A Disentangling Invertible Interpretation Network for Explaining Latent Representations

Patrick Esser* Robin Rombach* Björn Ommer
Heidelberg Collaboratory for Image Processing
IWR, Heidelberg University, Germany

Abstract

Neural networks have greatly boosted performance in computer vision by learning powerful representations of input data. The drawback of end-to-end training for maximal overall performance are black-box models whose hidden representations are lacking interpretability: Since distributed coding is optimal for latent layers to improve their robustness, attributing meaning to parts of a hidden feature vector or to individual neurons is hindered. We formulate interpretation as a translation of hidden representations onto semantic concepts that are comprehensible to the user. The mapping between both domains has to be bijective so that semantic modifications in the target domain correctly alter the original representation. The proposed invertible interpretation network can be transparently applied on top of existing architectures with no need to modify or retrain them. Consequently, we translate an original representation to an equivalent yet interpretable one and backwards without affecting the expressiveness and performance of the original. The invertible interpretation network disentangles the hidden representation into separate, semantically meaningful concepts. Moreover, we present an efficient approach to define semantic concepts by only sketching two images and also an unsupervised strategy. Experimental evaluation demonstrates the wide applicability to interpretation of existing classification and image generation networks as well as to semantically guided image manipulation.

1. Introduction

Deep neural networks have achieved unprecedented performance in various computer vision tasks [51, 15] by learning task-specific hidden representations rather than relying on predefined hand-crafted image features. However, the significant performance boost due to end-to-end learning comes at the cost of now having black-box models that are lacking interpretability: A deep network may have found a solution to a task, but human users cannot understand the causes for the predictions that the model makes [36]. Conversely, users must also be able to understand what the hidden representations have *not* learned and on which data the overall model will, consequently, fail. Interpretability is therefore a prerequisite for safeguarding AI, making its decisions transparent to the users it affects, and understanding its applicability, limitations, and the most promising options for its future improvement.

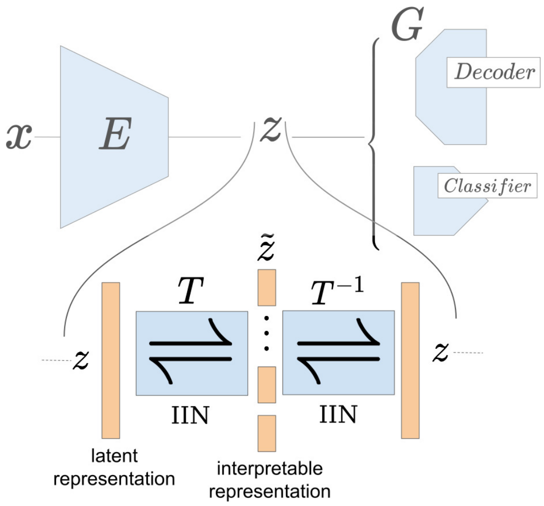

Figure 1: Our Invertible Interpretation Network T can be applied to arbitrary existing models. Its invertibility guarantees that the translation from z to $\tilde{z}$ does not affect the performance of the model to be interpreted. Code and results can be found at the project page https://compvis.github.io/iin/.

A key challenge is that learned latent representations typically do not correspond to semantic concepts that are comprehensible to human users. Hidden layer neurons are trained to help in solving an overall task in the output layer of the network. Therefore, the output neurons correspond to human-interpretable concepts such as object classes in semantic image segmentation [3] or bounding boxes in object detection [48]. In contrast, the hidden layer representa-

[*] Both authors contributed equally to this work.

tion of semantic concepts is a distributed pattern [9]. This distributed coding is crucial for the robustness and generalization performance of neural networks despite noisy inputs, large intra-class variability, and the stochastic nature of the learning algorithm [13]. However, as a downside of semantics being distributed over numerous neurons it is impossible to attribute semantic meaning to only an individual neuron despite attempts to backpropagate [37] or synthesize [47, 52] their associated semantic concepts. One solution has been to modify and constrain the network so that abstract concepts can be localized in the hidden representation [55]. However, this alters the network architecture and typically deteriorates overall performance [45].

Objective: Therefore, our goal needs to be an approach that can be transparently applied on top of arbitrary existing networks and their already learned representations without altering them. We seek a *translation* between these hidden representations and human-comprehensible semantic concepts—a non-linear mapping between the two domains. This translation needs to be invertible, i.e. an invertible neural network (INN) [5, 6, 19, 22], so that modifications in the domain of semantic concepts correctly alter the original representation.

To interpret a representation, we need to attribute meaning to parts of the feature encoding. That is, we have to disentangle the high-dimensional feature vector into multiple multi-dimensional factors so that each is mapped to a separate semantic concept that is comprehensible to the user. As discussed above, this disentangled mapping should be bijective so that modifications of the disentangled semantic factors correctly translate back to the original representation. We can now, without any supervision, disentangle the representation into independent concepts so that a user can post-hoc identify their meaning. Moreover, we present an efficient strategy for defining semantic concepts. It only requires two sketches that exhibit a change in a concept of interest rather than large annotated training sets for each concept. Given this input, we derive the invariance properties that characterize a concept and generate synthesized training data to train our invertible interpretation network. This network then acts as a translator that disentangles the original representation into multiple factors that correspond to the semantic concepts.

