Latent Space Factorisation and Manipulation via Matrix Subspace Projection

Xiao Li [1 2] Chenghua Lin [2] Ruizhe Li [2] Chaozheng Wang [1] Frank Guerin [3]

Abstract

We tackle the problem disentangling the latent space of an autoencoder in order to separate labelled attribute information from other characteristic information. This then allows us to change selected attributes while preserving other information. Our method, matrix subspace projection, is much simpler than previous approaches to latent space factorisation, for example not requiring multiple discriminators or a careful weighting among their loss functions. Furthermore our new model can be applied to autoencoders as a plugin, and works across diverse domains such as images or text. We demonstrate the utility of our method for attribute manipulation in autoencoders trained across varied domains, using both human evaluation and automated methods. The quality of generation of our new model (e.g. reconstruction, conditional generation) is highly competitive to a number of strong baselines.

1. Introduction

We investigate the problem of manipulating multiple attributes of data samples. This can be applied to image data, for example to manipulate a picture of a face to add a beard, change gender, or age. It can also be applied to text, for example to change the style or sentiment of a text. We assume that we have a training dataset where attributes are labelled. However there is an unsupervised aspect because we do not have samples of the same individual with different attribute combinations, e.g., the same person with and without a beard. Furthermore the training samples have some attribute combinations that are highly correlated, while other combinations are completely absent; e.g., in the CelebA dataset blond hair and earrings are highly correlated with female (Torfason et al., 2016), while a female with beard is absent. Nevertheless we would like our system to somehow isolate the explanatory factors in pixel space, to understand, e.g., that blond hair corresponds only to colour changes to hair pixels, and no change elsewhere in the face.

This challenge of isolating multiple explanatory factors poses interesting problems for generative models. In images of faces for example, even if the training data has no bearded lady, a good generative model should be able to 'imagine' novel examples that combine attributes in ways not present in training data. As noted by Higgins et al. (2016) "Models are unable to generalise to data outside of the convex hull of the training distribution ... unless they learn about the data generative factors and recombine them in novel ways." Ideally we should fully disentangle and isolate the data generative factors, so that we can represent the generative factors of a sample with a vector that has one part labelled attribute information, and another part with the other characteristic information of the sample. This is in a small way part of a general trend to try to move deep neural network research towards explanatory models of the world (LeCun, 2013; Lake et al., 2016; Yuille & Liu, 2018), which requires disentanglement. The problem is important because isolating explanatory factors is a way to overcome the combinatorial explosion of required training examples if such factors are not isolated (Yuille & Liu, 2018).

A typical approach to the problem uses an autoencoder (AE) which encodes a given input (e.g. picture, text, etc.) into a *latent vector*, and then restores (decodes) the latent vector to the given input (Lample et al., 2017; Hu et al., 2017; Xiao et al., 2018; Li et al., 2019). The latent vector contains the attribute information as well as other characteristic information of the input. If one can change the attribute information in the latent space, then one can generate examples with the altered attributes. The difficulty here is twofold: (1) learn a latent space representation which separates the attributes from all other characteristic information, and (2) fully disentangle the attributes. If we fail in the separation part, then efforts to generate with specific attributes may conflict with other information in the latent space (as in Kingma et al. (2014) etc., see §2). If we fail in the second part then examples generated with specified attributes will also be contaminated with spurious attributes (see Fig. 1 Left).

Many recent approaches make use of auxiliary neural net-

[1]Department of Computing Science, University of Aberdeen, UK [2]Department of Computer Science, University of Sheffield, UK [3]Department of Computer Science, University of Surrey, UK. Correspondence to: Chenghua Lin <c.lin@sheffield.ac.uk>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

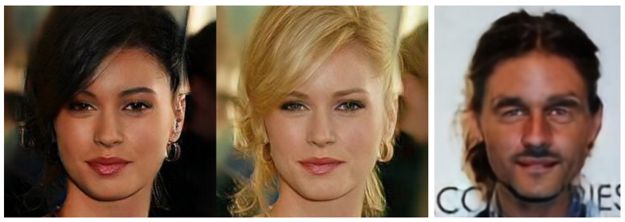


Figure 1. Left: from RelGAN (Wu et al., 2019), where the only attribute changed is hair colour, but we see significant changes in skin colour, eyebrows, eyes, and lips. Right: from Fader (Lample et al., 2017), where female was changed to male, but female eyebrows are retained above the male ones, due to skip connections.

work structures with adversarial training in the style of Generative Adversarial Networks (GANs). These new networks can be used to remove attribute information from the latent space (Lample et al., 2017), or to feedback a loss term to impose the attributes they want to appear in the output (He et al., 2019). These adversarial approaches have competing loss terms (for example reconstruction loss, attribute classification loss, realistic output loss), and require a careful choice of hyperparameters to weight these loss functions. In the case of Lample et al. (2017) a slowly increasing loss was critical. These hyperparameters and training schedules must be determined by trial and error, to avoid training instability. Even after successful training we have found that some models ignore the desired attributes and put too much weight on reconstruction and realistic output (see §4). This is partly because we push systems to the very difficult setting of training for multiple attributes together (e.g. 40 attributes for CelebA). This is a very demanding setting for disentanglement, e.g. to dissociate lipstick, make-up, and blond hair from female, and to dissociate beard, bushy eyebrows, and 5 o'clock shadow from male.

We propose a simple and generic method, Matrix Subspace Projection (MSP), which directly separates the attribute information from all other non-attribute information, without relying on weighting loss terms from auxiliary neural networks. Our variables representing attributes are fully disentangled, with one isolated variable for each attribute of the training set. Therefore, when we do conditional generation, we can assign pure attributes combined with other latent data which does not conflict, so that the generated pictures are of high quality and not contaminated with spurious attributes. Meanwhile, our model is a universal plugin. In theory, it can be applied to any existing AEs (if and only if the AEs use a latent vector). If the AE is a generative model (such as VAE), with our approach, it becomes a conditional generative model that generates content based on the given condition constraints. In the case of images, we add a PatchGAN at the end of our generator to sharpen the image, but this is not connected with the attribute manipulation task and is not core to our model; it could be replaced with any super resolution and sharpening method.

Our plugin has two uses: (1) samples can be generated from a random seed, but with given attributes; (2) a given sample can be modified to have desired specified attributes. Our key contributions are: (1) A simple and universal plugin for conditional generation and content replacement, directly applicable to any AE architectures (e.g., image or text). (2) Strong performance on learning disentangled latent representations of multiple (e.g. 40) attributes. (3) A principled weighting strategy for combining loss terms for training. The code for our model is available online[1].

