rate 0.002 as in Radford et al. (2016). Our different models take about half a day to train on four Nvidia P100 GPUs for 256×256 models and almost two days for the 512×512 ones. In our study, it was more convenient from a memory and computational resources standpoint to work with 256×256 images but we also provide 512×512 results in Fig. 3.1 to demonstrate the capabilities of our approach. For each setup, we manually select four

Figure 3.4: First column: random generations by the GAN MCE shape model. Four left columns: Retrieved Nearest Neighbors for each sample.

saved models after a sufficient number of iterations. Our models produce plausible results after training for 15000 iterations with a batch size of 64 images.

Table 3.2 presents shape and texture classifier confidence C scores, AM scores (for shape and texture), average NN distances computed for each model on a set of 100 randomly selected images. Our first observation is that the DCGAN model alone seems to perform worse than all other tested models with the highest NN distance and lower shape and texture scores. The value of the NN distance score may have different meanings. A high value could mean an enhanced creativity of the model, but also a higher failure rate.

For the RTW dataset, the two models having high AM texture score, and high NN distances scores are DCGAN with MCE loss models and Style CAN with texture novelty. On the handbags datasets, the models obtaining the best metrics overall are the DCGAN with MCE novelty losses texture alone or on shape and texture. To show that our models are not reproducing exactly samples from the dataset, we display in Fig. 3.4 results from the model having the lowest NN distance score, with its four nearest neighbors. We note that even if uniform bags tend to be similar to training data, complex prints show high differences. These differences are amplified on the RTW dataset.

Creating evaluation sets.

To automatically access the best generations from each model, we extract different clusters of images with particular visual properties that we want to associate with realism, overall appreciation and novelty. Given the selected models, 10000 images are generated from random numbers – or randomly selected masks for the styleGAN model – to produce 8 sets of 100 images each. Based on the shape entropy, texture entropy and mean nearest neighbors distance of each image we can rank the generations and select the ones with (i) high/low shape entropy, (ii) high/low texture entropy, (iii) high/low NN distance to real images. We also explore random and mixed sets such as *low shape entropy* and *high nearest neighbors distance*. We expect such a set to contain plausible generations since low shape entropy usually correlates with well defined shapes, while high nearest neighbor distance contains unusual designs. Overall, we have 8 different sets that may overlap. We choose to evaluate 100 images for each set.

Human study We perform a human study where we evaluate different sets of generations of interest on a designed set of questions in order to explore the correlations with

	C sh	AM sh	C tex	AM tex	NN
GAN	0.26	7.88	0.39	2.07	**14.2**
GAN classif	0.27	**9.78**	**0.58**	1.58	13
GAN MCE sh	**0.31**	8.22	**0.59**	1.69	13.1
GAN MCE tex	0.25	8.05	**0.59**	**2.33**	**13.8**
GAN MCE shTex	0.21	8.96	0.49	1.45	13.3
CAN sh	0.27	8.52	0.48	1.86	13.2
CAN tex	0.29	8.40	0.48	**2.24**	13.4
CAN shTex	0.19	**10.1**	0.46	**2.39**	13.2
StackGAN	0.25	8.82	0.52	1.95	12.9
StackGAN MCE sh	0.27	8.16	**0.64**	2.03	**13.6**
StackGAN MCE tex	0.26	8.55	**0.63**	1.68	13.2
StackGAN MCE shTex	0.27	7.90	**0.71**	1.91	**13.6**
StackCAN sh	0.20	8.96	**0.67**	1.46	12.7
StackCAN tex	0.22	**9.45**	0.49	**2.32**	13.2
StackCAN shTex	0.26	8.11	**0.67**	**2.07**	13.4
Style GAN	0.25	8.24	0.52	1.76	**13.7**
Style CAN tex	0.29	7.93	0.48	**2.05**	**13.9**
Style GAN MCE tex	**0.34**	7.35	0.49	1.49	13.4

(a) Ready To Wear dataset

	C sh	AM sh	C tex	AM tex	NN
GAN	**0.43**	3.65	1.71	1.57	20.8
CAN tex	0.35	**4.29**	1.79	1.60	**21.4**
CAN sh	0.39	4.23	1.88	1.26	21
CAN sh tex	0.39	3.93	1.89	1.56	21.1
GAN MCE tex	0.34	**4.38**	**1.99**	1.60	**21.6**
GAN MCE sh	0.38	4.15	**1.98**	**1.84**	20.8
GAN MCE sh tex	**0.42**	3.73	**2.00**	**1.80**	21

(b) Attribute bags dataset

Table 3.2: Quantitative evaluation on the RTW dataset and bag datasets. For better readability we only display metrics that correlate most with human judgment. Higher scores, highlighted in bold, are usually preferred.

each of the proposed automatic metrics in choosing best models and ranking sound generations. As our RTW garment dataset could not be made publicly available, we conducted two independent studies:

1. In our main human evaluation study, 800 images — selected as described in the previous section — per model were evaluated, each image evaluated by five different persons. There were on average 90 participants per model assessment, resulting on average to 45 images assessed per participant. Since the assessment was conducted given the same conditions for all models, we are confident that the comparative study is fair. Each subject is shown images from the 8 selected sets described in the previous section and is asked six questions:

    - Q1: how do you like this design overall on a scale from 1 to 5?

Figure 3.5: Best generations of handbags as rated by human annotators. Each question is in a row. Q1: overall score, Q2: shape novelty.

- Q2/Q3: rate the novelty of shape (Q2) and texture (Q3) from 1 to 5.
- Q4/Q5: rate the complexity of shape (Q4) and texture (Q5) from 1 to 5.
- Q6: Do you think this image was created by a fashion designer or generated by a computer? (yes/no)

Each image is annotated by taking the average rating of five annotators.

2. We conducted another study in our lab were we mixed both generations (500 total images picked randomly from 5 best models) and 300 real down-sampled images from the RTW dataset. We asked if the images were real or generated to about 45 participants who rated 20 images each on average. We obtain 20% of the generations thought to be real, and 21.5% of the original dataset images were considered to be generated.

Going from a score of 64% to more than 75% in likabity from classical GANs to our best model with shape novelty is a great improvement over the baseline model. Our proposed Style GAN and StackGAN models are producing competitive scores compared to the best DCGAN setups with high overall scores. In particular, our proposed StyleGAN model with novelty loss is ranked in the top-3, some results are presented in Fig. 3.3. We display images which obtained the best scores for each of the 6 questions in Fig. 3.6, and some results of handbags generation in Fig. 3.5.

Correlations between human evaluation and automatic scores.

From Table 3.3, we see that the automatic metrics that correlate the most with the overall likability are the classifiers confidence scores, the inception texture score, and the average intensity.

3.4 Image search and composition for visual blends creation

3.4.1 Related work

Visual blends creation. Many works have addressed the challenge of visual blends creation. Steinbrück (2013) describes a method based on geometrical shape correspondence and object semantics to replace objects with new retrieved ones. Similarly, Chilton

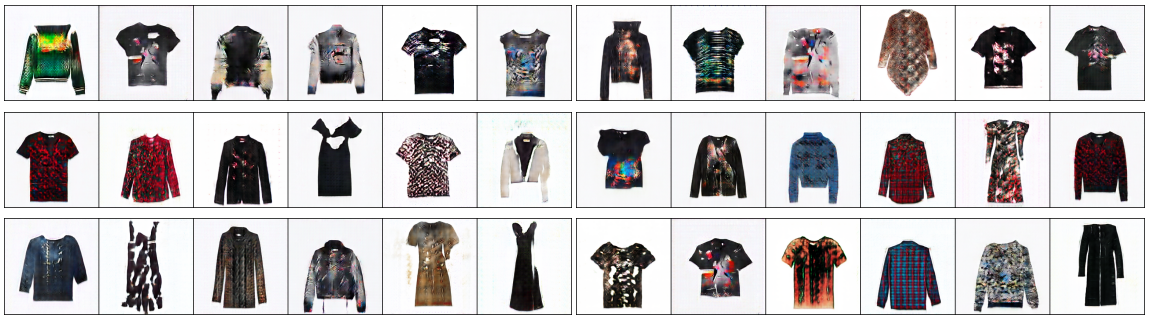


Figure 3.6: Best generations of RTW items as rated by human annotators. Each question is in a row. Left: Q1: overall score, Q2: shape novelty, Q3: shape complexity, Right: Q4: texture novelty, Q5: texture complexity, Q6: Realism.

et al. (2019) propose a shape based algorithm for finding and matching objects to blend together with a handcrafted blending synthesis method. Xiao et al. (2015) present *Vismantic*, a framework for generating image compositions based on a textual input, in order to express a specific meaning, using semantic associations and basic visual operations such as juxtaposition, replacement or fusion. Recently, Cun and Pun (2020) provide a detailed outline for building visual blendings with a strong conceptual grounding, without implementing that framework. In contrast to these methods, we do not use textual input to create visual compositions, but use both perceptual similarity and object semantics to suggest relevant compositions. Moreover, we propose a new image composition method based on recent advances in visual blending methods.

Searching for relevant objects to replace an existing one have been tackled in other works without the aim of creating visual blends. Tsai et al. (2016) present a pipeline for sky replacement to search for proper skies and perform a semantic-aware color transfer. Chen et al. (2009) construct a photomontage from a sketch by searching for candidate images matching the provided text label and performing the composition. Zhao et al. (2019) instead search for foreground objects that are semantically compatible with a background image given the category of the object to find.

Image blending. Early works on automatic image composition (Burt and Adelson, 1983; Milgram, 1975) use a multi-resolution image representation to create large mosaics of images. The seminal work of Poisson image blending (Pérez et al., 2003) proposes an elegant mathematical formulation based on solving Poisson equations to seamlessly blend images in the gradient domain. Several works improved the Poisson blending approach (Jia et al., 2006; Tao et al., 2010), which remains a very strong baseline for image composition.

Another line of work has tackled reducing the color discrepancy between composited images. Traditional image harmonization methods focused on better matching low level statistics between source and target images (Xue et al., 2012; Lalonde and Efros, 2007). Xue et al. (2012) identify image statistics that are correlated with composite realism such as luminance, saturation, contrast, while Lalonde and Efros (2007) study color statistics

Human / Auto	over-all	shape nov.	shape comp.	tex. nov.	tex. comp.	real fake
I sh score	0.43	**0.65**	**0.67**	0.33	0.60	0.30
AM sh score	**0.46**	**0.66**	**0.68**	0.38	**0.64**	0.28
C sh score	**0.48**	0.51	0.55	0.34	0.50	-0.06
I tex score	**0.48**	0.63	0.62	0.30	0.55	0.28
AM tex score	0.37	0.47	0.46	0.16	0.42	0.22
C tex score	**0.48**	**0.67**	**0.71**	**0.48**	**0.64**	0.27
N10	**0.46**	0.63	0.65	0.33	0.58	0.23

Human / Auto	over-all	shape nov.	shape comp.	tex. nov.	tex. comp.	real fake
coat	0.14	0.37	0.43	0.21	0.34	0.27
top	-0.15	0.12	0.16	0.18	0.07	-0.08
shirt	0.43	0.46	0.37	0.07	0.30	0.21
jacket	0.27	0.36	0.31	0.08	0.28	-0.02
t-shirt	0.44	0.49	0.43	0.31	0.49	0.08
dress	0.40	0.49	0.57	0.36	0.53	0.12
pullover	0.24	0.41	0.50	0.32	0.47	0.33
dotted	0.41	0.35	0.40	0.26	0.32	0.25
striped	0.20	0.16	0.15	0.08	0.21	0.11
print	0.42	0.31	0.26	0.30	0.35	-0.04
uniform	0.31	0.51	0.54	0.26	0.51	0.13
tiled	0.23	0.17	0.14	-0.08	-0.01	0.22
skin	-0.03	0.32	0.32	0.30	0.17	0.17
graphical	0.34	0.46	0.47	0.17	0.46	0.20

Table 3.3: Correlation scores between human evaluation ratings and automatic scores on the set of randomly sampled images of all models.

on a large dataset of realistic and unrealistic images to improve composites and discriminate unrealistic ones.

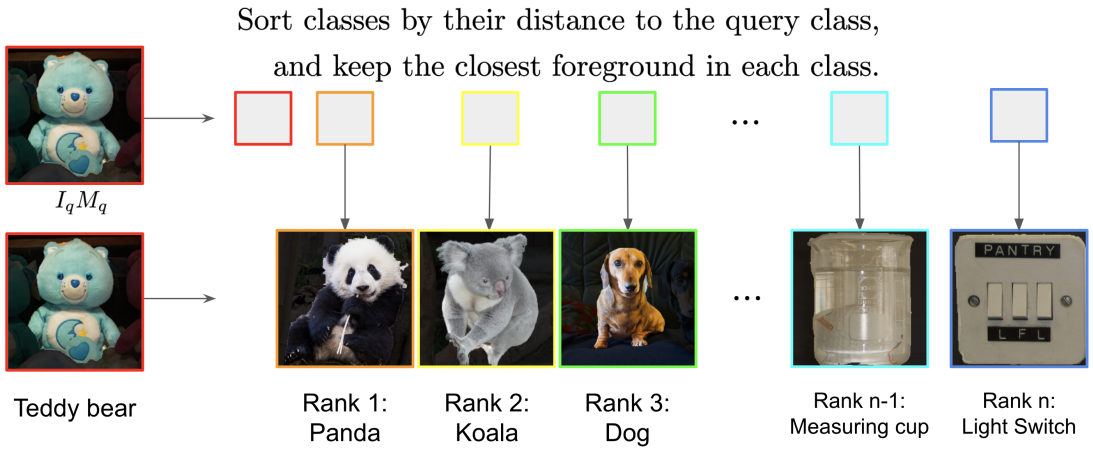
More recently, color harmonization (Cohen-Or et al., 2006) can be performed using deep learning methods Yan et al. (2015); Tsai et al. (2017); Cun and Pun (2020) that learn appearance adjustment using end-to-end networks. Recently, Cong et al. (2020) contributed a large-scale color harmonization dataset and a network to reduce foreground and background color inconsistencies.

In addition to color adjustment, some works study the geometric corrections necessary to place the new object in its new context. Using spatial transformer networks (Jaderberg et al., 2015), a differentiable module for sampling an image through an affine transformed grid, several works such as Lin et al. (2018) learn affine transformations to adjust the foreground position and reduce the geometric inconsistency between the source and the target images. While previous methods insert an object on an empty background image and focus on color harmonization, GCC-GAN (Chen and Kae, 2019) introduce a deep learning model based on predicting color and geometric adjustment for replacing a given object with a new one in addition to inpainting missing empty regions. Finally, performing using copy pasting for image composition has been enhanced with refined mask prediction

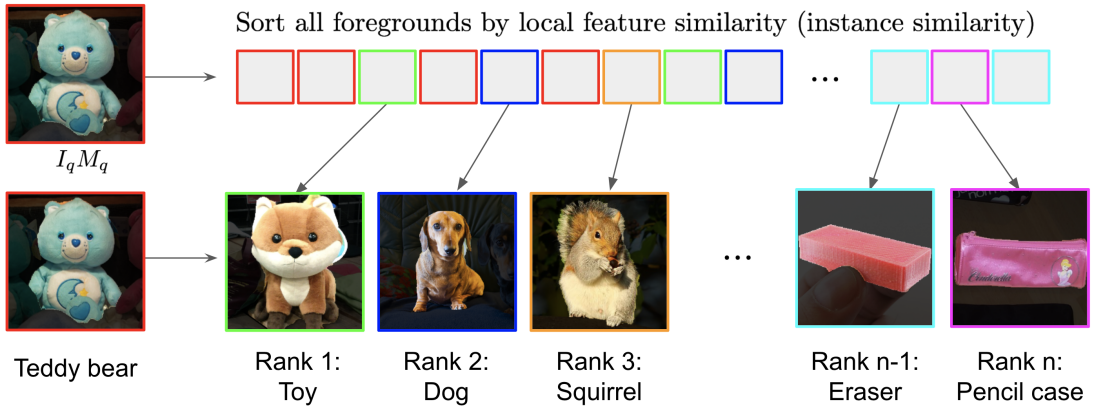
of the foreground as in Arandjelović and Zisserman (2019).

Assessing the realism of generated composite images is a challenge. RealismCNN (Zhu et al., 2015) propose a learning based approach to discriminate real images from composite ones by predicting a realism score while RGB-N (Zhou et al., 2018) introduce a two-stream Faster R-CNN network to detect the tampered regions given a manipulated image which we use in our study.

3.4.2 Foreground selection and image composition



(a) Class similarity



(b) Instance similarity

Figure 3.7: Overview of the class and instance similarity based foreground selection methods. From each class we keep the closest foreground. While instance similarity ranks the classes by local foreground similarity, class similarity instead ranks them with their distance to the query class.

Our approach relies on two key components; searching for suitable foregrounds to replace a selected one and performing image composition automatically. We first search for visually similar foregrounds from different classes, leading to placing objects in uncommon contexts. We then design an image composition model similar to the one proposed in GCC-GAN (Chen and Kae, 2019), where we apply affine geometric and color transforma-

tions to the foreground before pasting it on the inpainted background. In the following, we assume we have access to a dataset of centered segmented objects with class annotations.

Foreground selection To find visually similar but semantically different foregrounds for a given query image, we search foregrounds of different semantic classes with the most similar features. Using local features allows us to have an object similarity with more emphasis on the shape similarity than using global pooled features. We found that masking out the background of each object when computing local features leads to retrieving similar objects with similar masks, which is useful for visual blending through object replacement. We use the *layer3* features of a ResNet-50 trained on the images from ImageNet (Deng et al., 2009b) using MoCoV2 (Chen et al., 2020). To limit the memory footprint and computational cost, we reduce the dimension of each local feature from 1024 to 50 using Principal Component Analysis. Each local feature is ℓ_2 normalized. Each foreground is then represented by a $14 \times 14 \times 50$ feature map. Given a query foreground object, we search the index and keep only the closest foreground from each class for our analysis as visualized in Fig. 3.7.

More formally, given a query image I_q and the associated binary mask M_q, we select for each class c the image I_c and mask M_c defined by:

$$(I_c, M_c) = \arg\max_{(I,M)\in\mathcal{D}_c} \langle f(I_q M_q), f(IM) \rangle \tag{3.4}$$

where f is our feature extraction and $\mathcal{D}_c$ the set of pairs of image and mask associated to class c. To enable fast online search, we build an index from pre-computed local features using the FAISS library (Johnson et al., 2017) and search for similar foregrounds using the inner product between the flattened features.

In our analysis, we consider two ranking setups to select the pairs (I_c, M_c) to use for our composition, based on the visual similarity of foregrounds as described above and on a distance between the different classes, both setups are explained in Fig. 3.7. For the first one, dubbed *instance similarity*, we rank the images according to their distance to the query, similar to equation 3.4 and we select the closest foreground in each class. For the second one, dubbed *class similarity*, we instead use the similarity of the average feature of each class $\frac{1}{|\mathcal{D}_c|} \sum_{(I,M)\in\mathcal{D}_c} f(I_q M_q)$, where $|\mathcal{D}_c|$ is the number of images in $\mathcal{D}_c$ with the average feature of the query class. While in the first setup we focus on the visual foreground similarity to rank the images, in the second one instead we rank the closest objects according to the average similarity of the class, introducing a notion of semantics in the similarity.