Besides interpreting a network representation, we can also interpret the structure that is hidden in a dataset and explain it to the user. Applying the original representation and then translating onto the disentangled semantic factors allows seeing which concepts explain the data and its variability. Finally, the invertible translation supports semantically meaningful modifications of input images: Given an autoencoder representation, its representation is mapped onto interpretable factors, these can be modified and inverse translation allows to apply the decoder to project back into the image domain. In contrast to existing disentangled image synthesis [34, 8, 32, 24, 7], our invertible approach can be applied on top of existing autoencoder representations, which therefore do not have to be altered or retrained to handle different semantic concepts. Moreover, for other architectures such as classification networks, interpretability helps to analyze their invariance and robustness.

To summarize, *(i)* we present a new approach to the interpretability of neural networks, which can be applied to arbitrary existing models without affecting their performance; *(ii)* we obtain an invertible translation from hidden representations to disentangled representations of semantic concepts; *(iii)* we propose a method that allows users to efficiently define semantic concepts to be used for our interpretable representation; *(iv)* we investigate the interpretation of hidden representations, of the original data, and demonstrate semantic image modifications enabled by the invertibility of the translation network.

2. Interpretability

An interpretation is a translation between two domains such that concepts of the first domain can be understood in terms of concepts of the second domain. Here, we are interested in interpretations of internal representations of a neural network in terms of human-understandable representations. Examples for the latter are given by textual descriptions, visual attributes or images.

To interpret neural networks, some approaches modify network architectures or losses used for training to obtain inherently more interpretable networks. [55] relies on a global average pooling layer to obtain class activation maps, *i.e.* heatmaps showing which regions of an input are most relevant for the prediction of a certain object class. [54] learn part specific convolutional filters by restricting filter activations to localized regions. Invertible neural networks [5, 6, 19, 22] have been used to get a better understanding of adversarial attacks [18]. Instead of replacing existing architectures with invertible ones, we propose to augment them with invertible transformations. Using the invertibility, we can always map back and forth between original representations and interpretable ones without loss of information. Thus, our approach can be applied to arbitrary existing architectures without affecting their performance, whereas approaches modifying architectures always involve a tradeoff between interpretability and performance.

Most works on interpretability of existing networks focus on visualizations. [53] reconstruct images which activated a specific feature layer of a network. [47] uses gradient ascent to synthesize images which maximize class probabilities for different object classes. [52] generalizes this to arbitrary neurons within a network. Instead of directly optimizing over pixel values, [38] optimize over input codes of a generator network which was trained to reconstruct images from hidden layers. [55] avoid synthesizing images from scratch and look for regions within a given image that activate certain neurons. For a specific class of functions, [1] decompose the function into relevance scores which can be visualized pixel-wise. Layer-wise relevance propagation [37] is a more general approach to propagate relevance scores through a network based on rules to distribute the relevance among input neurons. [41] shows how saliency maps representing the importance of pixels for a classifier's decision can be obtained without access to the classifiers gradients. All these approaches assume that a fixed set of neurons is given and should be interpreted in terms of inputs which activate them. However, [2], [9] demonstrated that networks use distributed representations. In particular, semantic concepts are encoded by activation patterns of multiple neurons and single neurons are not concept specific but involved in the representation of different concepts. We directly address this finding by learning a non-linear transformation from a distributed representation to an interpretable representation with concept specific factors.

While [9] shows that for general networks we must expect internal representations to be distributed, there are situations where representations can be expected to have a simpler structure: Generative models are trained with the explicit goal to produce images from samples of a simple distribution, *e.g.* a Gaussian distribution. Most approaches are based either on Variational Autoencoders [23, 44], which try to reconstruct images from a representation whose marginal distribution is matched to a standard normal distribution, or on Generative Adversarial Networks [11, 14, 39], which directly map samples from a standard normal distribution to realistic looking images as judged by a discriminator network. The convexity of the Gaussian density makes linear operations between representations meaningful. Linear interpolations between representations enable walks along nonlinear data manifolds [42]. [26] finds visual attribute vectors which can be used to interpolate between binary attributes. To this end, two sets of images containing examples with or without an attribute are encoded to their representations, and the direction between their means is the visual attribute vector. Such attribute vectors have also been found for classifier networks [49], but because their representations have no linear structure, the approach is limited to aligned images. [42, 43] demonstrated that vector arithmetic also enables analogy making. [46] interprets the latent space of a GAN by finding attribute vectors as the normal direction of the decision boundary of an attribute classifier. [10] uses a similar approach to find attribute vectors associated with cognitive properties such as memorability, aesthetics and emotional valence. While these approaches provide enhanced interpretability through modifcation of attributes they are limited to representations with a linear structure. In contrast, we provide an approach to map arbitrary representations into a space of interpretable representations. This space consists of factors representing semantic attributes and admits linear operations. Thus, we can perform semantic modifications in our interpretable space and, due to the invertibility of our transformation, map the modified representation back to the original space.

3. Approach

3.1. Interpreting Hidden Representations

Invertible Transformation of Hidden Representations: Let f be a given neural network to be interpreted. We place no restrictions on the network f. For example, f could be an object classifier, a segmentation network or an autoencoder. f maps an image $x \in \mathbb{R}^{h \times w \times 3}$ through a sequence of hidden layers to a final output $f(x)$. Intermediate activations $E(x) \in \mathbb{R}^{H \times W \times C}$ of a hidden layer are a task-specific representation of the image x. Such hidden representations convey no meaning to a human and we must transform them into meaningful representations. We introduce the notation $z = E(x) \in \mathbb{R}^{H \cdot W \cdot C}$, *i.e.* z is the $N = H \cdot W \cdot C$ dimensional, flattened version of the hidden representation to be interpreted. E is the sub-network of f consisting of all layers of f up to and including the hidden layer that produces z, and the sub-network after this layer will be denoted by G, such that $f(x) = G \circ E(x)$ as illustrated in Fig. 1.

