Explaining Neural Networks by Decoding Layer Activations

Johannes Schneider
Institute of Information Systems,
University of Liechtenstein
Vaduz,Liechtenstein
johannes.schneider@uni.li

Michalis Vlachos
Department of Information Systems
University of Lausanne
Lausanne, Switzerland

***Abstract*—To derive explanations for deep learning models, ie. classifiers, we propose a 'CLAssifier-DECoder' architecture (*ClaDec*). *ClaDec* allows to explain the output of an arbitrary layer. To this end, it uses a decoder that transforms the non-interpretable representation of the given layer to a representation that is more similar to training data. One can recognize what information a layer maintains by contrasting reconstructed images of *ClaDec* with those of a conventional auto-encoder(AE) serving as reference. Our extended version also allows to trade human interpretability and fidelity to customize explanations to individual needs. We evaluate our approach for image classification using CNNs. In alignment with our theoretical motivation, the qualitative evaluation highlights that reconstructed images (of the network to be explained) tend to replace specific objects with more generic object templates and provide smoother reconstructions. We also show quantitatively that reconstructed visualizations using encodings from a classifier do capture more relevant information for classification than conventional AEs despite the fact that the latter contain more information on the original input.**

I. Introduction

Tacit or implicit knowledge, refers to the knowledge that explains how to perform a task. Such knowledge is difficult to transfer not only among humans but also between humans and machine learning models. Explaining models is important for many reasons, including: a) debugging or improving models, b) fulfilling legal obligations such as the "right to explain" as crystallized in the European GDPR data privacy law, c) increasing trust in models. Thus, it is not surprising that explaining neural networks has received a lot of attention [1], [2]. Explaining (and understanding) a neural network is a multi-faceted problem, ranging from understanding single decisions, single neurons, single layers onto understanding entire models. Often, methods touch on multiple of these aspects. In this work, we are primarily interested in explaining a decision with respect to a user-defined layer that originates from a complex feature hierarchy, as commonly found in a deep learning model. In a layered model, each layer corresponds to a transformed representation of the original input. Thus, the neural network succinctly transforms the input into representations that are more useful for the task at hand, eg. classification. From this point of view, we seek to answer the question: "Given an input X, what does the representation $L(X)$ produced in a layer L tell us about the decision and the network?"

To address this question, we propose a classifier-decoder architecture called *ClaDec*. It transforms the representation $L(X)$ produced by a layer L of the classifier to be explained into a human understandable representation, ie. one that is similar to the input domain, using a decoder. The layer in question provides the "code" that is fed into a decoder. The motivation for this architecture stems from the observation that auto-encoder (AE) architectures are good at (re)constructing (high-dimensional) data from a (low-dimensional) representation. The idea is that the classifier to be explained should encode aspects well that are relevant for classification and ignore information in inputs that do not impact decisions. Therefore, the decoder should be able to reconstruct parts and attributes of the inputs well, that are essential for classification and others that have no influence should not be reconstructed well. Attributes of inputs might refer to basic properties such as color, shape, sharpness but also more abstract, higher level concepts. That is, reconstructions of higher level constructs might be altered to be more similar to prototypical instances. We provide a theoretically founded motivation later.

Explanations should fulfill many partially conflicting objectives, potentially even depending on the individual receiving an explanation [2]. In particular, we are interested in the trade-off between fidelity (How accurately does the explanation express the model behavior?) and interpretability (How easy is it to make sense of the explanation?). While these properties of explanations are well-known, existing methods typically do not accommodate adjusting this trade-off. In contrast, we propose an extension of our base architecture *ClaDec* by adding a classification loss. It allows the balancing between producing reconstructions that are similar to the inputs, ie. training data that a user is probably more familiar with (easier interpretation), and reconstructions that are strongly influenced by the model to explain (higher fidelity) and likely more difficult to understand.

Our approach relies on an auxiliary model, ie. a decoder, to provide explanations. Like other methods that use auxiliary or proxy models, eg. to synthesize inputs [3] or approximate model behavior [4], we face the problem that explanation fidelity may be negatively impacted by a poor auxiliary model.