2. Related Work

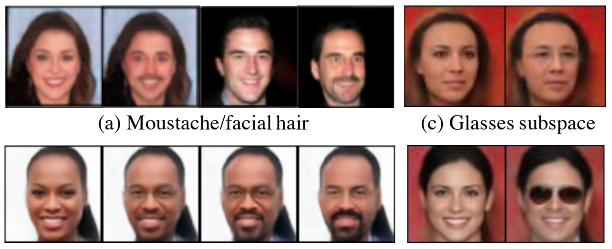
The first approaches to control of generation by attributes (conditional VAEs (Kingma et al., 2014; Sohn et al., 2015; Yan et al., 2016)) simply added attribute information as an extra input to the encoder or the decoder. These approaches generate using a latent vector $\mathbf{z}$ and also an attribute vector $\mathbf{y}$, where the $\mathbf{z}$ often conflicts with $\mathbf{y}$, because attribute information has not been removed from $\mathbf{z}$. With conflicting inputs the best the VAE can do is to produce a blurry image.

Generative Adversarial Networks (GANs) can be augmented with encoders. IcGAN trains separate encoders for the $\mathbf{y}$ and $\mathbf{z}$ vectors, but does not try to remove potentially conflicting information (Perarnau et al., 2016). The IcGAN authors also note that it can fail to generate unusual attribute combinations such as a woman with a moustache, because the GAN discriminator discourages the generator from generating samples outside the training distribution.

More recent work tackled the problem of separating the attribute information from the latent vector, using a new auxiliary network (like a GAN discriminator) (Lample et al., 2017; Creswell et al., 2017; Klys et al., 2018), which attempts to guess the attribute of the latent vector $\mathbf{z}$, and penalise the generator if attribute information remains. A significant drawback of these adversarial approaches is that great care must be taken in training so that the loss from the discriminator (which is trying to remove attribute information) does not disturb the training to produce a good reconstruction. In the case of Fader networks (Lample et al., 2017) it was necessary to start with a discriminator loss weight of zero, and linearly increase to 0.0001 over the first 500,000 iterations; the authors state "*This scheduling turned out to be critical in our experiments. Without it, we observed that the encoder was too affected by the loss coming from the discriminator, even for low values of [loss coefficient].*"

While this adversarial approach can successfully remove attribute information from $\mathbf{z}$, there is nothing to stop the

[1]Code: `https://xiao.ac/proj/msp`



(a) Moustache/facial hair (c) Glasses subspace

(b) Glasses and facial hair subspace (d) Glasses subspace

Figure 2. Figure showing the difficulty of disentangling attributes for supervised 'adversarial' approaches. (a) from Creswell et al. (2017) shows significant change in the eyebrows and eyes when adding facial hair. (b,c,d) from Klys et al. (2018). (b) moving across the glasses and facial hair subspace, from the female on the left, brings significant changes in eyebrows and eyes, and the shape of cheeks, making the face more masculine. (c) moving in glasses subspace shows changes around the eyes and mouth, looking older. (d) also moving in glasses subspace shows a narrower smile and a more masculine lower face. Note all of the pictures are generated by VAE, none are original photographs.

decoder (generator) from associating other spurious information with the attribute. For example the decoder might associate the attribute intended to be for 'glasses' with an older or more masculine face. This is what we see in the results of two of the adversarial approaches (see Fig. 2). Most of the results in Creswell et al. (2017) focus on the attribute 'smiling' (not reproduced here), and this is very well disentangled. It is only when the training dataset associates other attributes with the trained attribute that entanglement will arise. In Creswell et al. (2017) the attribute vector is a single binary variable so that the system can only be trained to control (or classify) one attribute. It is not unexpected that a generator will associate spurious information with an attribute if the association is present in the training data and the system has been trained only on examples labelling a single attribute, e.g., glasses. The system cannot know that it should isolate 'wearing glasses', and not 'wearing glasses and older'. Fader Networks (Lample et al., 2017) can train for multiple attributes together, however He et al. (2019) state that "Although Fader Networks is capable for multiple attribute editing with one model, in practice, multiple attribute setting makes the results blurry."

The most recent works (2018-19) are GAN-based. They do not try to remove attribute information from the latent space, but instead add an additional attribute classifier after generation, and impose an attribute classification loss. This is in addition to a typical GAN discriminator for realistic images. AttGAN (He et al., 2019) uses an endoder, decoder (generator), and the attribute classifier and discriminator applied to the output of the generator. StarGAN (Choi et al., 2018) and RelGAN (Wu et al., 2019) use no encoder, but use a singe generator twice, in a cycle; the first direction alters attributes like a conditional GAN, the second one attempts to reconstruct the image (using original attributes), and so requires that non-attribute information has been preserved. StarGAN uses a discriminator and attribute classifier, like AttGAN, while RelGAN adds a third network for interpolation.

All the works cited from 2017 to 2019 have an adversarial component (in the style of a GAN); they train auxiliary classifiers to feed back loss terms, to ensure they remove undesirable attributes, or enforce desired ones. They need a careful weighting among loss terms, but there is no principled method for determining these weighting hyperparameters. Our work does not rely on an adversarial component to manipulate attributes; we use a more direct method of matrix projection onto subspaces, in order to factorise the latent representation and separate attributes from other information. Furthermore, unlike the above works[2] we do not use any skip connections. Skip connections can introduce errors when a region of the source and target image is quite different, we illustrate this further in Fig. 1 Right.

In addition to the above works using labelled attributes there is also work on the more difficult problem of unsupervised learning of disentangled generative factors of data (Chen et al., 2016; Higgins et al., 2017; Kumar et al., 2018). However the supervised (labelled) approaches generate much clearer samples of selected attributes, and superior disentanglement. An alternative approach to controlled generation is to simply train a deep convolutional network and do linear interpolation in deep feature space (Upchurch et al., 2017). This shows surprisingly good results, but in changing an attribute that should only affect a local area it can affect more image regions, and can produce unrealistic results for more rare face poses.

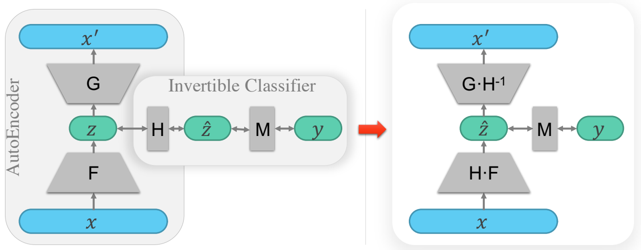
3. Method

3.1. Problem Formulation

We are interested in factorising and manipulating multiple attributes from a latent representation learned by an *arbitrary* Autoencoder (AE). Suppose we are given a dataset $\mathcal{D}$ of elements $(\mathbf{x}, \mathbf{y})$ with $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in Y = \{0, 1\}^k$ representing k attributes of $\mathbf{x}$.

