Disentangled Visual Attribute Manipulation in Fashion: Techniques in Generative Deep Neural Networks

---

Degree Dissertation

for the

Master Examination in Data Science in Business and Economics

at the

Faculty of Economic and Social Sciences

of the

Eberhard Karls Universität

Tübingen

Examiner:

Dr. Aseem Behl

Submitted by:

Malte Genschow

Born in Filderstadt

Date of submission: 28/08/2024

Faculty of Economics and Social Sciences
of the University of Tuebingen

Master Thesis

Disentangled Visual Attribute Manipulation in Fashion: Techniques in Generative Deep Neural Networks

Dr. Aseem Behl

Summer Term 2024

Malte Genschow
Hanna-Bernheim-Str. 46
72072 Tuebingen
Phone: 015784765887
E-Mail: malte.genschow@student.uni-tuebingen.de

Study Program:
Data Science in Business and Economics (M.Sc.)
5th semester
Matriculation Number: 3891245
Date of Submission: August 28, 2024

Contents

1 Introduction **1**

2 Background and Literature Review **3**

2.1 Architecture Selection 3

2.2 GANs 3

2.3 GAN Inversion 6

2.3.1 Latent Spaces 6

2.3.2 Inversion Methods 7

2.4 Manipulation 9

3 Dataset **10**

4 Method **11**

4.1 Generation 12

4.1.1 Training 12

4.1.2 Evaluation 13

4.2 Inversion 13

4.2.1 Training 14

4.2.2 Evaluation 14

4.3 Typicality Measures 15

4.4 Manipulation 17

4.4.1 InterFaceGAN 17

4.4.2 Training 18

5 Results **20**

5.1 Generator Training 20

5.2 Inversion 21

5.3 Typicality Measure 22

5.4 Manipulation 23

5.4.1 Preliminary Experiments 23

5.4.2 Selection of InterFaceGAN Boundary 24

5.4.3 Typicality Manipulation 24

6 Discussion **29**

7 Conclusion **30**

References **31**

Appendix **i**

List of Figures

1 Basic GAN architecture 4
2 StyleGAN2 Architecture 5
3 Random samples from the dataset 11
4 Inference Workflow 11
5 Training Workflow 12
6 Disentangled Embeddings Model for Typicality 17
7 Conditional manipulation via subspace projection 18
8 Generated Dresses from StyleGAN2-Ada Model 20
9 Qualitative Evaluation of Inversion Methods 22
10 Typicality Manipulations using DINOv2 Embeddings 25
11 Typicality Manipulations using Disentangled Embeddings 26
12 Typicality Manipulations using Disentangled Embeddings excl. sleeve length 27
A.1 StyleGAN Latent Spaces i
A.2 encoder4editing Training Loss Curves i
A.3 Schematic Overview of Inversion Methods ii
A.4 StyleGAN2-Ada Training Evaluation iii
A.5 Manipulations of Physical Attributes Used as Conditions iv
A.6 Most and Least Typical Dresses based on DINOv2 Embeddings v
A.7 Most and Least Typical Dresses based on Complete Disentangled Embeddings . . . vi
A.8 Most and Least Typical Dresses based on Disentangled Embeddings excluding Color vi
A.9 Most and Least Typical Dresses based on Disentangled Embeddings excluding Sleeve Length vi
A.10 Distribution of typicality scores vii
A.11 Preliminary Manipulation Tests using Hyperstyle Inversions viii
A.12 Decreasing Typicality for DINOv2 Embeddings x
A.13 Increasing Typicality for DINOv2 Embeddings xi
A.14 Decreasing Typicality for Disentangled Embeddings xii

A.15 Increasing Typicality for Disentangled Embeddings xiii
A.16 Increasing Typicality for DINOv2 Embeddings conditioned on Color xiv
A.17 Increasing Typicality for Disentangled Embeddings excl. Sleeve Length xv
A.18 Increasing Typicality for Disentangled Embeddings excl. Color xvi

List of Tables

1 Quantitative Evaluation of Inversion Methods 22
2 Validation Accuracy of InterfaceGAN Boundaries 24
3 Summary Statistics of Separation Performance for Physical Attributes v

Abbreviations

DCGAN . . . . . Deep Convolutional Generative Adversarial Network

e4e . . . . . Encoder 4 Editing

FID . . . . . Fréchet Inception Distance

GAN . . . . . Generative Adversarial Network

IS . . . . . Inception Score

KID . . . . . Kernel Inception Distance

LoRA . . . . . Low-Rank Adaptation

LPIPS . . . . . Learned Perceptual Image Path Similarity

MSE . . . . . Mean Squared Error

PCA . . . . . Principal Component Analysis

PGGAN . . . . . Progressive Generative Adversarial Network

PTI . . . . . Pivotal Tuning Inversion

SSIM . . . . . Structural Similarity Index Measure

SVM . . . . . Support Vector Machine

VAE . . . . . Variational Autoencoder

1 Introduction

The rapid development of generative neural networks for image generation has not only pushed the boundaries of computer vision research but also opened up new possibilities for productivity increases in many industries with drastic economic implications. McKinsey estimates, that only in product research and design, generative models could unlock $60 Billion in productivity (Chui et al., 2023). Using generative models as assistants in the design process and thus automating parts of the value creation chain can hugely increase industrial productivity (Regenwetter et al., 2022, p.1). At the same time, many products are highly dependent on innovative or brand-specific designs, and especially in the fashion industry, the designer's experience and professional knowledge play an important role in the design process (Sharma et al., 2021, p.3). However, generative models can augment these experts' design process and help them align their designs with the newest insights from consumer behavior research. In this work, one example of such a design augmentation for fashion designers is developed, focusing on the design typicality of fashion products. Research in consumer behavior has consistently demonstrated the significant role that typicality plays in product evaluation. Products are generally evaluated more positively, if they are more typical since typical products can be easier categorized by consumers (see Barsalou (1985) p.1, Sujan (1985) p.43) and processed more fluently (see e.g. Veryzer Jr and Hutchinson (1998) p.1, Winkielman et al. (2006), p.1). Additionally, more typical designs are more prevalent in consumers' everyday lives and can thus benefit from repeated exposure which can further increase consumer liking and familiarity (Zajonc, 1968, p.23). In addition, consumers perceive atypical products as less reliable and functional than typical ones (Schnurr, 2017, p.613). Interestingly, the relationship between typicality and consumer liking is not linear. While more typical products can be categorized more easily which leads to higher consumer preferences, this only works to a certain extent. Meyers-Levy and Tybout (1989) found that products which are moderately incongruent with the schema of their category (i.e. moderately atypical) are preferred over extremely incongruent products and completely congruent products. Products that are entirely congruent with the category schema are perceived as not noteworthy and exhibit lower consumer preference (Meyers-Levy and Tybout, 1989, p.40). Furthermore, Liu et al. (2017) found that moderate levels of typicality exhibit the highest consumer preferences with decreasing preferences for very high and very low typicality levels. Keeping product typicality in mind is therefore central to the design process, as it directly influences consumer perceptions and decision-making processes. While expert human designers can focus on their individual design process, novel technologies from the domain of generative models can support optimizing a finished design regarding its design typicality for increased consumer preferences. Such a model could edit a design such that it is more (or less) typical than the initial draft by the designer, thus increasing consumer liking. Since the overall design of the fashion item should not change significantly,

only minor and very targeted manipulations in the design are desired. This requires a very targeted and fine-grained image manipulation technique that can edit *aesthetic* attributes, e.g. typicality, of the design in a disentangled manner (i.e. without altering other attributes). There are multiple works that research design augmentation for *physical* attributes like sleeve length or color of fashion items (e.g. Choi et al. (2023), Chen et al. (2020), Ping et al. (2019), Kwon et al. (2022)). Furthermore, there is literature on *aesthetic* attribute manipulation in other domains using generative neural networks. For example, Goetschalckx et al. (2019) manipulates images regarding visual attributes like memorability or emotional valence. To the best of the author's knowledge, a disentangled and targeted manipulation technique for aesthetic attributes like design typicality does not exist yet. Therefore, the goal of this work is the

- development of a fine-grained typicality manipulation method for aesthetic attributes and
- the exploration of generative models, GAN inversion methods, and latent space manipulation techniques to achieve this goal.

The main contribution of this work is the development of a model that connects the research on aesthetic attribute manipulation using generative models and design augmentation in the fashion domain. While typicality is an *aesthetic* attribute, manipulation of the design typicality may involve implicit manipulation of *physical* attributes. The main difference to the works above, however, is the implicitness of *physical* attribute manipulation, as the manipulation is targeted towards an *aesthetic* attribute (typicality) which may result in manipulations of multiple *physical* attributes of the fashion item. To achieve the disentangled and targeted aesthetic attribute manipulation, a custom generative adversarial network (GAN), namely StyleGAN2-Ada (Karras et al., 2020a), is trained for the garment category of women's dresses. Furthermore, multiple GAN-inversion models are tested and the best models are selected. To assess the typicality of a dress, two alternative typicality measurements are developed and in a final step, real dresses are manipulated using a latent space editing technique. The remainder of this work is organized as follows: Section 2 reviews the necessary preliminaries and relevant literature. Section 3 details the training data while section 4 explains the methodology, covering generation, inversion, typicality measurement, and latent space manipulation. Section 5 presents the results of each sub-method and the overall typicality manipulation. Section 6 discusses the method and results, and Section 7 concludes the paper.

2 Background and Literature Review

2.1 Architecture Selection

In the landscape of deep generative modeling, Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs), and Diffusion models are the most commonly used architectures. While each of them has its strengths and weaknesses, I opt for the use of GANs. The main reasons for this will be described in this section. Since the goal of this thesis is to achieve highly controllable manipulations of visual attributes, having a well-defined, controllable, and interpolable latent space is a key requirement for the architecture. While both VAEs and GANs have such a latent space, VAEs lack the power to produce high-quality reconstructions (Müller et al., 2024, p.2) and clear images (Wang et al., 2020, p.1). Diffusion models on the other hand can produce high-fidelity images but have a highly unstructured latent space that lacks semantic meaning. Naively interpolating the latent space, as it is possible for GANs, leads to random and abrupt flickering and drastic changes in the generated image from diffusion models (Zhang et al., 2024, p.2). Making the latent space of diffusion models interpolable requires extensive efforts as e.g. Zhang et al. (2024) has shown by interpolating between the LoRA (Hu et al., 2021) adaptions of two images. Another approach would be to interpolate between encodings of text prompts obtained using textual inversion in text-guided diffusion models like Wang and Golland (2023) have done. In general, a controllable and interpolable latent space with effortless learning of semantic meaning is possible only using GAN architectures. Furthermore, although GANs generally achieve less mode coverage (i.e. diversity) in their generations than VAEs and diffusion models (Xiao et al., 2021, p.1), this does not pose a problem, since the application in this work builds on the narrow distribution of packshot images of dresses.

2.2 GANs

Generative adversarial networks (GANs), first proposed by Goodfellow et al. (2014), have revolutionized the field of generative modeling in deep learning. A GAN consists of two adversarial models, a *Generator* G and a *Discriminator* D. The generator aims to capture the data distribution and thus be able to produce data that mimics the real data. It takes data sampled from a prior distribution $\mathcal{Z}$, often a Gaussian distribution, and learns a function to map it into the data space. The discriminator model D simultaneously aims to distinguish between data from the real distribution and data generated by G. To enable G to learn the real data distribution and D to improve its discriminative power, a non-cooperative two-player minimax game is used in combination with simultaneous gradient descent (Saxena and Cao, 2021, p.5). The objective function of a basic GAN model uses two objectives: Parameters of D (θ_D) are chosen such that the negative log-likelihood for binary classification is minimized and parameters of G (θ_G) are chosen such that the generated examples have a high probability of being

real (Saxena and Cao, 2021, p.6). Thus, the overall objective function can be formulated as

$$\min_{\theta_G}\max_{\theta_D} V(G,D) = \min_{G}\max_{D} \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1-D(G(z)))] \tag{1}$$

where $V(G,D)$ is a binary cross entropy function (Saxena and Cao, 2021, p.6). In the beginning, none of the networks has any knowledge about the domain (Bermano et al., 2022, p.2). The models improve by learning from each other. Bermano et al. (2022) describe the training as a repetitive process, in which the generator tries to fool the discriminator which in turn learns to recognize this strategy. This training scheme allows the model to learn via self-supervision and interestingly, the generator learns to generate high-fidelity images without ever seeing a single real image (Bermano et al., 2022, p.2). As an extension, Mirza and Osindero (2014) introduced a conditional GAN architecture, where a conditioning of a single class of the domain can be achieved in both the generator and discriminator. Since in this work I focus on across-class manipulations, unconditional architectures are used which can capture the multimodal distribution of multiple classes in one model. The basic GAN architecture can be seen in figure 1.

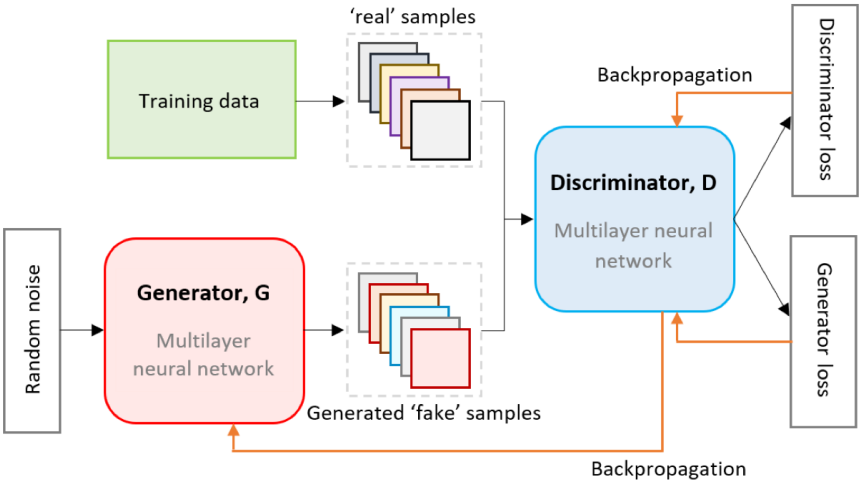


Figure 1: Basic GAN architecture - taken from Little et al. (2021)

While all GAN architectures share the concept of two adversarial networks, there are many combinations of generator architectures, discriminator architectures, loss functions, and optimization algorithms. Numerous architectures have gradually improved the image quality and training stability of GANs (e.g. DCGAN (Radford et al., 2015), WGAN (Arjovsky et al., 2017), PGGAN (Karras et al., 2017)). The final breakthrough of image generation using GANs, however, happened with the introduction of StyleGAN (Karras et al., 2019).