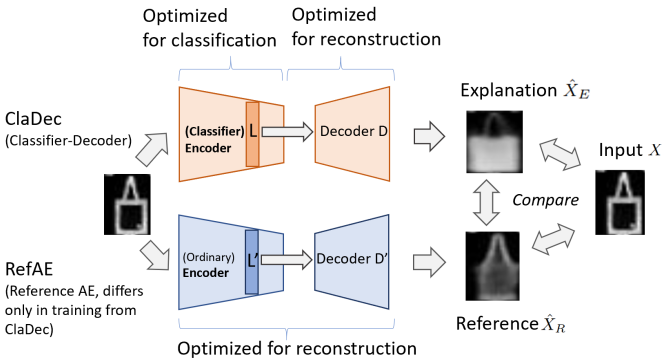

Fig. 1. Basic architecture of *ClaDec* and *RefAE* as well as illustrating the explanation process

That is, reconstructions of AEs (or GANs) might suffer from artifacts. For example, auto-encoders are known to produce images might appear more blurry than real images. GANs might produce crispier images but might suffer from other artifacts as shown in samples in [3]. Since evaluation of explainability methods still bears many open questions [5], it is no surprise that this problem has not been addressed to date, although it might have adverse impacts on understandability and even lead to wrong conclusions on model behavior. When looking at the reconstruction, a person not familiar with such artifacts might not attribute the distortion to the auxiliary model being used but she might believe that it is due to the model to be explained. To avoid any wrongful perceptions with respect to artifacts in reconstruction, we suggest to compare outcomes of auxiliary models to a reference architecture. We employ an auto-encoder *RefAE* with the exact same architecture as *ClaDec* to generate outputs for comparison as shown in Figure 1. The encoder of *RefAE* is not trained for classification, but the *RefAE* model optimizes the reconstruction loss of the original inputs as any conventional AE. Therefore, only differences in visualization that are visible in the reconstructions of *RefAE* and *ClaDec* can be attributed to the model to be explained. While our work focuses on explaining classifiers, we also briefly discuss how to explain an AE architecture itself, ie. the encoder in AE architecture.

The proposed comparison to a reference model might also be seen as a rudimentary sanity check, ie. if there are no differences then either the explanability method is of little value or the objective of the model to be explained is similar to that of the reference AE, as we shall elaborate more in our theoretical motivation. We believe that such sanity checks are urgently needed, since multiple explanation methods have been scrutinized for failing "sanity" checks and simple robustness properties [6]–[8]. For that reason, we also introduce a sanity check that formalizes the idea that inputs plus explanations should lead to better performance on downstream tasks than inputs alone. In our context, we even show that auxiliary classifiers trained on reconstructions from the reference AE *RefAE* and *ClaDec* perform better on the latter, although the reference AE leads to reconstructions that are closer to the original inputs. Thus, the reconstructions of *ClaDec* are more amendable for the task to be solved despite containing less information on the inputs.

Overall, we make the following contributions:

- Proposing a novel, theoretically grounded method to understand layers of a deep learning model. It relies on the idea to train a decoder to translate a (non-interpretable) layer outputs into a human understandable representation. It also allows to trade interpretability and fidelity.
- Introducing the idea to deal with artifacts created by auxiliary models (or proxies) through comparisons with adequate references.
- Adding to existing work on evaluation of explainability methods by formalizing the evaluation of different objectives of explanations, ie. fidelity and interpretability.

II. Theoretical Motivation of the Classifier-Decoder (*ClaDec*)

In this section we provide rational for our approach shown in Figure 1. That is, reconstructing explanations using a decoder from a layer of a classifier that should be explained, and comparing it to the output of a conventional AE, ie. *RefAE*. Auto-encoders perform a transformation of inputs to a latent space and then back to the original space. This comes with some information loss on the original inputs, because reconstructions are typically not identical to inputs. It may appear that this information loss is due to forcing high-dimensional data to be represented in a low dimensional space. However, as claimed in [9](p.505), a non-linear encoder and decoder (theoretically) only require a single dimension to encode arbitrary information without any loss. The deeper mathematical reason is that a dimension d is a real number, ie. $d \in \mathbb{R}$ and real numbers are uncountable infinite. Thus, there are (more than) enough options to encode an infinite amount of inputs.To provide some intuition, we focus on a simple architecture with a linear encoder (consisting of a linear model that should be explained), a single hidden unit and a linear decoder as depicted in Figure 2. An auto-encoder, ie. the reference AE *RefAE*, aims to find an encoding vector E and a reconstruction vector R, so that the reconstruction $R \cdot y$ of the encoding $y = E \cdot x$ is minimal using the L2-loss, ie.