Let an arbitrary AE be represented by $\mathbf{z} = F(\mathbf{x})$ and $\mathbf{x}' = G(\mathbf{z})$, where $F(\cdot)$ is the encoder, $G(\cdot)$ is the decoder, $\mathbf{z}$ is the latent vector encoding $\mathbf{x}$, and $\mathbf{x}'$ is the reconstruction of $\mathbf{x}$ (see Fig. 3). Note that when $\mathbf{x}'$ is a good approximation of $\mathbf{x}$ (i.e., $\mathbf{x}' \approx \mathbf{x}$), the attribute information of $\mathbf{x}$ represented in $\mathbf{y}$ will also be captured in the latent encoding $\mathbf{z}$. Attribute manipulation means that we replace the attributes $\mathbf{y}$ captured by $\mathbf{z}$ with new attributes $\mathbf{y_n}$. Let

[2]Not mentioned in the Fader networks paper, but in the published code: https://github.com/facebookresearch/FaderNetworks



(a) General Architecture (b) Simplified Architecture

Figure 3. (a) The general architecture of our model (MSP); (b) the simplified architecture of MSP.

$K(\cdot)$ be a replacement function, then we have new latent space $\mathbf{z_n} = K(\mathbf{z}, \mathbf{y_n})$ and $\mathbf{x_n} = G(K(\mathbf{z}, \mathbf{y_n}))$, where the attribute information encoded in $\mathbf{y_n}$ can be predicted from $\mathbf{x_n}$ and the non-attribute information of $\mathbf{x}$ will be *preserved*. To give a concrete example, given an image of a face, $\mathbf{x}$, we wish to manipulate $\mathbf{x}$ w.r.t. the presence or absence of a set of desired attributes encoded in $\mathbf{y_n}$ (e.g., a face with or without smiles, wearing or not wearing glasses), producing the manipulated image $\mathbf{x_n}$, without changing the identify of the face (i.e., preserving the non-attribute information of $\mathbf{x}$).

3.2. Learning Disentangled Latent Representations via Matrix Subspace Projection

To tackle the problem formulated in §3.1, we propose a generic method to factor out the information about attributes $\mathbf{y}$ from $\mathbf{z}$ based on the idea of performing orthogonal matrix projection onto subspaces. Our model works as a universal plugin and in theory, it can be applied to any existing AEs. The general architecture of the proposed MSP model is depicted in Fig. 3 (a). Given a latent vector $\mathbf{z}$ encoding $\mathbf{x}$ and an arbitrarily complex invertible function $H(\cdot)$, $H(\cdot)$ transforms $\mathbf{z}$ to a new linear space ($\hat{\mathbf{z}} = H(\mathbf{z})$) such that one can find a matrix $\mathbf{M}$ where (a) the projection of $\hat{\mathbf{z}}$ on $\mathbf{M}$ (denoted by $\hat{\mathbf{y}}$) approaches $\mathbf{y}$ (i.e., $\hat{\mathbf{y}}$ captures attribute information),

$$\mathbf{M} \cdot \hat{\mathbf{z}} = \hat{\mathbf{y}}; \quad \hat{\mathbf{y}} \to \mathbf{y} \tag{1}$$

and (b) there is an *orthogonal matrix* $\mathbf{U} \equiv [\mathbf{M}; \mathbf{N}]$, where $\mathbf{N}$ is the *null space* of $\mathbf{M}$ (i.e., $\mathbf{M} \perp \mathbf{N}$) and the projection of $\hat{\mathbf{z}}$ on $\mathbf{N}$ (denoted by $\hat{\mathbf{s}}$) captures non-attribute information. As $\mathbf{U}$ is orthogonal, we also have $\mathbf{U}^T \equiv \mathbf{U}^{-1}$.

Fig. 3 (b) presents a simplified architecture of our MSP model, which is equivalent to the general architecture. This simplification exists because as explained earlier $H(\cdot)$ is invertible. So when the encoder and decoder of an AE have enough capacity, they can essentially absorb H and H^{-1}. In other words, rather than fitting F and G, the encoder and decoder will fit $H \cdot F(\cdot)$ and $G \cdot H^{-1}(\cdot)$ instead. As $\mathbf{M}$ itself is an incomplete orthogonal matrix, similar operations cannot be applied to $\mathbf{M}$.

Our main learning objective, in addition to the original AE objective (i.e. reconstruction loss $\mathcal{L}_{AE}$; see §3.3 and Eq. 8), is then to estimate $\mathbf{M}$ which is nontrivial. We turn the problem of finding an optimal solution for $\mathbf{M}$ into an optimisation problem, in which we need to (1) enforce $\hat{\mathbf{y}}$ to be as close to $\mathbf{y}$ (i.e., the vector encoding the ground truth attributes) as possible; and (2) minimise $||\hat{\mathbf{s}}||^2$ so that $\hat{\mathbf{s}}$ contains as little information from $\hat{\mathbf{z}}$ as possible. This can be formulated into the following loss function

$$\mathcal{L}_{MSP} = \mathcal{L}_1 + \mathcal{L}_2 \tag{2}$$

$$\mathcal{L}_1 = ||\hat{\mathbf{y}} - \mathbf{y}||^2 = ||\mathbf{M} \cdot \hat{\mathbf{z}} - \mathbf{y}||^2 \tag{3}$$

$$\mathcal{L}_2 = ||\hat{\mathbf{s}}||^2 \tag{4}$$

Here $\mathcal{L}_1$ and $\mathcal{L}_2$ encode the above two constraints, respectively, and $\hat{\mathbf{y}}$ is the predicted attributes. Given that the AE relies on the information of $\hat{\mathbf{z}}$ to reconstruct $\mathbf{x}$, the optimisation constraints of $\mathcal{L}_{AE}$ and $\mathcal{L}_2$ essentially introduce an adversarial process: on the one hand, it discourages any information of $\hat{\mathbf{z}}$ to be stored in $\hat{\mathbf{s}}$ due to the penalty from $\mathcal{L}_2$; on the other hand, the AE requires information from $\hat{\mathbf{z}}$ to reconstruct $\mathbf{x}$. So, the best solution is to only restore the essential information for reconstruction (except the attribute information) in $\hat{\mathbf{s}}$. By optimising $\mathcal{L}_{MSP}$, we cause $\hat{\mathbf{z}}$ to be factorised, with the attribute information stored in $\hat{\mathbf{y}}$, while $\hat{\mathbf{s}}$ only retains non-attribute information.