The **StyleGAN family** (e.g., Karras et al. (2019), Karras et al. (2020b), Karras et al. (2020a)) has achieved an unprecedented level of image quality and fidelity (Tov et al., 2021, p.1). StyleGAN has become the state-of-the-art GAN architecture, and the golden standard for facial image editing (Bermano

et al., 2022, p.1). Its dominance is due to design choices that create a well-organized, smooth, and highly disentangled latent space that exhibits a semantic understanding of the target domain merely through the effect of inductive bias (Bermano et al., 2022, p.6). A key innovation in StyleGAN is the mapping network $f : \mathcal{Z} \rightarrow \mathcal{W}$, which transforms Gaussian-distributed latent codes from $\mathcal{Z}$ into an intermediate latent space $\mathcal{W}$, ensuring better alignment with the real data distribution. This mapping allows the model to account for the non-uniform distribution of image attributes by "unwrapping" the latent space, making it more linear and easier to manage (Karras et al., 2019, p.6). As a result, the generator can more effectively disentangle and recreate various image features, enhancing the quality and diversity of the generated images. The main component in StyleGAN's architecture is the style modulation layers that allow precise control of the "style" in generated images. The latent codes in w are transformed to "styles" $\boldsymbol{y} = (y_s, y_b)$ using learned affine transformations. Using these transformations, different dimensions of the StyleGAN latent space $\mathcal{W}$ are injected at different levels of the synthesis process, allowing targeted variation at the coarse, medium, and fine details of the generated image (Karras et al., 2019, p.4). Additionally, by injecting random noise into each layer of the synthesis network, stochastic variation in the output is ensured (Karras et al., 2020b, p.1).

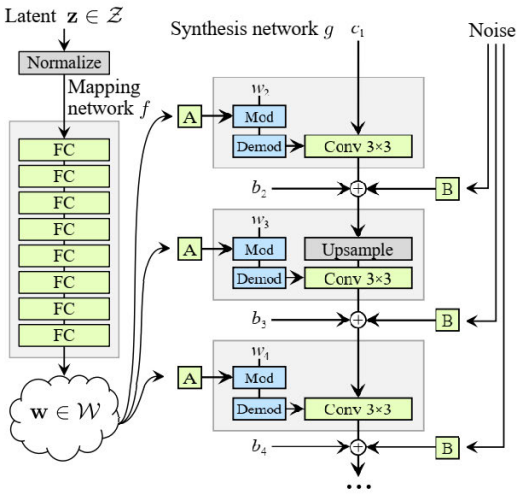


Figure 2: StyleGAN2 Architecture - taken from Hermosilla et al. (2021)

StyleGAN2, an improved version of the original StyleGAN, introduces enhancements that address artifacts and improve image quality by refining both the architecture and training processes (Karras et al., 2020b, p.2). A key change is the replacement of progressive growing with skip-connections in the generator and a residual discriminator (Karras et al., 2020b, p.6). This simplifies training by maintaining a fixed resolution throughout, eliminating the need for manual tuning as resolution increases. Another major improvement is the redesign of the style modulation layers through a modulation-demodulation operation. Here, style modulation is directly applied to the convolutional layer weights, followed by demodulation to normalize feature maps. This enhances the consistency and quality of the generated images. A detailed schema of the StyleGAN2 architecture is shown in figure 2. StyleGAN2 also introduces lazy regularization, computing regularization losses every 16 minibatches, which reduces computational cost and training time (Karras et al., 2020b, p.5). Additionally, the novel path length regularization ensures that small changes in the latent code $w \in \mathcal{W}$ lead to smooth, small changes in the generated image. This regularizer penalizes large variations in the perceptual space of consecutive latent codes during training, resulting in more reliable models and smoother generators, which are easier to invert (Karras et al.,

2020b, p.5). Since the model in this work needs to produce subtle changes when manipulating images, smooth latent space interpolation, achieved by StyleGAN2's path length regularizer, is crucial.

StyleGAN2Ada (Karras et al., 2020a) is a modified version of StyleGAN2 which introduces adaptive data augmentation techniques to address overfitting and improve generalization. Training GANs with a small dataset can easily lead to discriminator overfitting and thus training of the generator fails due to vanishing gradients. Using adaptive discriminator augmentations, the authors can stabilize training on small datasets and show that only a few thousand images are needed to train StyleGAN2Ada (Karras et al., 2020a, p.1). This is achieved through the use of an adaptive augmentation pipeline that introduces random transformations, such as translation, rotation, and color jittering, only when the discriminator begins to overfit. Since in this work, a limiting factor is the access to high-quality training data, using StyleGAN2Ada is the logical choice, as it allows training of a high-quality generator on a small sample.

2.3 GAN Inversion

While synthesizing arbitrary fake images from a pre-trained generator is straightforward, editing real images is a non-trivial task. For latent space editing techniques to be applied to real images, one must first find the corresponding latent code of the real image using a *GAN Inversion* technique. Initially introduced by Zhu et al. (2016), the goal of GAN inversion is to find a latent code $w \in \mathcal{W}$ which most closely reconstructs the real image. Following Bermano et al. (2022), inversion can be formally defined as: Given a pre-trained generator $G : \mathcal{W} \to \mathcal{X}$ and real image $x \in \mathcal{X}$, GAN inversion aims to find $w \in \mathcal{W}$ such that

$$w^* = \underset{w}{\operatorname{argmin}} \mathcal{L}(x, G(w)), \tag{2}$$

where $\mathcal{L}$ is any reconstruction loss. Due to the non-convexity of the generator, this leads to a non-convex optimization problem. Approaches to solve this problem can be grouped into learning-/encoder-based, optimization-based, and hybrid methods (Xia et al., 2022, p.6) and will be explained below. Note that inversion can be executed in any latent space ($\mathcal{Z}$, $\mathcal{W}$, etc.) of any GAN architecture. In the following, however, I will focus on the latent spaces of StyleGAN-based architectures.

2.3.1 Latent Spaces

Choosing the latent space for the inversion is an important design choice since this space should allow a precise reconstruction while facilitating disentangled editing and other downstream tasks (Xia et al., 2022, p.5). For StyleGAN-based architectures, most inversion techniques use $\mathcal{Z}$, $\mathcal{W}$, $\mathcal{W}^+$, $\mathcal{S}$, $\mathcal{P}_N$ or $\mathcal{P}_N^+$.

The first latent space in StyleGAN is $\mathcal{Z}$-space which follows a normal distribution and thus has a

closed-form definition. As the generator learns a mapping $\mathcal{Z} \rightarrow \mathcal{X}$, images from the GAN's latent space can easily be sampled from $\mathcal{Z}$ (Bermano et al., 2022, p.5). By using the nonlinear mapping network $f : \mathcal{Z} \rightarrow \mathcal{W}$ of the StyleGAN architecture (see section 2.2), a distribution of $\mathcal{W}$ is learned, which better captures the distribution of the real data (Karras et al., 2019, p.6) and has better disentanglement than $\mathcal{Z}$ (Shen et al., 2020, p.7). Abdal et al. (2019) propose an extension of $\mathcal{W}$, $\mathcal{W}^+$ to solve the limited expressiveness of $\mathcal{W}$ when representing real images in the inversion task. While in $\mathcal{W}$ space all rows of the 16x512-dimensional latent vector (for 512x512 output images) are identical, $\mathcal{W}^+$ allows for 16 different 512-dimensional style vectors that are inserted at different layers of the synthesis network. While allowing for more precise reconstructions due to higher expressiveness, using $\mathcal{W}^+$ can come at the expense of image quality, as operating in $\mathcal{W}^+$ makes regions in the latent space available, which are outside of the distribution the generator was trained on (Bermano et al., 2022, p.5). As almost all latent space manipulation methods work with the more common $\mathcal{Z}^{(+)}$- and $\mathcal{W}^{(+)}$-spaces, inversion methods that use $\mathcal{S}$-space (Wu et al. (2021)) or $\mathcal{P}_{\mathcal{N}}$- and $\mathcal{P}^+_{\mathcal{N}}$-space (Zhu et al. (2020b)) are not very relevant to this work. For a schematic overview of the StyleGAN latent spaces, see figure A.1 in the appendix. Note that in this work, inversion and manipulation in $\mathcal{Z}$, $\mathcal{W}$ and $\mathcal{W}^+$ have been performed and eventually it was decided to focus only on $\mathcal{W}^+$ due to its superior performance in the inversion process.

2.3.2 Inversion Methods

Optimization-based GAN inversion techniques (e.g. Yeh et al. (2017), Creswell and Bharath (2018), Gu et al. (2020), Bau et al. (2020), Lipton and Tripathi (2017)) derive the latent vector for a given image by solving equation 2 on an instance level. While this achieves high reconstruction quality of the real input images, this comes at the cost of computational efficiency (Alaluf et al., 2021b, p.2) and lower editability (Tov et al., 2021, p.3). Therefore, no optimization-based methods were implemented in this work.

Encoder-based GAN inversion approaches mostly train an encoder network $E(x,\theta_E)$ to obtain a mapping from the input image to its latent code. The objective function closely resembles this of an autoencoder pipeline where the decoder is the fixed generator for which the inversion network is obtained (Xia et al., 2022, p.6). The training objective can be defined following Bermano et al. (2022), p.9 as:

$$\theta_E^* = \underset{\theta_E}{\operatorname{argmin}} \sum_i \mathcal{L}(\boldsymbol{x}_i, G(E_{\theta_E}(\boldsymbol{x}_i))). \tag{3}$$

There have been extensive efforts to encoder-based architectures since Luo et al. (2017) first proposed an autoencoder-inspired approach (e.g. Kim et al. (2021), Wang et al. (2022), Shanyan et al. (2020), Pidhorskyi et al. (2020), Kang et al. (2021), Richardson et al. (2021), He et al. (2016)). The encoder

model used in this work is the state-of-the-art encoder model encoder4editing (e4e), introduced by Tov et al. (2021). e4e was designed with a particular focus on editing capabilities of the inversion. By analyzing the trade-offs between distortion, perceptual quality, and editability, the authors can design an encoder that ensures editability while maintaining a high reconstruction quality. They find that editability and perceptual quality are high when an image is inverted close to $\mathcal{W}$ but lives in $\mathcal{W}^+$ to simultaneously achieve low distortion in the generation (Tov et al., 2021, p.2). This is achieved by adding multiple loss terms to the encoder. First, distortion is minimized by adding a similarity loss based on a ResNet-50 (He et al., 2016) network, an $\mathcal{L}_2$ loss term, and a loss term based on LPIPS (Zhang et al., 2018). To maximize perceptual quality, a loss term is used to minimize the variation within the learned latent codes (Tov et al., 2021, p.5). Furthermore, a latent discriminator (Nitzan et al., 2020) is employed to ensure proximity of the latent code to $\mathcal{W}$ (Tov et al., 2021, p.5). Encoder4editing poses an important basis for many inversion techniques. See figure A.3a in the appendix for a schematic overview of the method. While encoder-based methods are considerably faster than optimization-based methods, their reconstructions are less accurate (Bermano et al., 2022, p.10). To combine the fast inference time of encoders and the generally better performance of optimization models, several hybrid models have been proposed.

Hybrid models obtain an initial latent code using an encoder and then further optimize this on a per-image level (e.g. Zhu et al. (2016), Zhu et al. (2020a)). Out of the universe of hybrid models, the most promising models based on the overall literature have been selected and tested in this work. These methods are explained below in more detail. Alaluf et al. (2021b) propose **ReStyle**, an iterative approach, in which latent codes obtained from an e4e encoder are iteratively improved by learning residuals. Instead of obtaining the inversion from a single forward pass, the encoder is iteratively refined using a small number of steps that leverage a feedback algorithm between the predicted and the real image (Alaluf et al., 2021b, p.2). Each iteration computes a residual vector correcting the previous latent code, enhancing high-frequency details and complex structures. This feedback mechanism allows intermediate reconstructions to better align with the target image, yielding lower LPIPS and MSE scores, while preserving semantic consistency and producing visually coherent results (Alaluf et al., 2021b, p.6). For a detailed schema of ReStyle see figure A.3b in the appendix.

Alaluf et al. (2022) propose **HyperStyle** which uses a hypernetwork-based encoder that learns offsets for the convolutional filters of the generator alongside the initial latent codes that are obtained from a fixed encoder (e.g. ReStyle). Thus, inference time is by a magnitude lower than for instance-level optimization while promising comparable reconstruction results (Alaluf et al., 2022, p.5). HyperStyle claims to bridge the gap between high-quality reconstructions that are obtained from generator-optimization approaches and fast inference time in encoder-based methods. Furthermore, the iterative refinement scheme introduced in ReStyle (Alaluf et al., 2021b) is adapted to the hyper-network training

to achieve stronger expressive power and more accurate inversions (Alaluf et al., 2022, p.5). For a detailed schema of HyperStyle see figure A.3c in the appendix.
Roich et al. (2022) propose **PTI**, a novel approach that combines initial inversions from an encoder and subsequent pivotal tuning of the generator, which achieved state-of-the-art performance in both distortion and editability. The initial latent code is obtained from an e4e model and thus lies within the well-behaved region of the latent space which allows editability (Tov et al., 2021, p.2). This code is used as a pivot around which the generator is trained for a few steps to better reconstruct the image while keeping nearby identities intact through a regularization term (Roich et al., 2022, p.1). While this achieves very high reconstruction quality and editability, it is also computationally expensive and complicates downstream tasks because each inversion needs its fine-tuned generator weights.

2.4 Manipulation

While good inversion performance allows to accurately reconstruct a given image, the main goal of this work is to be able to consistently manipulate image attributes. Latent space manipulation is a research field closely connected to GAN inversion that aims at finding techniques to manipulate latent space representations of images such that the generated image from the manipulated latent space exhibits the desired edited attributes. Since the StyleGAN latent space is smoothly interpolable, moving in a carefully chosen direction in this space should also result in smooth interpolations in the desired semantic attributes (Bermano et al., 2022, p.6). In general, semantic image manipulation using GANs can be performed using either specifically designed architectures based on conditional GANs or latent space manipulation techniques applied to pre-trained unconditional GANs. While models like StarGAN (Choi et al., 2018) and StarGANv2 (Choi et al., 2020) yield a more compact pipeline that does not require a specific model for each step, attributes have to be specified before training the generator. I chose to rely on an unconditional GAN model and additional latent space editing techniques since this approach is more general and yields better image quality (Abdal et al., 2021, p.2). Within those latent space navigation techniques, one can further differentiate between supervised (e.g. Goetschalckx et al. (2019), Shen et al. (2020), Wu et al. (2021), Yang et al. (2021)) and unsupervised (e.g. Ren et al. (2021), Yüksel et al. (2021), Härkönen et al. (2020), Shen and Zhou (2021)) approaches and between linear and non-linear (e.g. Abdal et al. (2021), Li et al. (2023), Chen et al. (2022)) approaches. The most important techniques will be briefly discussed below. Härkönen et al. (2020) use an unsupervised approach by applying PCA on the latent space of a specific dataset. The principal components then correspond to specific attributes, however this annotation has to be done by manual inspection of resulting image manipulations. Shen and Zhou (2021) proposed another unsupervised approach in which they use a semantic factorization method which is based on the

principal components from the weights of the first layer of a pre-trained GAN. While unsupervised approaches may yield good results in the absence of labeled data, supervised approaches can leverage the information contained in the labels to find more accurate directions in the latent space. In one of the earlier works, Goetschalckx et al. (2019) propose a method to manipulate cognitive image properties like memorability. They achieve this by learning a transformer function that adds a learned direction to the initial latent code in $\mathcal{Z}$-space such that the assessor model at the end of the pipeline detects the altered memorability of the generated image. While this is a rather simple approach, it can be adapted to more complex workflows and other latent spaces. Building on the idea of learning a latent space offset based on assessor scores, Shen et al. (2020) propose a technique that learns directions in latent space based on hyperplanes separating instances based on binary attributes. This method called InterFaceGAN will be explained in more detail in section 4.4.1. While the above-mentioned methods mostly learn directions in latent space such that by traversing the latent space in this direction, the desired attribute changes in the output, non-linear approaches also yield promising results. StyleFlow, one of the most promising techniques, was proposed by Abdal et al. (2021) and relies on normalizing flows that learn a mapping between $\mathcal{Z}$ and $\mathcal{W}$ conditioned on specific attributes. Manipulation is performed by reversing the mapping conditioned on the real attributes and then making a forward pass to the latent space using the manipulated attributes as the condition (Abdal et al., 2021, p.5). In this work, StyleFlow has been implemented and tested but was later disregarded due to its considerably higher computational cost and lower editing performance compared to InterFaceGAN.