$$\min_{R,E} ||x - R \cdot E \cdot x||^2$$

[10] showed that the optimal solution which minimizes the reconstruction loss stems from projecting onto the eigenvector space (as given by a Principal Component Analysis). That is, the optimal solution for $W = R \cdot E$ given there is just a single latent variable consists of the first eigenvector u_1. This is illustrated in Figure 2 in the upper part with $y = u_1 \cdot x$. Next, we discuss the *ClaDec* architecture, where the goal is to

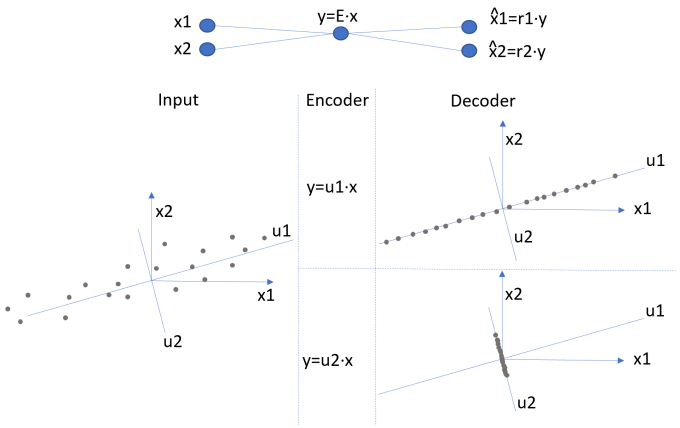

Fig. 2. Simple model: An AE with optimal encoder $y = u_1 \cdot x$ (and decoder) captures most information on the input. But an encoder (based on a regression/classification model, eg. $y = u_2 \cdot x$) combined with an optimized decoder, might capture some attributes of the input more accurately, eg. x_2.

explain a linear regression model $y = E \cdot x$.[1] The vector E is found by solving a regression problem. We fit the decoder R to minimize the reconstruction loss on the original inputs given the encoding, ie. $\min_R ||x - R \cdot y||^2$ with $y = E \cdot x$. Clearly, the more similar the regression problem is to the encoding problem of an AE, the more similar are the reconstructions. Put differently, the closer E is to u_1 the lower the reconstruction loss and the more similar are the optimal reconstructions for the reference AE and *ClaDec*. Assume that E differs strongly from u_1, ie. say that the optimal solution to the regression problem is the second eigenvector $y = u_2 \cdot x$. This is shown in the lower part of Figure 2. When comparing the optimal reconstruction of the *RefAE*, ie. using $y = u_1 x$, and the illustrated reconstruction of *ClaDec*, ie. using $y = u_2 x$, it becomes apparent that for the optimal encoding $y = u_1 x$ the reconstructions of both coordinates x_1 and x_2 are fairly accurate on average. In contrast, using $y = u_2 x$, coordinate x_2 is reconstructed more accurately (on average), whereas the reconstruction of x_1 is generally very poor.

In a more general context, this suggests that a representation obtained from a model (trained for some machine learning task) may capture some aspects of inputs in more or equal detail as an encoder optimized towards reconstructions of inputs. However, overall it will capture less information on inputs. Thus, for reconstructions from the classifier-decoder *ClaDec* we expect that they are "worse" overall in terms of similarity to the original input. However, for attributes relevant to classification, they should bear more similarity to inputs than for attributes that are irrelevant.

[1]From a practical point of view, there is limited value in explaining a linear regression model with few variables, since linear regression models are transparent. However, for more complex (linear) models involving many, potentially transformed input attributes x_i such as $\sin(x_i)$, x_i^2, explanations might still be helpful. Furthermore, linear regression exhibits nice properties for theoretical analysis and it is widely used.