The first part of our loss function $\mathcal{L}_1$ is relatively straightforward. The main obstacle here is to compute $\mathcal{L}_2$ as $\hat{\mathbf{s}}$ is unknown. We develop a strategy to compute $||\hat{\mathbf{s}}||^2$ indirectly. According to the definition of $\hat{\mathbf{y}}$ and $\hat{\mathbf{s}}$ we can derive:

$$\begin{aligned}\mathcal{L}_2 &= ||\hat{\mathbf{s}}||^2 = ||\hat{\mathbf{s}} - \mathbf{0}||^2 \\ &= ||[\hat{\mathbf{y}}; \hat{\mathbf{s}}] - [\hat{\mathbf{y}}; \mathbf{0}]||^2 \quad \text{Identical deformation} \\ &= ||\mathbf{U} \cdot \hat{\mathbf{z}} - [\hat{\mathbf{y}}; \mathbf{0}]||^2\end{aligned} \tag{5}$$

where the square brackets represent the vector concatenation. Because $\mathbf{U}$ is orthogonal, we have

$$\begin{aligned}\mathcal{L}_2 &= ||\mathbf{U} \cdot \hat{\mathbf{z}} - [\hat{\mathbf{y}}; \mathbf{0}]||^2 \\ &= ||\mathbf{U}^{-1} \cdot (\mathbf{U} \cdot \hat{\mathbf{z}} - [\hat{\mathbf{y}}; \mathbf{0}])||^2 \\ &= ||\hat{\mathbf{z}} - \mathbf{U}^{-1} \cdot [\hat{\mathbf{y}}; \mathbf{0}]||^2 = ||\hat{\mathbf{z}} - \mathbf{U}^T \cdot [\hat{\mathbf{y}}; \mathbf{0}]||^2 \\ &= ||\hat{\mathbf{z}} - [\mathbf{M}; \mathbf{N}]^T \cdot [\hat{\mathbf{y}}; \mathbf{0}]||^2 \\ &= ||\hat{\mathbf{z}} - \mathbf{M}^T \cdot \hat{\mathbf{y}}||^2 \approx ||\hat{\mathbf{z}} - \mathbf{M}^T \cdot \mathbf{y}||^2\end{aligned} \tag{6}$$

With Eq. 6 (which makes use of the properties of orthogonal matrices), we avoid computing $\hat{\mathbf{s}}$ and $\mathbf{N}$ directly when minimising $||\hat{\mathbf{s}}||^2$, and turn the minimisation problem into optimising $\mathbf{M}$ instead. Finally, we have:

$$\begin{aligned}\mathcal{L}_{MSP} &= \mathcal{L}_1 + \mathcal{L}_2 \\ &= ||\mathbf{M} \cdot \hat{\mathbf{z}} - \mathbf{y}||^2 + ||\hat{\mathbf{z}} - \mathbf{M}^T \cdot \hat{\mathbf{y}}||^2 \\ &\approx ||\mathbf{M} \cdot \hat{\mathbf{z}} - \mathbf{y}||^2 + ||\hat{\mathbf{z}} - \mathbf{M}^T \cdot \mathbf{y}||^2\end{aligned} \tag{7}$$

The loss function in Eq. 7 also guarantees that after training, the solution for $\mathbf{M}$ will be part of the orthogonal matrix $\mathbf{U}$ (see §4.5). When $\mathcal{L}_{MSP}$ is small, the transposition of $\mathbf{M}$ becomes the inverse of $\mathbf{M}$.

3.3. Applying the Matrix Subspace Projection Framework to an AE

To apply our matrix subspace projection (MSP) framework to an existing AE, one only needs to derive a final loss function by combining the loss of the AE and the loss of our MSP framework.

$$\mathcal{L} = \mathcal{L}_{AE} + \alpha \mathcal{L}_{MSP} \tag{8}$$

where α is the weight for $\mathcal{L}_{MSP}$. As illustrated in Fig. 3 (a) and (b), one should note that applying our framework will not change the structure of the AE, where our MSP component simply takes the latent vector $\hat{\mathbf{z}}$ of the AE as input. $\mathcal{L}_{AE}$ hopes that $\hat{\mathbf{s}}$ can store more information to reconstruct $\mathbf{x}$, but $\mathcal{L}_{MSP}$ wants $\hat{\mathbf{s}}$ to contain less information. When α is small, the model becomes a standard AE. When α is too large, the non-attribute information in $\hat{\mathbf{z}}$ is reduced excessively, resulting in the generated products tending to the average of the training samples. Therefore, another challenge we face is how to set α appropriately.

We propose a principled strategy for effectively determining the value of α (this strategy is used in all experiments in this paper). Since $\mathcal{L}_{AE}$ and $\mathcal{L}_{MSP}$ essentially represent a competing relationship for $\hat{\mathbf{z}}$ resources, we specify that $\mathcal{L}_{AE}$ and $\mathcal{L}_{MSP}$ have the same influence on updating $\hat{\mathbf{s}}$. We use α to represent the "intensity" with which the AE updates $\hat{\mathbf{z}}$ during each back-propagation process. This intensity depends on the structure of the model and the loss function used by the model. For example, suppose an AE (for picture generation) uses a CNN decoder and L2-loss. During the training process, the error of each pixel between the generated picture and the true picture is backpropagated to $\hat{\mathbf{z}}$ as the gradients of $\hat{\mathbf{z}}$. The sum of these gradients is the final gradient of $\hat{\mathbf{z}}$ (i.e., corresponding to $\mathcal{L}_{AE}$). For a picture with $h \times w$ pixels and c colour channels, there are $h \times w \times c$ parts of gradients accumulated, so the intensity is $h \times w \times c$. The intensity of $\hat{\mathbf{y}}$ for updating $\hat{\mathbf{z}}$ is the total amount of attributes (i.e. the dimension of $\hat{\mathbf{y}}$), because the error for each attribute is propagated back to $\hat{\mathbf{z}}$ and accumulated (i.e., corresponding to $\mathcal{L}_{MSP}$). Therefore, in order to balance the influence of $\mathcal{L}_{AE}$ and $\mathcal{L}_{MSP}$ on updating $\hat{\mathbf{z}}$ during training, we have:

$$\alpha \approx \frac{h \times w \times c}{\text{size}(attribute) + \text{size}(\hat{\mathbf{z}})} \tag{9}$$

When using the cross-entropy-loss (or NLL loss etc.), which is usually for textual generative models (e.g. the seq2seq model), each generated word produces only one intensity, regardless of the word embedding size. Meanwhile, loss values returned by the cross-entropy-loss are proportional to the error, but the loss values returned by the MSP loss (which is a MSE loss) are proportional to the error's square. Therefore, for a sentence of length k, the intensity of the entire sentence to $\hat{\mathbf{z}}$ is k^2, so that for cross-entropy-loss,

$$\alpha \approx \frac{k^2}{\text{size}(attribute) + \text{size}(\hat{\mathbf{z}})} \tag{10}$$

3.4. Content Replacement and Conditional Generation

After MSP is trained (i.e., $\mathbf{M}$ is estimated), there are two ways to perform content replacement or change of attributes. One way is to derive the orthogonal matrix $\mathbf{U} = [\mathbf{M}; \mathbf{N}]$ by solving the null space $\mathbf{N}$ of $\mathbf{M}$ (i.e., the null space is constituted of all the specific solutions for $\mathbf{n}$ w.r.t. equation $\mathbf{M} \cdot \mathbf{n} = 0$, where $\mathbf{n}$ is an independent variable). Given an input $\mathbf{x}$, we first encode it as $\hat{\mathbf{z}}$. We then use $\mathbf{U}$ to obtain the attribute vector $\hat{\mathbf{y}}$ of $\mathbf{x}$ and the non-attribute information vector $\hat{\mathbf{s}}$ as follows.