3 Dataset

All models are trained on a newly created dataset of dresses from Zalando, one of the most popular online fashion retailers in Europe (Freno, 2017, p.1). The initial dataset consists of article information and image URLs of all publicly available dresses listed on the website "zalando.de" at the time of creation in March 2024. As a first pre-processing step, all articles that do not offer a standardized and centered packshot image of the dress are removed. Afterwards, all images are downloaded, squared in 1024x1024 resolution, and centered. Additionally, the article information is subset to attributes that offer good data completeness and non-noisy labels. The subset consists of the attributes "brand", "price", "category", "fabric", "fit", "neckline", "pattern", "collar", "length", "shape" and "sleeve length". These attributes are cleaned such that similar but small categories are grouped and categories with very few samples are recoded as missing to reduce the dimensionality of each attribute. Since there is no standardized color attribute, the color of each dress is obtained using a pre-trained CLIP model (Radford et al., 2021, p.2) and a fixed list of 14 colors. The final dataset consists of 14,060 images with the corresponding article information from 643 unique brands. A random sample of images from

the dataset can be seen in figure 3.

Figure 3: Random samples from the dataset

4 Method

To achieve the goal of this work, the creation of a fine-grained typicality manipulation method, a modular and sequential workflow needs to be designed, which can be seen in figure 4. A given real image first needs to be inverted using a GAN inversion model (section 4.2) and the corresponding latent code obtained. Using a latent space manipulation technique (section 4.4), multiple new latent codes can be found that should exhibit the desired attributes. These latent codes are passed through a custom generator model and thus turned into images for which the typicality scores can be calculated using the custom typicality assessors described in section 4.3. In practice, the order of training the

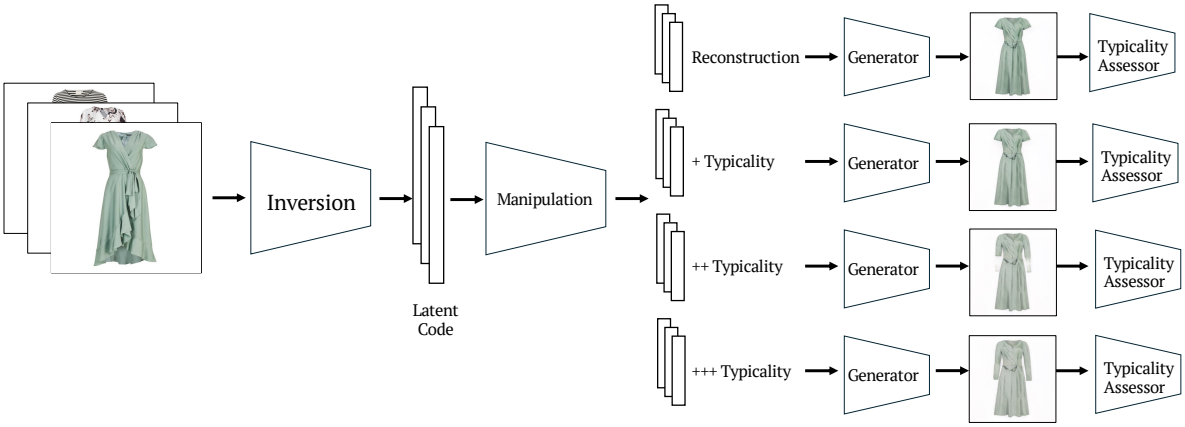


Figure 4: Inference Workflow

models differs from this inference workflow (see figure 5). As a first step, a generator model needs to be trained which is capable of generating images similar to the distribution of the images one wants to manipulate. This generator model is used in the second step when building an inversion model that achieves good reconstruction and editing performance. As a third step, the typicality assessor is built

using the reconstructions of the real images. In the last step, the manipulation model is trained based on the typicality scores and the latent codes from the inversion step. In the following sections, I will explain the different steps of this process in more detail and explain the methods applied in this work.

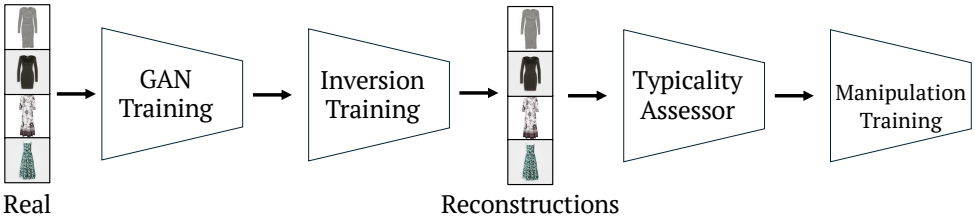


Figure 5: Training Workflow

4.1 Generation

While multiple GAN-based architectures might be suitable, StyleGAN-based architectures have become the golden standard for image editing (Bermano et al., 2022, p.1). This is mainly because they learn a highly disentangled and smooth latent space which allows for semantic attribute manipulation (see section 2.2). Within the family of StyleGAN-based architectures, StyleGAN2-Ada (Karras et al., 2020a) is chosen due to its remarkable capabilities to be trained on a small sample. Since the dataset acquired for training is limited by the availability of products on Zalando, training a traditional GAN model is not easily possible. Using StyleGAN2-Ada allows to train a high-fidelity generator even with a dataset of only 14k images.

4.1.1 Training

The StyleGAN2-Ada model used in this work has been trained for a resolution of 512x512 pixels per image using the "auto" configuration as defined in Karras et al. (2020a) and additional x-flip augmentations to increase the dataset size. This configuration uses a reduced depth of the mapping network from 8 to 2 linear layers as this is sufficient according to the authors (Karras et al., 2020b, Appendix p.35). A transfer learning approach from the FFHQ-dataset (Karras et al., 2019) is chosen since both datasets rely on centered images with non-relevant backgrounds. Using a pre-trained StyleGAN2-Ada model as the starting point speeds up the training process. The model has been trained for 26 hours on two NVIDIA Tesla V100 GPUs with 32GB VRAM. In total, 2000k images have been shown to the discriminator. Network snapshots have been saved every 40k images shown to the discriminator to later determine the best snapshot using the evaluation metrics described in section 4.1.2.

4.1.2 Evaluation

The two most widely used GAN evaluation metrics are the Fréchet Inception Distance (FID) and the Inception Score (IS) (Borji, 2022, p.2). The IS (Salimans et al., 2016) evaluates the quality of generated images using ImegNet (Deng et al., 2009) predictions from an InceptionV3 (Szegedy et al., 2016) model. It combines the confidence of the conditional class predictions (indicating image quality) with the diversity of the predicted classes across the dataset (indicating image diversity). FID (Heusel et al., 2017) on the other hand calculates features of the real and generated datasets using intermediate layers of the same InceptionV3 (Szegedy et al., 2016) network and computes the Fréchet Inception Distance distance between them. Heusel et al. (2017) define the FID using multivariate Gaussians over the extracted features as:

$$\mathrm{FID}(x,g) = \|\boldsymbol{\mu}_{\boldsymbol{x}} - \boldsymbol{\mu}_{\boldsymbol{g}}\|^2 + \mathrm{Tr}\left(\boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{g}} - 2(\boldsymbol{\Sigma}_{\boldsymbol{x}}\boldsymbol{\Sigma}_{\boldsymbol{g}})^{1/2}\right),$$

where $\boldsymbol{\mu}_{\boldsymbol{x}}$, $\boldsymbol{\mu}_{\boldsymbol{g}}$, $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{g}}$ are the mean vectors and covariance matrices of the real and generated image distributions, respectively, and Tr denotes the trace of a matrix.

FID is consistent with human inspection and can detect intra-class mode collapse (Borji, 2022, p.2). As a drawback, for FID to be accurate and comparable, it needs to be calculated on a large and comparable number of samples. Hence, in all my experiments, I calculate the FID based on 50k generated images. As an alternative method, Bińkowski et al. (2018) proposed the Kernel Inception Distance (KID) which promises to be an unbiased version of the FID. In practice, there is a very high Spearman rank-order correlation between FID and KID (Kurach et al., 2018, p.3). While both the IS and the FID are single-score metrics that evaluate the quality and diversity of generations together, Sajjadi et al. (2018) propose improved formulations of precision and recall as GAN evaluation metrics. Recall estimates the fraction of the training set that the generator can reproduce while precision denotes the fraction of generated images that are realistic (Kynkäänniemi et al., 2019, p.5). Thus, when training a GAN, one aims for high precision and recall such that the generated images are as realistic as possible while also being as diverse as possible. Naturally, there is a trade-off between the two since a model that can create more diverse generations spans a much larger latent space, thus possibly allowing latent codes from ill-defined regions of this latent space that exhibit less realistic generations. To get a holistic assessment of the generation quality, I calculate all the above-mentioned evaluation metrics and select the best model snapshot based on a comprehensive analysis of the metrics.

4.2 Inversion

After a thorough literature review of available inversion methods (see section 2.3), a set of suitable methods was selected. The baseline model chosen is e4e (Tov et al., 2021) from the encoder-based

models. Additionally, three hybrid models are tested: PTI (Roich et al., 2022), ReStyle (Alaluf et al., 2021b), and Hyperstyle (Alaluf et al., 2022). Among these, PTI promises the best inversion performance but with a high inference time. To reduce inference time, ReStyle and Hyperstyle were selected as they promise similar results to PTI at a fraction of the time.

4.2.1 Training

For training of the e4e model, the dataset has been randomly split into 90% training samples and 10% test samples. The model was trained with default parameters as defined in Tov et al. (2021) and checkpoints were saved every 2000 steps. In total, the model has been trained for 38 hours on one GPU. As can be seen in the loss curves for the overall loss, the LPIPS loss, and the L2 loss in figure A.2, the model has converged within this time. ReStyle and Hyperstyle have been trained with default parameters. ReStyle has initially been trained for 35 hours on one GPU but had not converged yet. Therefore it was trained for another 20 hours, totaling 55 hours of training time. Hyperstyle was trained for 40 hours and an additional 30 hours until convergence, totaling 70 hours of training time on one GPU. For all models, NVIDIA Tesla V100 GPUs with 32GB of VRAM were used.

4.2.2 Evaluation

Evaluating GAN inversion involves multiple dimensions. Generally, evaluation metrics can be grouped into three dimensions: Faithfulness (i.e. reconstruction quality), photorealism (i.e. perceptual quality), and editability of the latent code (Xia et al., 2022, p.3).
Faithfulness measures the quality of the reconstruction, i.e. how closely the generated image from the inversion resembles the real image. One of the early measures proposed was Peak signal-to-noise ratio (PSNR). Many works additionally consider the Structural Similarity Index (SSIM) (Wang et al., 2004) which measures the similarity using three key features: Luminance, Contrast, and Structure (Wang et al., 2004, p.6). It has been shown that SSIM more closely correlates with human perception than PSNR (Sheikh et al., 2006, p.19). For my evaluation, I use the advanced version of SSIM, the Multi Scale Structural Similarity Index Measures (MSSSIM) proposed by Wang et al. (2003). Furthermore, multiple pixel-wise distances (e.g. mean squared error, mean absolute error) have been proposed. Since close reconstruction of input images is important in this work, the pixel-wise $\mathcal{L}_2$ distance will be used as an additional measure of faithfulness.
Measuring the perceptual quality of images (i.e. photorealism) against a reference dataset is described in chapter 4.1.2 in more detail. Many works rely on the same measures to evaluate the quality of the inversion process. I use the FID (Heusel et al., 2017) to measure how close the generated images from the inversions resemble the distribution of the real data. Furthermore, I use Learned Perceptual

Image Patch Similarity (LPIPS) proposed by Zhang et al. (2018) which measures similarity between two images using features extracted from a VGG model (Simonyan and Zisserman, 2014) trained on ImageNet (Deng et al., 2009).

While the first two dimensions can be directly measured using multiple metrics, the editability of the latent code from an inversion method is not easily tested. Evaluating the editability involves yet another model, the editing technique. Thus, the evaluation of this dimension is very closely connected to the evaluation of the manipulation technique described in section 4.4. If the manipulation is not successful, it is difficult to find out whether the latent code itself is not editable or whether the manipulation method for the attribute does not work. As a qualitative measure for editing capabilities of inversions, Zhu et al. (2020b) propose to evaluate the quality of images generated from interpolating between two latent codes. They argue that an inversion is good if the interpolated images are of high quality (Zhu et al., 2020b, p.10). In general, preserving the original identity during edits and only editing localized attributes is of high importance. Alaluf et al. (2021a) and Richardson et al. (2021) use facial recognition networks as proposed in (Deng et al., 2019) to assess identity preservation via cosine similarity of facial identity representations. For non-facial data, using this technique is challenging since training identity recognition networks for other domains is difficult (Bermano et al., 2022, p.13). As Tov et al. (2021) point out, all popular quantitative metrics to evaluate editing performance contradict each other and often the human judgment (Tov et al., 2021, p.7).

Given the difficulty of evaluating the editing capabilities of inversion methods, I focus on quantitative metrics for faithfulness and photorealism when comparing the different inversion methods. To assess the editing capabilities of the latent codes, I rely on interpolation experiments as proposed by (Zhu et al., 2020b) and eventually on the results obtained from the downstream manipulation method.