III. Method and Architecture

In Figure 1 the top part shows the *ClaDec* architecture. It consists of an encoder and a decoder reconstructing the input. The encoder consists of all layers of a classifier up to a (user-specified) layer L. The entire classifier has been trained independently to optimize classification loss before the decoder. To explain layer L of the classifier for an input X, we use the activations of layer $L(X)$. The activations $L(X)$ are provided to the decoder. The decoder is trained while keeping the encoder, ie. classifier, fixed. It optimizes the reconstruction loss with respect to the original inputs X. The reference AE *RefAE* is identical to *ClaDec*. It differs only in the training process and the objective. For the reference AE, the encoder and decoder are trained jointly to optimize the reconstruction loss (of inputs X). In contrast, the encoder is treated as fixed in *ClaDec*. Once the training of all components is completed, explanations can be generated without further need for optimization. That is, for an input X, we compute the reconstruction $\hat{X}_E$ serving as explanation using *ClaDec*. We compute first the activation of the layer L, which serves as input to get the reconstruction from the decoder. However, comparing the reconstruction $\hat{X}_E$ to the input X might be difficult and even misleading, since the decoder can introduce distortions. Image reconstruction in general by AE or GANs is not perfect. Quality depends on the availability of training data, the chosen model and the computational effort (amount of training). Therefore, it is unclear, whether differences between the input and the reconstruction originate from the encoding of the classifier or the inherent limitations of the decoder. Thus, we propose to use both the *RefAE* (capturing unavoidable limitations of the model or data) and *ClaDec* (capturing model behavior). The evaluation proceeds by comparing the reconstructed "reference" from *RefAE*, the explanation from *ClaDec* and the input. Only differences between the input and the reconstruction of *ClaDec* that do not occur in the reconstruction of the reference can be attributed without doubt to the classifier. While our method is built to provide explanations for single examples, it is easy to derive generalizations from multiple explanations that hold for an entire class or capture general model behavior. While in principle, inductive reasoning to understand an entire model based on explanations of samples can be used for any method, we argue that our explanations are more amendable to do this because our explanations are easy to interpret, which is a consequence of both the reconstruction process and the idea to highlight differences to the reference model. We shall provide evidence for this claim highlighting general model behavior based on samples in the evaluation section.

An extension of the base architecture of *ClaDec* (Figure 1) using a second loss term for the decoder training is shown in Figure 3. It is motivated by the fact that *ClaDec* seems to yield reconstructions that capture more aspects of the input domain than of the classifier. That is, reconstructions might be easy to interpret, but in some cases it might be preferable to allow for explanations that are more fidel, ie. capturing more aspects of

the model that should be explained.

More formally, for an input X, a classifier CL (to be explained) and a layer L serving as explanation L, let $L(X)$ be the activations of layer L for input X, and $Loss(CL(X))$ the classification loss of X. The decoder D transforms the representation $L(X)$ into the reconstruction $\hat{X}$. For *ClaDec* the decoder loss is:

$$Loss_{ClaDec}(X) := \alpha \cdot \sum_i (X_i - \hat{X}_{i,E})^2 + (1-\alpha) \cdot Loss(CL(\hat{X}_{i,E}))$$
$$\text{with } \hat{X}_E := D(L(X)) \text{ and } \alpha \in [0,1] \tag{1}$$

The trade-off parameter α allows to control whether reconstructions $\hat{X}_E$ are more similar to inputs, ie. for which a domain expert is more familiar, or reconstructions that are more shaped by the classifier and, thus, they might look more different than training data a domain expert is familiar with. For the reference auto-encoder *RefAE*, the loss is simpler, ie. it is merely the reconstruction loss:

$$Loss_{RefAE}(X) := \sum_i (X_i - \hat{X}_{i,R})^2 \text{ with } \hat{X}_R := D'(L'(X))$$

It is possible to use a variational AE, which might be desirable if multiple explanations should be generated for the same input. We experimented with variational auto-encoders, which we constructed by adding an extra layer to generate a latent code from $L(X)$ that is used in the KL-divergence loss term enforcing a more Gaussian distribution of latent space variables. We found that they lead to worse outcomes in terms of reconstruction loss (for *ClaDec* and *RefAE*). This is expected given that variational AEs add a loss term unrelated to reconstruction and, therefore, the optimizer cannot just optimize towards reconstruction but has to strike a balance between the two loss terms.

