What do Deep Networks Like to See?

Sebastian Palacio[*] Joachim Folz[*] Jörn Hees Federico Raue Damian Borth Andreas Dengel
German Research Center for Artificial Intelligence (DFKI)
TU Kaiserslautern
first.last@dfki.de

Abstract

We propose a novel way to measure and understand convolutional neural networks by quantifying the amount of input signal they let in. To do this, an autoencoder (AE) was fine-tuned on gradients from a pre-trained classifier with fixed parameters. We compared the reconstructed samples from AEs that were fine-tuned on a set of image classifiers (AlexNet, VGG16, ResNet-50, and Inception v3) and found substantial differences. The AE learns which aspects of the input space to preserve and which ones to ignore, based on the information encoded in the backpropagated gradients. Measuring the changes in accuracy when the signal of one classifier is used by a second one, a relation of total order emerges. This order depends directly on each classifier's input signal but it does not correlate with classification accuracy or network size. Further evidence of this phenomenon is provided by measuring the normalized mutual information between original images and auto-encoded reconstructions from different fine-tuned AEs. These findings break new ground in the area of neural network understanding, opening a new way to reason, debug, and interpret their results. We present four concrete examples in the literature where observations can now be explained in terms of the input signal that a model uses.

1. Introduction

Diagnostics for Deep Neural Networks often rely on measurements taken at the end of the processing pipeline. Pinpointing issues with a network's architecture, learning process, and capacity typically depends on metrics based on the evolution of the loss function or on performance measurements like *top-k accuracy*. For example, to establish the presence of overfitting in a network, divergence of training and validation losses is considered as a good indicator. Under these circumstances, there are general guidelines to follow like early stopping, loss regularization, acquiring more data or reducing the number of parameters in the model [3]. Although these strategies are indeed effective against variance, they do not provide detailed insights on why the network failed to generalize in the first place. To further understand the transformation of input samples into predictions, strategies have been proposed to look at the internal signals of networks [35, 25]. These provide insightful properties, but they serve a descriptive purpose rather than a predictive one.

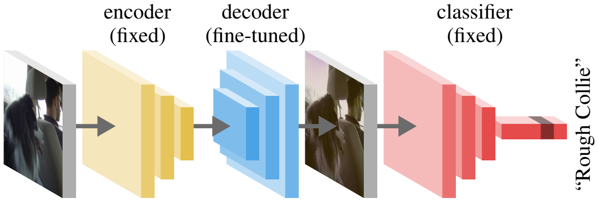


Figure 1. Overview of the proposed method and model. A pretrained AE is fine-tuned with gradients flowing through a pretrained image classifier whose parameters are fixed. After fine-tuning the combined network, images reconstructed by the AE preserve more information required by the classifier.

Recently, Zhang *et al.* [36] pointed out that our current understanding of how networks learn general features is incomplete and requires a different angle. We propose a way to study neural networks based on quantifying the information contained in input samples that classification networks rely on. In general, a deep convolutional neural network (DCNN) can be modeled as a function $y = f(x, \theta)$ where both the properties of the output y and parameters θ have been widely studied. In contrast, we introduce a novel way to examine the properties of the input x and, more precisely, the amount of signal that the network itself takes in from x.

Intuitively a convolution operation with kernel size k and striding s covers the entire input spatially as long as $s \leq k$. In other words, such operations incorporate the complete input space. In this work, we investigate how influential the input space is, not from the perspective of individual samples but rather of the model as a whole. We found that, in practice, information from input samples that image clas-

[*] Authors contributed equally

sifiers use for prediction differs greatly between architectures. These differences were measured by pre-training an AE on a large dataset and then fine-tuning its decoder using gradients from an image classifier with fixed parameters, as shown in Figure 1. Once the AE has been fine-tuned, predictions take place by passing samples through the AE first. The effect of this specialized compression and reconstruction phase is two-fold: First, information that is useful to the network gets preserved while irrelevant parts of the original input are canceled out. Second, AEs fine-tuned in conjunction with classifiers learn to reconstruct the input in a way that attenuates any distracting aspects of the original input, such as noise. Our approach offers the benefit that analysis happens in the input space and hence, any learned transformations by the AE are straightforward to visualize and interpret. We have applied our method to five well-known DCNNs trained for image classification: LeNet5, AlexNet, VGG, Inception v3, and ResNet-50. Analyzing the input signal that these networks take in (characterized by the transformation of their corresponding fine-tuned AE), we found that they rely on different amounts of input signal. Furthermore, this signal may be entirely different between classifiers.

Our main contributions are: First, a model architecture and learning scheme that allows quantification of the input signal used by DCNNs. Second, a relation of order that exists between high performance DCNNs concerning the input signal they use. Third, we present an extensive, comparative evaluation of the input signal used by multiple state-of-the-art DCNNs backed up by well-established measures of information theory such as *Mutual Information* (MI) [27].

2. Related Work

The need for interpretability in AI is growing with new laws and regulations [7] being introduced that govern its application. Given the ever growing body of evidence in favor of the effectiveness of Deep Networks, there is a pressing need for increased understanding how they work. One of the first insights about their properties was that their features were general enough to perform well in different classification tasks [34, 24]. Not only are these features transferable between tasks but they can also be distilled from an ensemble to a single model [9].

A second family of strategies has focused on understanding intermediate elements within networks like activation maps or convolutional filters. Valuable insights came from visualizing said elements [35], but also from inspecting the degree of correlation that network filters share [15].