4.3 Typicality Measures

Measuring typicality, i.e. how "dressy" a dress looks like, is a central aspect of the overall workflow. Having a consistent typicality measure allows to create supervision needed in the training of the manipulation method. Furthermore, it allows to validate the success of the manipulation ex-post quantitatively. Based on work by Landwehr et al. (2011), typicality is measured based on the cosine similarity of a product's features (i.e. its embedding) to a morph of the features of all products. With $f(\cdot)$ being any embedding model, the d-dimensional embedding for dress D_i is calculated as $\boldsymbol{v}_i = f(D_i)$. Furthermore, the morph of all dresses is

$$\boldsymbol{m} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{v}_i,$$

with $\boldsymbol{v}_i \in \mathbb{R}^d$ and $\boldsymbol{m} \in \mathbb{R}^d$. Finally, the typicality measure for dress D_i is

$$typicality = cosine\_similarity(\boldsymbol{v}_i, \boldsymbol{m}) = \frac{\boldsymbol{v}_i \cdot \boldsymbol{m}}{|\boldsymbol{v}_i||\boldsymbol{m}|}.$$

In the first step, as a simple embedding model the pre-trained Vision Transformer (ViT) DINOv2 (Oquab et al., 2023) is used. More specifically, the "Base" version is used with a patch size of 14, i.e. the "ViT-B/14" configuration. The resulting embeddings are 1x768 dimensional. The authors claim that the representations are task-agnostic and general-purpose (Oquab et al., 2023, p.1) and thus can be used for typicality analysis. While this approach may overfit to non-essential attributes or use shortcuts, the second embedding model I use aims to produce more holistic and robust representations. It does so by forcing the embedding model to learn multiple sub-embeddings, one for each relevant attribute of the dress. Thus, the decision about which features or attributes are important to represent a dress is not exclusively taken by the model as in the pure DINOv2 case but is defined ex-ante based on domain knowledge. The model starts with the simple DINOv2 embeddings and passes them through some fully connected layers. Afterwards, the embedding is split up into k subspaces $a_1, a_2, ..., a_k$ by individual classification heads $C_1, C_2, ..., C_k$, one for each attribute. The attributes selected consist of *Color, Fabric, Fit, Neckline, Pattern, Collar, Length, Shape* and *Sleeve Length*. Since the goal is to achieve disentangled sub-embeddings, a distance correlation (dCor) loss is added to the model training, enforcing the reduction of shared information across sub-embeddings. The distance correlation proposed by Müller et al. (2024) measures nonlinear dependencies between vectors of any dimension and has been used by the authors for disentangled representation learning. The dCor loss is defined as

$$\mathcal{L}_{dCor} = \sum_{i \neq j} \mathrm{dCor}(a_i, a_j),$$

with

$$\mathrm{dCor}(a_1, a_2) = \sqrt{\frac{\mathrm{dCov}^2(a_1, a_2)}{(\mathrm{dVar}^2(a_1)\mathrm{dVar}^2(a_2))^{1/2}}}.$$

Details about calculating dCov^2 and dVar^2 can be found in the original paper (Müller et al., 2024, p.5) and in the appendix in equation 4 to 11. A schematic overview of the disentangled embeddings model can be found in figure 6.

Each subspace embedding in this disentangled embedding model is 1x256 dimensional and thus the complete embedding, which is a concatenation of all subspace embeddings, has dimension 1x2304. Apart from the complete concatenation, I also construct embeddings that exclude one subspace at a time. This might be e.g. all subspace embeddings except the *Color* subspace such that the model can calculate the typicality of a dress but does not take the *Color* attribute of the dress into account.

It is important to note that both typicality measures are learned on the reconstructed images using the e4e inversion (Tov et al., 2021). Thus, the embeddings for each dress, as well as the morphs, are

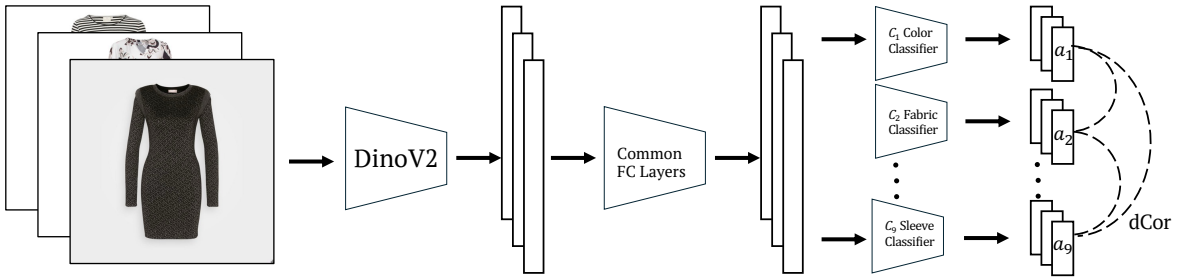


Figure 6: Disentangled Embeddings Model for Typicality: *Initial DINOv2 embeddings are divided into subspace embeddings a_1,a_2,...,a_k by classifiers. Disentanglement is forced by a distance correlation loss between all sub-embeddings*

calculated based on the respective reconstructions. Training the typicality measure on real images and using generated images at test time holds potential for problems. Although the reconstructions are very accurate, pixel values and low-level image features might differ between the generator outputs and the real image distribution, thus introducing possible mismatches in the typicality calculations.

4.4 Manipulation

As mentioned before, I tested a linear (*InterFaceGAN)* and a non-linear (*StyleFlow*) approach, both from the family of supervised latent space manipulation techniques. Due to its low editing performance and prohibitively high computational cost, *StyleFlow* has been disregarded after initial editing experiments on physical attributes like color or sleeve length of a dress. *InterFaceGAN* on the other hand was used throughout all experiments in this work.

4.4.1 InterFaceGAN

The latent space of StyleGAN models is smoothly interpolable (Karras et al., 2019, p.1) and gradual changes in one direction in the latent space of a well-trained generator can result in gradual changes in a specific semantic attribute (Bermano et al., 2022, p.6). Shen et al. (2020) propose a simple, yet effective method of finding this direction in a method called *InterFaceGAN*. The method is based on the assumption that for any binary semantic, a hyperplane in the latent space can be found that separates the two classes (Shen et al., 2020, p.3). Assuming a hyperplane with a unit normal vector $\mathbf{n} \in \mathbb{R}^d$, with d being the dimension of the latent space, the "distance" of the latent representation of sample $\mathbf{z}$ to the hyperplane can be defined as

$$d(\mathbf{n},\mathbf{z}) = \mathbf{n}^T \mathbf{z}.$$

If one moves towards and across the hyperplane in the latent space, the sign of the distance changes, and thus the semantic attribute in the generated image should have changed to the opposite class (Shen et al., 2020, p.3).

While this is straightforward for a single binary attribute, often multiple semantics are present and might not be independent of each other, although the latent space of StyleGAN models already enforces some degree of disentanglement (see section 2.2). In the case of *m* semantics, *m* separation boundaries $\mathbf{N} = [\mathbf{n_1},...,\mathbf{n_m}]$ can be found. To disentangle them, $\{\mathbf{n_1},...,\mathbf{n_m}\}$ need to be orthogonal to each other (Shen et al., 2020, p.4). In practice, conditional manipulation, i.e. editing one semantic without affecting any other semantic, can be performed using projection. As illustrated in figure 7, if one wants to manipulate the semantic connected with the hyperplane $\mathbf{n_1}$ without affecting the semantic connected with $\mathbf{n_2}$, the projection of $\mathbf{n_1}$ onto $\mathbf{n_2}$ is subtracted from $\mathbf{n_1}$. Thus, the conditional manipulation direction is

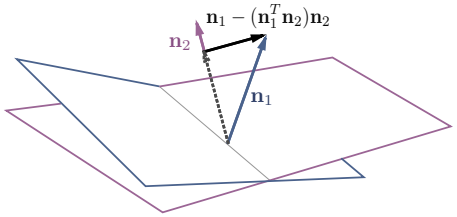


Figure 7: Conditional manipulation via subspace projection - taken from Shen et al. (2020), p.4

$$\mathbf{n_{cond}} = \mathbf{n_1} - (\mathbf{n_1}^T \mathbf{n_2})\mathbf{n_2}.$$

For the semantic editing, the authors apply a simple linear manipulation of the latent space in the positive or negative direction of the unit normal vector $\mathbf{n}$ of the separation boundary. Thus the manipulated latent representation is defined as

$$\mathbf{z}_{edit} = \mathbf{z} + \alpha\mathbf{n},$$

with the synthesis being more positive on the semantic for $\alpha > 0$ and more negative with $\alpha < 0$ (Shen et al., 2020, p.4).

Since *InterFaceGAN* is a supervised approach, either labels for the semantics or classifier scores need to be available. If there are labels available, those can directly be used to find a hyperplane that separates the latent space based on these labels. If only classifier scores are available, the authors propose to use the samples with the highest and lowest scores and assign binary labels. The hyperplane is then found by training a simple linear classifier on the latent representations. In this case, a linear Support Vector Machine (SVM) is trained. Linear SVMs learn a decision boundary which acts as the separation boundary in latent space that is used by InterFaceGAN.

4.4.2 Training

To make the *InterFaceGAN* approach usable for this work, some implementation details needed to be adapted. While the authors used the 512-dimensional $\mathcal{Z}$-space of StyleGAN, I used the

16x512-dimensional $\mathcal{W}^+$-space. Thus, finding a hyperplane in this higher dimensional space is not as straightforward as in $\mathcal{Z}$. To tackle this problem, one could either flatten the latent representation in $\mathcal{W}^+$-space by concatenating all 16 dimensions along one dimension or learn 16 separate hyperplanes, one for each of the dimensions. The latter results in individual manipulations in each of the 16 dimensions. As explained in section 2.2, the various dimensions of $\mathcal{W}$ and $\mathcal{W}^+$ are injected at different levels of the synthesis process and allow targeted variations at different scopes of the image. Thus, learning isolated separation boundaries allows to apply targeted manipulation in the specific semantic only at a certain coarseness level of the generation process. On the other hand, learning one separation boundary from the complete latent representation allows the linear classifier to consider all features at all levels at the same time when finding the manipulation direction. Both approaches have been implemented and tested and the results are very similar. Since the isolated boundaries approach requires another step of selecting the dimensions to consider in the manipulation, the method of flattening the latent space has been selected as the final approach, thus simplifying the process without sacrificing editing performance.

When constructing the training data for the typicality scores, I closely follow the authors of *InterFaceGAN* by using only the top n and bottom n scores in the classifier training. The idea is to create two distinct classes that can be clearly separated and that exhibit strong attributes of the two classes, i.e. high and low typicality. The boundaries have been computed using values of $n = 1000$, $n = 2000$, and $n = 3000$. Finally, the separation accuracy on a separate validation set has been computed and the boundary with the best accuracy was selected as the final manipulation direction.

Another difference in the *InterFaceGAN* implementation between this work and the original approach is the nature of the semantics. While the authors could show that the approach works well on naturally binary attributes like gender or the presence of glasses in facial images, or ordinal features like the age of a person, the labels present in the dresses dataset are different. Physical attributes like color or sleeve length, which are used as conditioning factors in the typicality manipulation, are nominal and have high cardinality. For those attributes, $\frac{n!}{2!(n-2)!}$ hyperplanes are calculated with n being the number of unique values an attribute has. Therefore, there are hyperplanes for each combination of binary directions in the feature space. Defining one attribute value as positive while assigning all the others to the negative class (one-vs-rest) leads to multiple problems. First of all, the linear classification problem becomes extremely imbalanced. Furthermore, according to Yang et al. (2021) that set of negative samples provides ambiguous or even misleading guidance to the linear classifier, resulting in entangled semantics and incorrect manipulation directions (Yang et al., 2021, p.1). When choosing a specific boundary as the conditional boundary, the set of boundaries to choose from is restricted to those that cover the real attribute value. So if the conditioning is on the color attribute and the dress is blue, all boundaries that include "blue" together with any other color are candidates. Out of the

candidates, the boundary that has the best accuracy on the validation set during the classifier training is chosen. To visually validate the quality of these physical attribute boundaries, some preliminary manipulation tests using physical attributes have been run. The results of these experiments can be found in the appendix in figure A.5. Furthermore, descriptive statistics for the multiple separation performances of the physical attributes can be found in table 3 in the appendix.

5 Results

5.1 Generator Training

To evaluate the performance of the StyleGAN2-Ada generator, multiple evaluation metrics have been computed for different snapshots saved during training. Training curves for the StyleGAN2-Ada model can be found in the appendix in figure A.4. FID and KID are at the lowest at step 1200, meaning after 1200k images have been shown to the discriminator. Interestingly, IS shows the highest value at step 1800. Since the value at step 1200 is also fairly high and IS is less meaningful for the task at hand as described in section 4.1.2, this discrepancy is ignored. Looking at the Precision and Recall curves, Precision, i.e. the fraction of realistic-looking images peaks early at step 600 while Recall, i.e. the diversity of generated images peaks at step 1200. Since there is an inherent trade-off between the two, I choose higher diversity while accepting a slightly lower fidelity in the generations. Consequently, step 1200 is chosen as the model to be used in all downstream applications.

Figure 8: Generated Dresses from StyleGAN2-Ada Model *The random generation showcases the diversity of dresses that the generator is capable of generating. At the same time, some generations may exhibit artifacts like the two different sleeve lengths in the upper left image.*

To validate the training results, both a quantitative evaluation against comparable models in literature as well as a qualitative evaluation are conducted. With a final FID score of 8.09, the generator performance

lies exactly in the range of 7.5-10.0 which has been established in the original StyleGAN2-Ada paper for datasets with comparable sizes (Karras et al., 2020b, p.8). Other applications of StyleGAN2-Ada have similar FID scores (e.g. 8.7 in Fu et al. (2022)) or much higher scores (e.g. 13.68 in Hermosilla et al. (2021) and 26.02 in Kim et al. (2022)). A similar application that aims at clothing generation could achieve an FID of 4.53 using a conditional StyleGAN2-Ada model and training on a much larger dataset of 62k images for ten days. Given those benchmark results, the FID achieved for my StyleGAN2-Ada model is well in the range of applications with similar dataset sizes and computational resources used. For a qualitative evaluation of the images generated by the final model, please refer to figure 8.

5.2 Inversion

In general, all inversion models implemented (e4e, ReStyle, Hyperstyle, PTI) achieve relatively good reconstruction quality as can be seen in figure 9. It becomes clear that even the encoder-based method e4e can relatively well reconstruct the original images. There are however some finer details missing in the e4e inversions that the more advanced models can capture. Looking at the samples in figure 9, one can see that the e4e model struggles with very fine fabric details as shown in the mesh fabric in the first and last row, and with more complex patterns as in the third and fourth row. The other methods seem to be able to capture those details, while it becomes clear that PTI seems most successful at reconstructing the dress with all details, especially the complex patterns. The results of the qualitative evaluation from figure 9 are also in line with the quantitative results in table 1. Here, e4e shows the highest LPIPS and L2 scores while having the lowest MSSSIM results. Interestingly, the reconstructions show a higher FID score than the random generations obtained from the StyleGAN2-Ada model. This is also the case for ReStyle, although all other metrics show Restyle as better performing than e4e. Hyperstyle performs strictly better than both e4e and Restyle based on these metrics. This is also obvious when looking a the reconstructions in figure 9. For PTI inversions, FID could not be calculated, as the model was run only for 500 dresses, which is not enough for FID calculations. The reason for the small sample size is the high inference time of 1.3 minutes per instance. The other metrics however can easily be calculated for PTI, since they are based on direct comparison of originals and reconstructions, regardless of the sample size. PTI shows superior performance in all metrics against Hyperstyle. The LPIPS score of PTI is almost half the magnitude as for Hyperstyle, indicating a considerable increase in reconstruction quality. This is also in line with the qualitative results from figure 9.