To turn z into an interpretable representation, we aim to translate the distributed representation z to a factorized representation $\tilde{z} = (\tilde{z}_k)_{k=0}^{K} \in \mathbb{R}^N$ where each of the $K + 1$ factors $\tilde{z}_k \in \mathbb{R}^{N_k}$, with $\sum_{k=0}^{K} N_k = N$, represents an interpretable concept. The goal of this translation is twofold: On the one hand, it should enable an analysis of the relationship between data and internal representations of f in terms of interpretable concepts; this requires a forward map T from z to $T(z) = \tilde{z}$. On the other hand, it should enable semantic modifications on internal representations of f; this requires the inverse of T. With this inverse map, T^{-1}, an internal representation z can be mapped to $\tilde{z}$, modified in semantically meaningful ways to obtain $\tilde{z}^*$ (*e.g.* changing a single interpretable concept), and mapped back to an internal representation of f. This way, semantic modifications, $\tilde{z} \mapsto \tilde{z}^*$, which were previously only defined on $\tilde{z}$ can be applied to internal representations via $z \mapsto z^* := T^{-1}(T(z)^*)$. See Fig. 2 for an example, where z is modified by replacing one of its semantic factors $\tilde{z}_k$ with that of another image.

Disentangling Interpretable Concepts: For meaningful analysis and modification, each factor $\tilde{z}_k$ must represent a specific interpretable concept and taken together, $\tilde{z}$ should support a wide range of modifications. Most importantly, it must be possible to analyze and modify different factors $\tilde{z}_k$ independently of each other. This implies a factorization of their joint density $p(\tilde{z}) = \prod_{k=0}^{K} p(\tilde{z}_k)$. To explore different

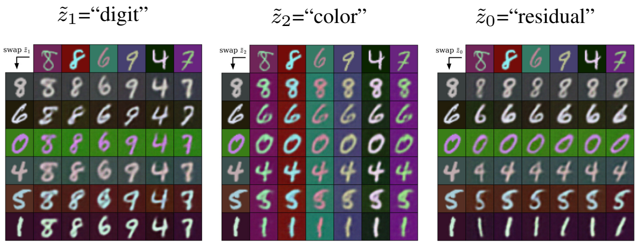

Figure 2: Applied to latent representations z of an autoencoder, our approach enables semantic image analogies. After transforming z to disentangled semantic factors $(\tilde{z}_k)_{k=0}^{K} = T(z)$, we replace $\tilde{z}_k$ of the target image (leftmost column), with $\tilde{z}_k$ of the source image (top row). From left to right: $k = 1$ (digit), $k = 2$ (color), $k = 0$ (residual).

factors, the distribution $p(\tilde{z}_k)$ of each factor must be easy to sample from to gain insights into the variability of a factor, and interpolations between two samples of a factor must be valid samples to analyze changes along a path. We thus specify each factor to be normally distributed which gives

$$p(\tilde{z}) = \prod_{k=0}^{K} \mathcal{N}(\tilde{z}_k|0, \mathbf{1}) \qquad (1)$$

Without additional constraints, the semantics represented by a factor $\tilde{z}_k$ are unspecified. To fix this, we demand that *(i)* each factor $\tilde{z}_k$ varies with one and only one interpretable concept and *(ii)* it is invariant with respect to all other variations. Thus, let there be training image pairs (x^a, x^b) which specify semantics through their similarity, *e.g.* image pairs containing animals of the same species to define the semantic concept of 'animal species'. Each semantic concept $F \in \{1, \dots, K\}$ defined by such pairs shall be represented by the corresponding factor $\tilde{z}_F$ and we write $(x^a, x^b) \sim p(x^a, x^b|F)$ to emphasize that (x^a, x^b) is a training pair for factor $\tilde{z}_F$. However, we cannot expect to have examples of image pairs for every semantic concept relevant in z. Still, all factors together, $\tilde{z} = (\tilde{z}_k)_{k=0}^{K}$, must be in one-to-one correspondence with the original representation, *i.e.* $z = T^{-1}(\tilde{z})$. Therefore, we introduce $\tilde{z}_0$ to act as a residual concept that captures the remaining variability of z which is missed by the semantic concepts $F = 1, \dots, K$.

For a given training pair $(x^a, x^b) \sim p(x^a, x^b|F)$, the corresponding factorized representations, $\tilde{z}^a = T(E(x^a))$ and $\tilde{z}^b = T(E(x^b))$, must now *(i)* mirror the semantic similarity of (x^a, x^b) in its F-th factor and *(ii)* be invariant in the remaining factors. This is expressed by a positive correlation factor $\sigma_{ab} \in (0, 1)$ for the F-th factor between pairs,

$$\tilde{z}^b_F \sim \mathcal{N}(\tilde{z}^b_F|\sigma_{ab}\tilde{z}^a_F, (1 - \sigma_{ab}^2)\mathbf{1}) \qquad (2)$$

and no correlation for the remaining factors between pairs,

$$\tilde{z}^b_k \sim \mathcal{N}(\tilde{z}^b_k|0, \mathbf{1}) \quad k \in \{0, \dots, K\} \setminus \{F\} \qquad (3)$$