This kind of intermediate analysis has been extended all the way down to the input domain (otherwise known as *activation maximization*). Results show that not only are intermediate features encoding enough information to reconstruct the original input [17, 5], but that it is also possible to identify areas within the input responsible for high prediction probabilities. These areas can be modeled as a generic subset of the input [38, 21] or as a collection of higher-level features [16, 2, 18]. Moreover, this idea can be refined by explicitly accounting for background and foreground areas [25, 6], target and non-target classes [37], and even for individual pixels [20, 4]. Interestingly, studying the influence of individual pixels on deep image classifiers, gave rise to the research area of adversarial examples [29].

The use of AEs to remap the input space into better suited latent-space has been explored with general-purpose architectures like the variational auto-encoder [11], as well as in specific tasks like noise suppression [31]. Moreover, it has been shown that initializing the weights of a network based on an auto-encoding scheme provides a good starting point for learning another task [19].

Despite all these advances in understanding neural networks, recent work shows that we are still far from having a comprehensive notion about the learned features [36]. This highlights the need for new ways to analyze the capacity of a neural network (*e.g.*, in terms of activation patterns or trajectory length [22]).

3. Methods

This section describes the selection of the AE architecture and training scheme, and analysis performed on the resulting networks. We pre-train a shared base AE and fine-tune it in conjunction with different classification networks to create tailored AEs. We quantify and compare the information contained in images reconstructed by these fine-tuned AEs through (1) relative changes of accuracy when input signals tailored to one classifier is used to measure the accuracy of another and (2) the amount of information that is present in images reconstructed by fine-tuned AEs.

3.1. Autoencoder Selection

For our purposes we need an AE architecture that is capable of reliably capturing a large portion of information contained in input samples. The reconstruction quality is primarily measured through the AE's loss. Additionally, we define a validation measure that depends on the classification accuracy of a pre-trained network. Informally, an AE produces good reconstructions if the accuracy of a pre-trained classifier does not change compared to the accuracy obtained by using the original inputs. We chose SegNet [1] as the architecture for our AE, since it meets all the aforementioned requirements. SegNet is a fully convolutional AE originally designed for semantic segmentation of RGB images. It consists of two VGG-16 [26] networks with batch normalization [10] where the second half of the network has its layers reversed. Max-pooling indices generated during the encoding stage are used to upsample activations

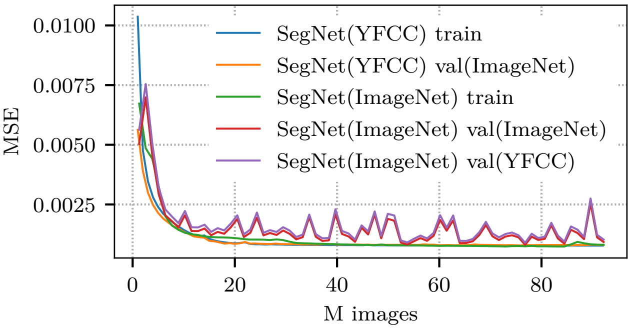


Figure 2. Training and cross-validation mean-square reconstruction error of SegNet network trained on either YFCC100m or ImageNet.

in the decoder. This enables the network to produce pixel-accurate reconstructions even though the smallest activation map is only $\frac{1}{32}$ of the input size.

3.2. Autoencoder Pre-training

As mentioned before, we first train the AE to minimize unsupervised reconstruction loss. Ultimately, we do this so that transformed samples do not significantly change the accuracy of a pre-trained classifier (which will be addressed in Section 3.3). Hence, we train the AE using the YFCC100m [30]. This dataset consists of 100 million media objects taken from Flickr with roughly 99.2 million images. This set presents a comprehensive selection of the kinds of photos taken by a large group of people. It constitutes the largest, publicly available image dataset to date. We remove potential sources of noise in the form of placeholder images (*i.e*., samples that are listed in the dataset but were later removed by the user) and other non-photos (*e.g*., very small file size, single color) for a final count of ≈ 92.1 million training images. Using the YFCC100m as training set comes with some unique advantages. First, the concept of training step or epoch does not apply, since given the scale images do not need to be reused. As shown in Figure 2, the reconstruction error during training, smoothly converges to good local minima before the dataset is used up. Second, learning without using any sample more than once makes the training loss an unbiased estimator of the model's ability to generalize at every step of the process. A separate validation set is no longer required, since new samples have never been seen by the network.

We also compare this first run to the learning behavior of a second SegNet AE using ImageNet [23] instead of YFCC100m. Both AEs were trained under similar conditions: Input size of 256×256 pixels, MSE loss, SGD with momentum 0.9 and initial learning rate of $\eta = 0.01$. When training on the YFCC100m, the reconstruction error is checked every 1 million images and learning rate is reduced to $\eta' = 0.2\eta$ if the loss does not improve after two consecutive checks. For comparability and fairness, when using ImageNet, we let the AE train for 72 epochs which amounts to ≈ 92.2 million images that will be seen by the network during learning. Although necessary for ImageNet but not for YFCC100m, we apply a common set of data augmentation operations (scale, rotation, shift, noise, blur, brightness, contrast, color, mirror) in both cases. We also measured validation in two different ways: For the AE trained on YFCC100m, we use ImageNet's validation set. For the AE trained on ImageNet, we computed a validation scored based on ImageNet's validation set and, additionally, also on a subset of 50000 random samples taken from YFCC100m. All training and validation curves are shown in Figure 2.