Figure 9: Qualitative Evaluation of Inversion Methods: *First column shows the original image, second to last columns show the inversions obtained by various GAN-inversion methods*

Metric	StyleGAN2-Ada	e4e	Restyle	Hyperstyle	PTI
FID	8.0966	8.13	8.4276	7.1874	-
LPIPS	-	0.114	0.1138	0.0988	0.0502
L2	-	0.0203	0.0147	0.0128	0.0085
MSSSIM	-	0.8702	0.8919	0.9053	0.945

Table 1: Quantitative Evaluation of Inversion Methods

5.3 Typicality Measure

Looking at samples with the highest and lowest typicality scores for both typicality measures in figure A.6 and A.7, it becomes clear that both measures are consistent. A typicality measure is consistent,

if the samples that show a high (low) typicality are highly similar in their attributes while they are highly dissimilar to the sample with low (high) typicality scores. For the typicality measure based on DINOv2 embeddings, the highly typical samples have variation in color and sleeve lengths but the neckline and the overall shape of the dress are very similar. In the non-typical dresses, there are mostly very untypical dress shapes. In the typicality measure based on disentangled embeddings, the variation between the most typical dresses seems higher at first glance which is expected, since this is a simple combination of multiple disentangled attribute embeddings. One attribute that seems dominant however is the sleeve length, since the most typical dresses all show long sleeves while the least typical dresses are all sleeveless. If one excludes the attribute sleeve length, the most typical dresses show a waist-fitted shape as can be seen in figure A.9 with varying sleeve lengths. Overall, the typicality measure based on the disentangled embeddings does not allow simple explanations like the one for the DINOv2-based measure, since the exact attributes, that determine that a dress is very typical can vary from dress to dress.

5.4 Manipulation

5.4.1 Preliminary Experiments

To test the overall suitability of inversion methods with the InterFaceGAN manipulation scheme, preliminary experiments are run on e4e-, Hyperstyle-, and PTI-inversions. In those experiments, easy-to-manipulate physical attributes like sleeve length or color are manipulated and the overall manipulation performance is visually assessed. Only inversion methods that perform sufficiently well on physical attributes will be considered for the experiments on aesthetic attribute manipulation. While these manipulations are reliably done for e4e- and PTI-inversions (see figure A.5), Hyperstyle inversions have a slightly worse manipulation performance. Although manipulation of the physical attributes works for most samples, some samples exhibit faulty artifacts as can be seen in figure A.11. The reason for this behavior lies in the structure of the Hyperstyle inversions. For Hyperstyle, the latent space of the inversion is only one component of the inversion model. The other component is the weights of the hyper-network which allow Hyperstyle to achieve this superior performance against Restyle and e4e. InterFaceGAN on the other hand operates exclusively on the latent space and incorporating the weights of the hyper-network in this technique is not possible. Since the reliance on the hyper-network weights for my custom Hyperstyle model is too high, the information contained in the latent codes alone is not sufficient to achieve similar manipulation performance as for e4e latents or PTI inversions. Interestingly, PTI also relies on fine-tuned generator weights to produce accurate inversions but achieves much better manipulation performance in the preliminary experiments. The latent codes that build the basis for PTI are the simple e4e latent codes. Therefore, they contain all the

semantic information needed to find manipulation directions but can simultaneously be improved by pivoted generator weights from PTI. Therefore, PTI inversions work better with InterFaceGAN than Hyperstyle inversions. Due to Hyperstyle's slightly less reliable manipulation performance in these preliminary tests, and the fact that it was outperformed by PTI in all inversion metrics, Hyperstyle was not considered for further experiments downstream. Consequently, all manipulation experiments below were carried out for e4e inversions using the basic StyleGAN2-Ada generator and PTI inversions using the pivoted generator architectures.

5.4.2 Selection of InterFaceGAN Boundary

As described in section 4.4.2, the dataset to train InterFaceGAN for typicality manipulation was created using the top and bottom n samples according to the typicality score. To make manipulation results comparable between all typicality measurements, n is fixed for all embedding types that are used for typicality score calculations. Besides the simple DINOv2 embeddings and the complete disentangled embeddings, InterFaceGAN boundaries were calculated for typicality scores based on disentangled embeddings excluding color and sleeve length attributes. These are used for conditional typicality manipulations.

	Simple Embeddings	**Disentangled Embeddings**		
n	DINOv2	All	Excl. color	Excl. sleeve length
1000	0.9817	1	0.9967	0.9733
2000	0.9825	0.9933	0.9967	0.9642
3000	0.9522	0.9967	0.9978	0.9394

Table 2: Validation Accuracy of InterfaceGAN Boundaries

As can be seen in table 2, the validation accuracy for the complete disentangled embeddings and the disentangled embeddings excluding sleeve length is highest for $n = 1000$. For the simple DINOv2 embeddings, validation accuracy is highest for $n = 2000$ but only slightly lower for $n = 1000$. For the disentangled embeddings excluding color, $n = 3000$ is optimal, but also only slightly higher than for $n = 1000$. Therefore, $n = 1000$ is selected for all boundary calculations in the InterFaceGAN methodology. In general, all boundaries achieve very high separation accuracy. Although this is not very surprising, given the training is performed only on the top and bottom n samples, this still indicates that the learned boundary can sensibly separate the two classes.

5.4.3 Typicality Manipulation

For the final typicality manipulation, I compare the two typicality measures against each other, i.e. typicality based on simple DINOv2 embeddings against typicality based on disentangled embeddings.

Figure 10: Typicality Manipulations using DINOv2 Embeddings

For both embedding types, only e4e and PTI inversions are considered, since those performed well enough in the preliminary experiments. The results of the typicality manipulation based on DINOv2 embeddings and disentangled embeddings for one article (SKU: KA321C13K-K11) are shown in figure 10 and 11 respectively. These figures depict manipulations using incremental InterFaceGAN steps, with each step representing a five-unit shift along the unit-normal vector of the separation boundary - positive for increasing typicality and negative for decreasing it. More examples of manipulations using DINOv2 embeddings can be found in figures A.12 and A.13 for decreased and increased typicality respectively. Analogously, figure A.14 and A.15 show further results for disentangled embeddings.

Manipulation smoothness comparisons of typicality scores reveal that typicality scores based on disentangled embeddings change less smoothly, as seen in the abrupt jumps in scores between

Figure 11: Typicality Manipulations using Disentangled Embeddings

manipulation steps. This contrasts with the much smoother transitions observed in DINOv2-based manipulations, and this pattern holds for both e4e and PTI inversions. Additionally, the distribution of typicality scores for all reconstructions in the dataset, shown in figure A.10, indicates that scores from disentangled embeddings are more dispersed and less smoothly distributed than those based on DINOv2. This likely accounts for the less smooth typicality score manipulations observed with disentangled embeddings.

The **physical attribute changes** that the manipulation does to the dresses, significantly differs between the two typicality measures. For DINOv2-based typicality scores, the changes in the dresses are mostly subtle, primarily involving fine-grained adjustments in shape or color. This observation is in line with the most and least typical dresses based on this embedding type, as shown in figure A.6 where the

most typical dresses are fitted around the waist. Consequently, the manipulation technique tends to make a dress more fitted around the waist when increasing typicality and less fitted when decreasing it. In contrast, typicality manipulation based on disentangled embeddings leads to less subtle changes, significantly altering physical attributes such as sleeve lengths. As shown in figure 11 and especially in figure A.15, to make a dress more typical, long sleeves are added to sleeveless dresses, while sleeves are shortened or transformed into spaghetti straps to make a dress less typical (figure A.14). This observation is consistent with the most and least typical dresses shown in figure A.7, where the most typical dresses have all long sleeves. Those results again hold for both e4e and PTI inversions equally.

Conditional manipulation using physical attributes like sleeve length can offer valuable insights

Figure 12: Typicality Manipulations using Disentangled Embeddings excl. Sleeve Length

into the manipulation behavior. By excluding the sleeve length sub-embedding from the disentangled

embeddings, the typicality calculation ignores this attribute. During manipulation, this conditioning ensures that the sleeve length cannot change when altering typicality. The results of this conditional typicality manipulation are shown in 12 and A.17. Since here the typicality manipulation does not entail changes in the sleeve lengths, conditioning works successfully. Additionally, without access to the sleeve length attribute, the manipulations become subtler, focusing instead on adjusting the fit around the waist and making slight color changes. These modifications are consistent with the most and least typical images based on disentangled embeddings excluding sleeve length, as seen in figure A.9. When applying the same conditioning approach using color as the attribute, the model again relies on sleeve length, as in the unconditional case (see figure A.18). These experiments reveal that the manipulation model based on disentangled embeddings always opts for the quickest path to typicality, which typically involves altering sleeve lengths. If this attribute is restricted, the model shifts to the next most effective path, adjusting shape and color. This demonstrates that typicality in disentangled embeddings is multi-dimensional, with each dimension contributing to the manipulation of a dress's typicality. Without conditioning, the model naturally selects the most dominant attribute for manipulation. To determine whether this behavior is unique to disentangled typicality manipulation, conditional manipulation was also tested with DINOv2-based typicality scores. Since the unconditional manipulations in this case primarily relied on shape and color, conditioning on color was examined. Conditioning on shape was not feasible due to a mismatch between the dataset's shape attribute and the actual changes seen in manipulation. Results of the color-conditioned typicality manipulation of DINOv2-based embeddings can be found in figure A.16. Here it becomes clear, that the conditioning approach in InterFaceGAN does not effectively constrain the color attribute, as the model continues to alter the color of dresses. Furthermore, no significant difference is observed compared to the unconditional manipulation. This indicates that DINOv2-based typicality is driven by a single dimension of typicality, in contrast to the richer, multi-dimensional nature of disentangled embeddings. It is also important to note that the two conditioning methods are not directly methodologically comparable, with one being a post-processing approach, while the other is a complete re-calculation of the typicality measures conditioned on an attribute. For the simple DINOv2 embeddings, conditioning entails ex-post transformation through a projection of the typicality boundary with the respective boundary of the physical attribute, as described in section 4.4.1. For the typicality measure based on disentangled embeddings, conditional manipulation works by calculating a separate typicality measure and InterFaceGAN boundary based on a concatenation of all sub-embeddings excluding the respective sub-embedding for the attribute at hand.

Differences between e4e and PTI inversions with respect to the manipulation performance can be observed in all result figures. Specifically, generations using PTI inversions tend to deteriorate in quality as the manipulation distance increases. Although PTI reconstructions are generally better than

e4e inversions, the last few manipulation steps, i.e. 15-20 unit-normal vector steps away from the reconstruction, often show a decline in image quality. This is likely because the fine-tuned generator in PTI is optimized for the area around the initial reconstruction. When manipulations are performed in the latent space, the resulting latent codes might fall outside the region for which the fine-tuned generator was optimized. This phenomenon is not equally present in all examples. Easy-to-invert dresses require fewer changes in the generator weights, thus being closer to the original generator that can operate well on the whole latent space. For those dresses, quality deterioration is not an issue. However, for dresses with complex patterns or shapes, image quality tends to decline as the distance from the initial reconstruction increases. This observation holds true for both typicality measures.

6 Discussion

The goal of this work was the development of a fine-grained typicality manipulation method and the exploration of various techniques to solve the various sub-tasks needed for this. While this goal and the desired main contribution have been achieved, a more detailed look at the implications of the results is necessary before concluding this work. Using two typicality measurements and two inversion techniques in the final experiments, the advantages and disadvantages of each combination of methods can be deducted when looking at the results presented in section 5. Comparing the manipulation performance between DINOv2-based typicality scores and those based on disentangled embeddings reveals a clear trade-off between disentanglement and interpolability. Disentangled embeddings offer a rich, multi-dimensional understanding of typicality, while DINOv2-based embeddings are simpler, allowing only one-dimensional typicality manipulations. Attempts to disentangle manipulations from DINOv2-based typicality scores through conditioning were unsuccessful, which can be seen as a limitation of this work, although the overall typicality manipulations still produced satisfactory results. The advantage of the richer disentangled embeddings lies in the possibility of conditional, attribute-specific typicality manipulation, as shown with the sleeve length attribute conditioning. On the other hand, typicality manipulation based on disentangled embeddings does not always allow smooth and thus controlled interpolation. When traversing the latent space, sudden jumps in typicality scores and drastic changes in physical attributes can occur for some samples. To address this problem, a smoother distribution of typicality scores, on which latent space editing directions are based, is needed. This could be achieved by significantly expanding the dataset size. However, this was not possible during this work due to the limited number of articles available on "zalando.de". To artificially increase the dataset size, one could add data augmentations by manipulating images using the techniques described in this work and learning typicality scores based on the augmented dataset. This approach could be explored in future research to further improve typicality manipulation. For practitioners,

this trade-off between disentanglement and interpolability is difficult to solve and the selection of the suitable approach often depends on the specific use-case. In high fashion, where a designer's creation is valued highly, often subtle changes in the design are desired. Here, using a simple typicality measure might be sufficient, since the changes applied can be much more subtle and the typicality scores change smoothly. However, for fast prototyping and automated design optimization, disentangled embeddings, that allow more control over the different attributes and that result in more drastic changes in the design are suitable.

Comparing the manipulation performance of the two tested inversion methods, a trade-off between reconstruction quality and editing performance becomes clear. While PTI is capable of reconstructing especially complicated samples much more realistically, its editing performance can degrade with the manipulation strength. The basic e4e inversion on the other hand lacks reconstruction quality but is capable of producing latent codes that are easily editable even into far regions of the latent space. Furthermore, PTI is significantly slower in producing inversions. Again, the selection of the correct approach depends on the specific use case and requirements a designer has for the design augmentation tool. For applications that require fast inference and strong manipulations, i.e. large steps in the latent space, e4e is the logical choice. However, if samples are difficult to invert or if the manipulation needs to be subtle, i.e. taking small steps in the latent space, using PTI is preferred.

Although the generated images from the GAN model and the inversion techniques are generally of high quality, artifacts can still occur in some generations. This is especially the case when the manipulation reaches ill-defined regions of the latent space. However, artifacts can also arise in any generation produced by the custom StyleGAN2-Ada model or any inversion technique. These quality issues highlight a weakness of using GANs compared to diffusion models, which typically produce images of higher quality. The main reason for using GAN-based techniques in this work was the well-behaved latent space of GANs. However, with the growing popularity of diffusion models for image generation, new methods for image manipulation using these models are being published more frequently. During the duration of this thesis, several promising approaches for attribute manipulation with diffusion models were published. New techniques claim to have found methods to create a disentangled latent space for diffusion models similar to the StyleGAN latent space (e.g. Dravid et al. (2024)). Exploring these techniques for typicality manipulation could be a promising direction for future research.

7 Conclusion

This thesis investigated the manipulation of design typicality in fashion with generative deep neural networks, with a particular focus on the category of women's dresses. By leveraging the capabilities of the StyleGAN2-Ada model, the study aimed to adjust the typicality of fashion designs, a key aesthetic

attribute linked to consumer preferences, while maintaining the overall integrity and coherence of the generated images. In this work, various techniques have been explored to achieve the sub-tasks of image generation, GAN inversion, typicality measurement, and latent space editing. The findings demonstrate that typicality can be effectively manipulated within the latent space of the StyleGAN2-Ada model, utilizing inversion techniques such as e4e and PTI. A supervised latent space manipulation method, InterFaceGAN, was successfully applied to achieve typicality editing. These results suggest that this approach could be a valuable tool for designers, enabling them to refine product designs in line with consumer expectations. However, the study also highlighted several limitations and trade-offs. Notably, a trade-off between disentanglement and interpolability was observed in the two typicality measurements used. Additionally, a trade-off between inversion quality and editing performance was identified when comparing e4e and PTI inversions. In summary, this work contributes to the understanding of how generative models can be used to manipulate aesthetic attributes in fashion design. While further research is needed, the findings offer a foundation for future exploration in this area and potential applications in the design process.