To fit this model to data, we utilize the invertibility of T to directly compute and maximize the likelihood of pairs $(z^a, z^b) = (E(x^a), E(x^b))$. We compute the likelihood with the absolute value of the Jacobian determinant of T, denoted $|T'(\cdot)|$, as

$$\begin{aligned} p(z^a, z^b|F) &= p(z^a)\, p(z^b|z^a, F) && (4)\\ &= |T'(z^a)|\, p(T(z^a))\, \cdot && (5)\\ &\quad\; |T'(z^b)|\, p\left(T(z^b)|T(z^a), F\right) && (6) \end{aligned}$$

To be able to compute the Jacobian determinant efficiently, we follow previous works [22] and build T based on ActNorm, AffineCoupling and Shuffling layers as described in more detail in Sec. A.1 of the supplementary. For training we use the negative log-likelihood as our loss function. Substituting Eq. (1) into Eq. (5), Eq. (2) and (3) into Eq. (6), leads to the per-example loss $\ell(z^a, z^b|F)$,

$$\begin{aligned} \ell(z^a, z^b|F) = &\sum_{k=0}^{K} \|T(z^a)_k\|^2 - \log|T'(z^a)| && (7)\\ &+ \sum_{k\neq F} \|T(z^b)_k\|^2 - \log|T'(z^b)| && (8)\\ &+ \frac{\|T(z^b)_F - \sigma_{ab}\, T(z^a)_F\|^2}{1 - \sigma_{ab}^2} && (9) \end{aligned}$$

which is optimized over training pairs (x^a, x^b) for all semantic concepts $F \in \{1, \dots, K\}$:

$$\mathcal{L} = \sum_{F=1}^{K} \mathbb{E}_{(x^a,x^b)\sim p(x^a,x^b|F)}\; \ell(E(x^a), E(x^b)|F) \qquad (10)$$

Note that we have described the case where image pairs share at least one semantic concept, which includes the case where they share more than one semantic concept. Moreover, our approach is readily applicable in the case where image pairs differ in a semantic concept. In this case, Eq. (2) holds for all factors $\tilde{z}^b_k, k \in \{0, \dots, K\} \setminus \{F\}$ and Eq. (3) holds for factor $\tilde{z}^b_F$. This case will also be used in the next section, where we discuss the dimensionality and origin of semantic concepts.

3.2. Obtaining Semantic Concepts

Estimating Dimensionality of Factors: Semantic concepts differ in complexity and thus also in dimensionality. Given image pairs $(x^a, x^b) \sim p(x^a, x^b|F)$ that define the F-th semantic concept, we must estimate the dimensionality of factor $\tilde{z}_F$ that represents this concept. Due to the invertibility of T, the sum of dimensions of all these factors equals the dimensionality of the original representation. Thus, semantic concepts captured by the network E require a larger share of the overall dimensionality than those E is invariant to.

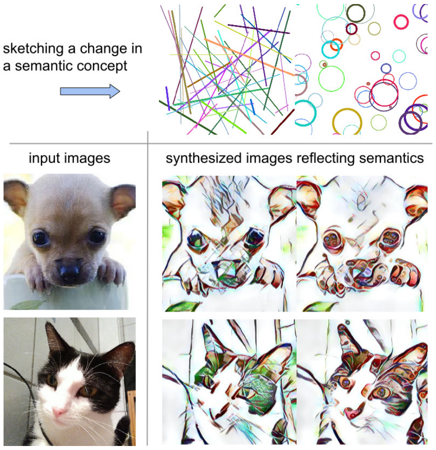
sketching a change in a semantic concept
input images
synthesized images reflecting semantics

Figure 3: Efficient generation of training examples for semantic concepts: A user must only provide two sketches (first row) for a change of a semantic concept, here: *roundness*. We then synthesize training images to reflect this semantic change.

The similarity of x^a, x^b in the F-th semantic concept implies a positive mutual information between them, which will only be preserved in the latent representations $E(x^a), E(x^b)$ if the F-th semantic concept is captured by E. Thus, based on the simplifying assumption that components of hidden representations $E(x^a)_i, E(x^b)_i$ are jointly Gaussian distributed, we approximate their mutual information with their correlation for each component i. Summing over all components i yields a relative score s_F that serves as proxy for the dimensionality of $\tilde{z}_F$ in case of training images $(x^a, x^b) \sim p(x^a, x^b|F)$ for concept F,

$$s_F = \sum_i \frac{\mathrm{Cov}\big(E(x^a)_i, E(x^b)_i\big)}{\sqrt{\mathrm{Var}(E(x^a)_i)\ \mathrm{Var}(E(x^b)_i)}}. \tag{11}$$

Since correlation is in $[-1, 1]$, scores s_F are in $[-N, N]$ for N-dimensional latent representations of E. Using the maximum score N for the residual factor $\tilde{z}_0$ ensures that all factors have equal dimensionality if all semantic concepts are captured by E. The dimensionality N_F of $\tilde{z}_F$ is then $N_F = \left\lfloor \frac{\exp s_F}{\sum_{k=0}^{K} \exp s_k} N \right\rfloor$. Tab. 1 demonstrates the feasibility of predicting dimensionalities with this approximation.

Dataset	Model	Latent z Dim.	Interpretable $\tilde{z}$ Dim.	Factor $\tilde{z}_F$
Color-MNIST	AE	64	12	Digit
			19	Color
	Classifier	64	22	Digit
			11	Color

Table 1: Estimated dimensionalities of interpretable factors $\tilde{z}_F$ representing different semantic concepts. Remaining dimensions are assigned to the residual factor $\tilde{z}_0$. Compared to an autoencoder, the color factor is smaller in case of a color-invariant classifier.