$$[\hat{\mathbf{y}}; \hat{\mathbf{s}}] = [\mathbf{M}; \mathbf{N}] \cdot \hat{\mathbf{z}} = \mathbf{U} \cdot \hat{\mathbf{z}} = \mathbf{U} \cdot \text{encoder}(\mathbf{x}) \tag{11}$$

At this point, we can directly replace $[\hat{\mathbf{y}}; \hat{\mathbf{s}}]$ with $[\mathbf{y_n}; \hat{\mathbf{s}}]$, where $\mathbf{y_n}$ is the new attribute vector. With $[\mathbf{y_n}; \hat{\mathbf{s}}]$ and $\mathbf{U}^T$ (note that $\mathbf{U}^T$ approaches $\mathbf{U}^{-1}$ during training), we can derive the new latent code $\mathbf{z_n}$ and then decode it into $\mathbf{x_n}$, which ultimately captures the desired new attributes.

$$\mathbf{x_n} = \text{decoder}(\mathbf{z_n}) = \text{decoder}(\mathbf{U}^T \cdot [\mathbf{y_n}; \hat{\mathbf{s}}]) \tag{12}$$

Alternatively, we can avoid explicitly calculating matrix $\mathbf{N}$ (i.e. avoid calculating $\hat{\mathbf{s}}$), for content replacement. According to Eqs.11 and 12, we define $\mathbf{d}$ as the distance between $\hat{\mathbf{z}}$ and $\mathbf{z_n}$.

$$\begin{aligned}\mathbf{d} = \hat{\mathbf{z}} - \mathbf{z_n} &= \mathbf{U}^T \cdot [\hat{\mathbf{y}}; \hat{\mathbf{s}}] - \mathbf{U}^T \cdot [\mathbf{y_n}; \hat{\mathbf{s}}] \\ &= \mathbf{U}^T \cdot ([\hat{\mathbf{y}}; \hat{\mathbf{s}}] - [\mathbf{y_n}; \hat{\mathbf{s}}]) = \mathbf{U}^T \cdot [\hat{\mathbf{y}} - \mathbf{y_n}; \mathbf{0}] \\ &= [\mathbf{M}; \mathbf{N}]^T \cdot [\hat{\mathbf{y}} - \mathbf{y_n}; \mathbf{0}] \\ &= \mathbf{M}^T \cdot (\hat{\mathbf{y}} - \mathbf{y_n}) = \mathbf{M}^T \cdot (\mathbf{M} \cdot \hat{\mathbf{z}} - \mathbf{y_n}) \end{aligned} \tag{13}$$

It should be noted that here $\hat{\mathbf{z}} \neq \mathbf{M}^T \cdot \mathbf{M} \cdot \hat{\mathbf{z}}$ because the reconstruction loss does not allows $\hat{\mathbf{s}}$ to be zero. Thus, we have:

$$\mathbf{z_n} = \hat{\mathbf{z}} - \mathbf{d} = \hat{\mathbf{z}} - \mathbf{M}^T \cdot (\mathbf{M} \cdot \hat{\mathbf{z}} - \mathbf{y_n}) \tag{14}$$

$$\mathbf{x_n} = \text{decoder}(\hat{\mathbf{z}} - \mathbf{M}^T \cdot (\mathbf{M} \cdot \hat{\mathbf{z}} - \mathbf{y_n})) \tag{15}$$

If the AE itself is a generative model (such as VAE), then the AE+MSP structure becomes a conditional generative model. Given a randomly sampled $\mathbf{s_r}$ and an attribute vector $\mathbf{y_r}$, the model can generate new sample $\mathbf{x_r}$ with the desired attributes with Eq.12.