References

Abdal, R., Qin, Y., and Wonka, P. (2019). Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4432–4441.

Abdal, R., Zhu, P., Mitra, N. J., and Wonka, P. (2021). Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (ToG)*, 40(3):1–21.

Alaluf, Y., Patashnik, O., and Cohen-Or, D. (2021a). Only a matter of style: Age transformation using a style-based regression model. *ACM Transactions on Graphics (TOG)*, 40(4):1–12.

Alaluf, Y., Patashnik, O., and Cohen-Or, D. (2021b). Restyle: A residual-based stylegan encoder via iterative refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6711–6720.

Alaluf, Y., Tov, O., Mokady, R., Gal, R., and Bermano, A. (2022). Hyperstyle: Stylegan inversion with hypernetworks for real image editing. In *Proceedings of the IEEE/CVF conference on computer Vision and pattern recognition*, pages 18511–18521.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

Barsalou, L. W. (1985). Ideals, central tendency, and frequency of instantiation as determinants of graded structure in categories. *Journal of experimental psychology: learning, memory, and cognition*, 11(4):629.

Bau, D., Strobelt, H., Peebles, W., Wulff, J., Zhou, B., Zhu, J.-Y., and Torralba, A. (2020). Semantic photo manipulation with a generative image prior. *arXiv preprint arXiv:2005.07727*.

Bermano, A. H., Gal, R., Alaluf, Y., Mokady, R., Nitzan, Y., Tov, O., Patashnik, O., and Cohen-Or, D. (2022). State-of-the-art in the architecture, methods and applications of stylegan. In *Computer Graphics Forum*, volume 41:2, pages 591–611. Wiley Online Library.

Bińkowski, M., Sutherland, D. J., Arbel, M., and Gretton, A. (2018). Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*.

Borji, A. (2022). Pros and cons of gan evaluation measures: New developments. *Computer Vision and Image Understanding*, 215:103329.

Chen, L., Tian, J., Li, G., Wu, C.-H., King, E.-K., Chen, K.-T., Hsieh, S.-H., and Xu, C. (2020). Tailorgan: making user-defined fashion designs. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3241–3250.

Chen, Z., Jiang, R., Duke, B., Zhao, H., and Aarabi, P. (2022). Exploring gradient-based multi-directional controls in gans. In *European Conference on Computer Vision*, pages 104–119. Springer.

Choi, W., Jang, S., Kim, H. Y., Lee, Y., Lee, S.-g., Lee, H., and Park, S. (2023). Developing an ai-based automated fashion design system: reflecting the work process of fashion designers. *Fashion and Textiles*, 10(1):39.

Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., and Choo, J. (2018). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797.

Choi, Y., Uh, Y., Yoo, J., and Ha, J.-W. (2020). Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8188–8197.

Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., and Zemmel, R. (2023). The economic potential of generative ai: The next productivity frontier. https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier. [Online; accessed 20-August-2024].

Creswell, A. and Bharath, A. A. (2018). Inverting the generator of a generative adversarial network. *IEEE transactions on neural networks and learning systems*, 30(7):1967–1974.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Deng, J., Guo, J., Xue, N., and Zafeiriou, S. (2019). Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699.

Dravid, A., Gandelsman, Y., Wang, K.-C., Abdal, R., Wetzstein, G., Efros, A. A., and Aberman, K. (2024). Interpreting the weight space of customized diffusion models. *arXiv preprint arXiv:2406.09413*.

Freno, A. (2017). Practical lessons from developing a large-scale recommender system at zalando. In *Proceedings of the eleventh ACM conference on recommender systems*, pages 251–259.

Fu, J., Li, S., Jiang, Y., Lin, K.-Y., Qian, C., Loy, C. C., Wu, W., and Liu, Z. (2022). Stylegan-human: A data-centric odyssey of human generation. In *European Conference on Computer Vision*, pages 1–19. Springer.

Goetschalckx, L., Andonian, A., Oliva, A., and Isola, P. (2019). Ganalyze: Toward visual definitions of cognitive image properties. In *Proceedings of the ieee/cvf international conference on computer vision*, pages 5744–5753.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.

Gu, J., Shen, Y., and Zhou, B. (2020). Image processing using multi-code gan prior. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3012–3021.

Härkönen, E., Hertzmann, A., Lehtinen, J., and Paris, S. (2020). Ganspace: Discovering interpretable gan controls. *Advances in neural information processing systems*, 33:9841–9850.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hermosilla, G., Tapia, D.-I. H., Allende-Cid, H., Castro, G. F., and Vera, E. (2021). Thermal face generation using stylegan. *IEEE Access*, 9:80511–80523.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Kang, K., Kim, S., and Cho, S. (2021). Gan inversion for out-of-range images with geometric transformations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13941–13949.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*.

Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T. (2020a). Training generative adversarial networks with limited data. *Advances in neural information processing systems*, 33:12104–12114.

Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2020b). Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119.

Kim, H., Choi, Y., Kim, J., Yoo, S., and Uh, Y. (2021). Exploiting spatial dimensions of latent in gan for real-time image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 852–861.

Kim, J.-Y., Tangriberganov, G., Jung, W., Kim, D. S., Koo, H. S., Lee, S., and Kim, S. M. (2022). Effective representation learning via the integrated self-supervised pre-training models of stylegan2-ada and dino for colonoscopy images. *BioRxiv*, pages 2022–06.

Kurach, K., Lucic, M., Zhai, X., Michalski, M., and Gelly, S. (2018). The gan landscape: Losses, architectures, regularization, and normalization. -.

Kwon, Y., Petrangeli, S., Kim, D., Wang, H., Swaminathan, V., and Fuchs, H. (2022). Tailor me: An editing network for fashion attribute shape manipulation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3831–3840.

Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., and Aila, T. (2019). Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32.

Landwehr, J. R., Labroo, A. A., and Herrmann, A. (2011). Gut liking for the ordinary: Incorporating design fluency improves automobile sales forecasts. *Marketing Science*, 30(3):416–429.

Li, B., Cai, S., Liu, W., Zhang, P., He, Q., Hua, M., and Yi, Z. (2023). Dystyle: Dynamic neural network for multi-attribute-conditioned style editings. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 189–197.

Lipton, Z. C. and Tripathi, S. (2017). Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*.

Little, C., Elliot, M., Allmendinger, R., and Samani, S. S. (2021). Generative adversarial networks for synthetic data generation: a comparative study. *arXiv preprint arXiv:2112.01925*.

Liu, Y., Li, K. J., Chen, H., and Balachander, S. (2017). The effects of products' aesthetic design on demand and marketing-mix effectiveness: The role of segment prototypicality and brand consistency. *Journal of Marketing*, 81(1):83–102.

Luo, J., Xu, Y., Tang, C., and Lv, J. (2017). Learning inverse mapping by autoencoder based generative adversarial nets. In *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24*, pages 207–216. Springer.

Meyers-Levy, J. and Tybout, A. M. (1989). Schema congruity as a basis for product evaluation. *Journal of consumer research*, 16(1):39–54.

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Müller, S., Koch, L. M., Lensch, H., and Berens, P. (2024). Disentangling representations of retinal images with generative models. *arXiv preprint arXiv:2402.19186*.

Nitzan, Y., Bermano, A., Li, Y., and Cohen-Or, D. (2020). Face identity disentanglement via latent space mapping. *arXiv preprint arXiv:2005.07728*.

Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al. (2023). Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*.

Pidhorskyi, S., Adjeroh, D. A., and Doretto, G. (2020). Adversarial latent autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14104–14113.

Ping, Q., Wu, B., Ding, W., and Yuan, J. (2019). Fashion-attgan: Attribute-aware fashion editing with multi-objective gan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Regenwetter, L., Nobari, A. H., and Ahmed, F. (2022). Deep generative models in engineering design: A review. *Journal of Mechanical Design*, 144(7):071704.

Ren, X., Yang, T., Wang, Y., and Zeng, W. (2021). Learning disentangled representation by exploiting pretrained generative models: A contrastive learning view. *arXiv preprint arXiv:2102.10543*.

Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., and Cohen-Or, D. (2021). Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2287–2296.

Roich, D., Mokady, R., Bermano, A. H., and Cohen-Or, D. (2022). Pivotal tuning for latent-based editing of real images. *ACM Transactions on graphics (TOG)*, 42(1):1–13.

Sajjadi, M. S., Bachem, O., Lucic, M., Bousquet, O., and Gelly, S. (2018). Assessing generative models via precision and recall. *Advances in neural information processing systems*, 31.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. *Advances in neural information processing systems*, 29.

Saxena, D. and Cao, J. (2021). Generative adversarial networks (gans) challenges, solutions, and future directions. *ACM Computing Surveys (CSUR)*, 54(3):1–42.

Schnurr, B. (2017). The impact of atypical product design on consumer product and brand perception. *Journal of Brand Management*, 24(6):609–621.

Shanyan, G., Ying, T., Bingbing, N., Feida, Z., Feiyue, H., and Xiaokang, Y. (2020). Collaborative learning for faster stylegan embedding. *arXiv preprint arXiv:2007.01758*.

Sharma, S., Koehl, L., Bruniaux, P., Zeng, X., and Wang, Z. (2021). Development of an intelligent data-driven system to recommend personalized fashion design solutions. *Sensors*, 21(12):4239.

Sheikh, H. R., Sabir, M. F., and Bovik, A. C. (2006). A statistical evaluation of recent full reference image quality assessment algorithms. *IEEE Transactions on image processing*, 15(11):3440–3451.

Shen, Y., Yang, C., Tang, X., and Zhou, B. (2020). Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE transactions on pattern analysis and machine intelligence*, 44(4):2004–2018.

Shen, Y. and Zhou, B. (2021). Closed-form factorization of latent semantics in gans. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1532–1540.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sujan, M. (1985). Consumer knowledge: Effects on evaluation strategies mediating consumer judgments. *Journal of consumer research*, 12(1):31–46.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Tov, O., Alaluf, Y., Nitzan, Y., Patashnik, O., and Cohen-Or, D. (2021). Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14.

Veryzer Jr, R. W. and Hutchinson, J. W. (1998). The influence of unity and prototypicality on aesthetic responses to new product designs. *Journal of consumer research*, 24(4):374–394.

Wang, C. and Golland, P. (2023). Interpolating between images with diffusion models. -.

Wang, L., Chen, W., Yang, W., Bi, F., and Yu, F. R. (2020). A state-of-the-art review on image synthesis with generative adversarial networks. *Ieee Access*, 8:63514–63537.

Wang, T., Zhang, Y., Fan, Y., Wang, J., and Chen, Q. (2022). High-fidelity gan inversion for image attribute editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11379–11388.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612.

Wang, Z., Simoncelli, E. P., and Bovik, A. C. (2003). Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee.

Winkielman, P., Halberstadt, J., Fazendeiro, T., and Catty, S. (2006). Prototypes are attractive because they are easy on the mind. *Psychological science*, 17(9):799–806.

Wu, Z., Lischinski, D., and Shechtman, E. (2021). Stylespace analysis: Disentangled controls for stylegan image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12863–12872.

Xia, W., Zhang, Y., Yang, Y., Xue, J.-H., Zhou, B., and Yang, M.-H. (2022). Gan inversion: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):3121–3138.

Xiao, Z., Kreis, K., and Vahdat, A. (2021). Tackling the generative learning trilemma with denoising diffusion gans. *arXiv preprint arXiv:2112.07804*.

Yang, H., Chai, L., Wen, Q., Zhao, S., Sun, Z., and He, S. (2021). Discovering interpretable latent space directions of gans beyond binary attributes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12177–12185.

Yeh, R. A., Chen, C., Yian Lim, T., Schwing, A. G., Hasegawa-Johnson, M., and Do, M. N. (2017). Semantic image inpainting with deep generative models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5485–5493.

Yüksel, O. K., Simsar, E., Er, E. G., and Yanardag, P. (2021). Latentclr: A contrastive learning approach for unsupervised discovery of interpretable directions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14263–14272.

Zajonc, R. B. (1968). Attitudinal effects of mere exposure. *Journal of personality and social psychology*, 9(2p2):1.

Zhang, K., Zhou, Y., Xu, X., Dai, B., and Pan, X. (2024). Diffmorpher: Unleashing the capability of diffusion models for image morphing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7912–7921.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595.

Zhu, J., Shen, Y., Zhao, D., and Zhou, B. (2020a). In-domain gan inversion for real image editing. In *European conference on computer vision*, pages 592–608. Springer.

Zhu, J.-Y., Krähenbühl, P., Shechtman, E., and Efros, A. A. (2016). Generative visual manipulation on the natural image manifold. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, pages 597–613. Springer.

Zhu, P., Abdal, R., Qin, Y., Femiani, J., and Wonka, P. (2020b). Improved stylegan embedding: Where are the good latents? *arXiv preprint arXiv:2012.09036*.

Appendix

GAN Inversion

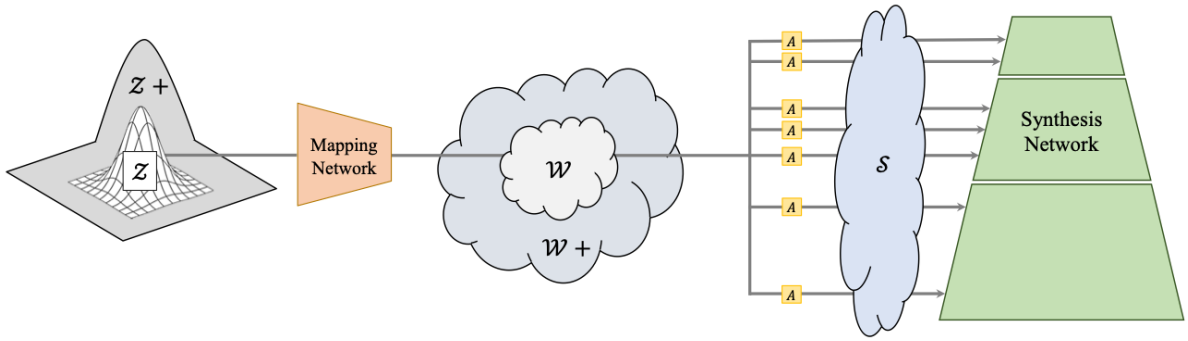


Figure A.1: StyleGAN Latent Spaces - taken from Bermano et al. (2022)