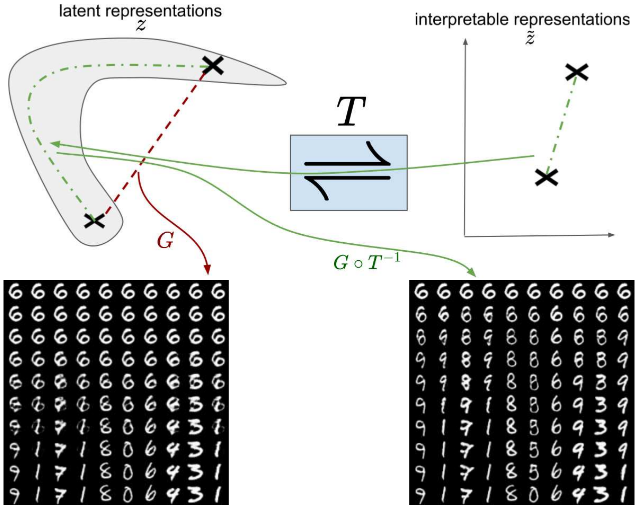

Figure 4: The inverse of our interpretation network T maps linear walks in the interpretable domain back to nonlinear walks on the data manifold in the encoder space, which get decoded to meaningful images (bottom right). In contrast, decoded images of linear walks in the encoder space contain ghosting artifacts (bottom left).

Sketch-Based Description of Semantic Concepts: Training requires the availability of image pairs that depict changes in a semantic concept. Most often, a sufficiently large number of such examples is not easy to obtain. The following describes an approach to help a user specify semantic concepts effortlessly.

Two sketches are worth a thousand labels: Instead of labeling thousands of images with semantic concepts, a user only has to provide two sketches, y^a and y^b which demonstrate a change in a concept. For example, one sketch may contain mostly round curves and another mostly angular ones as in Fig. 3. We then utilize a style transfer algorithm [40] to transform each x from the training set into two new images: x^a and x^b which are stylized with y^a and y^b, respectively. The combinations (x, x^a), (x, x^b) and (x^a, x^b) serve as examples for a change in the concept of interest.

Unsupervised Interpretations: Even without examples for changes in semantic factors, our approach can still produce disentangled factors. In this case, we minimize the negative log-likelihood of the marginal distribution of hidden representations $z = E(x)$:

$$\mathcal{L}_{\text{unsup}} = -\mathbb{E}_x \|T(E(x))\|^2 - \log|T'(E(x))| \tag{12}$$

As this leads to independent components in the transformed representation, it allows users to attribute meaning to this representation after training. Mapping a linear interpolation in our disentangled space back to E's representation space

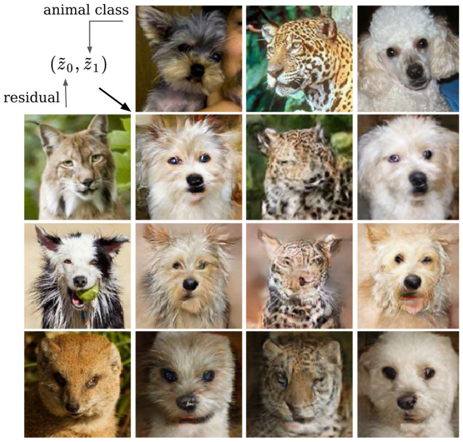

Figure 5: Transfer on AnimalFaces: We combine $\tilde{z}_0$ (residual) of the target image (leftmost column) with $\tilde{z}_1$ (animal class) of the source image (top row), resulting in a transfer of animal type from source to target.

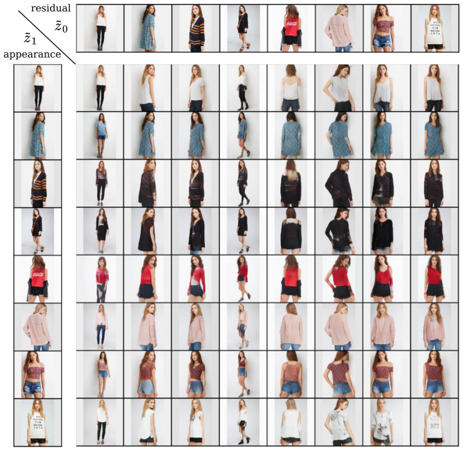

Figure 6: Transfer on DeepFashion: We combine $\tilde{z}_0$ (residual) of the target image (top row) with $\tilde{z}_1$ (appearance) of the source image (leftmost column), resulting in a transfer of appearances from source to target.

leads to a nonlinear interpolation on the data manifold embedded by E (see Fig. 4). This linear structure allows to explore the representations using vector arithmetics [49]. For example, based on a few examples of images with a change in a semantic concept, we can find a vector representing this concept as the mean direction between these images (see Eq. (14)). In contrast to previous works, we do not rely on disentangled latent representations but learn to translate arbitrary given representations into disentangled ones.

4. Experiments

The subsequent experiments use the following datasets: *AnimalFaces* [28], *DeepFashion* [29, 31], *CelebA* [30] *MNIST*[27], *Cifar10* [25], and *FashionMNIST* [50]. Moreover, we augment MNIST by randomly coloring its images to provide a benchmark for disentangling experiments (denoted *ColorMNIST*).

4.1. Interpretation of Autoencoder-Frameworks

Autoencoders learn to reconstruct images from a low-dimensional latent representation $z = E(x)$. Subsequently, we map z onto interpretable factors to perform semantic image modification. Note that z is only obtained using a given network; our invertible interpretation network has never seen an image itself.