IV. Explanation Properties and Their Measurement

The introduction discussed some desirable properties of explanations, namely interpretability, effort and fidelity. While there are more properties such as fairness or privacy [2], the chosen ones are among the most crucial metrics that are also commonly dealt with in the literature. Next, we discuss them in more detail, in particular, we state objective, quantifiable, necessary conditions that explainability methods must achieve to be able to hold those properties. The first condition relates to fidelity. It is of general nature, ie. it should be fulfilled by (any) explanation method. The second condition deals with interpretability. It is more tailored towards methods that provide explanations by synthesizing inputs.

A. Fidelity

Fidelity is the degree to which an explanation captures model behavior. That is, a "fidel" explanation captures the decision process of the model accurately.

The proposed evaluation (also to be used as sanity check) uses the rational that fidel explanations for decisions of a well-performing model should be helpful in performing the task the model addresses. Concretely, training a new classifier CL^E_{eval} on explanations and, possibly, inputs should yield a better performing classifier than relying on inputs only. That is, we train a baseline classifier $CL^R_{eval}(\hat{X}_R)$ on the reconstructions of the *RefAE* without explanations and a second classifier with identical architecture $CL^E_{eval}(\hat{X}_E)$ on explanations only. The latter classifier should achieve higher accuracy. This is a much stronger requirement than the common sanity check demanding that explanations must be valuable to perform a task better than a "guessing" baseline. One might use explanations in combinations with inputs, ie. (X, X_E). Generally, explanations might be also of different format or data type than inputs. This makes comparison more difficult, since the classifier architectures for evaluation cannot be identical in this case. Furthermore, one must be careful that explanations do not contain additional external knowledge (not present in the inputs or training data) that help in performing the task. For most methods (including ours), explanations are only computed using the model, the input and potentially the training data. This means that explanations cannot include additional information to the training data (and the input) that might be helpful for the task. Therefore, it is not obvious that training on explanations allows to improve on classification performance compared to training on original inputs. Improvements seem only be possible if an explanation is a more adequate representation to solve the problem than the original input. Furthermore, one might also (intentionally) distort inputs used to train the evaluation classifiers CL_{eval} and CL'_{eval} to various degrees to assess the value of explanations in a gradual manner. Clearly, once the distorted input resembles only noise, decisions are solely based on the explanations. This approach is most sound, the distorted input and (possible also distorted) explanation together, ie. (X^D, X^D_E), provide the same information on the original input X as the distorted input X^D alone. Our approach utilizes the latter idea, ie. we distort an input X through our reconstruction process. We measure the similarity between the reconstructions $\hat{X}_R$ (using *RefAE*) and $\hat{X}_E$ (of *ClaDec*) with the original inputs X. We show that explanations (from *ClaDec*) bear less similarity with original inputs than reconstructions from *RefAE*. Still, training on explanations $\hat{X}_E$ only yields classifiers with better performance than on the more informative outputs $\hat{X}_R$ from *RefAE*.

B. Interpretability (and Effort)

Interpretability is the degree to which the explanation is human understandable. Effort is the cognitive load that is required by a human to make sense of an explanation. We build upon the intuitive assumption that a human can better and more easily interpret explanations made of concepts that she is more familiar with. The assumption that familiarity results in reduced cognitive effort is well-justified. If unknown concepts are used in explanations then these novel concepts require additional explanations themselves, which would not be needed if familiar concepts were used. In addition, several studies have discussed upon the relationship between familiarity and trust in various contexts [11], [12] showing that "familiarity