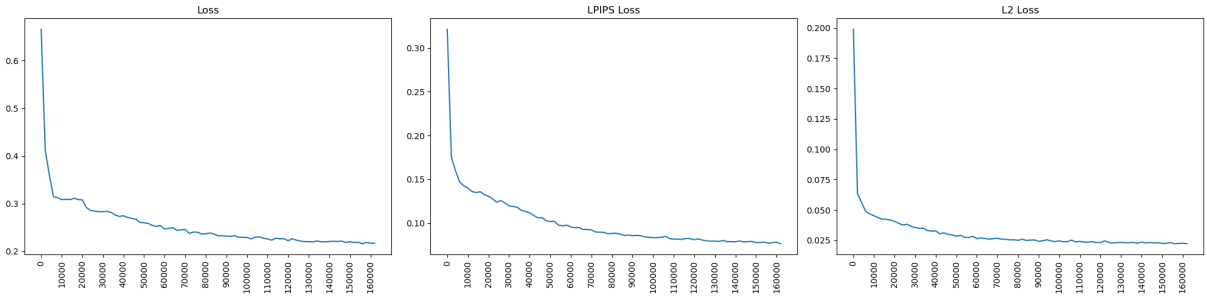
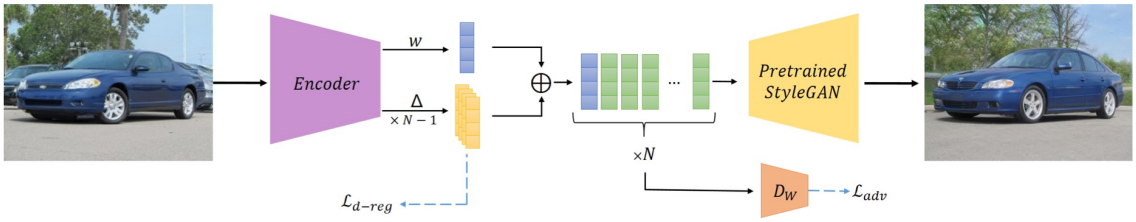
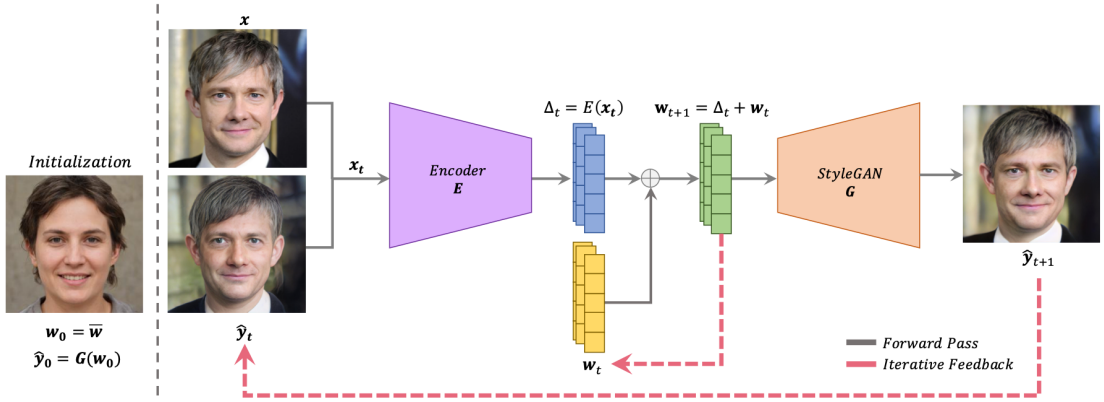


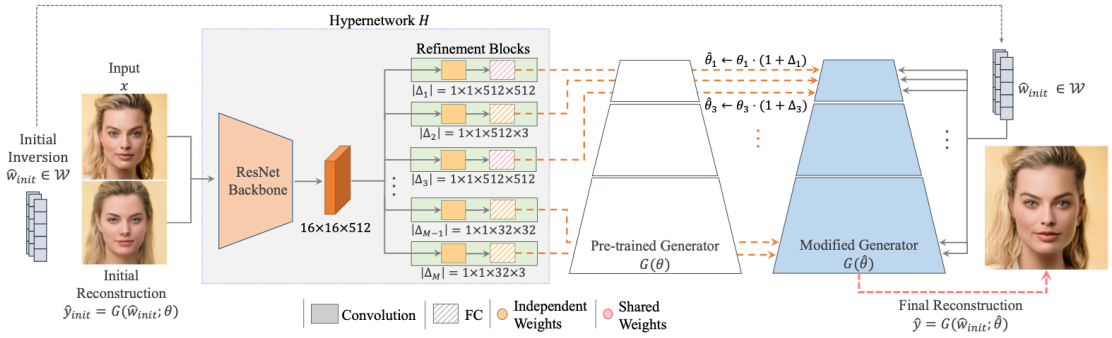
Figure A.2: encoder4editing Training Loss Curves



(a) encoder4editing - taken from Tov et al. (2021)



(b) ReStyle - taken from Alaluf et al. (2021b)



(c) HyperStyle - taken from Alaluf et al. (2022)

Figure A.3: Schematic Overview of Inversion Methods

StyleGAN2-Ada Training

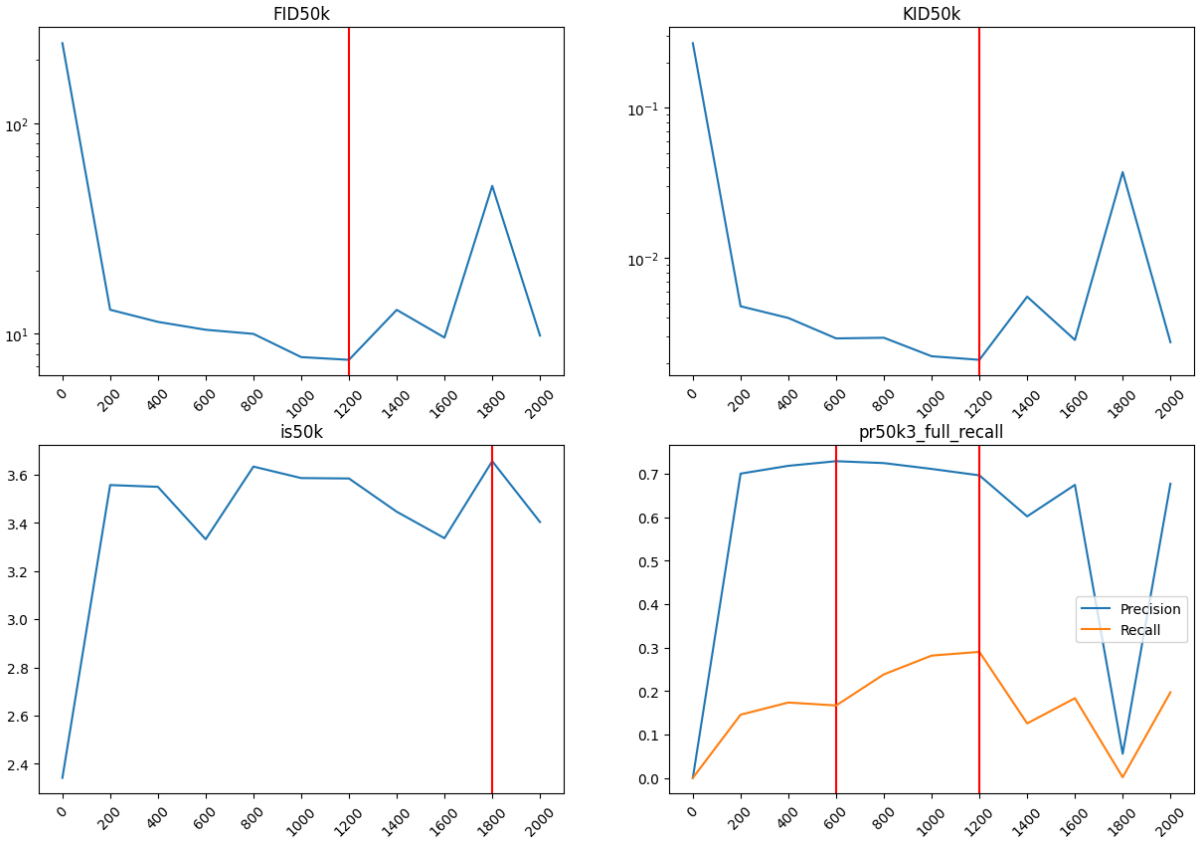


Figure A.4: StyleGAN2-Ada Training Evaluation: *Red lines indicate best values within each metric. FID50k, KID50k, and IS50k refer to the FID, KID, and IS calculated based on 50k generated images.*

InterFaceGAN

Figure A.5: Manipulations of physical attributes used as conditions: *Physical attributes are used as conditional boundaries for the typicality manipulation. To visually validate that the boundaries are meaningful, one can look at manipulations using the boundaries. All images are generated from e4e inversions.*

	Color	Sleeve Length
count	91	54
mean	0.9595	0.9673
std	0.0324	0.0506
min	0.8736	0.7647
25%	0.9359	0.965
50%	0.9694	0.9885
75%	0.9859	0.9962
max	1	1

Table 3: Summary Statistics of Separation Performance for Physical Attributes

Typicality Measurements

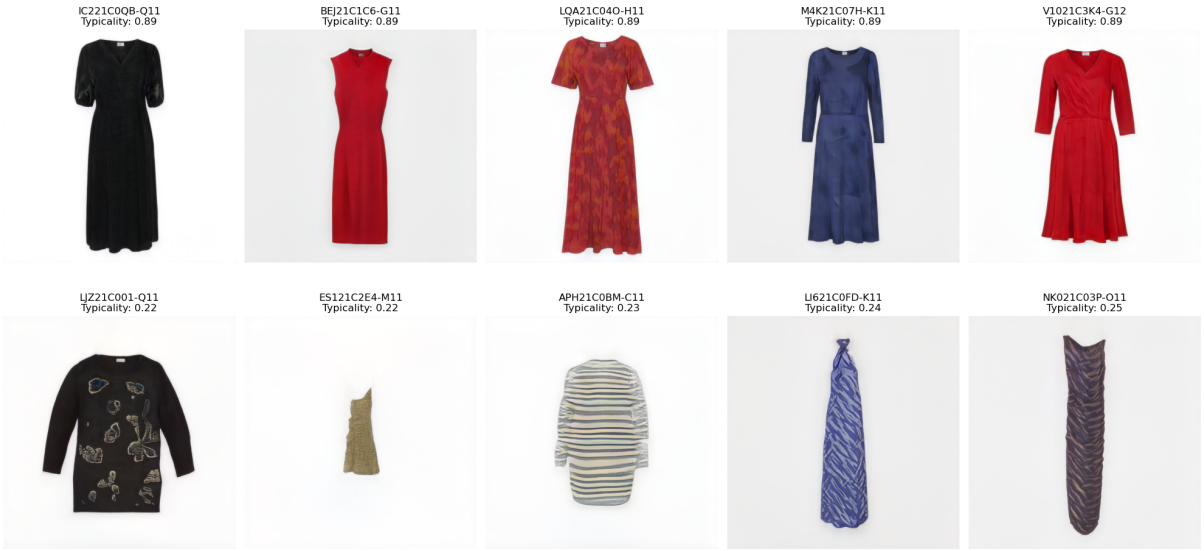


Figure A.6: Most and Least Typical Dresses based on DINOv2 Embeddings: *Top row shows samples with the highest typicality scores, bottom row shows samples with the lowest typicality scores*

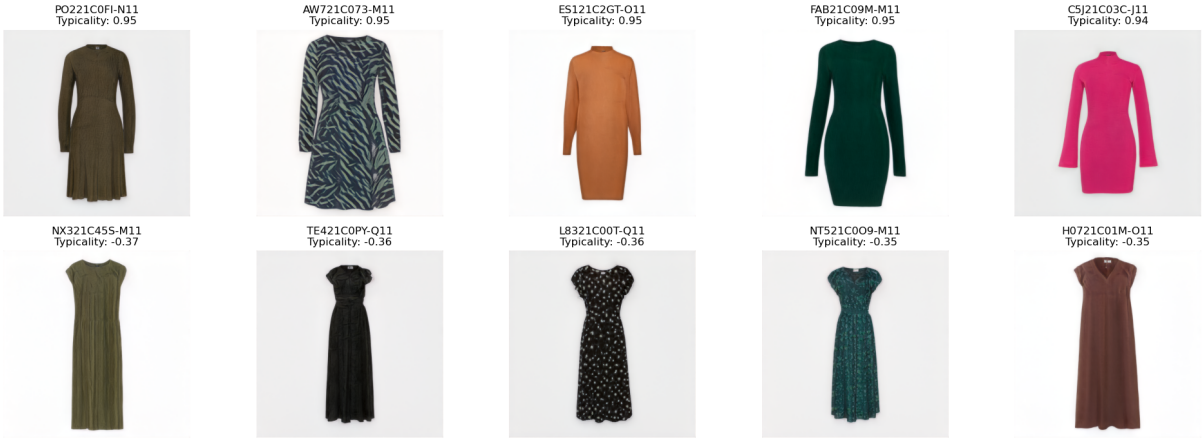
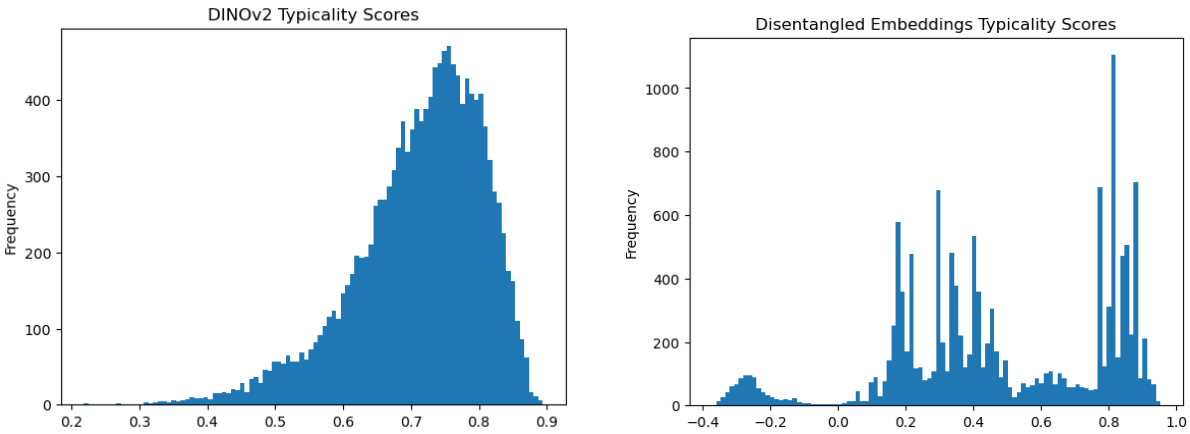


Figure A.7: Most and Least Typical Dresses based on Complete Disentangled Embeddings: *Top row shows samples with the highest typicality scores, bottom row shows samples with the lowest typicality scores*

Figure A.8: Most and Least Typical Dresses based on Disentangled Embeddings excluding Color: *Top row shows samples with the highest typicality scores, bottom row shows samples with the lowest typicality scores*

Figure A.9: Most and Least Typical Dresses based on Disentangled Embeddings excluding Sleeve Length: *Top row shows samples with the highest typicality scores, bottom row shows samples with the lowest typicality scores*