Disentangling Latent Codes of Autoencoders: Now we alter the $\tilde{z}_k$ which should in turn modify the z in a semantically meaningful manner. This tests two aspects of our translation onto mutually disentangled, interpretable representations: First, if its factors have been successfully disentangled, swapping factors from different images should still yield valid representations. Second, if the factors represent their semantic concepts faithfully, modifying a factor should alter its corresponding semantic concept.

To evaluate these aspects, we trained an autoencoder on the AnimalFaces dataset. As semantic concepts we utilize the animal category and a residual factor. Fig. 5 shows the results of combining the residual factor of the image on the left with the animal-class-factor of the image at the top. After decoding, the results depict animals from the class of the image at the top. However, their gaze direction corresponds to the image on the left. This demonstrates a successful disentangling of semantic concepts in our interpreted space.

The previous case has confirmed the applicability of our approach to roughly aligned images. We now test it on unaligned images of articulated persons on DeepFashion. Fig. 6 presents results for attribute swapping as in the previous experiment. Evidently, our approach can handle articulated objects and enables pose guided human synthesis.

Finally, we conduct this swapping experiment on ColorMNIST to investigate simultaneous disentangling of multiple factors. Fig. 2 shows a swapping using an interpretation of three factors: digit type, color, and residual.

Evaluating the Unsupervised Case: To investigate our approach in case of no supervision regarding semantic concepts, we analyze its capability to turn simple autoencoders into generative models. Because our interpretations yield normally distributed representations, we can sample them, translate them back onto the latent space of the autoencoder, and finally decode them to images,

$$\tilde{z} \sim \mathcal{N}(\tilde{z}|0, \mathbf{1}), \quad x = G(T^{-1}(\tilde{z})). \tag{13}$$

	MNIST	FashionMNIST	CIFAR-10	CelebA
TwoStageVAE	12.6 ± 1.5	29.3 ± 1.0	72.9 ± 0.9	44.4 ± 0.7
WGAN GP	20.3 ± 5.0	24.5 ± 2.1	55.8 ± 0.9	30.3 ± 1.0
WGAN	6.7 ± 0.4	21.5 ± 1.6	55.2 ± 2.3	41.3 ± 2.0
DRAGAN	7.6 ± 0.4	27.7 ± 1.2	69.8 ± 2.0	42.3 ± 3.0
BEGAN	13.1 ± 1.0	22.9 ± 0.9	71.9 ± 1.6	38.9 ± 0.9
Ours	$\mathbf{6.4 \pm 0.1}$	$\mathbf{16.0 \pm 0.1}$	$\mathbf{45.7 \pm 0.3}$	$\mathbf{20.2 \pm 0.5}$

Table 2: FID scores of various AE-based and GAN models as reported in [4].

We employ the standard evaluation protocol for generative models and measure image quality with Fréchet Inception Distance (FID scores). [4] presented an approach to generative modeling using two Variational Autoencoders and achieved results competitive with approaches based on GANs. We follow [4] and use an autoencoder architecture based on [33] and train on the same datasets with losses as in [26]. Tab. 2 presents mean and std of FID scores over three trials with 10K generated images. We significantly improve over state-of-the-art FID reported in [4]. Our approach can utilize a learned similarity metric similar to GANs, which enables them to produce high-quality images. In contrast to approaches based on GANs, we can rely on an autoencoder and a reconstruction loss. This enables stable training and avoids the mode-collapse problem of GANs, which explains our improvement in FID.

Besides sampling from the model as described by equation (13), our approach supports semantic interpolation in the representation space, since the invertible network constitutes a lossless encoder/decoder framework. We obtain semantic axes $\tilde{z}^{F \rightarrow \bar{F}}$ by encoding two sets of images $X^F = \{x^F\}, X^{\bar{F}} = \{x^{\bar{F}}\}$, showing examples with an attribute in X^F and without that attribute in $X^{\bar{F}}$. Note that these sets are only required after training, *i.e.* during test time. $\tilde{z}^{F \rightarrow \bar{F}}$ is then obtained as the average direction between examples of X_F and $X_{\bar{F}}$,

$$\tilde{z}^{F \rightarrow \bar{F}} = \frac{1}{|X^{\bar{F}}|} \sum_{x^{\bar{F}} \in X^{\bar{F}}} x^{\bar{F}} - \frac{1}{|X^F|} \sum_{x^F \in X^F} x^F. \tag{14}$$

Such vector arithmetic depends on a meaningful linear structure of our interpreted representation space. We illustrate this structure in Fig. 4. Linear walks in our interpretable space always result in meaningful decoded images, indicating that the backtransformed representations lie on the data manifold. In contrast, decoded images of linear walks in the encoder's hidden representation space contain ghosting artifacts. Consequently, our model can transform nonlinear hidden representations to an interpretable space with linear structure. Fig. 7 visualizes a 2D submanifold on CelebA.

Fig. 8 provides an example for an interpolation as described in Eq. 14 between attributes on the CelebA dataset. We linearly walk along the *beardiness* and *smiling* attributes, increasing the former and decreasing the latter.

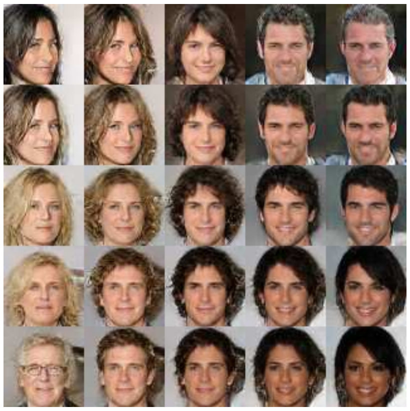

Figure 7: CelebA: Four randomly drawn samples (corners) and corresponding interpolations obtained with unsupervised training, see Sec. 4.1.