Image composition Here, we assume we want to create a composite image using the foreground object of image F associated with the mask M_F and the background image B excluding the object defined by the mask M_B. We consider a composition framework that predicts geometric and color corrections and applies them to the foreground object, similar to GCC-GAN (Chen and Kae, 2019). We use affine transformations for both the spatial and color components. Particularly, we use spatial transformers (Jaderberg et al., 2015) as a differentiable module to spatially transform the foreground object, and denote $\mathcal{T}$ and $\mathcal{C}$ respectively the spatial and color transformations applied to the foreground. We denote by g the network predicting the spatial and color transformation parameters θ_{ST} and θ_C, it takes as input the concatenation of the masked foreground FM_F and the masked

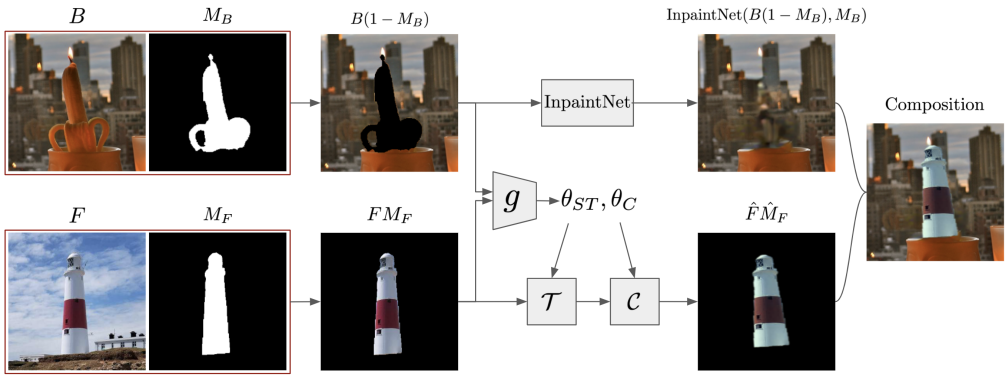


Figure 3.8: Overview of our image composition pipeline. We learn a model that predicts spatial and color corrections θ_{ST} and θ_C respectively on synthetic examples. We apply these parameters using spatial transformers (Jaderberg et al., 2015) and an affine color transformation in RGB space, before blending the transformed foreground onto the inpainted background.

background $B(1-M_B)$. We use the same architecture for g as in Lin et al. (2018). While, $\mathcal{T}$ and $\mathcal{C}$ are differentiable and have no trainable parameters, they take as input θ_{ST} and θ_C. Fig. 3.8 illustrates our entire composition pipeline.

At test time, to compose our final image, we first compute the spatial and color transformation parameters θ_{ST} and θ_C using the network g as shown in Eq. 3.5 to define a transformed foreground image $\hat{F}$ and a transformed foreground mask $\hat{M}_F$ as in Eq. 3.6.

$$\theta_{ST}, \theta_C = g(FM_F, B(1-M_B)) \tag{3.5}$$

$$\hat{F} = \mathcal{C}(\mathcal{T}(F, \theta_{ST}), \theta_C) \text{ and } \hat{M}_F = \mathcal{T}(M_F, \theta_{ST}) \tag{3.6}$$

We then use the network InpaintNet from Yu et al. (2019) to inpaint the background into InpaintNet$(B(1-M_B), M_B)$. Finally, we compose the transformed foreground image and the background image into a final composite image:

$$\hat{F}\hat{M}_F + (1-\hat{M}_F)\text{InpaintNet}(B(1-M_B), M_B) \tag{3.7}$$

We train g by creating synthetic examples as follows and as shown in Fig. 3.9: assuming we have access to segmented objects, we first extract an object and use its mask to create both foreground and background images; we then erode the border of both the foreground and background and jitter the foreground image using random affine color and spatial transformations obtained from a normal distribution $\mathcal{N}(0, 0.1)$ as perturbations from the identity of each transform. We use two different losses for the spatial and color transformation prediction. For the spatial part, we simply minimize the ℓ_2 distance between the predicted and target parameters to undo the spatial perturbation. For the color, such an error was not representative of the visual similarity between the transformed images. We use instead the ℓ_1 distance between the original foreground and the corrected one using the predicted color transformation. Note that mask erosion of the foreground and background is important to remove obvious visual clues and to make the training more challenging.

Figure 3.9: Examples of color and spatial random transformation used in our synthetic dataset to train our model. From left to right, we show the original image, the jittered one, and the overlap of eroded foreground and background that we input as six channels. The top row shows color modification associated with mask erosion. While the bottom one shows an example with spatial transformation jittering.

Note that our training and composition procedure are much simpler and more stable than the one proposed in Chen and Kae (2019), which uses multiple adversarial losses that we were unable to reproduce.

3.4.3 Experiments and results

In this section, we demonstrate the performance of our composition method by highlighting the importance of using spatial and color corrections through comparison with baselines both using a tampering detection metric and visually. We then present the human study that we perform to compare the two class sorting methods for selecting foregrounds of our composites, and show images that obtained unanimous ratings in our human study.

Dataset In order to demonstrate our search and composition method, we use Open-Images V6, a large collection of objects from diverse annotated classes with their mask annotations. We subsample a set of relevant segments by filtering out small objects ($< 64 \times 64$ pixels) and images of low quality computed using the image quality estimation network Koncept-512 (Hosu et al., 2020) leading to a dataset of 37 233 images from 319 object classes. Note that the quality of the obtained compositions heavily depend on the diversity of annotations in the dataset, and that using a larger image set would definitely lead to better image compositions.

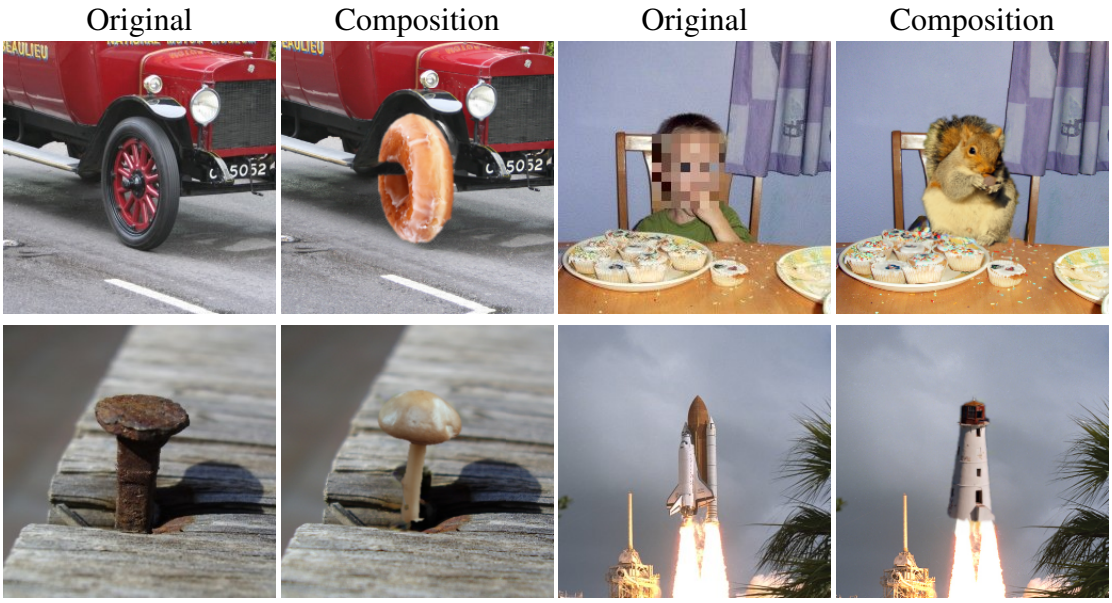


Figure 3.10: Examples of obtained visual blends (right) and their original images (left) using our search and composition method. Visual blends are usually used to challenge and trigger thoughts or simply entertain. From left to right: Bagel/Wheel, Squirrel/Boy, Nail/Mushroom, Lighthouse/Rocket.

Composition baselines We consider three baselines for our composition algorithm. The first one is based on simple object copy pasting, enhanced with inpainting the region of the removed object. The two other baselines are based on Poisson image blending (Pérez et al., 2003). While this algorithm is designed for inserting a foreground object on a background image, we adapt it with an inpainting step to fill in the removed initial object mask, we name this baseline "Poisson". In the last enhanced baseline that we name "ST+Poisson", we apply our learned spatial transformation module to adjust the foreground spatially, and blend it using Poisson blending.

Quantitative evaluation RGB-N score is a tampering detection score presented in Zhou et al. (2018), it represents how realistic an image is by detecting tampered regions and averaging their detection scores. In Table 3.4, we report this average over 1000 images sampled from the top-10 compositions obtained with our two foreground ranking strategies for our approach and the different baselines presented above. While copy pasting composites are systematically detected as tampered ones, our composition method obtains lower RGB-N score than all baselines both for top-10 compositions obtained with class similarity or using instance similarity. Also, we note the very clear boost given by our spatial transformation both with our composition method and the Poisson composition baselines.

Qualitative comparison to composition baselines In Fig. 3.10, we show examples of obtained visual blends and their original images. New foreground objects are visually similar to the original object but from a semantically different class, leading to surprising and original compositions. In Fig. 3.11, we show a comparison of our composition

Method	Class sim.	Instance sim.
Real images	59.24	
Copy-paste	97.49	97.45
Poisson	73.06	72.55
ST+Poisson	65.42	64.02
Ours	**58.01**	**56.95**

Table 3.4: Tampering RGB-N scores for real and composite images computed over 1000 samples. (lower is better)

algorithm with the baselines including a simple object copy paste. Our model is trained to undo synthetic affine color and spatial transformations, therefore, it predicts suitable geometric and color transformations to adjust the spatial arrangement of the foreground object and harmonize its appearance in the background image. On the contrary, the Poisson blending baseline suffers from color bleeding and is unable to resize and place the foreground object.

Human study We design an experiment where human raters are asked to evaluate different compositions obtained from the same original image. The goal is to understand how real, surprising and liked our compositions are given the class selection strategy for the new foreground. We thus rank the candidate classes either using our instance similarity or our class similarity strategy. For each annotation task, we sample four composite images from four groups defined by the rank of the selected composition (between 1 and 5, 6 and 10, 11 and 20 or above 20). For each of the class selection strategies, we randomly sample 200 tasks obtained from the same original images, and each task is presented to 5 different raters, leading to 1000 task evaluations per class selection strategy. Raters are shown the original image and four shuffled compositions and asked to select the most surprising composition, the one they like the most and the most realistic one independently.

In Fig. 3.12, we compare the ratings obtained by each group and for each search method; using class similarity or instance similarity to rank the selected foregrounds. We observe a much clearer correlation of the surprise and realism ratings with the rank groups from the class similarity selection - smaller ranks corresponding to more realistic and less surprising compositions - while little correlations are observed with the instance similarity. The observed correlations for class similarity are significant, as checked using a Pearson's Chi-squared test with p-values (0.002 for likeability, < 0.001 for surprise and < 0.001 for realism). Instead, using instance similarity foreground ranking method, only the correlation with realism is significant with a p-value of 0.002, the other p-values being larger than 0.05. We show examples of images with unanimous ratings in Fig. 3.13. We observe in these examples that composites generated by our method can be very realistic, by replacing foreground objects in similar contexts (birds or animals replacements). In contrast, when the foreground class is picked far from the original one, the context may be very different, resulting in surprising results (e.g. giraffe in the city, crocodile in plate). The most liked compositions, more difficult to analyze, can be explained in some cases by a judgment of the image aesthetic, or preference for some object class.

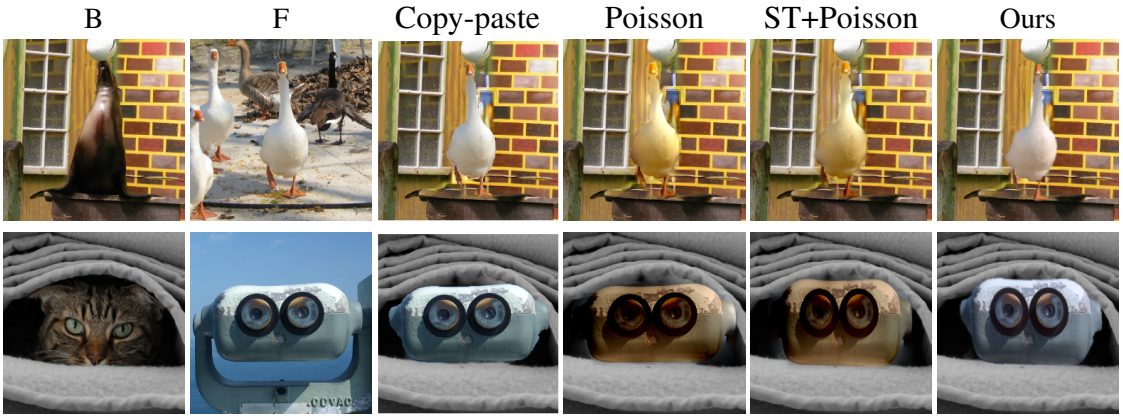


Figure 3.11: Comparison to baselines. Our model is able to place the foreground object and adjust its appearance so that it is blended seamlessly in the new context.

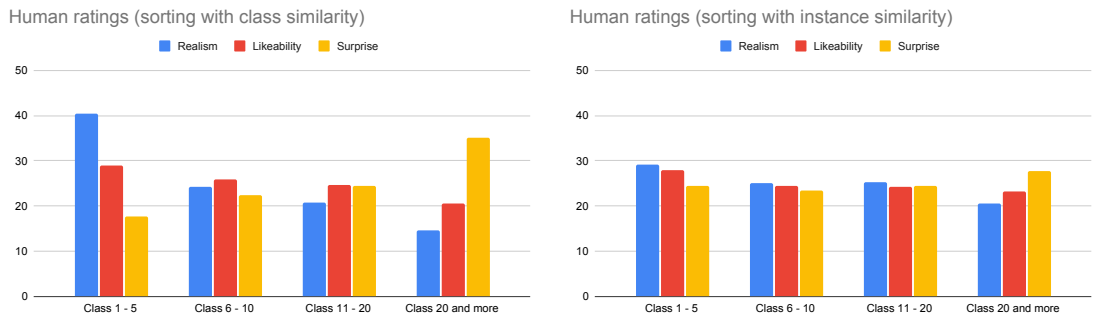


Figure 3.12: Human study: comparing realism, likeability, and surprise ratings for compositions obtained with class or instance similarity ranking. We represent the proportion of each group in being selected as real, liked or surprising on 1000 tasks.

3.5 Conclusion

We presented two different approaches for novel image generation by leveraging both generative models and image composition techniques. First, using generative adversarial networks, we propose a model for generating original garment images from a clothing dataset by introducing a specific conditioning on texture and shape elements. Adding novelty losses allows the learning to deviate from a reproduction of the training set and the generation of high resolution, realistic and novel designs, as reflected by our human study.

Second, using image search and an image composition model we are able to suggest realistic and surprising visual blends using visual similarity and blending methods. Our approach simplifies image composition by considering color and spatial adaptations that are trained synthetically in parallel with a state-of-the-art inpainting model for a seamless foreground blending.

Based on human studies, we observe that our proposed models are able to generate realistic images with a high likeability and surprise score. In visual blends generation, our human study shows that we can control the realism, likeability and surprise by considering

Figure 3.13: Composite images and their original ones selected through our human study with the most liked, most surprising or highest realism ratings. By pairs, the left columns represent the original images and the right ones our composites. Images that we show have at least 4 unanimous ratings among 5 raters.

class similarity instead of foreground similarity alone.

Finally, we believe that using recent image generation models conditioned on natural language could be a great advance in visual metaphor generation and novel generation of fashion designs.

Chapter 4

Image generation in multiple vector layers

4.1 Abstract

Deep image generation is becoming a tool to enhance artists and designers creativity potential. In this chapter, we aim at making the generation process more structured and easier to interact with. Inspired by vector graphics systems, we propose a new deep image reconstruction paradigm where the outputs are composed from simple layers, defined by their color and a vector transparency mask. This presents a number of advantages compared to the commonly used convolutional network architectures. In particular, our layered decomposition allows simple user interaction, for example to update a given mask, or change the color of a selected layer. From a compact code, our architecture also generates vector images with a virtually infinite resolution, the color at each point in an image being a parametric function of its coordinates. We validate the efficiency of our approach by comparing reconstructions with state-of-the-art baselines given similar memory resources on CelebA and ImageNet datasets. Most importantly, we demonstrate several applications of our new image representation obtained in an unsupervised manner, including editing, vectorization and image search.

Keywords: vector graphics, image reconstruction, image editing, image generation.

4.2 Introduction

Deep image generation models demonstrate breathtaking and inspiring results, e.g. (Zhu et al., 2017a,b; Karras et al., 2019; Brock et al., 2019), but usually offer limited control and little interpretability. It is indeed particularly challenging to learn end-to-end editable image decomposition without relying on either expensive user input or handcrafted image processing tools. In contrast, we introduce and explore a new deep image generation paradigm, which follows an approach similar to the one used in interactive design tools. We formulate image generation as the composition of successive layers, each associated to a single color. Rather than learning high resolution image generation, we produce a decomposition of the image in vector layers, that can easily be used to edit images at any resolution. We aim at enabling designers to easily build on the results of deep

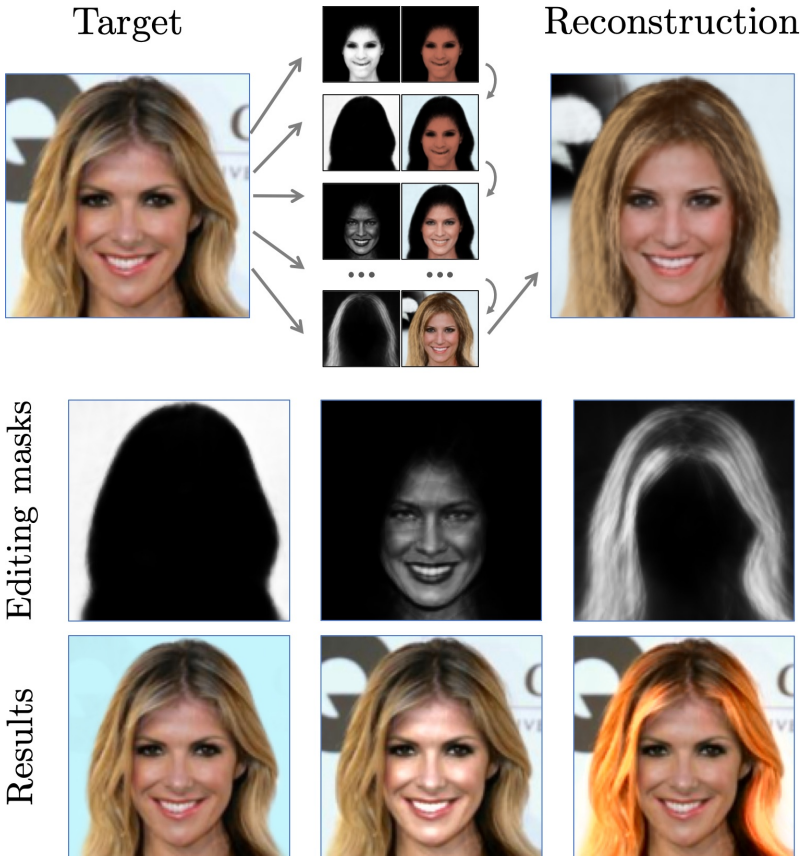


Figure 4.1: Our system learns in an unsupervised manner a decomposition of images as superimposed α-channel masks (top) that can be used for quick image editing (bottom).

image generation methods, by editing layers individually, changing their characteristics, or intervening in the middle of the generation process.

Our approach is in line with the long standing Computer Vision trend to look for a simplified and compact representation of the visual world. For examples, in 1971 Binford (Binford, 1971) proposes to represent the 3D world using generalized cylinders and in 1987 the seminal work of Biederman (Biederman, 1987) aims at explaining and interpreting the 3D world and images using geons, a family of simple parametric shapes. These ideas have recently been revisited using Neural Networks to represent a 3D shape using a set of blocks (Tulsiani et al., 2017) or, more related to our approach, a set of parametric patches (Groueix et al., 2018). The idea of identifying elementary shapes and organizing them in layers has been successfully applied to model images (Adelson, 1991; Isola and Liu, 2013) and videos (Wang and Adelson, 1994). A classical texture generation method, the dead leaves model (Lee et al., 2001) which creates realistic textures by relying on the iteration of simple patterns addition, is particularly related to our work.

We build on this idea of composing layers of simple primitives in order to design a deep image generation method, relying on two core ingredients. First, the learning of vector transparency masks as parametric continuous function defined on the unit square. In practice, this function is computed by a network applied at 2D coordinates on a square grid, to output mask values at each pixel coordinates. Second, a mask blending module which we use to iteratively build the images by superimposing a mask with a given color to the previous generation. At each step of our generation process, a network predicts both parameters and color for one mask. Our final generated image is the result of blending a fixed number of colored masks. Figure 4.1 shows the reconstruction from the mask decomposition and sample image editing we can perform. One of the advantages of this approach is that, differently to most existing deep generation setups where the generation is of fixed size, our generations are vector images defined continuously, and thus have virtually infinite resolution. Another key aspect is that the generation process is easily interpretable, allowing simple user interaction.

To summarize, our main contribution is a new deep image generation paradigm which:

- builds images iteratively from masks corresponding to meaningful image regions, learned without any semantic supervision.
- is one of the first to generate *vector* images from a compact code.
- is useful for several applications, including image editing using generated masks, image vectorization, and image search in mask space.