At a glance we observe that both learning schemes reach good local minima for both training and validation. However, using YFCC100m produces a faster-converging network which also reaches a consistently well-behaved lower bound for the loss function. In contrast, training on ImageNet yields an asymptotic, unstable learning curve. Validation losses closely followed the training curves with the AE trained on ImageNet and validated on a subset of YFCC100m producing the highest error.

These experiments indicate that initial training using the YFCC100m is optimal for the AE, as it produces consistent, low error image reconstructions that generalize better than using a smaller dataset. Having a well trained AE as a starting point for further training has been recommended [3, 19] and proven to improve convergence.

3.3. Autoencoder Fine-Tuning

Using the AE from Section 3.1, we now fine-tune it with gradients that originate from a pre-trained classification network with fixed parameters. Intuitively, by letting the AE adapt the reconstruction function to produce samples that are likely to enhance classification performance, the AE will be rewarded for keeping the parts of the input that are relevant for inference. Conversely, the fine-tuned AE will disregard any portions of the original input that do not have positive impact on the performance of the classifier. An AE could then learn to represent all the information that is used by a classifier, if the accuracy of the latter does not decrease by using input reconstructions from the former.

Fine-tuning the AE occurs by feeding the reconstructed sample from the AE into a classifier, computing the prediction loss and, backpropagating the gradients all the way down to the input pixels and, further down the AE architecture. Since the target of analysis is the classifier and not the combined AE-classifier network, the parameters of the classifier are not updated by the flowing gradients. Only parameters belonging to the AE architecture get updated once gradients from the classifier start flowing into the AE itself. Incoming gradients can be used to update parameters either on the encoding, the decoding, or both sides of the AE. We found that the best results were obtained by updating only the weights belonging to the decoder.

To test the influence of architectural elements from the classification network in the fine-tuning process, we fine-tune copies of the pre-trained AE on four different classifiers pre-trained on ImageNet, as provided by the torchvision project[1]: AlexNet [12], VGG-16 [26] with batch normalization, Inception v3 [28], and ResNet 50 [8]. We also added a version of LeNet-5 [14] modified to take input images of size 224 × 224 pixels as a simple, lower bound in terms of classification accuracy. These networks have been recognized as high performing models when they were first proposed. Furthermore, they include a series of different structural elements that have played an important role to push the state of the art on image classification *e.g.*, higher depth, batch normalization, inception modules, and residual connections. We measure changes in accuracy when these networks use the unaltered images from the ImageNet validation set and when they use reconstructions from the AE of the same samples. Moreover, we check if the reconstructed samples from the original pre-trained AE already encode all the information that each classifier uses for inference.

For the remainder of this paper, classifiers will be referenced by their first letter (L, A, V, I, and R) where applicable. Furthermore, we will use the shorthand notation $j \circ \mathbb{A}_i$ to indicate that the classifier j is using input samples from an AE that has been fine-tuned with gradients provided by classifier i. Additionally, $\mathbb{A}_S$ will refer to the SegNet AE pre-trained on YFCC100m and, $\mathbb{A}_i(x)$ will refer to a reconstruction of input sample x using $\mathbb{A}_i$. We also define $\mathscr{C} = \{L, A, V, I, R\}$, the set of all classifiers evaluated.

Table 1 shows top-1 and top-5 accuracies for the aforementioned classifiers before and after pre-pending them with their corresponding fine-tuned AE. As described above, we also computed the baseline $i \circ \mathbb{A}_S, i \in \mathscr{C}$ for reference. We see that the baseline lays between 1.2 and 3.7 points below the accuracy of the classifiers alone. Although this is not a dramatic drop, it does indicate that some information has been lost during the reconstruction of input samples. Taking into account that the difference between x and $\mathbb{A}_S(x)$ is quite small, we can infer that the missing signal that is relevant for inference has to be small as well. Notwithstanding, fine-tuning $\mathbb{A}_S$ on any classifier, already makes up for the initial loss of information and, for most cases, even surpasses the performance of the classifiers alone.

Table 1. Center-crop, single-scale accuracies on ImageNet validation set for: original classifier, classifier using reconstructions from $\mathbb{A}_S$ and, classifier using reconstructions from the AE that was fine-tuned in combination with the classifier.

Network	top-1	diff	top-5	diff
L	32.30		54.63	
$L \circ \mathbb{A}_S$	31.08	-1.22	53.01	-1.62
$L \circ \mathbb{A}_L$	34.85	$+2.55$	57.79	$+3.61$
A	54.96		77.98	
$A \circ \mathbb{A}_S$	51.89	-3.07	75.52	-2.46
$A \circ \mathbb{A}_A$	56.13	$+1.17$	78.96	$+0.98$
V	71.35		90.50	
$V \circ \mathbb{A}_S$	67.59	-3.76	87.95	-2.55
$V \circ \mathbb{A}_V$	71.65	$+0.30$	90.55	$+0.05$
R	74.02		92.01	
$R \circ \mathbb{A}_S$	71.19	$-2,83$	90.23	-1.78
$R \circ \mathbb{A}_R$	74.94	$+0.92$	92.27	$+0.26$
I	77.12		93.25	
$I \circ \mathbb{A}_S$	74.42	-2.70	91.87	-1.38
$I \circ \mathbb{A}_I$	76.71	-0.41	93.03	-0.22