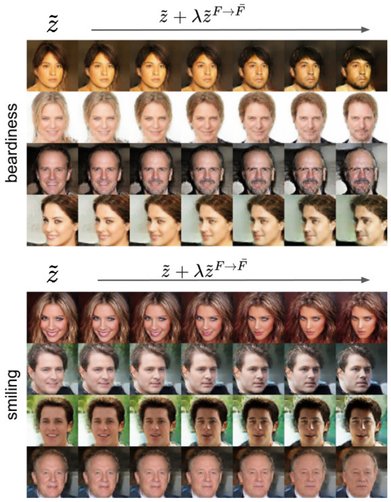
$\tilde{z}$
$\tilde{z} + \lambda \tilde{z}^{F \rightarrow \bar{F}}$
beardiness
$\tilde{z}$
$\tilde{z} + \lambda \tilde{z}^{F \rightarrow \bar{F}}$
smiling

Figure 8: Interpolating along semantic directions in disentangled representation space: First four rows show interpolations along *beardiness*, while the last four depict interpolations along *smiling* attribute. Note the change of gender in rows 1,2,4, reflecting the strong correlation of *beard* and *gender* in the original data.

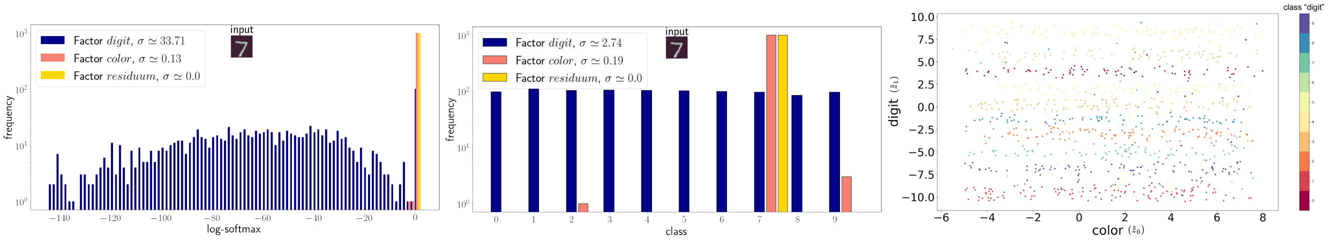

Figure 9: *Left:* Output variance *per class* of a digit classifier on ColorMNIST, assessed via distribution of log-softmaxed logits and class predictions. T disentangles $\tilde{z}_0$ (residual), $\tilde{z}_1$ (digit) and $\tilde{z}_2$ (color). *Right:* 1d disentangled UMAP embeddings of $\tilde{z}_1$ and $\tilde{z}_2$. See Sec. 4.2.

4.2. Interpretation of Classifiers

After interpreting autoencoder architectures we now analyze classification networks: *(i)* A digit classifier on ColorMNIST (accuracy $\sim 97\%$). To interpret this network, we extract hidden representations $z \in \mathbb{R}^{64}$ just before the classification head. *(ii)* A ResNet-50 classifier [12] trained on classes of AnimalFaces. Hidden representations $z \in \mathbb{R}^{2048}$ are extracted after the fully convolutional layers.

Network Response Analysis: We now analyze how class output probabilities change under manipulations in the interpretation space: First, we train the translator T to disentangle K (plus a residual) distinct factors $\tilde{z}_k$. For evaluation we modify a single factor $\tilde{z}_k$ while keeping all others fixed. More precisely, we modify $\tilde{z}_k$ by replacing it with samples drawn from a random walk in a harmonic potential (an Ornstein-Uhlenbeck process, see Sec. B of the supplementary), starting at $\tilde{z}_k$. This yields a sequence of modified factors $(\tilde{z}_k^{(1)}, \tilde{z}_k^{(2)}, \ldots, \tilde{z}_k^{(n)})$ when performing n modification steps. We invert every element in this sequence back to its hidden representation and apply the classifier. We analyze the response of the network to each modified factor k through the distribution of the logits and class predictions.

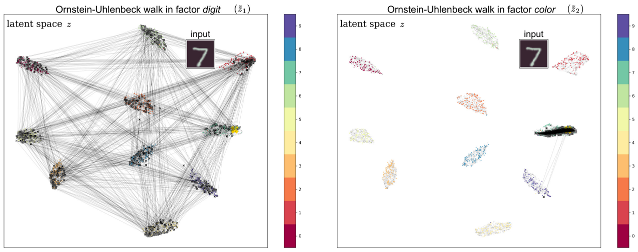

Figure 10: UMAP embedding of a ColorMNIST classifier's latent space $z = E(x)$. Colors of dots represent classes of test examples. We map latent representations z to interpretable representations $\tilde{z} = T(z)$, where we perform a random walk in one of the factors $\tilde{z}_k$. Using T^{-1}, this random walk is mapped back to the latent space and shown as black crosses connected by gray lines. On the left, a random walk in the digit factor jumps between digit clusters, whereas on the right, a random walk in the color factor stays (mostly) within the digit cluster it starts from.