Our code is available[1].

4.3 Related work

We begin this section by presenting relevant works on image vectorization, then focus on most related unsupervised image generation strategies and finally discuss applications of deep learning to image manipulation.

[1] http://imagine.enpc.fr/~sbaio/pix2vec/

Vectorization. Many vector-based primitives have been proposed to allow shape editing, color editing and vector image processing ranging from paths and polygons filled with uniform color or linear and radial gradients (Richardt et al., 2014; Favreau et al., 2017), to region based partitioning using triangulation (Demaret et al., 2006; Liao et al., 2012; Duan and Lafarge, 2015), parametric patches (Bezier patches) (Xia et al., 2009) or diffusion curves (Orzan et al., 2008). We note that traditionally, image vectorization techniques were handcrafted using image smoothing and edge detectors. In contrast, our approach parametrizes the image using a function defined by a neural network.

Differentiable image parametrizations with neural networks were first proposed in Stanley (2007) which introduced Compositional Pattern Producing Networks (CPPNs) that are simply neural networks that map pixel coordinates to image colors at each pixel. The architecture of the network determines the space of images that can be generated. Since CPPNs learn images as functions of pixel coordinates they provide the ability to sample images at high resolution. The weights of the network can be optimized to reconstruct a single image (Karpathy, 2015) or sample randomly in which case each network results in abstract patterns (Ha, 2016a). In contrast with these approaches, we propose to *learn* the weights of this mapping network and condition it on a an image feature so that it can generate any image without image-specific weight optimization. Similarly, recent works have modeled 2D and 3D shapes using parametric and implicit functions (Groueix et al., 2018; Mescheder et al., 2019; Park et al., 2019a; Chen and Zhang, 2019). While previous attempts to apply this idea on images has focused on directly generating images on simple datasets such as MNIST (Ha, 2016b; Chen and Zhang, 2019), we obtain a layer decomposition allowing various applications such as image editing and retrieval on complex images.

Deep, unsupervised, sequential image generation.

We now present deep unupervised sequential approaches to image generation, the most related to our work. Rolfe and LeCun (2013) use a recurrent auto-encoder to reconstruct images iteratively, and employs a sparsity criterion to make sure that the image parts that are added at each iteration are simple. A second line of approaches (Gregor et al., 2015; Eslami et al., 2016; Gregor et al., 2016) are designed in a VAE framework. Deep Recurrent Attentive Writer (DRAW) (Gregor et al., 2015) frame a recurrent approach using reinforcement learning and a spatial attention mechanism to mimic human gestures. A potential application of DRAW arises in its extension to conceptual image compression (Gregor et al., 2016), where a recurrent convolutional and hierarchical architecture allows to obtain various levels of lossy compressed images. Attend, Infer, Repeat (Eslami et al., 2016) models scenes by latent variables of object presence, content, and position. The parameters of presence and position are inferred by an RNN and a VAE decodes the objects one at a time to reconstruct images. A third strategy for learning sequential generative models is to employ adversarial networks. Ganin et al. (2018) employ adversarial training in a reinforcement learning context. Specifically, their method dubbed SPIRAL, trains an agent to synthesize programs executed by a graphic engine to reconstruct images. The Layered Recursive GANs (Yang et al., 2017) learn to generate foreground and background images that are stitched together using STNs to form a consistent image. Although presented in a generic way that generalizes to multiple steps, the experiments are limited to foreground and background separation, made possible by the definition of a prior on the object size contained in the image. In contrast, our method (i) does not rely

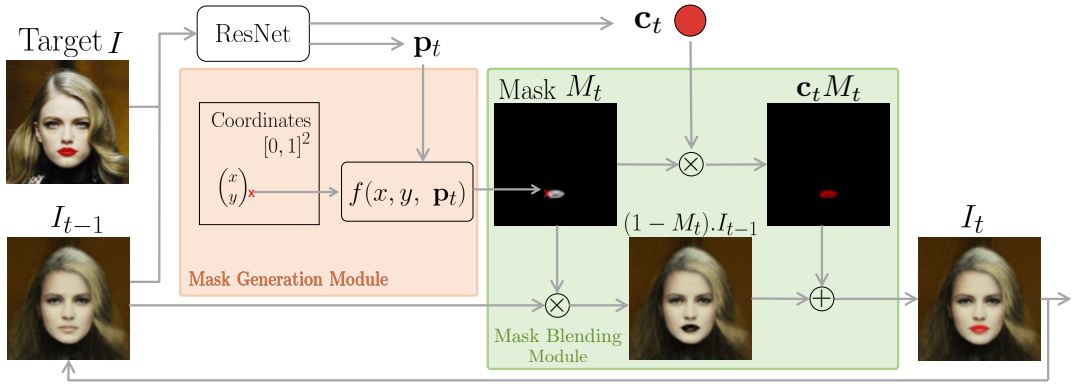


Figure 4.2: Our iterative generation pipeline for image reconstruction of target I. The previous canvas I_{t-1} (I_0 can be initialized to a random uniform color) is concatenated with I and forwarded through a ResNet feature extractor, to obtain a color $\mathbf{c}_t$ and mask parameters $\mathbf{p}_t$. A Multi Layer Perceptron f generates a parametric mask M_t from pixel-wise coordinates of a 2D grid and mask parameters $\mathbf{p}_t$. Our Mask Blending Module (in green) finally blends this mask with its corresponding color to the previous output I_{t-1}.

on STNs; (ii) extends to tens of steps as demonstrated in our experiments; (iii) relies on simple architectures and losses, without the need of LSTMs or reinforcement learning.

Image manipulation. Some successful applications of deep learning to image manipulation have been demonstrated, but they are usually specialized and offer limited user interaction. Image colorization (Zhang et al., 2016) and style transfer (Gatys et al., 2016) are two popular examples. Most approaches that allow user interaction are supervised. Zhu et al. (2016) integrate user constraints in the form of brush strokes in GAN generations. More recently Park et al. (2019b) use semantic segmentation layouts and brush strokes to allow users to create new landscapes. In a similar vein, Bau et al. (2019) locate sets of neurons related to different visual components of images, such as trees or artifacts, and allows their removal interactively. Approaches specialized in face editing, such as Shen et al. (2016) and Portenier et al. (2018) demonstrate the large set of photo-realistic image manipulations that can be done to enhance quality, for instance background removal or swapping, diverse stylization effects, changes of the depth of field of the background, etc. These approach typically require precise label inputs from users, or training on heavily annotated datasets. Our approach provides an unsupervised alternative, with similar editing capacities.

4.4 Layered Vector Image Generation

We frame image generation as an alpha-blending composition of a sequence of layers starting from a canvas of random uniform color I_0. Given a fixed budget of T iterations, we iteratively blend T generated colored masks onto the canvas. In this section, we first present our new architecture for vector image generation, then the training loss and finally discuss the advantages of our new architecture compared to existing approaches.

4.4.1 Architecture

The core idea of our approach is visualized in Fig. 4.2. At each iteration $t \in \{1...T\}$, our model takes as input the concatenation of the target image $I \in \mathbb{R}^{3\times W\times H}$ and the current canvas I_t, and iteratively blends colored masks on the canvas resulting in I_t:

$$I_t = g(I_{t-1}, I), \tag{4.1}$$

where g consists of:

(i) a Residual Network (ResNet) that predicts mask parameters $\mathbf{p}_t \in \mathbb{R}^P$, with the corresponding color triplet $\mathbf{c}_t \in \mathbb{R}^3$,

(ii) a mask generator module f, which generates an alpha-blending mask M_t from the parameters $\mathbf{p}_t$, and

(iii) our mask blending module that blends the masks M_t with their color $\mathbf{c}_t$ on the previous canvas I_{t-1}.

We represent the function f generating the mask M_t from $\mathbf{p}_t$ as a standard Multi-Layer Perceptron (MLP), which takes as input the concatenation of the mask parameters $\mathbf{p}_t$ and the two spatial coordinates (x, y) of a point in image space. This MLP f defines the continuous 2D function of the mask M_t by:

$$M_t(x, y) = f(x, y, \mathbf{p}_t). \tag{4.2}$$

In practice, we evaluate the mask at discrete spatial locations corresponding to the desired resolution to produce a discrete image. We then update I_t at each spatial location (x, y) using the following blending:

$$I_t(x, y) = I_{t-1}(x, y).(1 - M_t(x, y)) + \mathbf{c}_t.M_t(x, y), \tag{4.3}$$

where $I_t(x, y) \in \mathbb{R}^3$ is the RGB value of the resulting image I_t at position (x, y). We note that, at test time, we may perform a different number of iterations N than the one during training T. Choosing $N > T$ may help to model accurately images that contain complex patterns, as we show in our experiments.

All the design choices of our approach are justified in detail in Section 4.4.3 and supported empirically by experiments and ablations in Section 4.5.3.

4.4.2 Training losses

We learn the weights of our network end-to-end by minimizing a reconstruction loss between the target I and our result $R = I_T$. We perform experiments either using an ℓ_1 loss, which enables simple quantitative comparisons, or a perceptual loss (Johnson et al., 2016), leading to visually improved results. Our perceptual loss $\mathcal{L}_{perc}$ is based on the Euclidean norm $\|.\|_2$ between feature maps $\phi(.)$ extracted from a pre-trained VGG16 network and the Frobenius norm between the Gram matrices obtained from these feature maps $G(\phi(.))$:

$$\mathcal{L}_{perc} = \mathcal{L}_{content} + \lambda\mathcal{L}_{style},$$

where

$$\mathcal{L}_{content}(I, R) = \|\phi(I) - \phi(R)\|_2 ,$$

$$\mathcal{L}_{style}(I, R) = \|G(\phi(I)) - G(\phi(R))\|_F ,$$

and λ is a non-negative scalar that controls the relative influence of the style loss. To obtain even sharper results, we may optionally add an adversarial loss. In this case, a discriminator D is trained to recognize real images from generated ones, and we optimize our generator G to fool this discriminator. We train D to minimize the non saturating GAN loss from Goodfellow et al. (2014) with R1 Gradient Penalty loss (Mescheder et al., 2018). The architecture of D is the patch discriminator defined in Isola et al. (2017).

4.4.3 Discussion

Architecture choices. Our architecture choices are related to desirable properties of the final generation model:

Layered decomposition: This choice allows us to obtain a mask decomposition which is a key component of image editing pipelines. Defining one color per layer, similar to image simplification and quantization approaches, is important to obtain visually coherent regions. We further show that a single layer baseline does not perform as well.

Vectorized layers: By using a lattice input for the mask generator, it is possible to perform local image editing and generation at any resolution without introducing upsampling artifacts or changing our model architecture. This vector mask representation is especially convenient for HD image editing.

Recursive vs one-shot: We generate the mask parameters recursively to allow the model to better take into account the interaction between the different masks. We show that a one-shot baseline, where all the mask parameters are predicted in a single pass leads to worse results. Moreover, as mentioned above and demonstrated in the experiments, our recursive procedure can be applied a larger number of times to model more complex images.

Number of layers vs. size of the mask parameter. Our mask blending module iteratively adds colored masks to the canvas to compose the final image. The size of the mask parameter $\mathbf{p}$ controls the complexity of the possible mask shapes, while the number of masks controls the amount of different shapes that be used to compose the image. Since we aim at producing a set of layers that can easily be used and interpreted by a human, we use a limited number of strokes and masks.

Complexity of the mask generator network. Interestingly, if the network generating the masks from the parameters was very large, it could generate very complex patterns. In fact, one could show using the universal approximation theorem (Cybenko, 1989; Hornik, 1991) that, with a large number of hidden units in the MLP f, an image could be approximated with only three layers ($N = 3$) of our generation process, using one mask for each color channel. Thus it is important to control the complexity of f to obtain meaningful primitive shapes. For example, we found that replacing our MLP by a ResNet leads to less interpretable masks (see Section 4.5.3 and Fig. 4.10).

4.5 Experiments

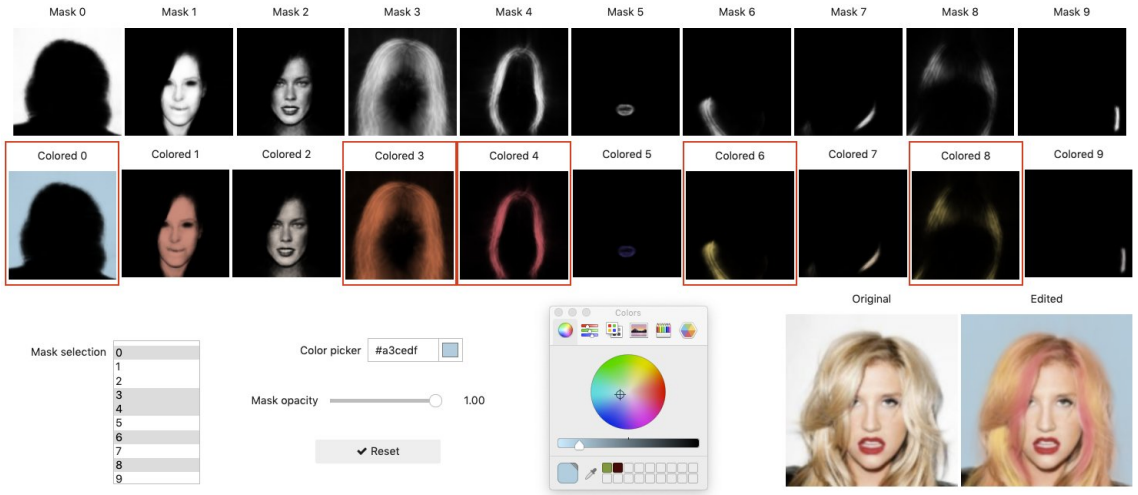


Figure 4.3: Our editing interface using automatically extracted masks to bootstrap the editing process.

In this section we first introduce the datasets, the training and network architecture details, then we demonstrate the practical interest of our approach in several applications, and justify the architecture choices in extensive ablation studies.

4.5.1 Datasets and implementation details

Datasets. Our models are trained on two datasets, CelebA (Liu et al., 2015) (202k images of celebrity faces) and ImageNet (Deng et al., 2009a) (1.28M natural images of 1000 classes), using images downsampled to 128×128.

Training details. The parameters of our generator g are optimized using Adam (Kingma and Ba, 2014) with a learning rate of 2×10^{-4}, $\beta_1 = 0.9$ and no weight decay. The batch size is set to 32 and training image size is fixed to 128×128 pixel images.

Network architectures. The mask generator f consists of an MLP with three hidden layers of 128 units with group normalization (Wu and He, 2018), tanh non-linearities, and an additional sigmoid after the last layer. f takes as input a parameter vector $\mathbf{p}$ and pixel coordinates (x, y), and outputs a value between 0 and 1. The parameter $\mathbf{p}$ and the color $\mathbf{c}$ are predicted by a ResNet-18 network.

4.5.2 Applications

We now demonstrate how our image decomposition may serve different purposes such as image editing, retrieval and vectorization.

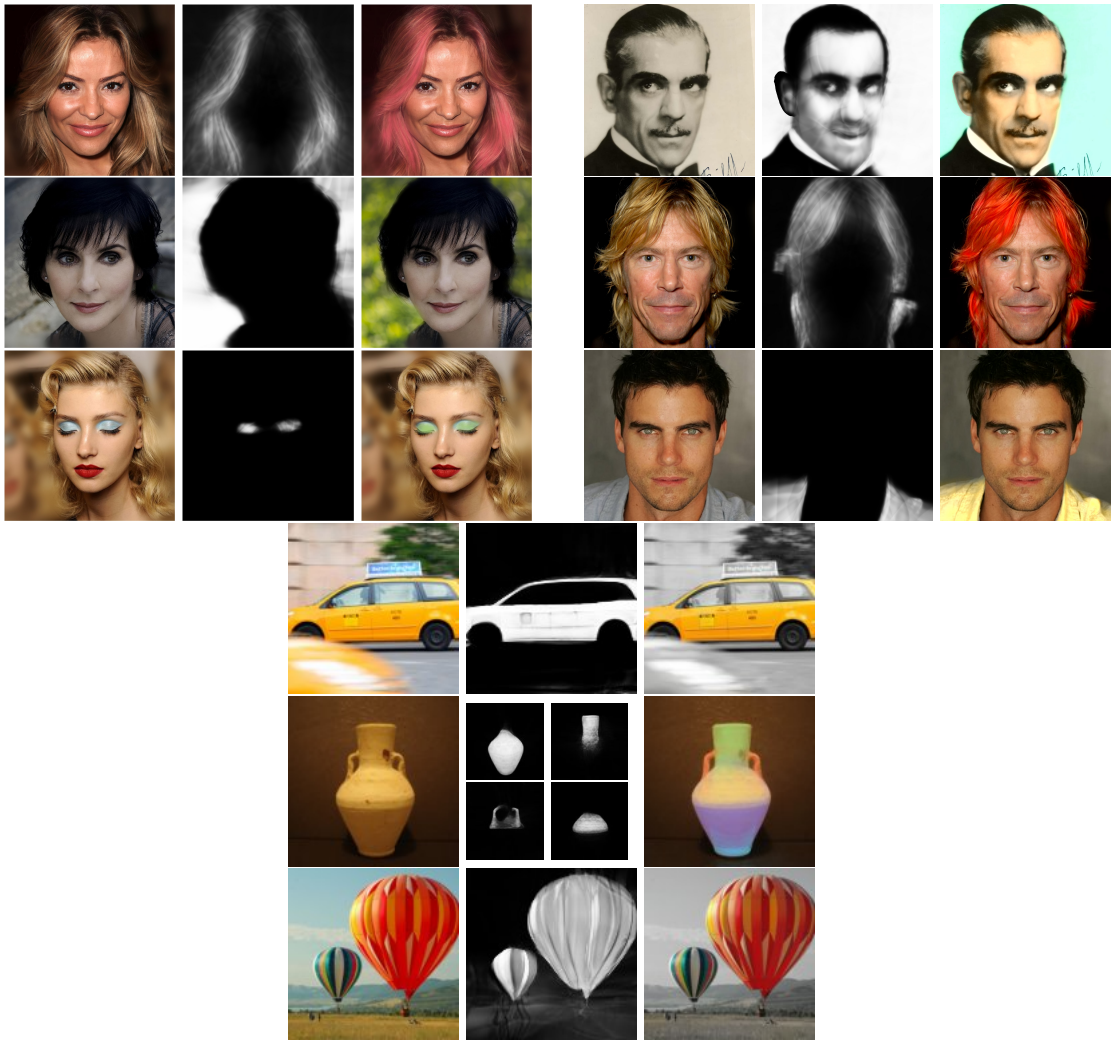


Figure 4.4: Some editings on CelebA and ImageNet, using little supervision (mask selection in one click and new style/color selection). Note that the CelebA editings are performed on 1024×1024 images. Left: original; center: mask; right: edit.

Image editing. Image editing from raw pixels can be time consuming. Using our generated masks, it is possible to alter the original image by applying edits such as luminosity or color modifications on the region specified by a mask. Fig. 4.3 shows an interface we designed for such editing showing the masks corresponding to the image. It avoids going through the tedious process of defining a blending mask manually. The learned masks capture the main components of the image, such as the background, face, hairs, lips. Fig. 4.4 demonstrate a variety of editing we performed and the associated masks. Our approach works well on the CelebA dataset, and allows to make simple image modifications on the more challenging ImageNet images. To optimize our results on ImageNet, the edits of Fig. 4.4 are obtained by finetuning our model on images of each object class.