3.3.1 Encoded Representations of Fine-Tuned AEs

Once we obtain all $\mathbb{A}_i, i \in \mathscr{C} \cup \{S\}$ by fine-tuning, we can have a look at the reconstructed images. Figure 3 shows an original image and the corresponding reconstructions by each $\mathbb{A}_i$. Below each reconstructed image, we visualize channel-wise histograms in L*a*b* color space. These histograms allow us to identify global changes in perceived brightness and opposing colors. The L* channel represents the luminance, a* encodes color changes between green and magenta and, b* encodes changes between blue and yellow. We chose this colorspace because it relates closely to how human vision works. It neatly separates perceived brightness from color and does not suffer from range singularities like HSV. Distances between colors are approximately linear with respect to perception.

The observed transformations are unique for each classifier and are easily identified as large changes in all three channels when compared to the original image. Changes introduced by $\mathbb{A}_V$ are the smallest among all AEs. $\mathbb{A}_A$ and $\mathbb{A}_R$ introduce consistent checkerboard artifacts over the entire input space. These artifacts appear as peaks in the histograms in channels a* and b*. $\mathbb{A}_R$ and $\mathbb{A}_I$ compress the range of the luminance, making dark areas brighter and bright areas darker. $\mathbb{A}_I$ introduces a strong shift in b* values that manifest as a lack of blues and a yellow/brown tint. $\mathbb{A}_A$ produces images with a distinct pink hue.

3.3.2 Emergent Resilience to Noise

We noticed that a consequence of fine-tuning AEs, as proposed in this work, is that their reconstructions are preserving more information under noise than the original inputs. We tracked the top-1 accuracy of the classifiers in $\mathscr{C}$ on ImageNet's validation set under an increasing additive uniform noise drawn from $\mathbb{U}(-s, s)$ with strength $s \in [0, 1]$. We com-

[1] `https://github.com/pytorch/vision`, commit 10a387a

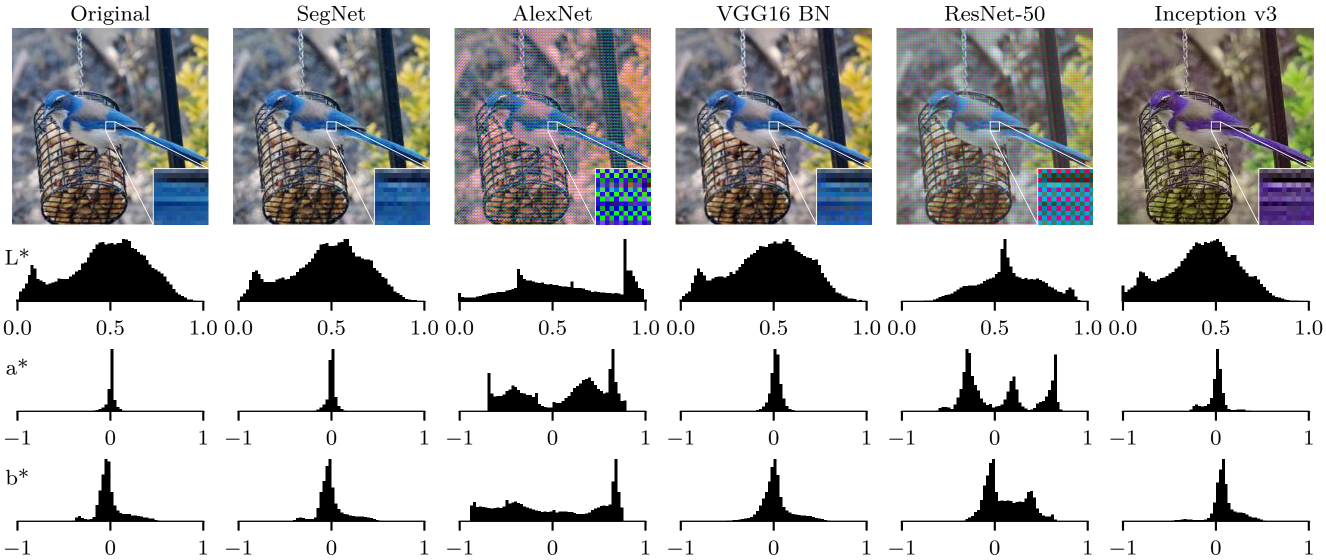


Figure 3. Example taken from ImageNet's validation set. Each image corresponds to a reconstruction produced by one of the fine-tuned AEs. Below each reconstruction, global histograms of the image in L*a*b* colorspace are shown. Zoomed portions of the image reconstructions are provided, showing the emergent patterns in more detail.

pared the results to a similar setup where the accuracy was computed using reconstructed samples from $\mathbb{A}_{i\in\mathscr{C}\cup\{S\}}$.