(a) DINOv2 Embeddings

(b) Disentangled Embeddings

Figure A.10: Distribution of typicality scores

Hyperstyle Manipulation Results

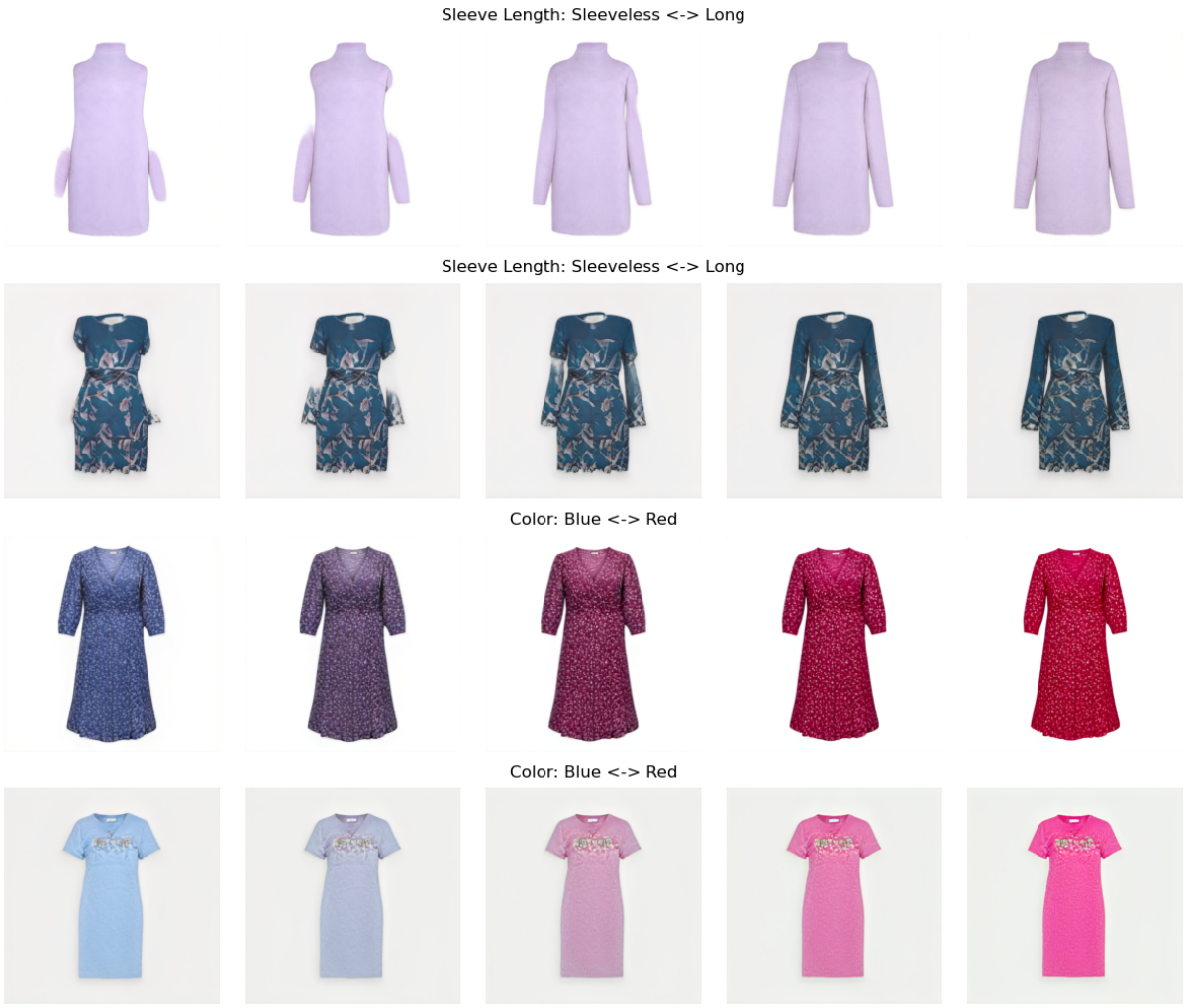


Figure A.11: Preliminary Manipulation Tests using Hyperstyle Inversions: *To assess the overall suitability of Hyperstyle Inversions for the InterFaceGAN scheme, manipulation of visual attributes is tested. Although some manipulations seem to work well, others, especially the sleeve length manipulations, produce faulty artifacts.*

Distance Correlation Formulas

Below, detailed formulas for the calculation of the distance correlation are given. All formulas and explanations are directly taken or slightly adapted from Müller et al. (2024), p.5.

$$\mathrm{dCor}(w_1,w_2)=\sqrt{\frac{\mathrm{dCov}^2(w_1,w_2)}{(\mathrm{dVar}^2(w_1)\mathrm{dVar}^2(w_2))^{1/2}}} \tag{4}$$

Assuming n-dimensional batches of subspace vectors $w_1\in\mathbb{R}^{n\times d_1}$ and $w_2\in\mathbb{R}^{n\times d_2}$, a $n\times n$ dimensional Euclidean space containing all pairwise distances can be calculated as

$$A\in\mathbb{R}^{n\times n},\quad a_{j,k}=\|w_1^{(j)}-w_1^{(k)}\|_2,\quad j,k=1,2,...,n \tag{5}$$

$$B\in\mathbb{R}^{n\times n},\quad b_{j,k}=\|w_2^{(j)}-w_2^{(k)}\|_2,\quad j,k=1,2,...,n \tag{6}$$

Normalization of the distance matrices was performed by subtracting the row mean, column mean, and grand mean. Formally, this can be expressed as

$$A_{j,k}:=a_{j,k}-\bar{a}_{j,\cdot}-\bar{a}_{\cdot,k}+\bar{a}_{\cdot,\cdot} \tag{7}$$

$$B_{j,k}:=b_{j,k}-\bar{b}_{j,\cdot}-\bar{b}_{\cdot,k}+\bar{b}_{\cdot,\cdot} \tag{8}$$

Finally, dCov^2 can be calculated as the arithmetic mean of the element-wise matrix product.

$$\mathrm{dCov}^2(w_1,w_2)=\frac{1}{n^2}\sum_{j=1}^{n}\sum_{k=1}^{n}A_{j,k}B_{j,k} \tag{9}$$

dVar^2 can be calculated as the arithmetic mean of the squared matrix elements of the normalized distance matrices.

$$\mathrm{dVar}^2(w_1)=\frac{1}{n^2}\sum_{j,k}A_{j,k}^2 \tag{10}$$

$$\mathrm{dVar}^2(w_2)=\frac{1}{n^2}\sum_{j,k}B_{j,k}^2 \tag{11}$$

Figure A.12: Decreasing Typicality for DINOv2 Embeddings: *First row shows the original image, second row is the inversion. All following rows are each +5 InterFaceGAN steps towards less typicality*

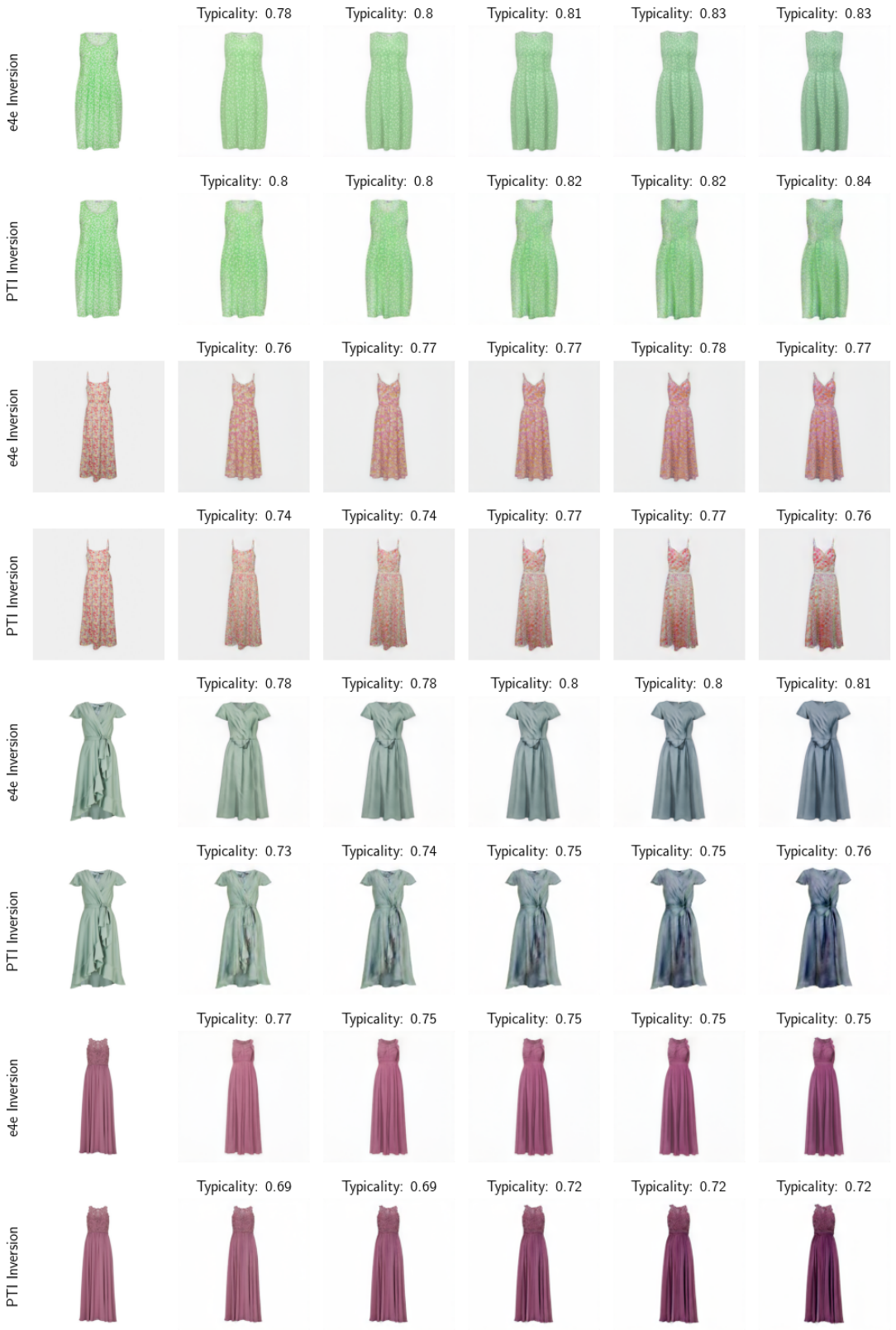


Figure A.13: Increasing Typicality for DINOv2 Embeddings: *First row shows the original image, second row is the inversion. All following rows are each +5 InterFaceGAN steps towards more typicality*

Figure A.14: Decreasing Typicality for Disentangled Embeddings: *First row shows the original image, second row is the inversion. All following rows are each +5 InterFaceGAN steps towards less typicality*

Figure A.15: Increasing Typicality for Disentangled Embeddings: *First row shows the original image, second row is the inversion. All following rows are each +5 InterFaceGAN steps towards more typicality*

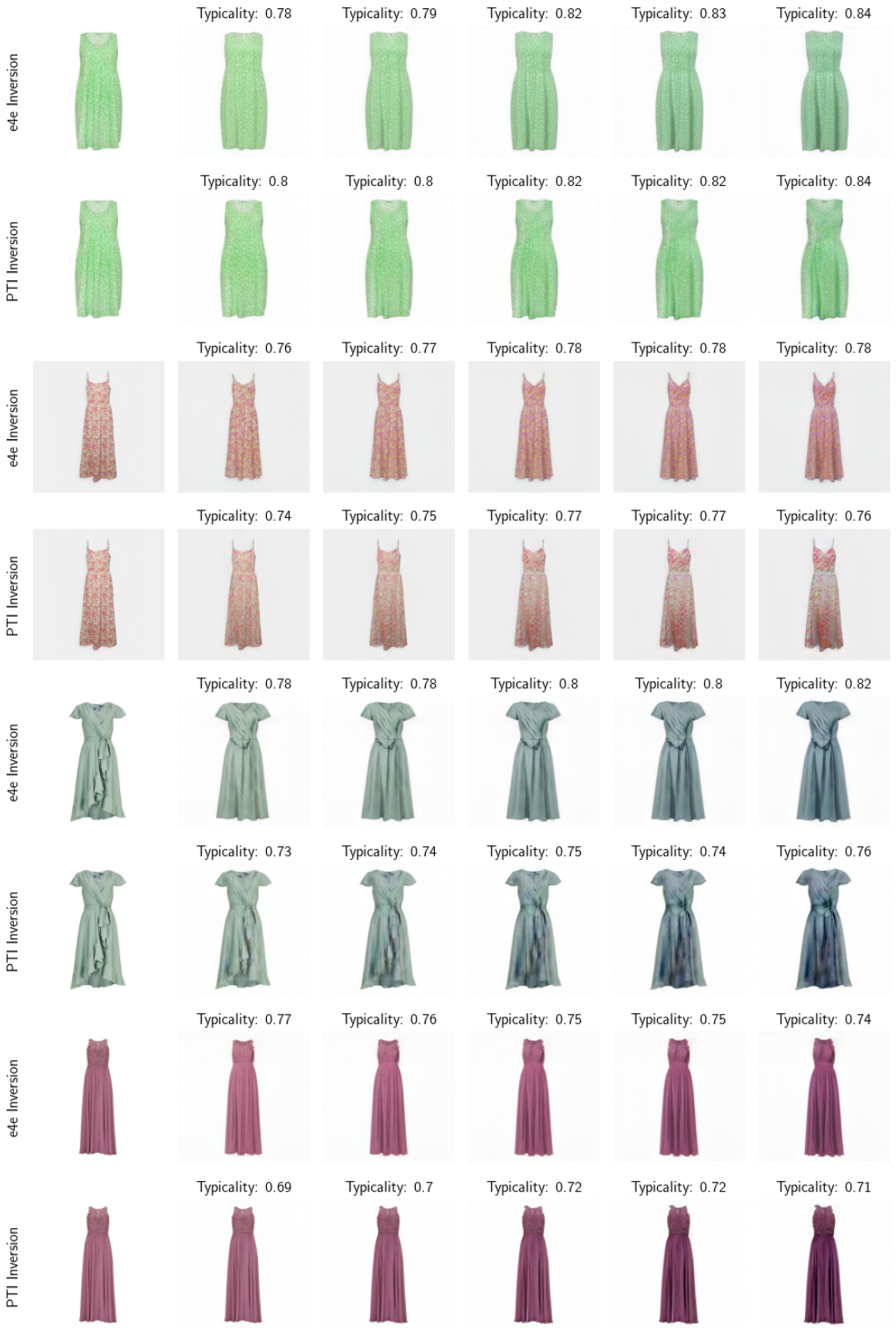


Figure A.16: Increasing Typicality for DINOv2 Embeddings conditioned on Color: *First row shows the original image, second row is the inversion. All following rows are each +5 InterFaceGAN steps towards more typicality*

Figure A.17: Increasing Typicality for Disentangled Embeddings excl. Sleeve Length: *First row shows the original image, second row is the inversion. All following rows are each +5 InterFaceGAN steps towards more typicality*

Figure A.18: Increasing Typicality for Disentangled Embeddings excl. Color: *First row shows the original image, second row is the inversion. All following rows are each +5 InterFaceGAN steps towards more typicality*

Eidesstattliche Erklärung

Ich versichere an Eides Statt, dass ich die Arbeit selbständig verfasst, keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe, alle wörtlich oder sinngemäß aus anderen Werken übernommenen Aussagen als solche gekennzeichnet habe und dass die Arbeit weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens gewesen ist und dass ich die Arbeit weder vollständig noch in wesentlichen Teilen bereits veröffentlicht habe, sowie dass das in Dateiform eingereichte Exemplar mit den eingereichten gebundenen Exemplaren übereinstimmt.

Datum Unterschrift