Attribute-based image retrieval. A t-SNE (Maaten and Hinton, 2008) visualization of the mask parameters obtained on CelebA is shown in Fig. 4.5. Different clusters of masks are clearly visible, for backgrounds, hairs, face shadows, etc. This experiment highlights

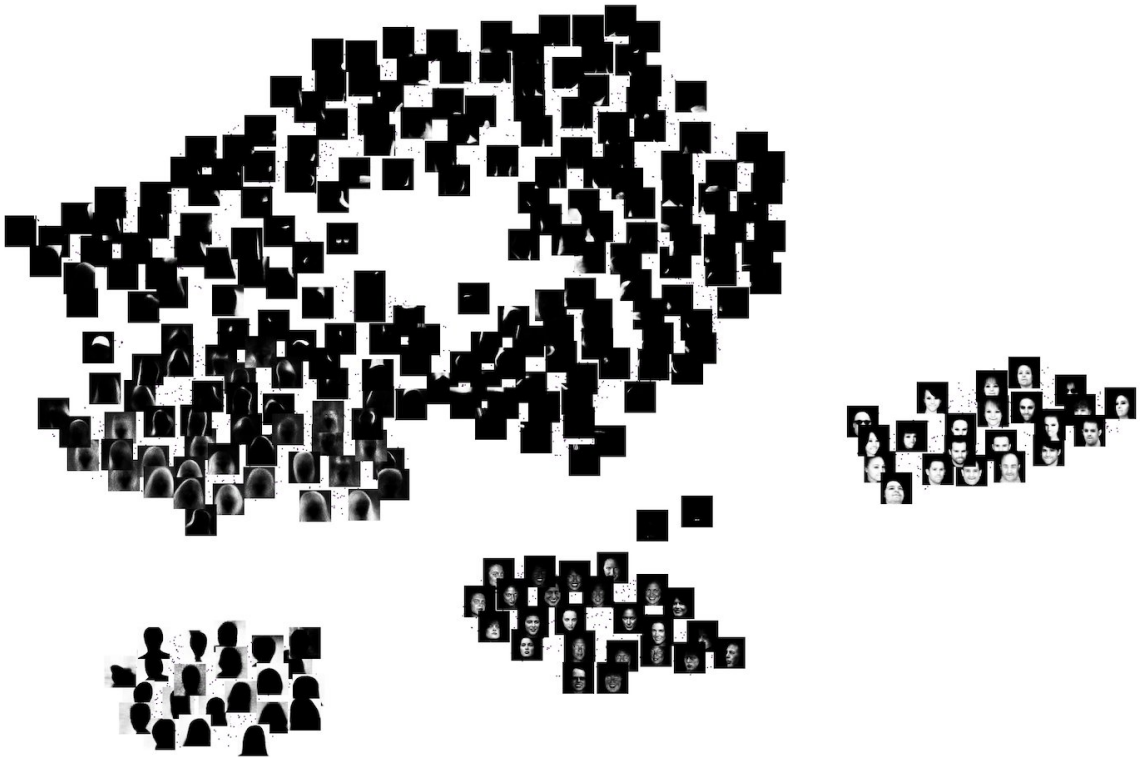


Figure 4.5: t-SNE visualization of masks obtained from 5000 reconstructions on CelebA.

the fact our approach naturally extract semantic components of face images.

Our approach may be used in an image search content: given a query image, a user can select a mask that displays a particular attribute of interest and search for images which decomposition includes similar masks. Suppose we would like to retrieve pictures of people wearing a hat as displayed in a query image, we can easily extract the mask that corresponds to the hat in our decomposition and its parameters. Nearest neighbor for different masks, using a cosine similarity distance between mask parameters $\mathbf{p}$ are provided in Fig. 4.6. Note how different masks extracted from the same query image lead to very different retrieved images. Such a strategy could potentially be used for efficient image annotation or few-shot learning. We evaluated oneshot nearest neighbor classification for the "Wearing Hat" and "Eyeglasses" categories in CelebA using the hat and glasses examples shown in Fig. 4.6, and obtained respectively 34% and 49% average precision. Results for eyeglasses attribute were especially impressive with 33% recall at 98% precision, compared to a low recall (less than 10% at 98% precision) for a baseline using cosine distance between features of a Resnet18 trained on ImageNet.

Vector image generation. Producing vectorized images is often essential for design applications. We demonstrate in Fig. 4.7(a) the potential of our approach for producing a continuous vector image from a low resolution bitmap. Here, we train our network on the MNIST dataset (28×28), but generate the output at resolution 1024×1024. Compared to bilinear interpolation, the image we generate presents less artifacts.

We finally compare our model with SPIRAL (Ganin et al., 2018) on a few images from CelebA dataset published in Ganin et al. (2018). SPIRAL is the approach the most closely

related to ours in the sense that it is an iterative deep approach for reconstructing an image and extracting its structure only using a few color strokes and that it can produce vector results. We report SPIRAL results using 20-step episodes. In each episode, a tuple of 8 discrete decisions is estimated, resulting in a total of 160 parameters for reconstruction. Our results shown in Fig. 4.7(b) are obtained with a model using 10 iterations and 10 mask parameters. Although we do not reproduce the stroke gesture for drawing each mask as it is the case in SPIRAL, our results reconstruct the original images much better.

4.5.3 Architecture and training choices

L1, perceptual and adversarial loss. In Fig. 4.8, we show how the perceptual loss allows to obtain qualitatively better reconstructions than these obtained with an ℓ_1 loss. Training our model with an additional adversarial loss enhances further the sharpness of the reconstructions.

In the remainder of this section, we trained our models with an ℓ_1 loss which results in easier quality assessment using standard image similarity metrics.

Comparison to baselines. As discussed in Section 4.4.3, every component of our model is important to obtain reconstructions similar to the target. To show that, we provide comparisons between different versions of our model and baselines using PSNR and MS-SSIM metrics. Each baseline consists of an auto-encoder where the encoder is a residual network (ResNet-18, same as our model) producing a latent code z and different types of decoders. The different baselines, depicted in Fig. 4.9 with a summary of their properties, are designed to validate each component of our architecture:

A. *ResNet AE:* using as a decoder a ResNet with convolutions, residual connections, and upsampling similarly to the architecture used in Miyato et al. (2018); Kurach et al. (2019).

B. *MLP AE:* using as decoder an MLP with a $3 \times W \times H$ output.

C. *Vect. AE:* the decoder computes the resulting image R as a function f of the coordinates (x, y) of a pixel in image space and the latent code z as $R(x, y) = f(x, y, z)$. Here f is an MLP similar to the one used in our mask generation network, but with a 3-channel output instead of a 1-channel as for the mask.

D. *Ours One-shot:* generates all the mask parameters $\mathbf{p_t}$ and colors $\mathbf{c}_t$ in one pass, instead of recursively. The MLP then processes each $\mathbf{p_t}$ separately leading to different masks to be assembled in the blending module as in our approach.

E. *Ours ResNet:* using a ResNet decoder to generate masks M_t and otherwise similar to our method, iteratively blending the masks with one color onto the canvas, in our experiments we started with a black canvas.

Table 4.1 shows a quantitative comparison of results obtained by our model and baselines trained with ℓ_1 loss and for the same bottleneck $|z| = 320$. This corresponds to a size of parameters of $P = |z|/N - 3$ where 3 is the number of parameters used for color prediction. On both datasets, our approach (F) clearly outperforms the baselines

which produce vector outputs, either in one layer (C) or with one-shot parameters prediction (D). Interestingly, a parametric generation (C) is itself better than directly using an MLP to predict pixel values (B). Finally, our approach (F) has quantitative reconstruction results similar to the ResNet baselines (A and E).

However, our method has two strong advantages over Resnet generations. First, it produces vector outputs. Second, it produces more interpretable masks. This can be seen in Fig. 4.10 where we compare the masks resulting from (E) and (F). Our method (F) captures much better the different components of face images, notably the hairs, while the masks of (E) include several different component in the image, with a first mask covering both hairs and faces.

	ImageNet		CelebA	
	PSNR	MSSIM	PSNR	MSSIM
A. MLP AE	16.45	0.46	19.69	0.78
C. Vect. AE	17.95	0.62	20.99	0.82
D. Ours One-shot	20.00	0.77	23.13	0.89
E. Ours Resnet	**21.05**	**0.82**	**24.67**	**0.92**
F. Ours	**21.03**	**0.82**	24.02	0.90

Table 4.1: Comparing the quality of reconstruction on ImageNet and CelebA using a bottleneck z of size 320 (10 masks for iterative approaches).

	$T = 5$		$T = 10$		$T = 20$	
	One-shot	Ours	One-shot	Ours	One-shot	Ours
PSNR	21.97	23.07	22.25	24.2	22.37	24
Time(h) 95% PSNR	7.6	9.8	12.1	16.9	19.8	36.5
Testing time (ms)	12	32	18	65	31	129

Table 4.2: Comparison of our recursive strategy with the One-shot approach, in terms of reconstruction quality (PSNR) and training time required to reach 95% of its best achievable PSNR at full convergence on CelebA. The inference time does not exceeds 0.2 seconds.

Recursive setup and computational cost. There is of course a computational cost to our recursive approach. In Table 4.2, we compare the PSNR and computation time for the same total number of parameters (320) but using different number of masks T, both for our approach and the one-shot baseline. Interestingly, the quality of the reconstruction improves with the number of masks for both approaches, our approach being consistently more than a point PSNR better than the one-shot baseline. However, as expected our approach is slower than the baseline both for training and testing, and the cost increases with the number of masks.

Table 4.3 evaluates the reconstruction quality when recomposing images at test time with a larger number of masks N than the $T = 10$ masks used at training time. On both

datasets, the PSNR increases by almost a point with additional masks. This is another advantage of our recursive approach.

	ImageNet				CelebA			
N	10	20	40	80	10	20	40	80
PSNR	20.97	21.72	21.83	21.84	24.02	24.82	24.86	24.86

Table 4.3: Forwarding more masks at test time improves reconstruction. These results with N masks forwards are obtained with a model trained using T=10 masks ($|z| = 320$).

4.6 Conclusion

We have presented a new paradigm for image reconstruction using a succession of single-color parametric layers. This decomposition, learned without supervision, enables image editing from the masks generated for each layer.

We also show how the learned mask parameters may be exploited in a retrieval context. Moreover, our experiments prove that our image reconstruction results are competitive with convolution-based auto-encoders.

Our work is the first to showcase the potential of a deep vector image architecture for real world applications. Furthermore, while our model is introduced in an image reconstruction setting, it may be extended to adversarial image generation, where generating high resolution images is challenging. We are aware of risks surrounding manipulated media but we believe the importance of publishing this work openly may have benefits in augmented reality filters or more realistic virtual reality.

We think that because of its differences and its advantages for user interaction, our method will inspire new approaches.

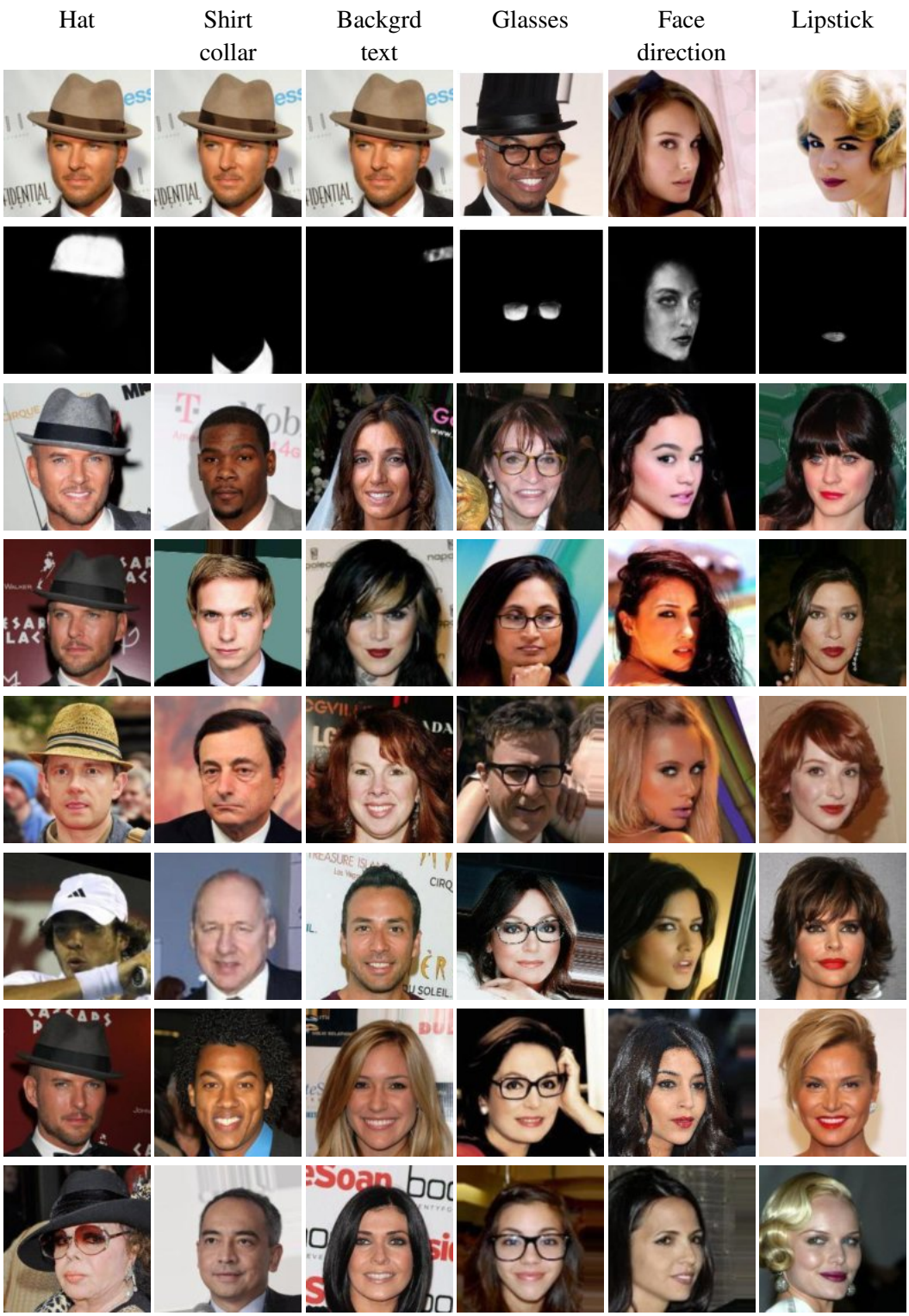
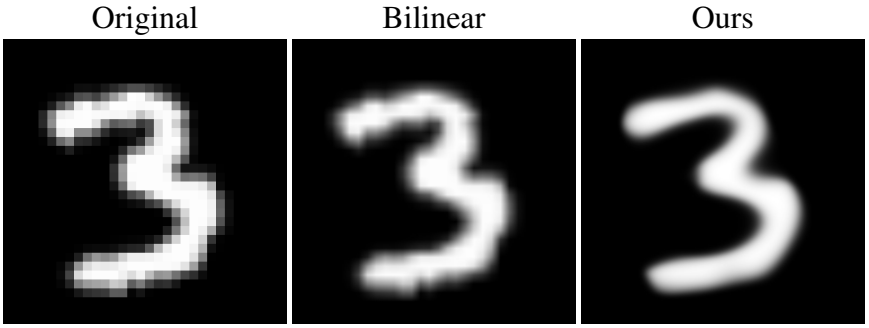
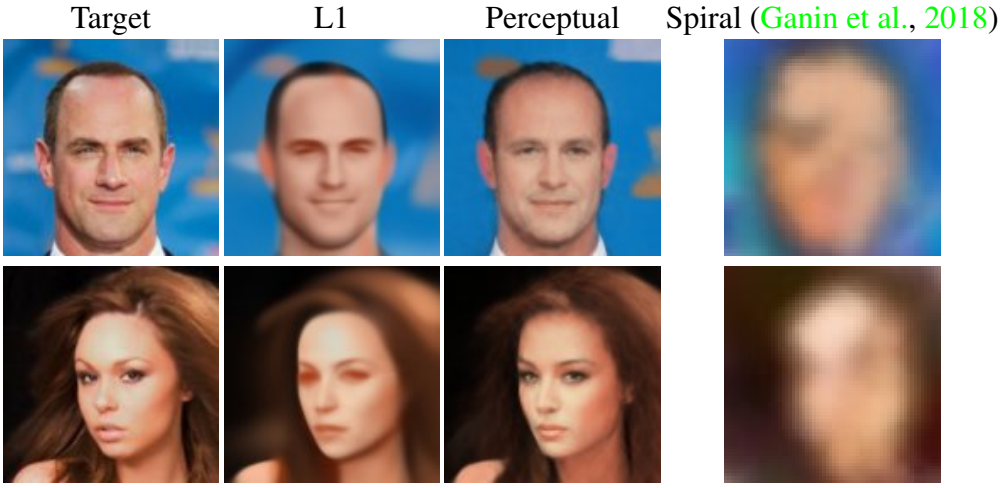


Figure 4.6: Given a target image and a mask of an area of interest extracted from it, a nearest neighbor search in the learned mask parameter space allows the retrieval of images sharing the desired attribute with the target.



(a) Vectorization: reconstructions of MNIST images.



(b) Comparison with SPIRAL (Ganin et al., 2018) on CelebA.

Figure 4.7: Our model learns a vectorized mask representation that can be generated at any resolution without interpolation artifacts.

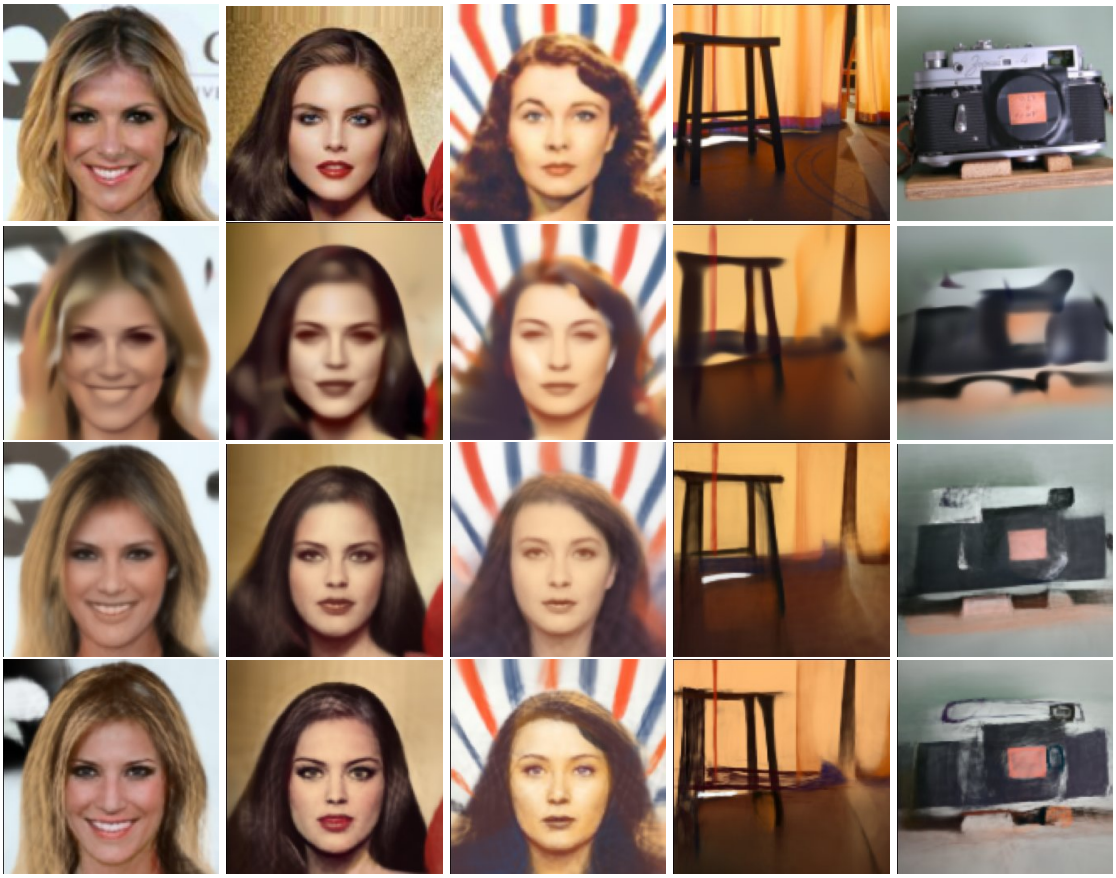


Figure 4.8: Training with perceptual and adversarial loss allows our model to reach more convincing details in the reconstructions. From top to bottom: Original images, ℓ_1 reconstruction, using perceptual loss, adding adversarial loss.