For the first experiment (dashed lines in Figure 4) accuracy decays steadily as noise increases. Using reconstructions from $\mathbb{A}_S$ (Figure 4; left) worsens performance even more, suggesting that the AE itself has not learned a denoising transformation. Note that these results are aligned with initial findings presented in Section 3.3.1 where classification with reconstructed samples of $\mathbb{A}_S$ alone were already lower than the corresponding baseline. In contrast, reconstructions from fine-tuned AEs are consistently more resilient to noise, even when compared to the stand-alone classifiers (Figure 4; right). Accuracy for classifiers of the form $i \circ \mathbb{A}_i$ present higher accuracies for all levels of noise, indicating that all $\mathbb{A}_{i\in\mathscr{C}}$ have indeed learned a representation of the information that is useful to the classifier it was fine-tuned on. The most pronounced difference can be seen with AlexNet, which falls below the accuracy of LeNet-5 at $s = 0.3$, yet it manages to remain above LeNet-5 when using inputs from $\mathbb{A}_A$. This measurable resilience to noise is also visually perceptible as shown in Figure 5.

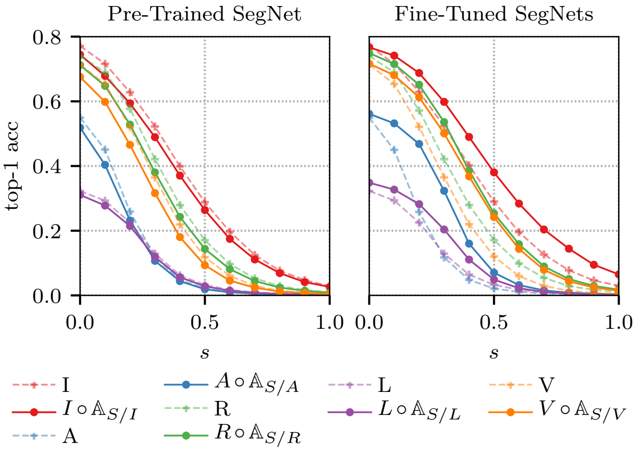


Figure 4. Top-1 accuracy on ImageNet's validation set under increasing noise strength s. Dashed lines correspond to networks evaluated on the original inputs. Solid lines depict evaluation of the networks using reconstructed inputs from AEs. Left: reconstructions from $\mathbb{A}_S$. Right: reconstructions from $\mathbb{A}_i$, $i \in \mathscr{C}$.

3.4. Measuring Information through Classifiers

We explored the relationship between different encodings defined by fine-tuned AEs. By observing the wide visual differences between image reconstructions from all $\mathbb{A}_i$, $i \in \mathscr{C}$, it is clear that each classifier prefers different re-

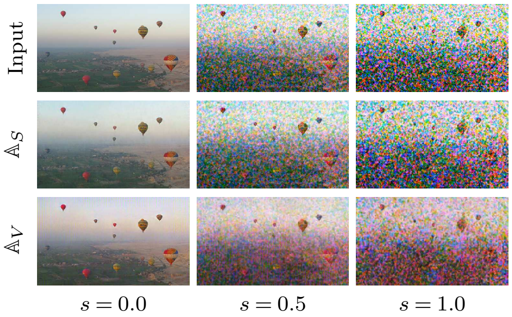


Figure 5. Example for additive noise used to evaluate performance in noisy environments for different values of the strength parameter s. Top is the original input image. Middle and bottom rows are reconstructions of $\mathbb{A}_S$ and $\mathbb{A}_V$.

Table 2. Cross-validation accuracies on ImageNet validation set for classifiers with AEs fine-tuned on different models.

	L	A	V	I	R
$\mathbb{A}_L$	0.3484	0.3077	0.0416	0.4352	0.4730
$\mathbb{A}_A$	0.0211	0.5613	0.0097	0.5375	0.0925
$\mathbb{A}_V$	0.2929	0.5362	0.7163	0.7400	0.7300
$\mathbb{A}_I$	0.1829	0.3024	0.4555	0.7671	0.4540
$\mathbb{A}_R$	0.0163	0.4972	0.0710	0.7249	0.7494

constructions and thus, different information. Hence, we measured changes in accuracy for each classifier when using input reconstructions from AEs that were fine-tuned on other classifiers. More formally, we evaluate the accuracy of $j \circ \mathbb{A}_i, \forall i, j \in \mathscr{C}$. Results are summarized in Table 2.

We observed that accuracy drops consistently for each classifier when they use other AEs. However, some combinations of $j \circ \mathbb{A}_i$ tend to preserve the accuracy of the lowest performing model in that combination *i.e.*, $\exists_{i \neq j}$: $acc(j \circ \mathbb{A}_i) \approx min(acc(i), acc(j))$. Informally, this effect can be interpreted as networks that use *at least the signal that the lowest performing model uses*. To quantify this, we define the *relative rate of change (RRC)* of an AE-classifier pair as follows: $RRC(j \circ \mathbb{A}_i) = \frac{acc(j \circ \mathbb{A}_i)}{m(i,j)}$ where $m(i,j) = min(acc(i), acc(j))$. Computing RRC values on the cross-validation experiment reveals which combinations of AEs and classifiers preserve more signal, as shown in Figure 6. From this curve, we see how the input signal used by VGG is enough to make both ResNet 50 and Inception v3 perform better than VGG itself. Additionally, the signal from Inception v3 seems to be quite different from the one used by any other model, as none of the models in $\mathscr{C} \setminus \{I\}$ performed well (*i.e.*, below 0.65 of their original accuracy) using $\mathbb{A}_I$.