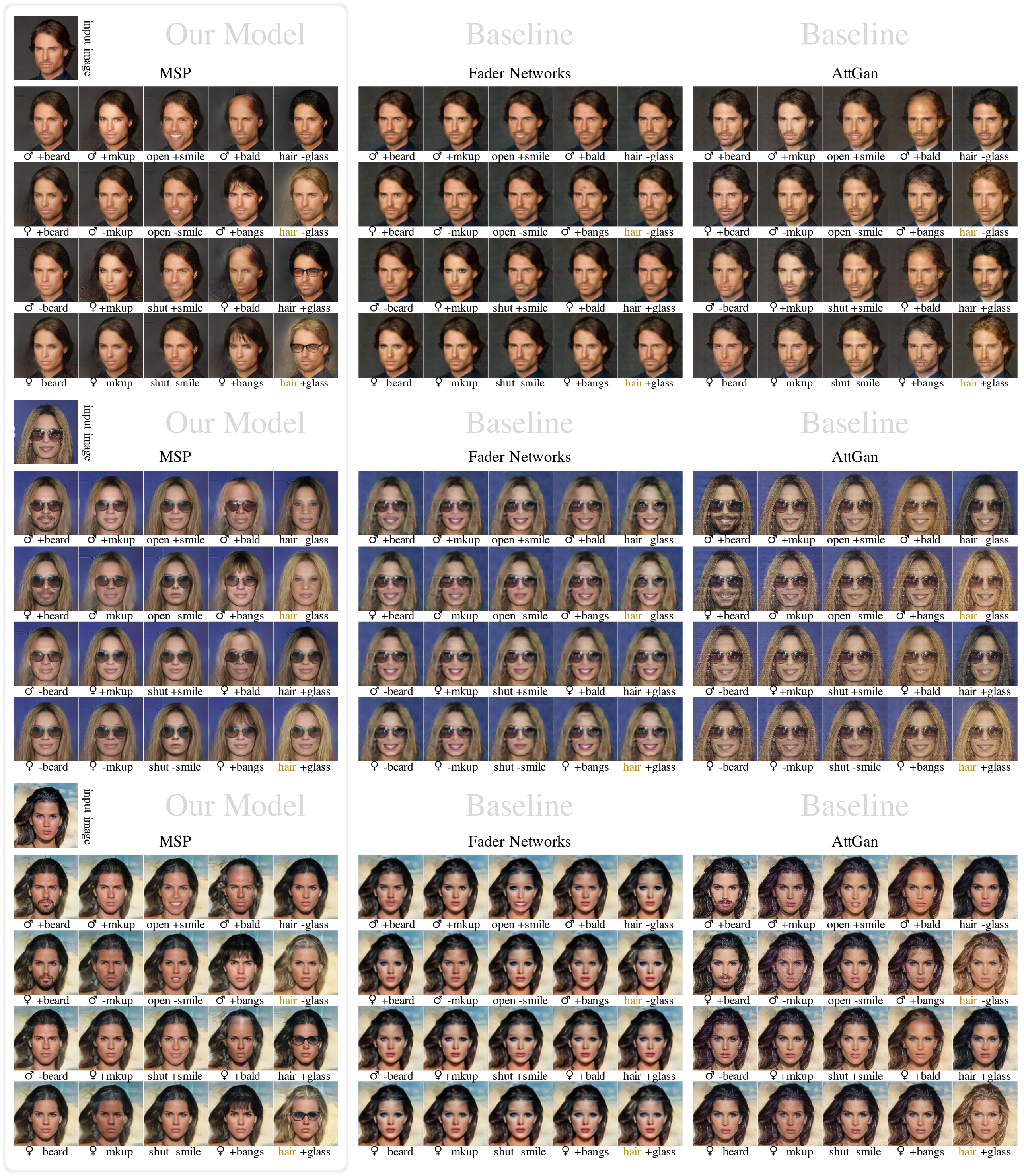


Figure 4. Examples of image attributes transformations using MSP (our model), Fader Networks and AttGAN.

4. Evaluation

Here we evaluate the ability to disentangle. We also evaluate the orthogonality of $\mathbf{M}$ as it is an important indicator of how well our algorithm can approximate $\mathbf{M}$.

4.1. Matrix Subspace Projection in VAE

We apply our model on a *vanilla VAE* (Kingma & Welling, 2013) with the standard CNN encoder and decoder (the architectures are same as Lample et al. (2017)). We used the

	Seq2seq	VAE	VAE+GAN
Better quality with AE only	34.5%	12.6%	17.9%
Better quality with AE+MSP	37.6%	12.0%	15.2%
both similar quality	27.9%	75.4%	66.9%

Table 1. Evaluation results of generation quality. Numbers in the table denote percentage of participants under the column heading who felt images were better with or without MSP.

	MSP(ours)	Fader	AttGAN
male x beard	**0.78**	0.42	0.45
female x beard	**0.52**	0.03	0.41
male x no-beard	**0.86**	0.40	0.42
female x no-beard	**0.90**	0.61	0.63
male x makeup	**0.52**	0.02	0.35
male x no-makeup	**0.89**	0.50	0.47
female x makeup	**0.87**	0.63	0.52
female x no-makeup	**0.67**	0.42	0.47
smile x open-mouth	**0.89**	0.59	0.63
no-smile x open-mouth	**0.66**	0.11	0.29
smile x calm-mouth	**0.95**	0.34	0.33
no-smile x calm-mouth	**0.76**	0.43	0.38
male x bald	**0.78**	0.10	0.29
male x bangs	**0.56**	0.05	0.19
female x bald	**0.29**	0.01	0.17
female x bangs	**0.68**	0.21	0.20
no-glasses x black-hair	**0.74**	0.38	0.53
no-glasses x golden-hair	**0.86**	0.36	0.79
glasses x black-hair	**0.82**	0.21	0.32
glasses x golden-hair	**0.77**	0.19	0.33

Table 2. The classification accuracy (ResNet-CNN classifier) of generated images using MSP, Fader Networks and AttGAN.

ADAM optimiser with learning rate = 0.0002, mini-batch size of 256, and images are upsampled to 256 $\times$ 256. We add an additional PatchGAN (Li & Wand, 2016) to make the produced images sharp. The architecture of the PatchGAN discriminator also adopts the version of Lample et al. (2017). Our baselines are Fader networks (Lample et al., 2017) and AttGAN (He et al., 2019), based on their published code and settings. We did not compare StarGAN (Choi et al., 2018) because we feel it is superseded by AttGAN, which demonstrated better performance. We did not compare RelGAN (Wu et al., 2019) as it does not disentangle attributes (see Fig. 1 (left), also RelGAN cannot add a moustache or beard to a female face, instead it will transform it to a male face with beard).

We evaluated on the CelebA dataset (Liu et al., 2015) (202,600 images) and trained one model on all 40 labelled attributes. The generated examples are shown in Fig. 4. Qualitatively we see clear examples of what Fader networks and AttGan cannot do: For the woman with glasses (middle) FaderNetwork and AttGan show complete inability to remove the glasses; FaderNetwork completely fails to add glasses to the other two faces, and AttGan can only manage weak rims on the final woman (bottom). FaderNetwork in general struggles to change attributes, especially for the two women, while AttGan does better, but struggles with certain attributes, e.g. mostly it fails to change the final woman to male, and struggles to remove makeup.

Target attribute changed	Influence on other attributes	MSP (ours)	Fader	AttGAN
gender	beard	0.01	0.28	0.09
beard	gender	0.07	0.11	0.02
gender	makeup	0.02	0.07	0.05
makeup	gender	0.05	0.09	0.14
smile	mouth-open	0.01	0.20	0.07
mouth-open	smile	0.02	0.07	0.09

Table 3. Quantitative evaluation of disentanglement (using ResNet-CNN classifier).

mouth open / smiling attributes morphing			
	Fader Network	AttGAN	VAE+GAN MSP (ours)
perfect	36.7%	47.5%	68.3%
recognizable	20.8%	15.3%	4.9%
unreco/unchang	42.5%	37.2%	26.8%

male / beard attributes morphing			
	Fader Network	AttGAN	VAE+GAN MSP (ours)
perfect	38.3%	55.9%	74.4%
recognizable	8.3%	11.2%	11.6%
unreco/unchang	53.3%	32.9%	14.0%

Table 4. Manual evaluation results of disentanglement. Numbers in the table denote percentage of participants under the column heading who felt the images represented the specified attribute (e.g. smiling) in a way that was perfect, recognisable, or unrecognisable/unchanged.

For a quantitative evaluation we trained a classifier (ResNet-CNN) to measure the accuracy with which attributes are changed. Table 2 shows that our MSP approach outperforms the competitors. Finally we calculated the average Fréchet Inception Distance (FID) (Heusel et al., 2017) for each method: MSP=35.0, Fader=26.3, AttGAN=7.3 (lower is better, 0 is the best). The FID score tries to calculate the similarity of original images and generated images. Clearly AttGAN is a winner for quality while our MSP is a winner for accuracy of attribute modification. When AttGAN cannot handle the attribute modification it generates unchanged images and can get lower FID scores.