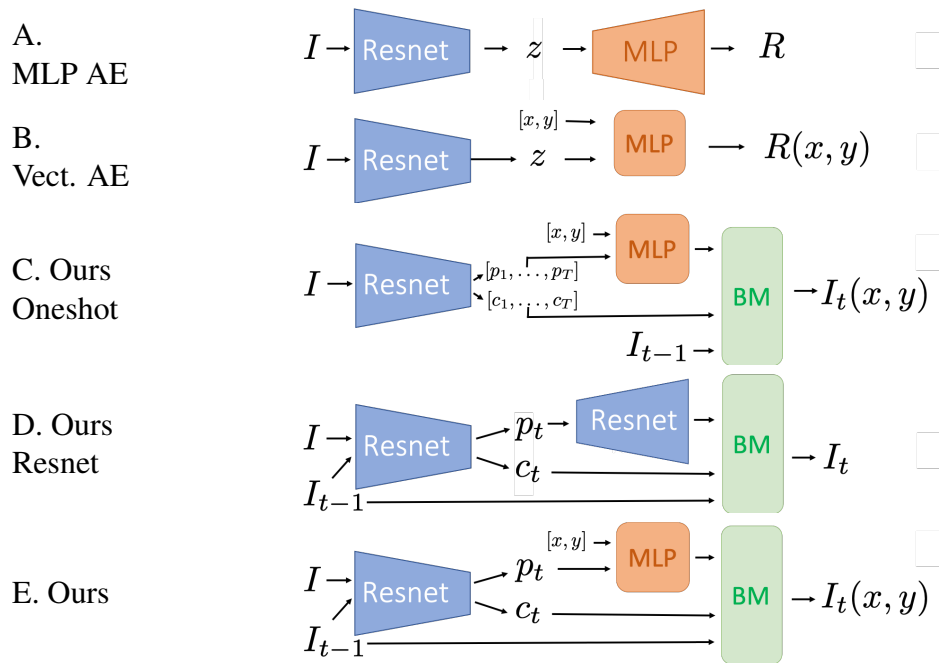


Figure 4.9: Considered baselines. R: reconstructed image; BM: Mask blending module of Fig. 4.2.

Figure 4.10: Comparison of masks obtained with our approach (F) (bottom), with these obtained by our iterative ResNet baseline (E) (top).

Chapter 5

Dataset impact on image representation

5.1 Abstract

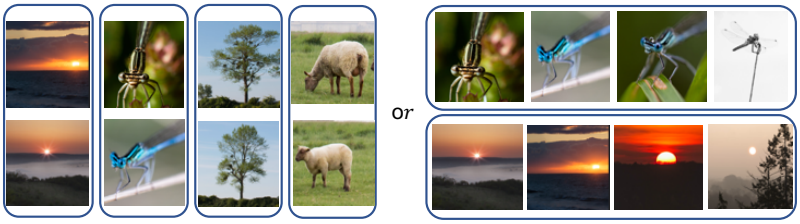
Learning good image representations is crucial for multiple computer vision tasks such as image similarity search and transfer learning for downstream tasks. In chapter 3, our introduced model for generating original image compositions relies mainly on an image search of similar foregrounds to a given object. Deep image features provide a good measure of object similarity whether trained in a supervised or weakly-supervised manner. However, learning good features with few labelled data remains a major challenge. There have been many works in the deep metric learning literature that aim for better features both in terms of image retrieval and generalization to new unseen concepts.

The quality and generality of deep image features is crucially determined by the data they have been trained on, but little is known about this often overlooked effect. In this chapter, we systematically study the effect of variations in the training data by evaluating deep features trained on different image sets in a few-shot classification setting. The experimental protocol we define allows us to explore key practical questions. What is the influence of the similarity between base and test classes? Given a fixed annotation budget, what is the optimal trade-off between the number of images per class and the number of classes? Given a fixed dataset, can features be improved by splitting or combining different classes? Should simple or diverse classes be annotated? In a wide range of experiments, we provide clear answers to these questions on the mini-ImageNet, ImageNet and CUB-200 benchmarks. We also show how the base dataset design can improve performance in few-shot classification more drastically than replacing a simple baseline by an advanced state of the art algorithm.

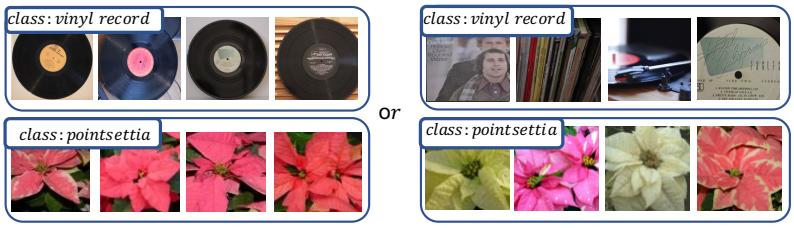
Keywords: Dataset labeling, few-shot classification, meta-learning.

5.2 Introduction

Deep features can be trained on a base dataset and provide good descriptors on new images (Sharif Razavian et al., 2014; Oquab et al., 2014). The importance of large scale image annotation for the base training is now fully recognized and many efforts are dedicated to creating very large scale datasets. However, little is known on the desirable properties of such dataset, even for standard image classification tasks. To evaluate the



(a) Annotate more classes or more examples per class?



(b) Build classes using more or less diverse images?

Figure 5.1: How should we design the base training dataset and how will it influence the features? a) Many classes with few examples / few classes with many examples; b) Simple or diverse base training images.

impact of the dataset on the quality of learned features, we propose an experimental protocol based on few-shot classification. In this setting, a first model is typically trained to extract features on a base training dataset, and in a second classification stage, features are used to label images of novel classes given only few exemplars. Beyond the interest of few-shot classification itself, our protocol is well suited to vary specific parameters in the base training set and answer specific questions about its design, such as the ones presented in Fig. 5.1.

We believe this work is the first to study, with a consistent approach, the importance of the similarity of training and test data, the suitable trade-off between the number of classes and the number of images per class, the possibility of defining better labels for a given set of images, and the optimal diversity and complexity of the images and classes to annotate. Past studies have mostly focused on feature transfer between datasets and tasks (Huh et al., 2016; Zamir et al., 2018). The study most related to ours is likely Huh et al. (2016), which asks the question "What makes ImageNet good for transfer learning?". The authors present a variety of experiments on transferring features trained on ImageNet to SUN (Xiao et al., 2010) and Pascal VOC classification and detection (Everingham et al., 2010), as well as a one-shot experiment on ImageNet. However, using AlexNet fc7 features (Krizhevsky et al., 2012), and often relying on the WordNet hierarchy (Fellbaum, 1998), the authors find that variations of the base training dataset do not significantly affect transfer performance, in particular for the balance between image-per-class and classes. This is in strong contrast with our results, which outline the importance of this trade-off in our setup. We believe this might partially be due to the importance of the effect of transfer between datasets, which overshadows the differences in the learned features. Our few-shot learning setting precisely allows to focus on the influence of the training data without considering the complex issues of domain or task transfer.

Our work also aims at outlining data collection strategies and research directions that might lead to new performance boosts. Indeed, several works (Chitta et al., 2019; Tri-

antafillou et al., 2020) have recently stressed the limitations of performance improvements brought when training on larger datasets, obtained for example by aggregating datasets (Triantafillou et al., 2020). On the contrary, Ge and Yu (2017) show that performance can be improved using a "Selective Joint Fine-Tuning" strategy for transfer learning, selecting only images in the source dataset with low level feature similar to the target dataset and training jointly on both. Our results give insights on why it might happen, showing in particular that a limited number of images per class is often sufficient to obtain good features. Code is available at `imagine.enpc.fr/~sbaio/fewshot_dataset_design`.

Contribution. Our main contribution is an experimental protocol to systematically study the influence of the characteristics of the base training dataset on the resulting deep features for few-shot classification. It leads us to the following key conclusions:

- The similarity of the base training classes and the test classes has a crucial effect and standard datasets for few-shot learning consider only a very specific scenario.
- For a fixed annotation budget, the trade-off between the number of classes and the number of images per class has a major effect on the final performance. The best trade-off usually corresponds to much fewer images per class (~ 60) than collected in most datasets.
- If a dataset with a sub-optimal class number is already available, we demonstrate that a performance boost can be achieved by grouping or splitting classes. While oracle features work best, we show that class grouping can be achieved using self-supervised features.
- Class diversity and difficulty also have an independent influence, easier classes with lower than average diversity leading to better few-shot performances.

While we focus most of our analysis on a single few-shot classification approach and architecture backbone, key experiments for other methods and architectures demonstrate the generality of our results.

5.3 Related Work

5.3.1 Data selection and sampling

Training image selection is often tackled through the lens of **active learning** (Cohn et al., 1994). The goal of active learning is to select a subset of samples to label when training a model, while obtaining similar performance as in the case where the full dataset is annotated. A complete review of classical active learning approaches is beyond the scope of this work and can be found in Settles (2009). A common strategy is to remove redundancy from datasets by designing acquisition functions (entropy, mutual information, and error count) (Gal et al., 2017; Chitta et al., 2019) to better sample training data. Specifically, Chitta et al. (2019) introduce an "Adaptive Dataset Subsampling" approach designed to remove redundant samples in datasets. It predicts the uncertainty of ensemble of models to encourage the selection of samples with high "disagreement". Another approach is to select samples close to the boundary decision of the model, which in the case of deep networks can be done using adversarial examples (Ducoffe and Precioso, 2018). In Sener

and Savarese (2017), the authors adapt active learning strategies to batch training of neural networks and evaluate their method in a transfer learning setting. While these approaches select specific training samples based on their diversity or difficulty, they typically focus on performance on a fixed dataset and classes, and do not analyze performance of learned features on new classes as in our few-shot setting.

Related to active learning is the question of online **sampling strategies** to improve the training with fixed, large datasets (Fan et al., 2017; London, 2017; Buda et al., 2017; Katharopoulos and Fleuret, 2018). For instance, the study of Buda et al. (2017) on class imbalance highlights over-sampling or under-sampling strategies that are privileged in many works. Fan et al. (2017) and Katharopoulos and Fleuret (2018) propose respectively reinforcement learning and importance sampling strategies to select the samples which lead to faster convergence for SGD.

The spirit of our work is more similar to studies that try to understand key properties of good training samples to **remove unnecessary samples** from large datasets. Focusing on the deep training process and inspired by active SVM learning approaches, Vodrahalli et al. (2018) explore using the gradient magnitude as a measure of the importance of training images. However using this measure to select training examples leads to poor performances on CIFAR and ImageNet. Birodkar et al. (2019) identify redundancies in datasets such as ImageNet and CIFAR using agglomerative clustering (Defays, 1977). Similar to us, they use features from a network pre-trained on the full dataset to compute an oracle similarity measure between the samples. However, their focus is to demonstrate that it is possible to slightly reduce the size of datasets (10%) without harming test performance, and they do not explore further the desirable properties of a training dataset.

5.3.2 Few-shot classification

The goal of few-shot image classification is to be able to classify images from novel classes using only a few labeled examples, relying on a large base dataset of annotated images from other classes. Among the many deep learning approaches, the pioneer Matching networks (Vinyals et al., 2016) and Prototypical networks (Snell et al., 2017) tackle the problem from a metric learning perspective. Both methods are meta-learning approaches, i.e. they train a model to learn from sampled classification episodes similar to those of evaluation. MatchingNet considers the cosine similarity to compute an attention over the support set, while ProtoNet employs an ℓ_2 between the query and the class mean of support features.

Recently, Chen et al. (2019) revisited few-shot classification and showed that the simple, meta-learning free, Cosine Classifier baseline introduced in Gidaris and Komodakis (2018) performs better or on par with more sophisticated approaches. Notably, its results on the CUB and Mini-ImageNet benchmarks were close to the state-of-the-art (Antoniou and Storkey, 2019; Lee et al., 2019). Many more approaches have been proposed even more recently in this very active research area (e.g. (Rusu et al., 2019; Li et al., 2019)), including approaches relying on other self-supervised tasks (e.g. (Gidaris et al., 2019a)) and semi-supervised approaches (e.g. (Kim et al., 2019; Liu et al., 2019; Hu et al., 2019)), but a complete review is outside the scope of this work, and exploration of novel methods orthogonal to our goal.

The choice of the base dataset remains indeed largely unexplored in previous studies,

whereas we show that it has a huge impact on the performance, and different choices of base datasets might lead to different optimal approaches. The Meta-dataset (Triantafillou et al., 2020) study is related to our work from the perspective of analyzing dataset impact on few-shot performance. However, it investigates the effect of meta-training hyperparameters, while our study focuses on how the base dataset design can improve few-shot classification performance. More recently, Zhou et al. (2020) investigate the same question of selecting base classes for few-shot learning, leading to a performance better than that of random choice, while highlighting the importance of base dataset selection in few-shot learning.

Since a Cosine Classifier (CC) with a Wide ResNet backbone is widely recognized as a strong baseline (Gidaris and Komodakis, 2018; Gidaris et al., 2019a; Chen et al., 2019; Wang et al., 2019), we use it as reference, but also report results with two other classical algorithms, namely MatchingNet and ProtoNet.

The classical benchmarks for few-shot evaluation on which we build and evaluate are listed below. Note this is not an exhaustive review, but a selection of diverse datasets which are suited to our goals.

Mini-ImageNet benchmark. Mini-ImageNet is a common benchmark for few-shot learning of small resolution images (Vinyals et al., 2016; Ravi and Larochelle, 2017). It includes 600K images from 100 random classes sampled from the ImageNet-1K (Deng et al., 2009b) dataset and downsampled to 84×84 resolution. It has a standard split of base training, validation and test classes of 64, 16, and 20 classes respectively.

ImageNet benchmark. For high-resolution images, we consider the few-shot learning benchmark proposed by Hariharan and Girshick (2017); Wang et al. (2018b). This benchmark splits the ImageNet-1K dataset into 389 base training, 300 validation and 311 novel classes. The base training set contains 497350 images.

CUB benchmark. For fine-grained classification, we experiment with the CUB-200-2011 dataset (Wah et al., 2011). It contains 11,788 images from 200 classes, each class containing between 40 to 60 images. Following Hilliard et al. (2018); Chen et al. (2019) we resize the images to 84×84 pixels and use the standard splits in 100 base, 50 validation and 50 novel classes and use exactly the same evaluation protocol as for mini-ImageNet.

5.4 Method

In this section, we present the different components of our analysis. First, we explain in detail the main few-shot learning approach that we use to evaluate the influence of training data. Second, we present the large base dataset we use to sample training sets. Third, we discuss the different descriptors of images and classes that we consider, the different splitting and grouping strategies we use for dataset relabeling and the class selection methods we analyze. Finally we give details on architecture and training.

5.4.1 Dataset evaluation using few-shot classification

Few-shot image classification aims at classifying test examples in novel categories using only a few annotated examples per category and typically relying on a larger base training set with annotated data for training categories. We use the simple but efficient nearest neighbor based approach, visualized in Fig. 5.2.

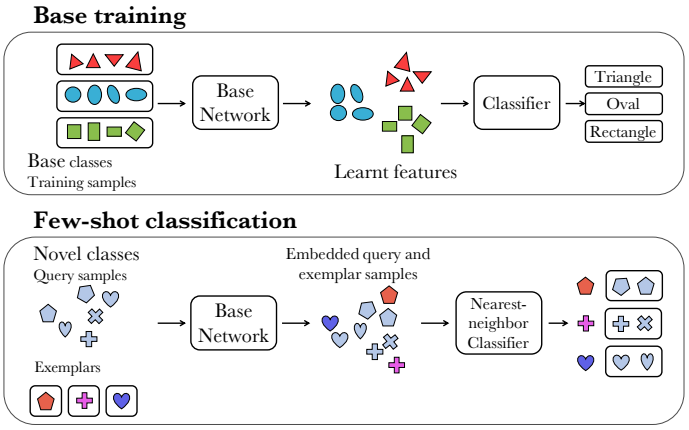


Figure 5.2: Illustration of our few-shot learning framework. We train a feature extractor together with a classifier on base training classes. Then, we evaluate the few-shot classification performance of this learned feature extractor to classify novel unseen classes with few annotated examples using a nearest neighbor classifier.

More precisely, we start by training a feature extractor f with a cosine classifier on base categories (Fig. 5.2 top). Then, we define a linear classifier for the novel classes as follows: if z_i for $i = 1...N$ are the labelled examples for a given novel class, we define the classifier weights w for this class as:

$$w = \frac{1}{N} \sum_{i=1}^{N} \frac{f(z_i)}{\|f(z_i)\|}. \tag{5.1}$$

In other words, we associate each test image to the novel class for which its average cosine similarity with the examples from this novel class is the highest. Previous work on few-shot learning focuses on algorithm design for improving the classifier defined on new labels. Instead, we explore the orthogonal dimension of base training dataset and compare the same baseline classifier using features trained on different base datasets.

5.4.2 A large base dataset, ImageNet-6K

To investigate a wide variety of base training datasets, we design the ImageNet-6K dataset from which we sample images and classes for our experiments. We require both a large number of classes and a large number of images per class, to allow very diverse image selections, class splittings or groupings. We define ImageNet-6K as the subset from the ImageNet-22K dataset (Russakovsky et al., 2015; Deng et al., 2009b) containing the largest 6K classes, excluding ImageNet-1K classes. Image duplicates are removed automatically as done in Sablayrolles et al. (2018). Each class has more than 900 images, with a total number of 7135116 images. For experiments on mini-ImageNet and CUB, we downsample the images to 84×84, and dub the resulting dataset MiniIN6K. For CUB experiments, to avoid training on classes corresponding to the CUB test set, we additionally look for the most similar images to each of the 2953 images of CUB test set using our oracle features (see Section 5.4.3), and completely remove the 296 classes they belong to. We denote this base dataset MiniIN6K*.

5.4.3 Class definition and sampling strategies

Image and class representation. In most experiments, we represent images by what we call *oracle features*, i.e. features trained on our IN6k or miniIN6K datasets. These features can be expected to provide a good notion of distance between images, but can of course not be used in a practical scenario where no large annotated dataset is available. Each class is represented by its average feature as defined in Equation 5.1. This class representation can be used for examples to select training classes close or far from the test classes, or to group similar classes.

We also report results with several alternative representations and metrics. In particular, we experiment with *self-supervised features*, which could be computed on a new type of images from a non-annotated dataset. We tried using features from RotNet (Gidaris et al., 2019b), DeepCluster (Caron et al., 2018), and MoCo (He et al., 2019a) approaches, and obtained stronger results with MoCo features which we report. MoCo exploits the self-supervised feature clustering idea and builds a feature dictionary using a contrastive loss. As an additional baseline we report results using deep features with randomly initialized weights and updated batch normalization layers during 1 epoch of miniIN6k. Finally, similar to several prior works, we experiment using the WordNet (Fellbaum, 1998) hierarchy to compute similarity between classes based on the shortest path that relates their synsets and on their respective depths.

Defining new classes. A natural question is whether for a fixed set of images, different labels could be used to train a better feature extractor.

Given a set of images, we propose to use existing class labels to define new classes by splitting or merging them. Using K-means to cluster images or classes would lead to unbalanced classes, we thus used different strategies for splitting and grouping, which give consistently better performance than using K-means:

- *Class splitting.* We iteratively split in half every class along the principal component computed over the features of the class images. We refer to this strategy as BPC (Bisection along Principal Component).
- *Class grouping.* To merge classes, we use a simple greedy algorithm which defines meta-classes by merging the two closest classes using their mean features, and repeat the same process for unprocessed classes recursively.

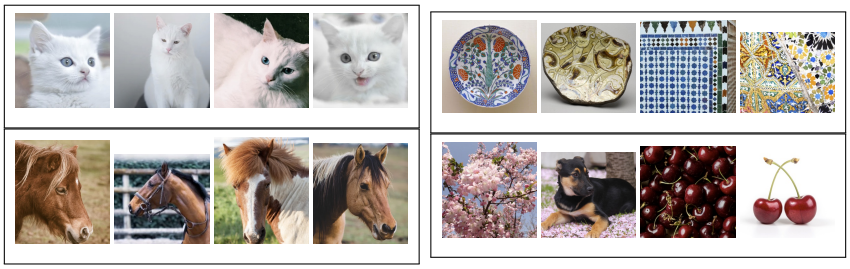
We display examples of resulting grouped and splitted classes in Fig. 5.3.

Measuring class diversity and difficulty. One of the questions we ask is whether class diversity impacts the trained features' few-shot performance. We therefore analyze results by sampling classes more or less according to their diversity and difficulty:

- *Class diversity.* We use the variance of the normalized features as a measure of class diversity. We show in Fig. 5.3 (c,d) examples of least and most diverse classes. Classes with low feature variance consist of very similar looking objects or simple visual concepts while the ones with high feature variance represent abstract concepts or include very diverse images.
- *Class difficulty.* To measure the difficulty of a class, we use the validation accuracy of our oracle classifier.

(a) Images from two meta-classes obtained by grouping four dataset classes.

(b) Images from sub-classes obtained by splitting classes “Sunflower” and “Tulips”.



c) Least diverse classes (Horse, Angora cat ...)

d) Most diverse classes (Arabesque, cherry...)