To further examine the relation between AEs and classifiers, we use formal concept analysis (FCA) [32] to derive a hierarchical ontology between all possible combinations of them. FCA is of particular interest here because it allows us to model partially order sets (under the inclusion operation $\subseteq$). To this end, we need to define a *formal context* $\mathbf{K} = (\mathbf{G}, \mathbf{M}, \mathbf{I})$ where $\mathbf{G}$ is a set of *objects*, $\mathbf{M}$ is a set of *attributes* and $\mathbf{I}$ is a binary relation between elements of $\mathbf{G} \times \mathbf{M}$ that expresses whether $\mathbf{G}$ has the attribute $\mathbf{M}$ or not. Formal concepts of $\mathbf{K}$ are object-attribute subset pairs (S_A, S_B) such that $S'_A = S_B$ and $S'_B = S_A$, where $S'_A = \{m \in \mathbf{M} | \forall_{g \in \mathbf{G}} g\mathbf{I}m\}$ and $S'_B = \{g \in \mathbf{G} | \forall_{m \in \mathbf{M}} g\mathbf{I}m\}$. The lattice of formal concepts for $\mathbf{K}$ is constructed by ordering pairs of formal concepts $(S_A, S_B), (S_C, S_D)$ under the operation $\leq$: $(S_A, S_B) \leq (S_C, S_D) \leftrightarrow S_A \subseteq S_C$. Please refer to [33] for a more intuitive introduction on FCA. Let $\mathbf{G} = \mathscr{C}$ and $M = \{\mathbb{A}_{i \in \mathscr{C}}\}$. Finally, let $\mathbf{I} = \{(i, j) : RRC(j \circ \mathbb{A}_i) \geq t\}$ for a given threshold t. In other words, we convert the table of RRC values into a binary relationship between AEs and classifiers by applying a threshold to it. We generate lattices for the FCs at thresholds $t \in \{0.1, 0.2, 0.8, 0.9\}$, shown in Figure 7. For any two nodes connected by an edge, the upstream connection can be interpreted as *the signal encoded by* $\mathbb{A}_i$ *is used by classifier* j. Looking at $t \in \{0.1, 0.9\}$ gives an idea of the most sensitive and robust changes in signal behavior since they are close to the upper and lower bound in the range of RRCs. Similarly, FCs for $t \in \{0.2, 0.8\}$ characterize the largest changes in signal intake (*i.e.*, they lay in between the largest gaps) among classifiers. Note that any value of t between those intervals ($[0.17 - 0.52]$ and $[0.66 - 0.84]$) yield the same FC, thus the same lattice. This is important for establishing more precise lower and upper bounds between signals and classifiers later on.

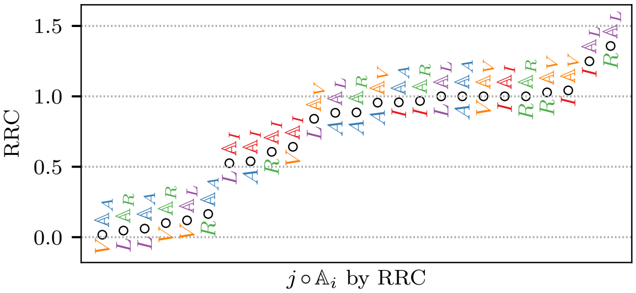
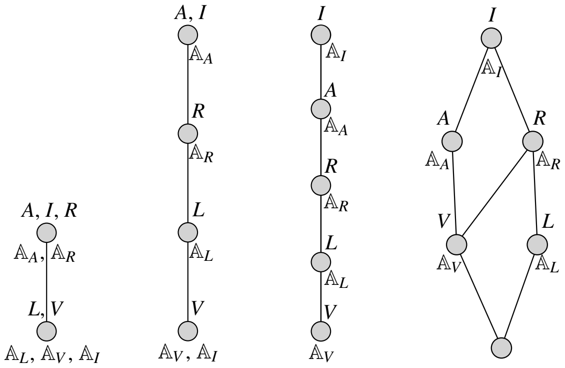


Figure 6. Every tested $j \circ \mathbb{A}_i$ combination, sorted by RRC. Parts are color coded according to their associated classifier: L purple, A blue, V orange, I red, R green.



(a) $t = 0.1$ (b) $t = 0.2$ (c) $t = 0.8$ (d) $t = 0.9$

Figure 7. FC lattices for different RRC thresholds. Attributes (signal) are always below the nodes, objects (classifiers) are on top.

Looking at the first lattice in Figure 7a, we can see that the signal from ResNet 50's AE is already not enough to make VGG or LeNet reach an accuracy that can go above 10% of what either of them achieves on their own, using their corresponding AE. This is especially surprising for VGG, considering how similar its performance is compared to ResNet 50. Conversely, in Figure 7d we see how the signal of VGG is enough for making ResNet 50's performance at least 90% of what VGG–the lowest performing of the

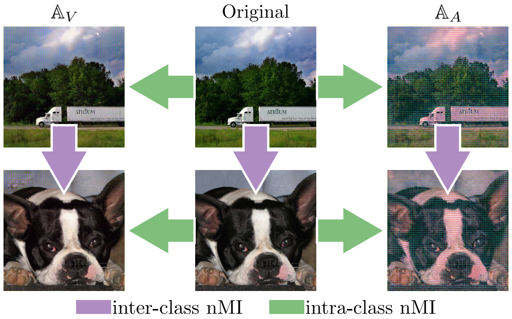


Figure 8. Difference between intra-class nMI and inter-class nMI. The former measures information between the original sample and reconstructions from different AEs. The latter measures reconstructions of different samples using the same AE.