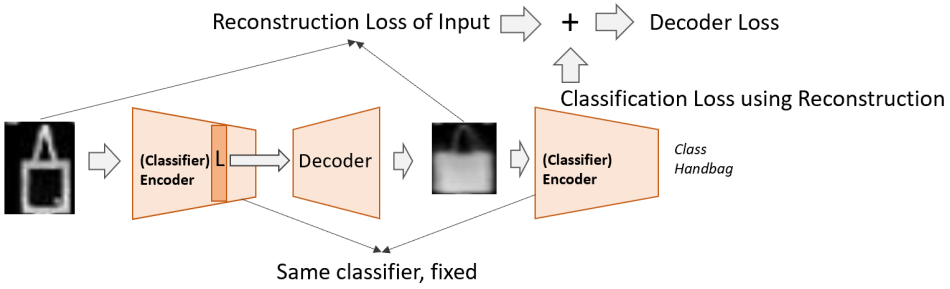

Fig. 3. Architecture extension of *ClaDec*, where decoder is optimized for reconstruction and classification loss

breeds trust". We argue that a user is more familiar with real-world phenomena and concepts as captured in the training data than possibly unknown concepts captured in representations of a neural network. In our scenario, this implies that explanations that are more similar to the training data are more interpretable than those with strong deviation from the training data. Therefore, we quantify interpretability by measuring the distance to the original input, ie. the reconstruction loss. That is, if explanations show concepts that are highly fidelitous, but completely unintuitive for a user (high reconstruction loss) a user can experience difficulties in making sense of the explanation. In contrast, a trivial explanation (showing the unmodified input) is easy to understand but it might not reveal any insights into model behavior, ie. it lacks fidelity.

V. Evaluation

We perform a qualitative and quantitative evaluation focusing on image classification using convolutional neural networks (CNN). This task is suitable not only because of its practical relevance, but also because of its accessibility for a qualitative interpretation. We also assess explanations using quantitative measures proposed in Section IV. We perform three experiments: (i) explaining different layers, (ii) Varying the fidelity and interpretability tradeoff, (iii) Assessing the impact of performance of model to explain on explanations and (together with (iii)) explaining the encoder in a conventional AE rather than a classifier using random transformations (as in extreme learning) as reference.

Encoder		Decoder	
Type/Stride	Filter Shape	Type/Stride	Filter Shape
C/s2	3×3×1×16	FC	nClasses
C/s2	3×3×16×32	DC/s2	3×5×5×256
C/s2	3×3×32×64	DC/s2	3×5×5×128
C/s2	3×3×64×128	DC/s2	3×5×5×64
C/s2	3×3×128×256	DC/s2	3×5×5×32
FC/s1	256×nClasses	DC/s2	3×5×5×1
Softmax/s1	Classifier		

TABLE I
Encoder/Decoder, where "C" is a convolution, "DC" a deconv; a BatchNorm and a ReLu layer follow each "C" layer; a ReLu layer follows each "DC" layer

VI. Setup

The classifier (and encoder) is a VGG-style architecture [13]. The decoder follows a standard design, ie. using 5x5 deconvolutional layers. Encoder (= classifier) and decoder are shown in Table I. Note, that the same classifier architecture (but trained with different input data) serves as encoder in *RefAE*, classifier in *ClaDec* and for evaluation of reconstructions from *ClaDec* training classifier CL^E_{Eval} and *RefAE* training classifier CL^R_{Eval}. The evaluation setup is shown in Figure 4 for *ClaDec*. We only report on validation accuracies, since training accuracies were above 99.5 percent.[2] Thus, we denote by "Acc Enc *ClaDec*" the validation accuracy of the encoder, ie. classifier, of the *ClaDec* architecture and by "Acc Eval *RefAE*" the validation accuracy of the classifier CL^R_{Eval} used for evaluation as shown in Figure 4 trained on reconstructions from the reference AE. Other combinations are analogous. Note that the decoder architecture varies depending on which layer is to be explained. The exact architecture as in Table I allows to either obtain reconstructions from the last convolutional layer or the fully connected layer. For a lower layer, the highest deconvolutional layers from the decoder have to be removed, so that the reconstructed image $\hat{X}$ has the same width and height as the original input X. We employ three datasets namely Fashion MNIST [14], MNIST and TinyImageNet[3]. For TinyImageNet, we doubled the number of conv layers in Table I of the encoder and increased the number of neurons of all layers (encoder and decoder) by four. For MNIST, we used the cross-entropy loss instead of the L2-loss.[4] Since all datasets behaved similarly, we shall discuss primarily results on FASHION-MNIST consisting of 70000 28x28 images of clothing stemming from 10 classes that we scaled to 32x32. 10000 samples are used for testing. We discuss results for other data in a more summarized form