Figure 5.3: a) Images from **meta-classes obtained by grouping** dataset classes using pre-trained features. Each line represents a meta-class. b) Examples of **sub-classes obtained by splitting** dataset classes using pre-trained features. Each column represents a sub-class. c), d) Images from least or most diverse classes from miniIN6k, with one line per class.

5.4.4 Architecture and training details

We use different architectures and training methods in our experiments. Similar to previous works (Wang et al., 2019; Chen et al., 2019), we perform our experiments using the WRN28-10, ResNet10, ResNet18 and Conv4 architectures. The ResNet architectures are adapted to handle 84×84 images by replacing the first convolution with a kernel size of 3 and stride of 1 and removing the first max pooling layer. In addition to the cosine classifier described in Section 5.4.1, we experiment with the classical Prototypical Networks (Snell et al., 2017) and Matching Networks (Vinyals et al., 2016).

Since we compare different training datasets, we adapt the training schedule depending on the size of the training dataset and the method. For example on MiniIN-6k, we train Prototypical Networks and Matching Networks for 150k episodes, while when training on smaller size datasets we use 40k episodes as in Chen et al. (2019). We also use fewer query images per class when training on classes with not enough images per class for Prototypical and Matching Networks.

When training a Cosine Classifier, we train using an SGD optimizer with momentum

of 0.9 and weight decay of 5.10^{-4} for 90 epochs starting with an initial learning rate of 0.05 and dividing it by 10 every 30 epochs. We also use a learning rate warmup for the first 6K iterations, that we found beneficial for stabilizing the training and limiting the variance of the results. For large datasets with more than 10^6 images, we use a batch size of 256 and 8 GPUs to speed up the training convergence, while for smaller datasets (most of our experiments are done using datasets of 38400 images, as in MiniIN training set), we use a batch size of 64 images and train on a single GPU. During training, we use a balanced class sampler that ensures sampled images come from a uniform distribution over the classes regardless of their number of images.

On the ImageNet benchmark, we use a ResNet-34 network and trained for 150K dividing the learning rate by 10 after 120K, 135K and 145K iterations using a batch size of 256 on 1 GPU.

Following common practices, during evaluation, we compute the average top-1 accuracy on 15 query examples over 10k episodes sampled from the test set on 5-way tasks for miniIN and CUB benchmarks, while we compute the top-5 accuracy on 6 query examples over 250-way tasks on the ImageNet benchmark.

5.5 Analysis

In this section, we provide an analysis of the components of our method by evaluating the influence of the training data in multiple setups. We first start by validating the impact of the base dataset size on the quality of learned features and its similarity with test data. Second, we consider the tradeoff between the number of classes and images per class for a fixed number of annotations. Third, we explore how redefining class labels through splitting and grouping allows to confirm the previously observed tradeoff before finally considering the class selection bias from the perspective of diversity and difficulty.

5.5.1 Importance of base data and its similarity to test data

We start by validating the importance of the base training dataset for the few-shot classification, both in terms of size and of the selection of classes. In table 5.1, we report five shot results on the CUB and MiniIN datasets, the one shot results are available in table 5.2. We write N the total number of images in the dataset and C the number of classes. On the miniIN benchmark, we observe that our implementation of the strong CC baseline using a WRN backbone yields slightly better performance using miniIN base classes than the ones reported in Gidaris et al. (2019a); Lee et al. (2019)(76.59). We validate the consistency of our observations by varying algorithms and architectures using the codebase of Chen et al. (2019).

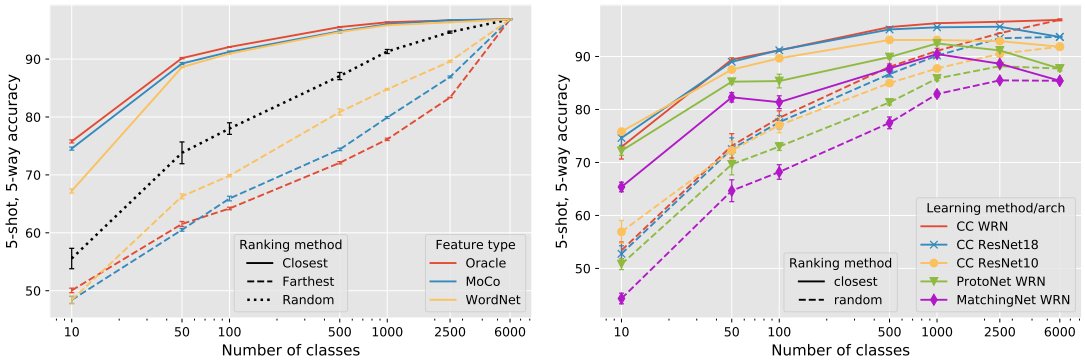
Our first finding is that using the whole miniIN-6K dataset for the base training boosts the performance on miniIN by a very large amount, 20% and 18% for 1-shot and 5-shot classification respectively, compared to training on 64 miniIN base classes. Training on IN-6K images also results in a large 10% boosts in 5-shot top-5 accuracy on ImageNet benchmark. Another interesting result is that sampling random datasets of 64 classes and 600 images per class leads to a 5-shot performance of 75.48% on MiniIN clearly below the one using the base classes from miniIN 78.95%. A similar observation can be made for

	MiniIN			CUB		
	MiniIN N=38400 $C = 64$	MiniIN6K Random N=38400 $C = 64$	MiniIN6K $N\approx$7,1.10^6 C=6000	CUB N=5885 $C = 100$	MiniIN6K* Random N=38400 $C = 64$	MiniIN6K* $N\approx$6,8.10^6 $C = 5704$
WRN PN	73.64±0.84	70.26±1.30	85.14±0.28	87.84±0.42	52.51±1.57	68.62±0.5
WRN MN	69.19±0.36	65.45±1.87	82.12±0.27	85.08±0.62	46.32±0.72	59.90±0.45
WRN CC	**78.95**±0.24	**75.48**±1.53	**96.91**±0.14	**90.32**±0.14	**58.03**±1.43	**90.89**±0.10
Conv4	65.99±0.04	64.05±0.75	74.56±0.12	80.71±0.15	56.44±0.63	66.81±0.30
ResNet10 CC	76.99±0.07	74.17±1.42	91.84±0.06	89.07±0.15	57.01±1.44	82.20±0.44
ResNet18 CC	78.29±0.05	75.14±1.58	93.36±0.19	89.99±0.07	56.64±1.28	88.32±0.23

Table 5.1: 5-shot, 5-way accuracy on MiniIN and CUB test sets using different base training data, algorithms and backbones. PN: Prototype Networks Snell et al. (2017). MN: Matching Networks Vinyals et al. (2016). CC: Cosine Classifier. WRN: Wide ResNet28-10. MiniIN6K (resp. MiniIN6K*) Random: 600 images from 64 classes sampled randomly from MiniIN6K (resp. MiniIN6K*). We evaluate the variances over 3 different runs.

	MiniIN			CUB		
	MiniIN N=38400 $C = 64$	MiniIN6K Random N=38400 $C = 64$	MiniIN6K $N\approx$7,1.10^6 C=6000	CUB N=5885 $C = 100$	MiniIN6K* Random N=38400 $C = 64$	MiniIN6K* $N\approx$6,8.10^6 $C = 5704$
WRN	61.62±0.17	58.49±2.29	85.40±0.15	76.73±0.40	41.62±0.93	73.51±0.21
Conv4	48.62±0.09	46.87±0.70	56.09±0.16	61.21±0.16	39.65±0.71	47.01±0.26
ResNet10	59.06±0.35	56.06±1.74	74.42±0.20	74.48±0.42	40.92±0.51	57.81±0.43
ResNet18	60.85±0.17	57.51±1.79	81.42±0.20	76.13±0.39	40.90±0.86	63.14±0.93

Table 5.2: 1-shot, 5-way accuracy on MiniIN and CUB using a Cosine Classifier (CC) on different base training data and backbones. WRN: Wide ResNet28-10. MiniIN6K Random: 600 images from 64 classes sampled randomly from MiniIN6K. We evaluate the variances over 3 different runs, each run compute the few-shot performance on 10k sampled episodes. MiniIN6K*: MiniIN6K without images from bird categories.



(a) Different selection criteria (b) Different backbone and algorithms

Figure 5.4: Five-shot accuracy on miniIN when sampling classes from miniIN-6K closest/farthest to the miniIN test set or randomly using 900 images per class. (a) Comparison between different class selection criteria for selecting classes closest or farthest from the test classes. (b) Comparison of results with different algorithms and backbones using oracle features to select closest classes.

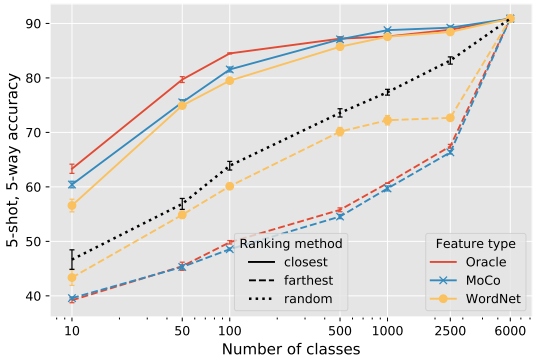


Figure 5.5: Five-shot accuracy on CUB when sampling classes from miniIN-6K **closest/farthest** to the CUB test set or randomly.

different backbones (Conv4, ResNets) and algorithms tested (ProtoNet, MatchingNets), as well as on the ImageNet benchmark.

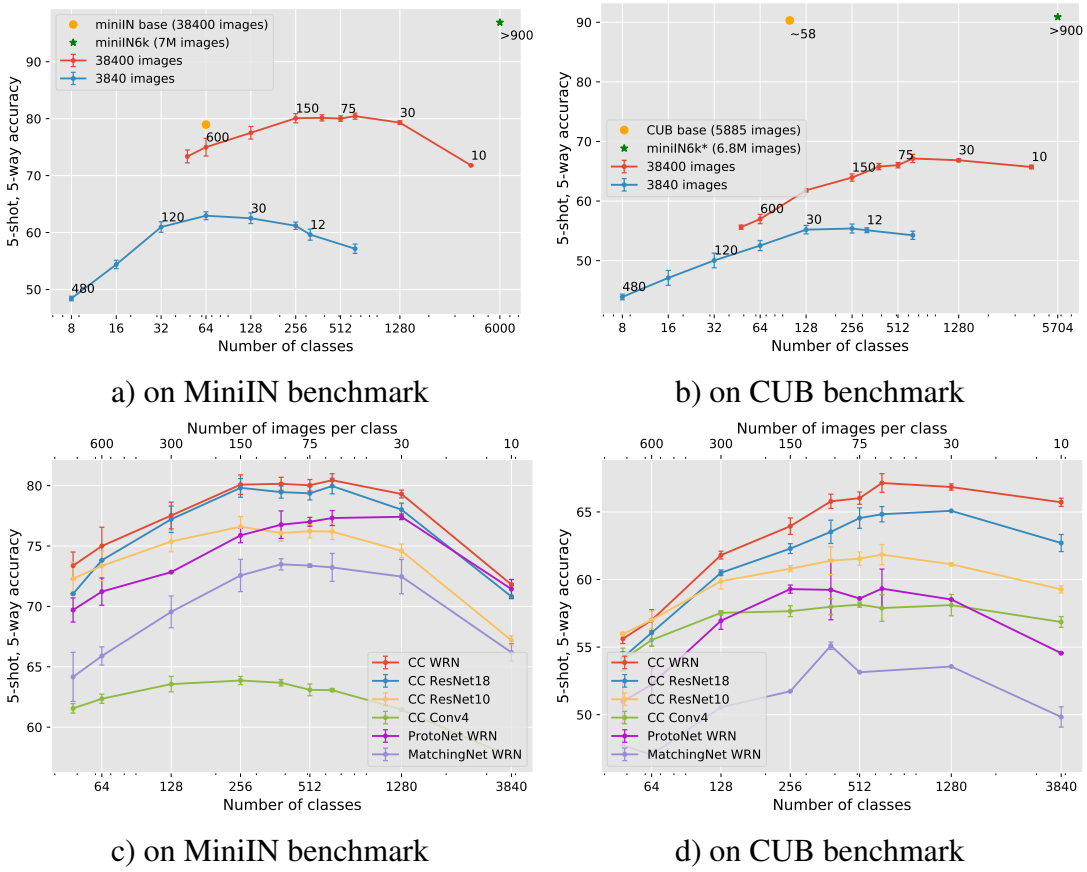
A natural explanation for these differences is that the base training classes from the benchmarks are correlated to the test classes.

To validate this hypothesis, we selected a varying number of base training classes from miniIN-6K closest and farthest to miniIN test classes using either oracle features, MoCo features, or the WordNet hierarchy, and report the results of training using a cosine classifier with the WRN architecture in Fig. 5.4. Similar experiment on CUB is shown in Fig. 5.5. We used 900 random images for each class.

For computing MoCo features, we use the self-supervised features on ImageNet using a ResNet-50 backbone from Tian et al. (2019)[1] unofficial implementation of Momentum Contrast for unsupervised visual representation learning He et al. (2019a).

While all features used for class selection yield similarly superior results for closest class selection and worst results for farthest class selection, we observe that using oracle

[1]Moco features from `https://github.com/HobbitLong/CMC/`



a) on MiniIN benchmark

b) on CUB benchmark

c) on MiniIN benchmark

d) on CUB benchmark

Figure 5.6: Trade-off between the number of classes and images per class for a fixed image budget. In (a,b) we show the trade-off for different dataset sizes and points are annotated with the corresponding number of images per class. In (c,d) we consider a total budget of 38400 annotated images and show the tradeoff for different architectures and methods. The top scale shows the number of images per class and the bottom scale the number of classes.

features leads to larger differences than than MoCo features and Wordnet hierarchy. In Fig. 5.4, we study the influence of the architecture and training method on the previously observed importance of class similarity to test classes. Similar gaps can be observed in all cases. Note however that for ProtoNet, MatchingNet and smaller backbones with CC, the best performance is not obtained with the largest number of classes.

While these findings themselves are not surprising, the amplitude of performance variations demonstrates the importance of studying the influence of training data and strategies for training data selection, especially considering that most advanced few-shot learning strategies only increase performance by a few percentage points compared to strong nearest neighbor based baselines such as CC (Chen et al., 2019; Qiao et al., 2018).

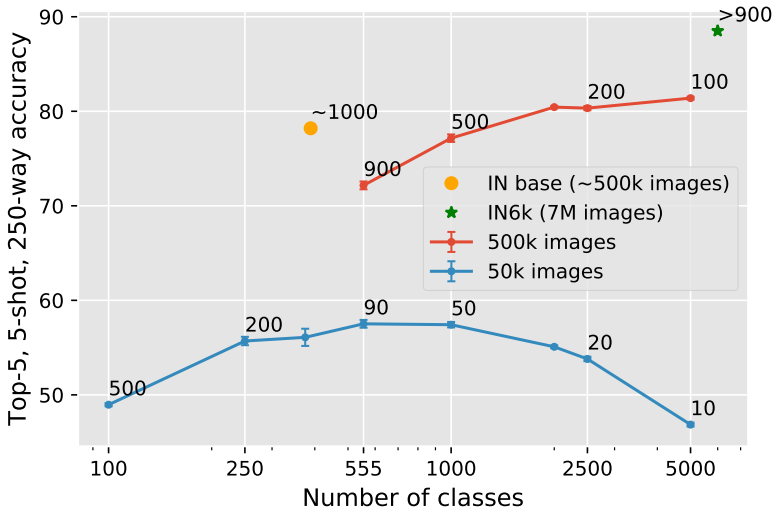


Figure 5.7: Trade-off between the number of classes and images per class for a fixed image budget on the IN benchmark. Each point is annotated with its corresponding number of images per class.

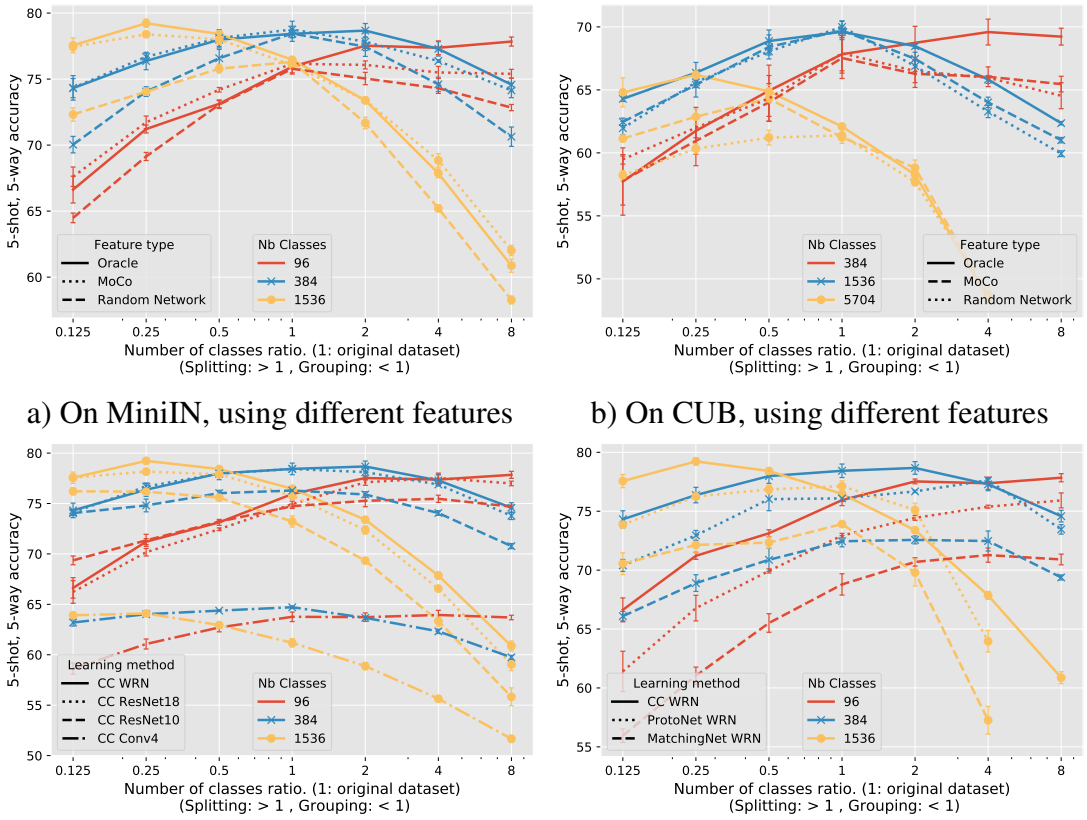
5.5.2 Effect of the number of classes for a fixed number of annotations

An important practical question when building a base training dataset is the number of classes and images to annotate, since the constraint is often the cost of the annotation process. We thus consider a fixed number of annotated images and explore the effect of the trade-off between the number of images per class and the number of classes. In Fig. 5.6, we visualize the 5-shot performance resulting from this trade-off in the base training classes on the miniIN and CUB benchmarks. In all cases, we select the classes and images randomly from our miniIN6K and miniIN6k* dataset respectively, and plot the variance over 3 runs.

First, in Fig. 5.6 (a,b) we compare the trade-off for different numbers of annotated images. We sample randomly datasets of 38400 or 3840 images with different number of classes and the same number of image in each class. We also indicate the performance with the standard benchmarks base dataset and the full miniIN6K data. The same graph on ImageNet benchmark can be seen in Fig. 5.7 using 50k and 500k images datasets.

As expected, the performance decreases when too few classes or too few images per classes are available. Interestingly, on the miniIN test benchmark (Fig. 5.6a) the best performance is obtained around 384 classes and 100 images per class with a clear boost (around 5%) over the performance using 600 images for 64 classes which is the trade-off chosen in the miniIN benchmark. In Fig. 5.6b, we observe that the best trade-off is very different on the CUB benchmark, corresponding to more classes and very few images per class. We believe this is due to the fine-grained nature of the dataset.