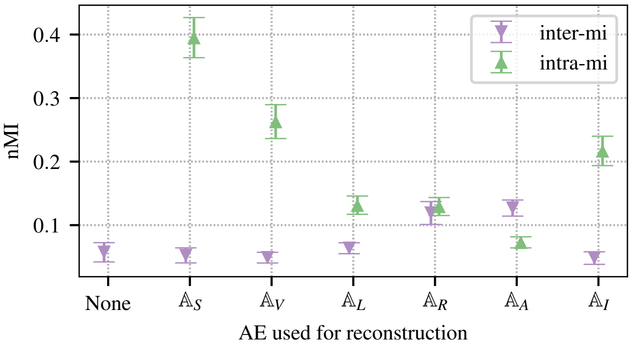


Figure 9. Normalized mutual information for input samples reconstructed from different AEs. *Intra-class* nMI measures information between reconstructions of the same sample

two–is originally capable of. For Inception, we see that it shares up to 50% of the signal that the other classifiers are picking up on. However, the remaining half of the signal used by all other networks appears to be completely different from the one Inception uses to achieve its high performance. As a general trend, VGG stands out as the classifier making the most exhaustive use of the input signal. The AE fine-tuned on VGG preserves a signal that enables all other classifiers retain at least 80% of their performance.

This analysis shows that all networks extract features based on a common portion of the input signal, but said portion can be as small as 10%, as pointed out earlier for ResNet 50. Note how any FC for thresholds between 0.2 and 0.8 yield lattices describing a totally ordered set. The only changing element is the AE fine-tuned on Inception. Such a total order exposes an unexplored aspect that networks are sensitive to, namely that DCNNs are only extracting features from a reduced portion of the input signal. We interpret this hierarchy as the amount of general or specialized signal used by a network.

3.5. Measuring Information through Image Reconstructions

We validate the pattern found in the cross-validation experiments from Section 3.4 by measuring loss and preservation of information between input reconstructions. We use the normalized mutual information (nMI) [27] measure to calculate bounded values reflecting relative changes in the information that is preserved or lost when input samples are passed through each AE. There are two complementary cases to be considered, as shown in Figure 8:

Intra-class nMI: compare reconstructions from each fine-tuned AE to the original sample. Reconstructed samples should preserve as much information as possible from the original image and hence, a high correspondence is expected. To compute the intra-class nMI, an input sample is passed through all AEs and the nMI is computed with respect to the original sample. In other words, $\texttt{intra-nMI}(x, \mathbb{A}_i) = \texttt{nMI}(x, \mathbb{A}_i(x))$. For each AE, the average nMI and the standard deviation of all samples in the validation set of ImageNet (50000 samples) are reported.

Inter-class nMI: compare reconstructions of two different samples using a single fine-tuned AE. Reconstructed samples of two independent images should yield low nMI values. Therefore, a low correspondence is expected. To compute the inter-class nMI, two samples are drawn at random, passed through the same AE, and the nMI is calculated between those two reconstructions. More formally, $\texttt{inter-nMI}(x_1, x_2, \mathbb{A}_i) = \texttt{nMI}(\mathbb{A}_i(x_1), \mathbb{A}_i(x_2))$. For each AE, the average nMI and the standard deviation of consecutive sample pairs over the entire set of ImageNet (25000 pairs) are reported.

Results in Figure 9 show that $\mathbb{A}_S$ is able to preserve the highest amount of the input signal. Moreover, there is a well-defined order with respect to classifiers whose fine-tuned AEs preserve more information. Regardless of the variant of nMI used for comparison, all fine-tuned AEs can be sorted as follows: $\mathbb{A}_V > \mathbb{A}_I > \mathbb{A}_L > \mathbb{A}_R > \mathbb{A}_A$. Note that both $\mathbb{A}_A$ and $\mathbb{A}_R$ show comparatively high amounts of inter-class nMI, which is consistent with earlier observations of highly regular patterns in their reconstructions as described in Section 3.3.1. The resulting totally ordered set aligns closely to the attributes in the lattice of Figure 7b which also describes a totally ordered set.

This analysis provides further evidence of the existence of a pattern, where different networks are using of more or less information from the original input. Furthermore, considering that classification accuracy dropped consistently for all classifiers when the original AE was used, we can infer that the lost information, though irrelevant in most cases, is indeed used by all classifiers. The nature of information reconstructed by $\mathbb{A}_S$ and its fine-tuned versions suggests that classifiers are using parts of the inputs that are less relevant for accurate reconstruction.

3.6. Understanding Previous Work based on Signal

We have so far been able to quantify the amount of signal different classifiers use for their predictions. Another way of interpreting our findings resulted in lattices that describe (partial) orders of classifiers according to (non-) overlapping parts of the input signal they process. These results coincide with several prior publications on understanding the operations performed by deep neural network. Zeiler *et al*. [35] discovered that AlexNet is highly sensitive to local structures. This can be understood in part as a consequence of the reduced input signal that AlexNet uses. Raghu *et al*. [22] showed that parameters on shallower layers have a higher impact on the final prediction than deeper layers. Shallower layers are closer and hence more exposed to the entire signal and can therefore drop more of it. Modifying shallow layers alters the amount and the kind of signal that goes into the network, affecting prediction scores significantly. The work of Montavon *et al*. [20] shows relevance reconstruction maps that are coarser for CaffeNet than for GoogleNet. This phenomenon is consistent with other work [13] and can now be understood from the point of view of the input signal. In one of their latest experiments, Bau *et al*. [2] trained a variant of AlexNet with wider layers and global average pooling to explore its interpretability. Despite all those changes, the accuracy was similar to the original architecture even after increasing the number of filters of the last convolutional layer by a factor of 4 and 8. They suggest that the capacity of the network has been exhausted although, by definition, more filters indicate a higher capacity. We propose a complementary idea: adding more filters in deeper layers do not affect the performance not because the capacity of the model is exhausted, but because the input signal is. In other words, deeper layers already interpret all the signal that is available to them.