The results of attribute manipulation (both qualitative and quantitative) are surprisingly bad for Fader networks and

AttGAN, especially relative to the examples displayed in their original papers. The primary reason for this is that we trained those models on all 40 attributes together. Fader networks works best when trained on a single attribute, as noted in Sec. 2. The original AttGAN paper trained on 13 attributes, and indeed it performs better at attribute manipulation than Fader in our pictures. For the 40-attribute-together version, when any attribute is changed all others must be unchanged. For example, when we transition from male to female (Fig. 4 left), it is implicit that the female should keep no make-up or lipstick, etc. In the direction from female to male the male should keep no bushy eyebrows or 5 o'clock shadow. The original Fader network paper displays a beautiful example of interpolating between male and female, but the female does have make-up and lipstick and the male does have bushy eyebrows and 5 o'clock shadow. Our difficult 40-attribute setting is central to our aims, as stated in our introduction: we want to fully disentangle multiple attributes, because this gives a generative model the ability to 'imagine' novel examples that combine attributes in ways not present in training data.

4.2. Human Evaluation of Generated Example Quality

We evaluated whether our MSP model reduces the quality of generated examples, using human evaluation via Amazon Mechanical Turk (hiring 150 participants in total).

For each model, we randomly generated 1,000 example pairs. Each pair contains a reconstructed example (from AE) and an example generated by AE+MSP with one or two random attribute modification (attributes were changed to -1 if they were originally > 0, or changed to 1 if they were < 0)[3]. The participants were shown the examples pair-by-pair in the blind test, and they were asked to *please choose the one with better text/image quality, or choose both if you think they perform similarly*. The participants were told that the text quality means the fluency, semantic accuracy, and syntactic accuracy, and the image quality means the clarity and (face) recognisability. The results are shown in Table 1. We treat the scores (i.e. participants' choices) of the result as a Likert scale, and we set our null hypothesis to be H_0: *generation quality of AE+MSP is worse than using the AE only*, and, H_1: *generation quality of AE+MSP is equal or higher than using the AE only*. The hypotheses are tested by a discrete Mann-Whitney U-test, rejecting H_0 with $p < 0.03$.

4.3. Evaluation of Disentanglement

Disentanglement is also an important feature of our model. It means that when an attribute is modified, other attributes remain unchanged. We make the three models (VAE+GAN+MSP, AttGAN, and Fader Networks) generate images by manipulating two groups of highly correlated attributes, openness of mouth / smiling, and male / beard. For the two groups, the three models should respectively generate the images with closed mouth $\times$ no smiling, closed mouth $\times$ smiling, opened mouth $\times$ no smiling, opened mouth $\times$ smiling, female $\times$ no beard, female $\times$ beard, male $\times$ no beard, and male $\times$ beard. We hired 50 participants in Amazon Mechanical Turk; each of them was given 40 image blocks. A block contains four images, which were from the mouth / smiling group or the male / beard group, and which were generated by one of the three models. The participants were told which image should represent which attributes, and the participants evaluated whether it did for each image in the block by using a 3-level Likert scale (perfect, recognisable, and unrecognisable/unchanged). The results are shown in Table 4. It shows that our model performs significantly better than the baseline. ($p < 0.0001$).

In addition to human evaluation, we also conducted a quantitative evaluation to test how well a model can change an attribute in isolation. For some selected highly correlated attributes we change one target attribute, and measure the change in another non-target attribute. For instance (the row of gender/beard in Table 3), when the gender attribute is changed manually, we measure the amount by which the beard attribute is consequently changed. The results are shown in Table 3. Note that, the scores show how much the non-target attributes are affected, but not whether the target attributes are correctly changed in the generated pictures. Therefore the scores need to be read in conjunction with Table 2. According to both Table 2 and Table 3, we can conclude that in both of the aspects of the manipulation of attributes and avoiding influence on non-target attributes, the performance of our model exceeds the baselines.

4.4. Matrix Subspace Projection in Seq2seq

We apply our model to a classic *seq2seq* model for textual content replacement, in order to determine if we can replace words according to the given attributes and keep other words unchanged. In this task, we adopt the E2E corpus (Dušek et al., 2019), which contains 50k+ reviews of restaurants (E2E dataset is developed for Natural Language Generation, but here we use it for content replacement). Each review is a single sentence that is labelled by the attribute-value pairs, for example, "name=[The Eagle]", "food=[French]", and "customerRating=[3/5]". We regard each attribute-value pair as a unique label. All the attributes constitute y whose entries are 1 or 0 to represent each value (the correct texts of the attribute name or value are NOT used).

Both the encoder and decoder of the seq2seq model are formed by two-layer LSTMs. The model is trained for 1000

[3]The generated examples were automatically filtered to prevent conflict attributes; e.g. images of woman with beard are not provided to the participants in this experiment.

	Example 1	Example 2
Orig-attribute	eatType[pub], customer-rating[5-out-of-5], name[Blue-Spice], near[Crowne-Plaza-Hotel]	familyFriendly[yes], area[city-centre], eatType[pub], food[Japanese], near[Express-by-Holiday-Inn], name[Green-Man]
Orig-text	the blue spice pub , near crowne plaza hotel , has a customer rating of 5 out of 5 .	near the express by holiday inn in the city centre is green man . it is a japanese pub that is family-friendly .
New-attribute	eatType[coffee-shop], customer-rating[5-out-of-5], name[Blue-Spice], near[Avalon]	familyFriendly[no], area[riverside], eatType[coffee-shop], food[French], near[The-Six-Bells], name[Green-Man]
New-text	the blue spice coffee shop , near avalon has a customer rating of 5 out of 5 .	near the six bells in the riverside area is a green man . it is a french coffee shop that is not family-friendly .

Table 5. Results of changing attributes in E2E corpus.

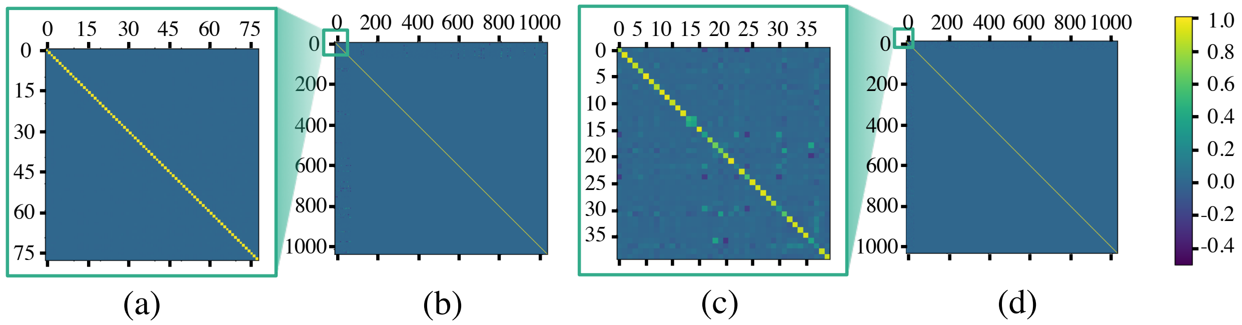


Figure 5. Measuring orthogonality: Heat map of $\mathbf{M}^T \cdot \mathbf{M}$ and $\mathbf{U}^T \cdot \mathbf{U}$ for Seq2seq+MSP (a,b), and VAE+GAN+MSP (c,d)

epochs (on a Tesla T4 around 12 hours). After training, we reconstruct the review sentences with randomly replaced attributes, for example replacing “name=[The Eagle]” by “name=[Burger King]”, “customerRating=[3/5]” by “customerRating=[1/5]”. 50% of attributes are changed in each sentence. The outcomes are shown in Table 5.