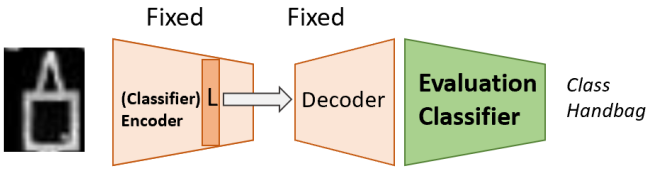

Fig. 4. Architecture with evaluation classifier

[2]Except for the benchmark, where we trained for less epochs
[3]https://tiny-imagenet.herokuapp.com/
[4]Otherwise results were often of poor quality.

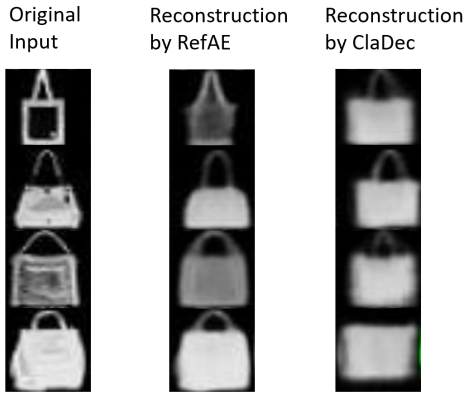

Fig. 5. Comparison of original inputs and reconstructions using the FC layer of the encoder in Table I for handbags. Both reconstructions do not recover detailed textures. The classifier does not rely on graytunes, but focuses on prototypical shapes of features.

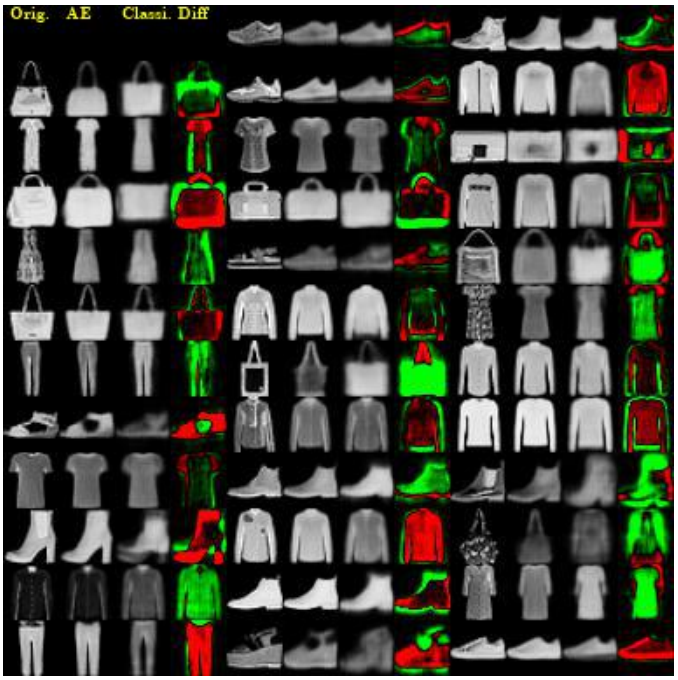

Fig. 6. Comparison of original inputs and reconstructions using the last layer, ie. FC, of the encoder in Table I. Differences between reconstructions are shown in the last column.

in the end (Section VI-C).

We train all models using the Adam optimizer for 64 epochs. That is, the reference AE, the decoder of the *ClaDec*, the classifier serving as encoder in *ClaDec* as well as the classifiers used for evaluation. We conducted 10 runs for each reported number. We show both averages and standard deviations.