Second, in Fig. 5.6 (c,d), we study the consistency of these findings for different architectures and few-shot algorithms with a 38400 annotated images budget. While the trade-off depends on the architecture and method, there is always a strong effect, and the optimum tends to correspond to much fewer images per class than in standard bench-



a) On MiniIN, using different features

b) On CUB, using different features

c) MiniIN, using different backbones

d) MiniIN, using different algorithms

Figure 5.8: Impact of class grouping or splitting on few-shot accuracy on miniIN and CUB depending on the initial number of classes. Starting from different number of classes C, we group similar classes together into meta-classes or split them into sub-classes to obtain $\alpha \times C$ ones. $\alpha \in \{\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8\}$ is the x-axis. Experiments in a) and b) use CC WRN setup.

marks. For example, the best performance with ProtoNet and MatchingNet on the miniIN benchmark is obtained with as few as 30 images per class. This is interesting since it shows that the ranking of different few-shot approaches may depend on the trade-off between number of base images and classes selected in the benchmark.

The importance of this balance, and the fact that it does not necessarily correspond to the one used in the standard datasets is also important if one wants to pre-train features with limited resources. Indeed, better features can be obtained by using more classes and less images per class compared to using all available images for the classes with the largest number of images as is often done, with the idea to avoid over-fitting. Again, the boost observed for few-shot classification performance is very important compared to the ones provided by many advanced few-shot learning approaches.

5.5.3 Redefining classes

There are two possible explanations for the improvement provided by the increased number of classes for a fixed number of annotated images discussed in the previous paragraph.

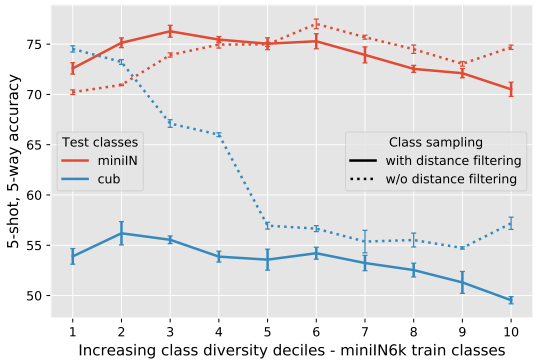
The first one is that the images sampled from more random classes cover better the space of natural images, and thus provide images more likely similar to the test images. The second one is that learning a classifier with more classes is itself beneficial to the quality of the features. To investigate whether for fixed data increasing the number of classes can boost performances, we relabel images inside each class as described in Section 5.4.3.

In Fig. 5.8, we compare the effect of grouping and splitting classes on three dataset configurations sampled from miniIN-6K and miniIN6K* with a total number of images 38400 for different number of classes $C \in \{96, 384, 1536\}$ for miniIN and $C \in \{384, 1536, 5704\}$ for CUB. Given images originally labeled with C classes, we relabel images of each class to obtain $\alpha \times C$ sub-classes. The x-axes represent the class ratio $\alpha \in \{\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8\}$. For class ratios lower than 1, we group classes using our greedy iterative grouping, while for ratios α greater than 1, we split classes using our BCP method. In Fig 5.8 (a,b), we show three possible behaviors on miniIN and CUB when using our oracle features: (i) if the number of initial classes is higher than the optimal tradeoff, grouping is beneficial and splitting hurts performances (yellow curves); (ii) if the number of initial classes is the optimal one, both splitting and grouping hurt decrease performances (blue curves); (iii) if the number of initial classes is smaller than the optimal tradeoff, splitting is beneficial and grouping hurts performance (red curves). This is a strong result, since it shows there is potential to improve performances with a fixed training dataset by redefining new classes. This can be done for grouping using the self-supervised MoCo features. However, we found they were not sufficient to split classes in a way that improves performances. Using random features on the contrary did not lead to any significant improvements. Fig. 5.8c confirms the consistency of results with various architecture on miniIN benchmark. Fig. 5.8d compares these results to the ones obtained with ProtoNet and MatchingNet. Interestingly, we see that since the tradeoff for this methods was with much fewer images per class, class splitting can increase performances in all the scenarios we considered.

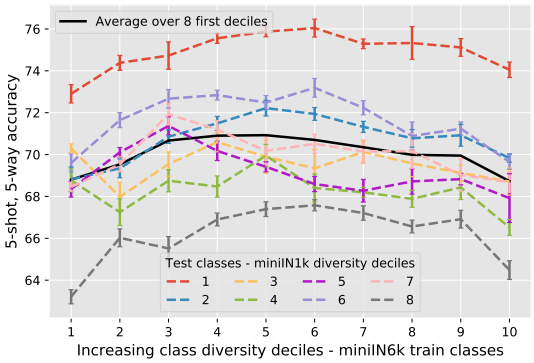
These results outline the need to adapt not only the base training images but also the base training granularity to the target few-shot task and algorithm. They also clearly demonstrate that the performance improvements we observe compared to standard tradeoffs by using more classes and less images per class is not only due to the fact that the training data is more diverse, but also to the fact that training a classifier with more classes leads to improved features for few-shot classification.

5.5.4 Selecting classes based on their diversity or difficulty

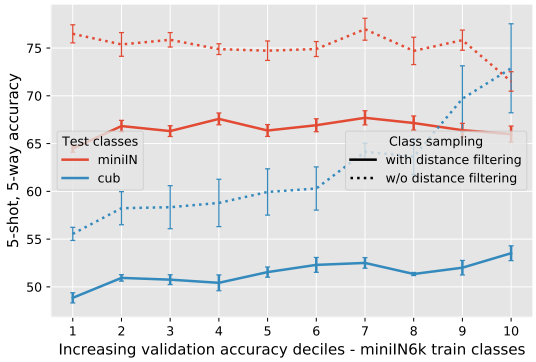
In Section 5.5.1, we observed the importance of the similarity between base training classes and the test classes. We now study whether the diversity of the base classes or their difficulty is also an important factor. To this end, we compute the measures described in Section 5.4.3 for every miniIN-6K classes and rank them by in increasing order. Then, we split the ranked classes into 10 bins of similar diversity or validation accuracy. We found that the bins obtained in this way were correlated to similarity to the test classes and thus introduces a bias in the performance due to this similarity instead of the diversity or difficulty we want to study (see Fig. 5.10, we show the similarity of classes in each bin to the test classes). To avoid this sampling bias, we associate to each class its distance to test classes, and sample base classes in each bin only in a small range of similarities, so that the



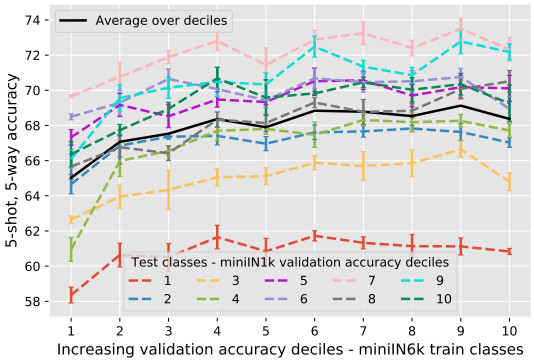
(a) Class diversity on miniIN and CUB



(b) Class diversity on miniIN1k diversity deciles



(c) Val. acc. on miniIN and CUB



(d) Val. acc. on miniIN1k val. acc. deciles

Figure 5.9: Impact of **class selection using class diversity and validation accuracy** on few-shot accuracy on miniIN and CUB benchmarks and benchmarks sampled from miniIN-1K. For training, we rank the classes of miniIN-6K in increasing feature variance or validation accuracy and split them into 10 bins from which we sample $C = 64$ classes that we use for base training. Fig. a) and c) show the importance of selecting classes in each bin while considering their distance to test classes to disentangle both selection effects. Fig. b) and d) show impact of class selection method on different benchmarks from miniIN1k sampled as deciles of increasing class diversity or validation accuracy.

average distance to the test classes is constant over all bins. In Fig. 5.9 we show the performances obtained by sampling using this strategy 64 classes and 600 images per class for a total of 38400 images in each bin. The performances obtained are shown on miniIN and CUB in Fig. 5.9 (a,c) both using random sampling from the bin and using sampling with distance filtering as explained before. It can be seen that the effect of distance filtering is very strong, decreasing significantly the range of performance variation especially on the CUB dataset, however the difference in performance is still significant, around 5% in all experiments. Both for CUB and miniIN, moderate class diversity - avoiding both the most and least diverse classes - seem beneficial, while using the most difficult classes seem to harm performances. To validate and provide additionnal insight on this experiment, we also use test benchmarks sampled from miniIN1k with classes grouped by their diversity or validation accuracy deciles from 1 to 10 in Fig. 5.9 (b, d). The curve in black shows the average over all the bins. While the range of performances highly depends on the test class selection criteria, the tendency seem very consistent on each of them. For class

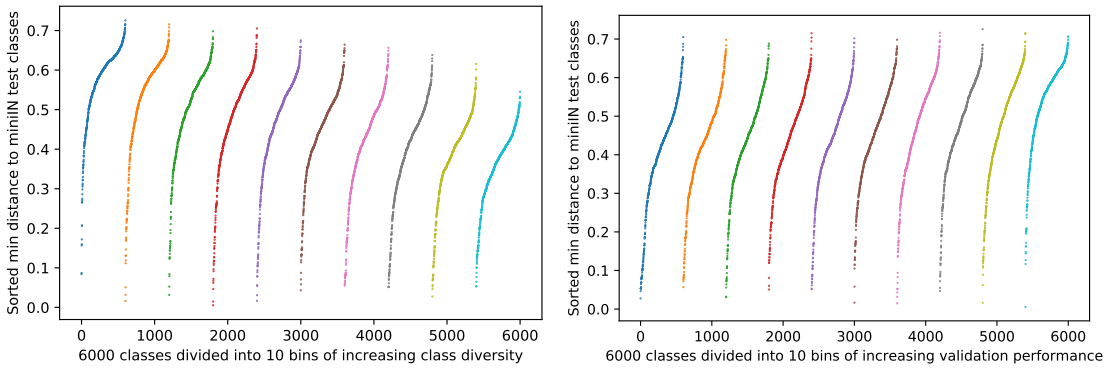


Figure 5.10: Class similarity between miniIN6k classes and miniIN test classes. MiniIN6k classes (x-axis) are grouped in 10 bins of increasing class diversity or validation accuracy. We observe that class similarity to test classes correlates with both class diversity and class validation accuracy, thus the importance of avoiding this bias during class selection.

diversity, we observe a inversed U shape average curve, i.e. using most or least diverse classes can hurt the few-shot performance, with optimal performances corresponding to slightly lower than average diversity. For validation accuracy, better few-shot classification performance is correlated with higher class validation performance, i.e. using classes that are easier to classify lead to better feature for few-shot classification.

5.6 Conclusion

Our empirical study outlines the key importance of the base training data in few-shot learning scenarios, with seemingly minor modifications of the base data resulting in large changes in performance, and carefully selected data leading to much better accuracy. We also show that few-hot performance can be improved by automatically relabelling an initial dataset by merging or splitting classes. We hope the analysis and insights that we present will:

- impact dataset design for practical applications, e.g. given a fixed number of images to label, one should prioritize a large number of different classes and potentially use class grouping strategies using self-supervised features. In addition to base classes similar to test data, one should also prioritize simple classes, with moderate diversity.

- lead to new evaluations of few-shot learning algorithm, considering explicitly the influence of the base data training in the results: the current mini-ImageNet setting of 64 classes and 600 images per class is far from optimal for several approaches. Furthermore, he optimal trade-off between number of classes and number of images per class can be different for different few-shot algorithms, suggesting taking into account different base data distributions in future few-shot evaluation benchmarks.

- inspire advances in few-shot learning and in particular the design of practical ap-

proaches to adapt base training data automatically and efficiently to the target few-shot tasks.

Chapter 6

Conclusion

The application of machine learning advances to artistic creativity is promising yet challenging. Deep learning particularly has allowed leveraging large datasets and weak supervision to achieve high level understanding, generation and manipulation of different modalities (images, 3D, text, etc.). With the progress of generative models, not only samples have become more realistic, but also the control over the generations has become simple and diverse, thus opening the way to useful creativity support tools.

In this work, we have focused on applying generative models advances to artistic creativity from the perspective of fashion and visual blends creation while addressing some fundamental related computer vision problems. We contributed several deep learning tools for image generation and manipulation aimed at assisting human artists. We proposed two different generative models for original image generation with a great potential for inspiring artists in a co-creative context, and studied multiple improvement directions for related challenges. We explored how novelty and originality can be enforced using additional loss functions or through image composition while keeping realistic generations.

First, using generative models, in particular generative adversarial networks, we developed an original fashion item generation model by using novelty losses that encourage the generation of garments that deviate from known shape and texture classes. Using simple architectures and a relatively small training dataset, our model is able to generate high resolution and original images. Our human study shows that our models with novelty losses achieve higher likeability and realism compared to the baseline ones. While this work was the first to frame fashion image generation using GANs and to suggest ways to make the generations realistic and novel, it can benefit from the recent developments in image generation from new architectures to better control over the generations.

Second, using image retrieval and composition methods, we built a model able to suggest novel image compositions based on visual similarities between foreground objects while performing the composition seamlessly through object replacement. Our human study shows that we can select foregrounds to obtain composites with more or less realism and surprise using the rank of the foreground object with respect to class similarity. While this study does not provide a solution to the complex visual blends creation problem, it remains the only work using visual similarity to suggest new plausible visual blends. This method could be extended by giving artists more control over the selected classes to ground the obtained composites in a given meaning. Generating intentful visual metaphors by providing the underlying meaning of the generated image remains a chal-

lenge to tackle. Also, leveraging learned generative models for high level image blending could improve over the copy pasting approach considered.

In addition to these two original image generation methods, we proposed a model for vector image generation aimed at high resolution image generation. Using neural networks as implicit functions over the 2D space, we defined a novel mask generator that we use to represent an image in a layered way. Our model for image decomposition in multiple colored vector layers allowed us to learn a meaningful embedding of masks from a given dataset. We also showed how we can leverage this learned decomposition for various applications spanning image editing, vectorization and image retrieval. Recent work on image generation have focused on exploring novel architectures from style conditioned architectures allowing manipulation of the generated images (Karras et al., 2019), to architecture for image generation from discrete vectors (Esser et al., 2021), and architectures using implicit functions to represent images in a resolution independent way similar to our proposed method (Lin et al., 2021) which present a promising direction for generating images with a hybrid model with a multi-layer perceptron for resolution independent generation and convolutional layers for texture generation in a local scale.

Finally, we studied the importance of the training data on the quality of image representations used for image search and classification. We developed an evaluation protocol to measure the influence of the training data on the generalizaion capacity of learned features from a few-shot classification perspective. By comparing multiple datasets of different sizes, classes, and labelings, we highlighted important tradeoffs in the dataset construction that directly affect the quality of the learned features. We also showed that these findings generalize to multiple architectures, benchmarks and learning algorithms.

In sum, future work on artistic creativity should leverage multi-modal approaches combining text and images i.e. (Radford et al., 2021), while also focusing on novel architectures that enable simple human interaction.

Bibliography

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *ICCV*, 2019.

Edward Adelson. *Layered representations for image coding*. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1991.

Yağiz Aksoy, Tunç Ozan Aydin, Aljoša Smolić, and Marc Pollefeys. Unmixing-based soft color segmentation for image manipulation. *ACM Transactions on Graphics (TOG)*, 2017.

Ziad Al-Halah, Rainer Stiefelhagen, and Kristen Grauman. Fashion forward: Forecasting visual style in fashion. In *Proceedings of the IEEE international conference on computer vision*, pages 388–397, 2017.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *CVPR*, 2016.

Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. Image generators with conditionally-independent pixel synthesis. *arXiv:2011.13775*, 2020.

Antreas Antoniou and Amos J. Storkey. Learning to learn via self-critique. *NeurIPS*, 2019.

Relja Arandjelović and Andrew Zisserman. Object discovery with a copy-pasting gan. *arXiv preprint arXiv:1905.11369*, 2019.

Martín Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *ArXiv*, abs/1701.04862, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

David Bau, Jun-Yan Zhu, Hendrik Strobelt, Bolei Zhou, Joshua B Tenenbaum, William T Freeman, and Antonio Torralba. Gan dissection: Visualizing and understanding generative adversarial networks. *ICLR*, 2019.

Tamara L. Berg, Alexander C. Berg, and Jonathan Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV*, 2010.

David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *ArXiv*, 2017.

Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 1987.

I Binford. Visual perception by computer. In *IEEE Conference of Systems and Control*, 1971.

Damian P Birney and Robert J Sternberg. The development of cognitive abilities. *Psychology Press*, 2011.

Vighnesh Birodkar, Hossein Mobahi, and Samy Bengio. Semantic redundancies in image-classification datasets: The 10% you don't need. *ArXiv preprint 1901.11409*, 2019.

Margaret A Boden et al. *The creative mind: Myths and mechanisms*. Psychology Press, 2004.

Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. Deep learning techniques for music generation-a survey. *arXiv:1709.01620*, 2017.

Jean-Pierre Briot, Gaëtan HADJERES, and François-David Pachet. *Deep Learning Techniques for Music Generation – A Survey*. HAL, 2019. URL `https://hal.sorbonne-universite.fr/hal-01660772`.

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

T. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, J. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. Henighan, R. Child, A. Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, J. Clark, Christopher Berner, Sam McCandlish, A. Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *ArXiv*, 2020.

Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *ArXiv preprint 1710.05381*, 2017.

Peter J. Burt and Edward H. Adelson. A multiresolution spline with application to image mosaics. *Computer Graphics*, 1983.

Alexandre Carlier, Martin Danelljan, Alexandre Alahi, and Radu Timofte. Deepsvg: A hierarchical generative network for vector graphics animation. *NeurIPS*, 2020.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.

L. Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, M. Rivière, Kevin Tran, J. Heras-Domingo, Caleb Ho, Weihua Hu, Aini Palizhati, Anuroop Sriram, Brandon Wood, J. Yoon, Devi Parikh, C. L. Zitnick, and Zachary W. Ulissi. The open catalyst 2020 (oc20) dataset and community challenges. *ArXiv*, 2020.

Bor-Chun Chen and Andrew Kae. Toward realistic image compositing with adversarial learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2photo: Internet image montage. *ACM Trans. Graph.*, 2009.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *ICLR*, 2019.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019.

Anton Cherepkov, Andrey Voynov, and Artem Babenko. Navigating the gan parameter space for semantic image editing. *arXiv preprint arXiv:2011.13786*, 2020.

Lydia B Chilton, Savvas Petridis, and Maneesh Agrawala. Visiblends: A flexible workflow for visual blends. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2019.

Kashyap Chitta, Jose M. Alvarez, Elmar Haussmann, and Clément Farabet. Less is more: An exploration of data redundancy with active dataset subsampling. *ArXiv preprint 1905.12737*, 2019.

Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018.

Christies.com. A collaboration between two artists one human one a machine. `https://www.christies.com/features/A-collaboration-between-two-artists-one-human-one-a-machine-9332-1.aspx`, 2018.

Daniel Cohen-Or, Olga Sorkine, Ran Gal, Tommer Leyvand, and Ying-Qing Xu. Color harmonization. *ACM Trans. Graph.*, 2006.

David Cohn, Richard Ladner, and Alex Waibel. Improving generalization with active learning. In *Machine Learning*, pages 201–221, 1994.

Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In *CVPR*, 2020.

Ronan Collobert, J. Weston, L. Bottou, Michael Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 2011.

Wenyan Cong, Jianfu Zhang, Li Niu, Liu Liu, Zhixin Ling, Weiyuan Li, and Liqing Zhang. Dovenet: Deep image harmonization via domain verification. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

Antonia Creswell and Anil Anthony Bharath. Inverting the generator of a generative adversarial network. *IEEE transactions on neural networks and learning systems*, 2018.

Xiaodong Cun and Chi-Man Pun. Improving the harmony of the composite image by spatial-separated attention module. *IEEE Trans. Image Process.*, 2020.

J. Cunha, Pedro Martins, and P. Machado. Let's figure this out: A roadmap for visual conceptual blending. In *ICCC*, 2020.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 1989.

Prutha Date, Ashwinkumar Ganesan, and Tim Oates. Fashioning with networks: Neural style transfer to design clothes. In *KDD ML4Fashion workshop*, 2017.

Daniel Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977.

Laurent Demaret, Nira Dyn, and Armin Iske. Image compression by linear splines over adaptive triangulations. *Signal Processing*, 2006.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009a.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2009b.

Emily L Denton, Soumith Chintala, arthur szlam, and Rob Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

Liuyun Duan and Florent Lafarge. Image partitioning into convex polygons. In *CVPR*, 2015.

Melanie Ducoffe and Frederic Precioso. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018.

Vincent Dumoulin, Jonathon Shlens, Manjunath Kudlur, Arash Behboodi, Filip Lemic, Adam Wolisz, Marco Molinaro, Christoph Hirche, Masahito Hayashi, Emilio Bagan, et al. A learned representation for artistic style. *ICLR*, 2017.