Interpreting Classifiers to Estimate their Invariance: Network interpretation also identifies the invariance properties of a learned representation. Here we evaluate invariances of a digit classifier to color. We learn a translation T to disentangle *digit* $\tilde{z}_1$, *color* $\tilde{z}_2$, and a *residual* $\tilde{z}_0$. Fig. 9 shows the network response analysis. The distribution of $\log$ softmax-values and predicted classes is indeed not sensitive to variations in the factor *color*, but turns out to be quite responsive when altering the *digit* representation. We additionally show a UMAP [35] of the reversed factor manipulations in Fig. 10 (in black). Since the entire modification occurs within one cluster, this underlines that T found a disentangled representation and that the classifier is almost invariant to color. Additionally, we employ another 1D-UMAP dimensionality reduction to each factor seperately and then plot their pair-wise correlation in Fig. 9.

Next, we trained a transformer T to evaluate interpretability in case of the popular ResNet-50. The analysis of three factors, grayscale value $\tilde{z}_1$, roundness $\tilde{z}_2$, and a residual $\tilde{z}_0$ reveals an invariance of the classifier towards grayness but not roundness. More details can be found in Sec. B of the supplementary.

5. Conclusion

We have shown that latent representations of black boxes can be translated to interpretable representations where disentangled factors represent semantic concepts. We presented an approach to perform this translation without loss of information. For arbitrary models, we provide the ability to work with interpretable representations which are equivalent to the ones used internally by the model. We have shown how this provides a better understanding of models and data as seen by a model. Invertibility of our approach enables semantic modifications and we showed how it can be used to obtain state-of-the-art autoencoder-based generative models.

This work has been supported in part by the German federal ministry BMWi within the project "KI Absicherung", the German Research Foundation (DFG) projects 371923335 and 421703927, and a hardware donation from NVIDIA corporation.

References

[1] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 3

[2] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017. 3

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs, 2014. 1

[4] Bin Dai and David Wipf. Diagnosing and enhancing vae models, 2019. 7, 12

[5] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation, 2014. 2

[6] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp, 2016. 2, 11

[7] Patrick Esser, Johannes Haux, and Bjorn Ommer. Unsupervised robust disentangling of latent characteristics for image synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2699–2709, 2019. 2

[8] Patrick Esser, Ekaterina Sutter, and Björn Ommer. A variational u-net for conditional appearance and shape generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8857–8866, 2018. 2

[9] Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. 2, 3

[10] Lore Goetschalckx, Alex Andonian, Aude Oliva, and Phillip Isola. Ganalyze: Toward visual definitions of cognitive image properties. *arXiv preprint arXiv:1906.10112*, 2019. 3

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 3

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 8, 12

[13] Geoffrey E. Hinton, James L. McClelland, and David E. Rumelhart. Distributed representations. 1986. 2

[14] Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. MGAN: Training generative adversarial nets with multiple generators. In *International Conference on Learning Representations*, 2018. 3

[15] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017. 1

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 11

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 11

[18] Jörn-Henrik Jacobsen, Jens Behrmann, Richard Zemel, and Matthias Bethge. Excessive invariance causes adversarial vulnerability, 2018. 2

[19] Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. i-revnet: Deep invertible networks, 2018. 2

[20] Sergey Karayev, Matthew Trentacoste, Helen Han, Aseem Agarwala, Trevor Darrell, Aaron Hertzmann, and Holger Winnemoeller. Recognizing image style. *arXiv preprint arXiv:1311.3715*, 2013. 12

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 12

[22] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018. 2, 4, 11

[23] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[24] Dmytro Kotovenko, Artsiom Sanakoyeu, Sabine Lang, and Bjorn Ommer. Content and style disentanglement for artistic style transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4422–4431, 2019. 2

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 6

[26] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric, 2015. 3, 7

[27] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 6

[28] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. *arXiv preprint arXiv:1905.01723*, 2019. 6

[29] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1096–1104, 2016. 6

[30] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 6

[31] Ziwei Liu, Sijie Yan, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Fashion landmark detection in the wild. *Lecture Notes in Computer Science*, page 229–245, 2016. 6

[32] Dominik Lorenz, Leonard Bereska, Timo Milbich, and Bjorn Ommer. Unsupervised part-based disentangling of object

shape and appearance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10955–10964, 2019. 2

[33] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study, 2017. 7, 11

[34] Liqian Ma, Qianru Sun, Stamatios Georgoulis, Luc Van Gool, Bernt Schiele, and Mario Fritz. Disentangled person image generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 99–108, 2018. 2

[35] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018. 8

[36] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, Feb 2019. 1

[37] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, May 2017. 2, 3

[38] Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks, 2016. 2

[39] Lili Pan, Shen Cheng, Jian Liu, Yazhou Ren, and Zenglin Xu. Latent dirichlet allocation in generative adversarial networks, 2018. 3

[40] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5880–5888, 2019. 5, 12

[41] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models, 2018. 3

[42] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015. 3

[43] Scott E Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1252–1260. Curran Associates, Inc., 2015. 3

[44] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning-Volume 32*, pages II–1278. JMLR. org, 2014. 3

[45] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “why should i trust you?”. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’16*, 2016. 2

[46] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. *arXiv preprint arXiv:1907.10786*, 2019. 3

[47] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 2

[48] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. In *Advances in neural information processing systems*, pages 2553–2561, 2013. 1

[49] Paul Upchurch, Jacob Gardner, Geoff Pleiss, Robert Pless, Noah Snavely, Kavita Bala, and Kilian Weinberger. Deep feature interpolation for image content changes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7064–7073, 2017. 3, 6

[50] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. 6

[51] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 1

[52] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization, 2015. 2

[53] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science*, page 818–833, 2014. 2

[54] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. 2

[55] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016. 2, 3