4.5. Orthogonality Evaluation

The ability to disentangle attributes is ensured by the orthogonality of $\mathbf{M}$ in our model. Instead of directly using an orthogonal matrix, we train $\mathbf{M}$ to be orthogonal. Thus, we evaluate how close $\mathbf{M}$ is to the orthogonal matrix. Fig. 5 shows the heat map of $\mathbf{M}^T \cdot \mathbf{M}$ and $\mathbf{U}^T \cdot \mathbf{U}$, which indicates that the production is fairly close to a unit matrix. It visualises $\mathbf{M}^T \cdot \mathbf{M}$ in the seq2seq version of our model (Fig. 5 (a)) and in the VAE version (Fig. 5 (c)). The matrix $\mathbf{U}^T \cdot \mathbf{U}$ ($\mathbf{U}$ is formed by $\mathbf{M}$ concatenating its null space) is also visualised (in Fig. 5 (b) and (d)). It is clear that when the matrices are multiplied by their transposes, the products do approximate the unit matrix. Although Fig. 5 (c) shows that a small number of attributes remain slightly entangled (by the green and deep purple pixels), this is mainly caused by the few conflicting attributes in CelebA, for example the receding-hairline $\times$ bald $\times$ bangs, and the straight-hair $\times$ wavy-hair. Thus, $\mathbf{M}$ is indeed trained to be a (partial) orthogonal matrix.

5. Conclusion

We propose a matrix projection plugin that can be attached to various autoencoders (e.g. Seq2seq, VAE) to make the latent space factorised and disentangled, based on labelled attribute information, which ensures that manipulation in the latent space is much easier. We test the attribute manipulation ability of our model on an image dataset and text corpus, obtaining results that show clean disentanglement. In addition our model involves a simpler training process than adversarial approaches which need a long training with a very low weight on the loss coming from the discriminator that removes attribute information, to avoid the encoder being too affected by this loss term (Lample et al., 2017).

Acknowledgement

We would like to thank all the anonymous reviewers for their insightful comments. This work is supported by the award made by the UK Engineering and Physical Sciences Research Council (Grant number: EP/P011829/1).

References

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 2172–2180. Curran Associates, Inc., 2016.

Choi, Y., Choi, M., Kim, M., Ha, J., Kim, S., and Choo, J. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 8789–8797. IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00916.

Creswell, A., Bharath, A. A., and Sengupta, B. Conditional autoencoders with adversarial information factorization. *CoRR*, abs/1711.05175, 2017. URL http://arxiv.org/abs/1711.05175.

Dušek, O., Novikova, J., and Rieser, V. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG Challenge. *arXiv preprint arXiv:1901.11528*, January 2019. URL https://arxiv.org/abs/1901.11528.

He, Z., Zuo, W., Kan, M., Shan, S., and Chen, X. Attgan: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing*, 28(11):5464–5478, Nov 2019. ISSN 1941-0042. doi: 10.1109/TIP.2019.2916751.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6629–6640, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Higgins, I., Matthey, L., Glorot, X., Pal, A., Uria, B., Blundell, C., Mohamed, S., and Lerchner, A. Early visual concept learning with unsupervised deep learning. *CoRR*, abs/1606.05579, 2016. URL http://arxiv.org/abs/1606.05579.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Hu, Z., Yang, Z., Liang, X., Salakhutdinov, R., and Xing, E. P. Toward controlled generation of text. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1587–1596, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/hu17e.html.

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

Kingma, D. P., Mohamed, S., Jimenez Rezende, D., and Welling, M. Semi-supervised learning with deep generative models. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27*, pp. 3581–3589. Curran Associates, Inc., 2014.

Klys, J., Snell, J., and Zemel, R. Learning latent subspaces in variational autoencoders. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 6444–6454. Curran Associates, Inc., 2018.

Kumar, A., Sattigeri, P., and Balakrishnan, A. Variational inference of disentangled latent concepts from unlabeled observations. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018. URL https://openreview.net/forum?id=H1kG7GZAW.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. Building machines that learn and think like people. *CoRR*, abs/1604.00289, 2016.

Lample, G., Zeghidour, N., Usunier, N., Bordes, A., Denoyer, L., et al. Fader networks: Manipulating images by sliding attributes. In *Advances in Neural Information Processing Systems*, pp. 5967–5976, 2017.

LeCun, Y. The power and limits of deep learning. *ResearchTechnology Management*, 61(6):22–27, 2013. doi: 10.1080/08956308.2018.1516928.

Li, C. and Wand, M. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision*, pp. 702–716. Springer, 2016.

Li, R., Li, X., Lin, C., Collinson, M., and Mao, R. A stable variational autoencoder for text modelling. In *Proceedings of the 12th International Conference on Natural Language Generation*, pp. 594–599, 2019.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738, 2015.

Perarnau, G., van de Weijer, J., Raducanu, B., and Álvarez, J. M. Invertible conditional gans for image editing. *CoRR*, abs/1611.06355, 2016. URL `http://arxiv.org/abs/1611.06355`.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28*, pp. 3483–3491. Curran Associates, Inc., 2015.

Torfason, R., Agustsson, E., Rothe, R., and Timofte, R. From face images and attributes to attributes. In *13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24*, 11 2016. doi: 10.1007/978-3-319-54187-7_21.

Upchurch, P., Gardner, J. R., Pleiss, G., Pless, R., Snavely, N., Bala, K., and Weinberger, K. Q. Deep feature interpolation for image content changes. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 6090–6099, 2017. doi: 10.1109/CVPR.2017.645. URL `https://doi.org/10.1109/CVPR.2017.645`.

Wu, P.-W., Lin, Y.-J., Chang, C.-H., Chang, E. Y., and Liao, S.-W. Relgan: Multi-domain image-to-image translation via relative attributes. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

Xiao, Y., Zhao, T., and Wang, W. Y. Dirichlet variational autoencoder for text modeling. *CoRR*, abs/1811.00135, 2018. URL `http://arxiv.org/abs/1811.00135`.

Yan, X., Yang, J., Sohn, K., and Lee, H. Attribute2image: Conditional image generation from visual attributes. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, pp. 776–791, 2016. doi: 10.1007/978-3-319-46493-0_47. URL `https://doi.org/10.1007/978-3-319-46493-0_47`.

Yuille, A. L. and Liu, C. Deep nets: What have they ever done for vision? *CoRR*, abs/1805.04025, 2018.