4. Discussion & Future Work

In this work, we introduced a novel, alternative way to understand the behavior of deep neural networks by studying the reconstruction performed by autoencoders that were fine-tuned to suit their needs. This setup allowed us to analyze the amount of signal that a network uses before it enters the model itself. Our approach is fundamentally different from previous work since we did not focus on measuring the behavior of intermediate or end results (*e.g*., through bias-variance metrics or activation maximization analysis).

We propose a training scheme for the autoencoder that ensures excellent generalization and reproducible results by using the YFCC100m as dataset for pre-training. Furthermore, we use the resulting model as basis for further fine-tuning of the decoder with gradients from different pre-trained image classifiers. By looking at the response of classifiers when different auto-encoded images are fed, we were able to establish a relation of order between classifiers that depends on the input signal. We presented evidence of this underlying pattern by using formal concept analysis and validate our findings by measuring the information contained in the different image reconstructions.

Additional Findings: There are some further observations that spawn from our proposed method that we like to highlight.

The two-stage training strategy for autoencoders has influenced these networks to learn denoising operations. Said function is effective because it favors the preservation of parts of the input signal that are used by the classifier, increasing the overall tolerance to noise within individual samples.

The amount of signal used by most classifiers is small compared to the amount of signal that is available from the input. High performing image classifiers like ResNet can use as little or less than 10% of the original input. This can be seen as a beneficial property, as less evidence is required to make a correct prediction. Possible downside of this reliance on little evidence is that small changes to relevant parts of the input, also known as adversarial examples, can change the prediction. Classifiers that take advantage of redundant information are more robust to changes that were unaccounted for during training. Note also that the amount of input signal does not correlate with the number of parameters or performance.

However, the relevant portion of the input signal does follow a general, distributed pattern as seen by the global checkerboard artifacts that appear in reconstructions produced by some AEs. In other words, these patterns do not depend on the specific content of each image, but rather to general patterns (*e.g*. darken bright areas, increased distance between colors).

Future Work: As next steps, we want to compare the denoising properties of fine-tuned autoencoders with other architectures and training strategies that are explicitly designed to denoise. Moreover, we would like to explore alternatives to SegNet as autoencoder that are more efficient and light-weight. Finally, we are not yet able to point to specific points in the architecture of a deep network that are responsible for losing signal. A better understanding of how the flow of signal through a network can be controlled will allow for a more principled approach to design future architectures.

Acknowledgments: This work was supported by the BMBF project DeFuseNN (Grant 01IW17002) and the NVIDIA AI Lab (NVAIL) program. We thank all members of the Deep Learning Competence Center at the DFKI for their comments and support.

References

[1] V. Badrinarayanan, A. Handa, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv preprint arXiv:1505.07293*, 2015.

[2] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017.

[3] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

[4] A. Binder, S. Bach, G. Montavon, K.-R. Müller, and W. Samek. Layer-wise relevance propagation for deep neural network architectures. In *Information Science and Applications (ICISA) 2016*, pages 913–922. Springer, 2016.

[5] A. Dosovitskiy and T. Brox. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4829–4837, 2016.

[6] R. C. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[7] B. Goodman and S. Flaxman. European Union regulations on algorithmic decision-making and a "right to explanation". *ArXiv e-prints*, June 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[9] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[10] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

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

[12] A. Krizhevsky. One weird trick for parallelizing convolutional neural networks. *CoRR*, abs/1404.5997, 2014.

[13] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Muller, and W. Samek. Analyzing classifiers: Fisher vectors and deep neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[15] Y. Li, J. Yosinski, J. Clune, H. Lipson, and J. Hopcroft. Convergent learning: Do different neural networks learn the same representations? In *Feature Extraction: Modern Questions and Challenges*, pages 196–212, 2015.

[16] J. L. Long, N. Zhang, and T. Darrell. Do convnets learn correspondence? In *Advances in Neural Information Processing Systems*, pages 1601–1609, 2014.

[17] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.

[18] A. Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120(3):233–255, Dec 2016.

[19] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. *Artificial Neural Networks and Machine Learning–ICANN 2011*, pages 52–59, 2011.

[20] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[21] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in Neural Information Processing Systems*, pages 3387–3395, 2016.

[22] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein. On the expressive power of deep neural networks. *arXiv preprint arXiv:1606.05336*, 2016.

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[24] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

[25] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[27] A. Strehl and J. Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002.

[28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[29] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[30] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L. Li. The new data and new challenges in multimedia research. *CoRR*, abs/1503.01817, 2015.

[31] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

[32] R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In *Ordered sets*, pages 445–470. Springer, 1982.

[33] K. E. Wolff. A first course in formal concept analysis. *StatSoft*, 93:429–438, 1993.

[34] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[35] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[36] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[37] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, Dec 2017.

[38] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.