Scott Eaton. Human allocation of space. *nvidia.com*, 2019. URL `https://www.nvidia.com/en-us/deep-learning-ai/ai-art-gallery/artists/scott-eaton/`.

Ahmed Elgammal, Bingchen Liu, Mohamed Elhoseiny, and Marian Mazzone. Creative adversarial networks. In *ICCC*, 2017.

S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey Hinton. Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. *NIPS*, 2016.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. Beyond english-centric multilingual machine translation. *arXiv preprint*, 2020.

Yang Fan, Fei Tian, Tao Qin, and Tie-Yan Liu. Neural data filter for bootstrapping stochastic gradient descent. *ICLR Workshop*, 2017.

Jean-Dominique Favreau, Florent Lafarge, and Adrien Bousseau. Photo2clipart: image abstraction and vectorization using layered linear gradients. *ACM Transactions on Graphics (TOG)*, 2017.

Christiane Fellbaum. Wordnet: An electronic lexical database and some of its applications, 1998.

Charles Forceville. Pictorial metaphor in advertisements. *Metaphor and symbol*, 1994.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *ICML*, 2017.

Yaroslav Ganin, Tejas Kulkarni, Igor Babuschkin, SM Eslami, and Oriol Vinyals. Synthesizing programs for images using reinforced adversarial learning. *ICML*, 2018.

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

Weifeng Ge and Yizhou Yu. Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. In *CVPR*, 2017.

Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018.

Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. *ICCV*, 2019a.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *CVPR*, 2019b.

Lampros Gkiouzepas and Margaret K Hogg. Articulating a new framework for visual metaphors in advertising. *Journal of Advertising*, 2011.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

Leo Grady. Random walks for image segmentation. *Trans. Pattern Anal. Mach. Intell.*, 2006.

Jeanine Graf and Wolfgang Banzhaf. Interactive evolution of images. In *Evolutionary Programming*, pages 53–65, 1995.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv:1502.04623*, 2015.

Karol Gregor, Frederic Besse, Danilo Jimenez Rezende, Ivo Danihelka, and Daan Wierstra. Towards conceptual compression. In *NIPS*, 2016.

Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *CVPR*, 2018.

David Ha. Generating abstract patterns with tensorflow. *blog.otoro.net*, 2016a. URL `http://blog.otoro.net/2016/03/25/generating-abstract-patterns-with-tensorflow/`.

David Ha. Generating large images from latent vectors. *blog.otoro.net*, 2016b. URL `http://blog.otoro.net/2016/04/01/generating-large-images-from-latent-vectors/`.

David Ha and Douglas Eck. A Neural Representation of Sketch Drawings. *arXiv:1704.03477*, 2017.

David Ha and Douglas Eck. A neural representation of sketch drawings. *ICLR*, 2018.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *arXiv preprint arXiv:1809.01999*, 2018.

Gaëtan Hadjeres and François Pachet. Deepbach: a steerable model for bach chorales generation. *ICML*, 2017.

Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3018–3027, 2017.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *NeurIPS*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv:1512.03385*, 2015a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015b.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. *ICCV*, 2017.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *ArXiv preprint 1911.05722*, 2019a.

Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Attgan: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing*, 2019b.

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

Nathan Hilliard, Lawrence Phillips, Scott Howland, Artëm Yankov, Courtney D Corley, and Nathan O Hodas. Few-shot learning with metric-agnostic conditional embeddings. *arXiv preprint 1802.04376*, 2018.

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 1991.

V. Hosu, H. Lin, T. Sziranyi, and D. Saupe. Koniq-10k: An ecologically valid database for deep learning of blind image quality assessment. *IEEE Trans. Image Process.*, 2020.

Dirk Hovy, Shashank Srivastava, Sujay Kumar Jauhar, Mrinmaya Sachan, Kartik Goyal, Huying Li, Whitney Sanders, and Eduard Hovy. Identifying metaphorical word use with tree kernels. In *Proceedings of the First Workshop on Metaphor in NLP*, 2013.

Shell Xu Hu, Pablo Moreno, Yang Xiao, Xi Shen, Guillaume Obozinski, Neil Lawrence, and Andreas Damianou. Empirical bayes transductive meta-learning with synthetic gradients. In *ICLR*, 2019.

Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

Zhewei Huang, Wen Heng, and Shuchang Zhou. Learning to paint with model-based deep reinforcement learning. *ICCV*, 2019.

Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. What makes imagenet good for transfer learning? *NeurIPS LSCVS 2016 Workshop*, 2016.

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

Phillip Isola and Ce Liu. Scene collaging: Analysis and synthesis of natural images with semantic layers. In *ICCV*, 2013.

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

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Adv. Neural Inform. Process. Syst.*, 2015.

Tahereh Javaheri, Morteza Homayounfar, Zohreh Amoozgar, Reza Reiazi, Fatemeh Homayounieh, Engy Abbas, Azadeh Laali, Amir Reza Radmard, Mohammad Hadi Gharib, Seyed Ali Javad Mousavi, et al. Covidctnet: An open-source deep learning approach to identify covid-19 using ct image. *arXiv preprint arXiv:2005.03059*, 2020.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33 (1):117–128, 2010.

Se-Hoon Jeong. Visual metaphor in advertising: Is the persuasive effect attributable to visual argumentation or metaphorical rhetoric? *Journal of Marketing Communications*, 2008.

Jiaya Jia, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Drag-and-drop pasting. *ACM Trans. Graph.*, 2006.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

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

Anna Kantosalo and Tapio Takala. Five c's for human–computer co-creativity—an update on classical creativity perspectives. In *International Conference on Computational Creativity*, 2020.

Pegah Karimi, Mary Lou Maher, Kazjon Grace, and Nicholas Davis. A computational model for visual conceptual blends. *IBM Journal of Research and Development*, 2018.

Andrej Karpathy. Image 'painting'. *cs.stanford.edu*, 2015. URL `https://cs.stanford.edu/people/karpathy/convnetjs/demo/image_regression.html`.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv:1710.10196*, 2017.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. *arXiv preprint arXiv:1803.00942*, 2018.

Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D Yoo. Edge-labeling graph neural network for few-shot learning. In *CVPR*, 2019.

Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *ICML*, 2017.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik Kingma and Max Welling. Auto-encoding variational Bayes. In *ICLR*, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. The GAN landscape: Losses, architectures, regularization, and normalization. *ICML*, 2019.

Jean-Francois Lalonde and Alexei A Efros. Using color compatibility for assessing image realism. In *Int. Conf. Comput. Vis.*, 2007.

Guillaume Lample and Franccois Charton. Deep learning for symbolic mathematics. *ArXiv*, 2020.

Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. In *NeurIPS*, 2019.

Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, and Marc'Aurelio Ranzato. Fader networks: Manipulating images by sliding attributes. *arXiv preprint arXiv:1706.00409*, 2017.

Christoph Lassner, Gerard Pons-Moll, and Peter V. Gehler. A generative model of people in clothing. *ICCV*, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

Ann B Lee, David Mumford, and Jinggang Huang. Occlusion models for natural images: A statistical study of a scale-invariant dead leaves model. *International Journal of Computer Vision*, 2001.

Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019.

Hongyang Li, David Eigen, Samuel Dodge, Matthew Zeiler, and Xiaogang Wang. Finding task-relevant features for few-shot learning by category traversal. In *CVPR*, 2019.

Pengyong Li, Yuquan Li, Chang-Yu Hsieh, Shengyu Zhang, Xianggen Liu, Huanxiang Liu, Sen Song, and Xiaojun Yao. Trimnet: learning molecular representation from triplet messages for biomedicine. *Briefings in Bioinformatics*, 2020.

Zicheng Liao, Hugues Hoppe, David Forsyth, and Yizhou Yu. A subdivision-based representation for vector image editing. *Trans. on vis. and comp. graph.*, 2012.

Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey. St-gan: Spatial transformer generative adversarial networks for image compositing. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

Chieh Hubert Lin, Hsin-Ying Lee, Yen-Chi Cheng, Sergey Tulyakov, and Ming-Hsuan Yang. Infinitygan: Towards infinite-resolution image synthesis. *arXiv preprint arXiv:2104.03963*, 2021.

Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. In *ICLR*, 2019.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.

Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Ben London. A pac-bayesian analysis of randomized learning with application to stochastic gradient descent. *NeurIPS*, 2017.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

Raphael Gontijo Lopes, David Ha, Douglas Eck, and Jonathon Shlens. A learned representation for scalable vector graphics. In *ICCV*, 2019.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 2004.

Mario Lucic, Karol Kurach, Marcin Michalski, S. Gelly, and O. Bousquet. Are gans created equal? a large-scale study. In *NeurIPS*, 2018.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 2008.

Xudong Mao, Q. Li, Haoran Xie, Raymond Y. K. Lau, Z. Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

Scott Mayer McKinney, M. Sieniek, Varun Godbole, Jonathan Godwin, N. Antropova, H. Ashrafian, T. Back, Mary Chesus, Greg C. Corrado, A. Darzi, M. Etemadi, Florencia Garcia-Vicente, F. Gilbert, M. Halling-Brown, D. Hassabis, S. Jansen, A. Karthikesalingam, Christopher J. Kelly, Dominic King, J. Ledsam, David S. Melnick, H. Mostofi, Lily Peng, J. Reicher, B. Romera-Paredes, R. Sidebottom, Mustafa Suleyman, Daniel Tse, K. Young, J. Fauw, and S. Shetty. International evaluation of an ai system for breast cancer screening. *Nature*, 2020.

John FJ Mellor, Eunbyung Park, Yaroslav Ganin, Igor Babuschkin, Tejas Kulkarni, Dan Rosenbaum, Andy Ballard, Theophane Weber, Oriol Vinyals, and SM Eslami. Unsupervised doodling and painting with improved spiral. *arXiv preprint arXiv:1910.01007*, 2019.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? *ICML*, 2018.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *arXiv preprint arXiv:2003.08934*, 2020.

David L. Milgram. Computer methods for creating photomosaics. *ACM Transactions on Computers*, 1975.

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

Michael Mohler, David Bracewell, Marc Tomlinson, and David Hinote. Semantic signatures for example-based linguistic metaphor detection. In *Proceedings of the First Workshop on Metaphor in NLP*, 2013.

Matthew Muckley, Bruno Riemenschneider, A. Radmanesh, Sunwoo Kim, Geunu Jeong, Jingyu Ko, Yohan Jun, Hyungseob Shin, D. Hwang, M. Mostapha, S. Arberet, D. Nickel, Zaccharie Ramzi, P. Ciuciu, Jean-Luc Starck, Jonas Teuwen, Dimitrios Karkalousos, C. Zhang, Anuroop Sriram, Zhengnan Huang, N. Yakubova, Y. Lui, and F. Knoll. State-of-the-art machine learning mri reconstruction in 2020: Results of the second fastmri challenge. *ArXiv*, 2020.

Reiichiro Nakano. Neural painters: A learned differentiable constraint for generating brushstroke paintings. *arXiv preprint arXiv:1904.08410*, 2019.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *ICML*, 2017.

Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

Alexandrina Orzan, Adrien Bousseau, Holger Winnemöller, Pascal Barla, Joëlle Thollot, and David Salesin. Diffusion curves: a vector representation for smooth-shaded images. In *ACM Transactions on Graphics (TOG)*. ACM, 2008.

Devi Parikh and C Lawrence Zitnick. Exploring crowd co-creation scenarios for sketches. *arXiv preprint arXiv:2005.07328*, 2020.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *CVPR*, 2019a.

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

Adam Paszke, S. Gross, Francisco Massa, A. Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Z. Lin, N. Gimelshein, L. Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach 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 *NeurIPS*, 2019.

Guim Perarnau, Joost Van De Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *NIPS Workshop on Adversarial Training*, 2016.

Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Trans. Graph.*, 2003.

S. Petridis and L. Chilton. Human errors in interpreting visual metaphor. *Proceedings of the 2019 on Creativity and Cognition*, 2019.

Barbara J Phillips and Edward F McQuarrie. Beyond visual metaphor: A new typology of visual rhetoric in advertising. *Marketing theory*, 2004.

Tiziano Portenier, Qiyang Hu, Attila Szabo, Siavash Arjomand Bigdeli, Paolo Favaro, and Matthias Zwicker. Faceshop: Deep sketch-based face image editing. *Trans. on Graphics (TOG)*, 2018.

Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L Yuille. Few-shot image recognition by predicting parameters from activations. In *CVPR*, 2018.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017.

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

Scott E Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. Learning what and where to draw. *NIPS*, 2016.

Christian Richardt, Jorge Lopez-Moreno, Adrien Bousseau, Maneesh Agrawala, and George Drettakis. Vectorising bitmaps into semi-transparent gradient layers. In *Computer Graphics Forum*, 2014.

Graeme Ritchie. Computational mechanisms for pun generation. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*, 2005.

Jason Tyler Rolfe and Yann LeCun. Discriminative recurrent sparse auto-encoders. *ICLR*, 2013.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, J. Xu, Myle Ott, Kurt Shuster, Eric Michael Smith, Y.-Lan Boureau, and J. Weston. Recipes for building an open-domain chatbot. *ArXiv*, 2020.

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*, 2015.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 1958.

N. Rostamzadeh, Seyedarian Hosseini, Thomas Boquet, Wojciech Stokowiec, Y. Zhang, Christian Jauvin, and C. Pal. Fashion-gen: The generative fashion dataset and challenge. *ArXiv*, abs/1806.08317, 2018.

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)*, 2015.

Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *ICLR*, 2019.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. Déja vu: an empirical evaluation of the memorization properties of convnets. *ArXiv preprint 1809.06396*, 2018.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NIPS*, 2016.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. In *INTERSPEECH*, 2019.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2017.

A. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, Tim Green, C. Qin, Augustin Zídek, Alexander W. R. Nelson, A. Bridgland, Hugo Penedones, Stig Petersen, K. Simonyan, Steve Crossan, P. Kohli, D. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis. Improved protein structure prediction using potentials from deep learning. *Nature*, 2020.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *CVPR workshops*, 2014.

Xiaoyong Shen, Aaron Hertzmann, Jiaya Jia, Sylvain Paris, Brian Price, Eli Shechtman, and Ian Sachs. Automatic portrait segmentation for image stylization. In *Computer Graphics Forum*, 2016.

Pao Siangliulue, Kenneth C Arnold, Krzysztof Z Gajos, and Steven P Dow. Toward collaborative ideation at scale: Leveraging ideas from others to generate more creative and diverse ideas. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 2015.

Pao Siangliulue, Joel Chan, Steven P Dow, and Krzysztof Z Gajos. Ideahound: improving large-scale collaborative ideation with crowd-powered real-time semantic modeling. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 2016.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *NeurIPS*, 2020.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 2007.

Alexa Steinbrück. Conceptual blending for the visual domain. *Ph. D. dissertation, Masters thesis*, 2013.

Jian Sun, Lin Liang, Fang Wen, and Heung-Yeung Shum. Image vectorization using optimized gradient meshes. *ACM Transactions on Graphics (TOG)*, 2007.

Ruoyu Sun, Tiantian Fang, and A. Schwing. Towards a better global loss landscape of gans. *NeurIPS*, 2020.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

Yu-Wing Tai, Jiaya Jia, and Chi-Keung Tang. Soft color segmentation and its applications. *Trans. Pattern Anal. Mach. Intell.*, 2007.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*. PMLR, 2019.

Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020.

Michael W Tao, Micah K Johnson, and Sylvain Paris. Error-tolerant image compositing. In *Eur. Conf. Comput. Vis.*, 2010.

Eu Wern Teh, Terrance DeVries, and Graham W Taylor. Proxynca++: Revisiting and revitalizing proxy neighborhood component analysis. In *European Conference on Computer Vision (ECCV)*. Springer, 2020.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. *ICLR*, 2020.

Yi-Hsuan Tsai, Xiaohui Shen, Zhe Lin, Kalyan Sunkavalli, and Ming-Hsuan Yang. Sky is not the limit: semantic-aware sky replacement. *ACM Trans. Graph.*, 2016.

Yi-Hsuan Tsai, Xiaohui Shen, Zhe Lin, Kalyan Sunkavalli, Xin Lu, and Ming-Hsuan Yang. Deep image harmonization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014.

Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *Proc. CVPR*, volume 2, 2017.

Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, 2016.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016.

Kailas Vodrahalli, Ke Li, and Jitendra Malik. Are all training examples created equal? an empirical study. *ArXiv preprint 1811.12569*, 2018.

Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. In *ICML*, 2020.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD birds-200-2011 dataset. *California Institute of Technology*, 2011.

John Wang and Edward Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 1994.

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*, 2018a.

Xiaolong Wang and Abhinav Gupta. Generative image modeling using style and structure adversarial networks. *ECCV*, 2016.

Y. Wang, W.-L. Chao, K. Q. Weinberger, and L. van der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *ArXiv preprint 1911.04623*, 2019.

Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *CVPR*, 2018b.

David Warde-Farley and Yoshua Bengio. Improving generative adversarial networks with denoising feature matching. *ICLR*, 2017.

wikiart.org. Wikiart: Visual art encyclopedia. `https://www.wikiart.org/`, 2010.

Yuxin Wu and Kaiming He. Group normalization. *ECCV*, 2018.

Tian Xia, Binbin Liao, and Yizhou Yu. Patch-based image vectorization with automatic curvilinear feature alignment. *Trans. on Graphics (TOG)*, 2009.

Wenqi Xian, Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. TextureGAN: Controlling deep image synthesis with texture patches. *CVPR*, 2018.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

Ping Xiao, Simo Matias Linkola, et al. Vismantic: Meaning-making with images. In *ICCC*, 2015.

Ning Xie, Hirotaka Hachiya, and Masashi Sugiyama. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. *IEICE TRANSACTIONS on Information and Systems*, 2013.

Su Xue, Aseem Agarwala, Julie Dorsey, and Holly Rushmeier. Understanding and improving the realism of image composites. *ACM Trans. Graph.*, 2012.

Kota Yamaguchi, M Hadi Kiapour, Luis E Ortiz, and Tamara L Berg. Parsing clothing in fashion photographs. In *2012 IEEE Conference on Computer vision and pattern recognition*, 2012.

Zhicheng Yan, Hao Zhang, Baoyuan Wang, Sylvain Paris, and Yizhou Yu. Automatic photo adjustment using deep neural networks. *ACM Trans. Graph.*, 2015.

Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh. LR-GAN: layered recursive generative adversarial networks for image generation. *ICLR*, 2017.

Gokhan Yildirim, Nikolay Jetchev, Roland Vollgraf, and Urs Bergmann. Generating high-resolution fashion model images wearing custom outfits. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Int. Conf. Comput. Vis.*, 2019.

Zhiwei Yu, Jiwei Tan, and Xiaojun Wan. A neural approach to pun generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, 2018.

Andrew Zhai and Hao-Yu Wu. Classification is a strong baseline for deep metric learning. *arXiv preprint arXiv:1811.12649*, 2018.

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

Han Zhang, I. Goodfellow, Dimitris N. Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, 2019.

Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016.

Yinan Zhao, Brian Price, Scott Cohen, and Danna Gurari. Unconstrained foreground object search. In *Int. Conf. Comput. Vis.*, 2019.

Liang Zheng, Yi Yang, and Alexander G Hauptmann. Person re-identification: Past, present and future. *arXiv preprint arXiv:1610.02984*, 2016.

Ningyuan Zheng, Yifan Jiang, and Dingjiang Huang. Strokenet: A neural painting environment. In *ICLR*, 2018.

Linjun Zhou, Peng Cui, Xu Jia, Shiqiang Yang, and Qi Tian. Learning to select base classes for few-shot classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4624–4633, 2020.

Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Learning rich features for image manipulation detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

Zhiming Zhou, Weinan Zhang, and Jun Wang. Inception score, label smoothing, gradient vanishing and -log(d(x)) alternative. *arXiv:1708.01729*, 2017.

Jun-Yan Zhu, Philipp Krahenbuhl, Eli Shechtman, and Alexei A Efros. Learning a discriminative model for the perception of realism in composite images. In *Int. Conf. Comput. Vis.*, 2015.

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. *ECCV*, 2016.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *ICCV*, 2017a.

Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A. Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. *NIPS*, 2017b.

Shizhan Zhu, Sanja Fidler, Raquel Urtasun, Dahua Lin, and Chen Change Loy. Be your own prada: Fashion synthesis with structural coherence. *ICCV*, 2017c.