A. Qualitative Evaluation

1) Varying Explanation Layers: Figures 5 and 6 show reconstructions based on *RefAE* and *ClaDec* for the last layer, ie. the fully connected (FC) layer (see Table I). For this layer, there is only one value per class, implying a representation of 10 dimensions for FASHION-MNIST. For the handbags depicted in Figure 5, comparing the original inputs and the reconstructions by *RefAE* and *ClaDec* shows that both reconstructions do not capture detailed textures. Overall, *RefAE* is able to reconstruct shapes and graytunes fairly well. Comparing reconstructions from *ClaDec* to those of *RefAE* and the original inputs, it becomes apparent that reconstructions from *ClaDec* have more uniform graytunes, ie. they are poor approximations of the actual graytunes compared to both the original and reconstructions from *RefAE*. This hints that knowledge of precise graytunes is not crucial to classify objects. Reconstructions from *ClaDec* seem to resemble more prototypical, abstract features of handbags. Looking closer into multiple samples of handbags shows that handbags might be characterized by having a handle or not. Handbags without handles often have a rectangular shape. Figure 6 shows that reconstructions capture this well: The reconstructed handbags that have a handle exhibit typically more of a square shape, whereas the handbags without handles are more of a rectangular shape. Reconstructions from *ClaDec* are more blurry than for *RefAE*. Blurriness indicates that the representation of the layer does not contain information needed to recover the details. However, the reason is not (primarily) distortions inherent in the decoder architecture, since *RefAE* produces significantly sharper images. For handbags as shwon in Figure 5, *ClaDec* performs stronger alterations of shape compared to the original than for other classes shown in Figure 6 such as sweatshirts. This can indicate that precise shape information for sweatshirts plays a more crucial role than for handbags. This should, in particular, be the case if both classes exhibit similar variation in shape.

When computing reconstructions from a lower layer, both the *RefAE* and the *ClaDec* reconstructions are closer to the original inputs – in terms of shape, sharpness, graytunes and details. Most observations distinguishing the two reconstructions made for the very last layer still hold (but are not as profound). The more subtle difference, when using lower layers, also motivates to reconstruct images that are more influenced by the classifier as done in the extended architecture (Figure 3) and illustrated next.

2) Fidelity and Interpretability Tradeoff: Figure 7 shows for the last conv. layer (second to last overall) the impact of adding a classification loss to modulate how much the model impacts reconstructions (see Figure 3). Most notably is the observation that neglecting any reconstruction loss pushing the decoding towards the original inputs yields images consisting of black and white patterns that are completely unrelated to the original input, which we deem to be of poor interpretability. However, already modest reconstruction loss leads to human recognizable shapes. The quality of reconstructions in terms of sharpness and amount of captured detail constantly improves the more emphasis is put on reconstruction loss. It also becomes evident that the deep learning network seems to learn "prototypical" samples (or features) towards which reconstructed samples are being optimized. For example, the shape of handbag handles is fairly uniform for low values of α, slightly varying in thickness and length. It shows much more

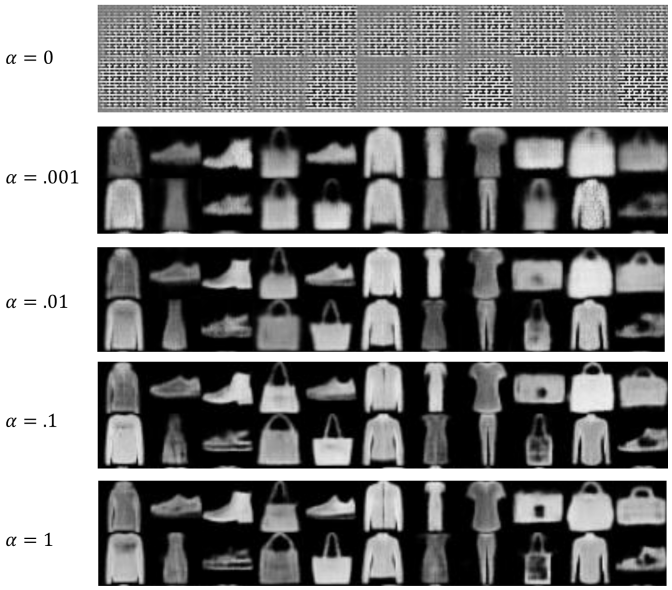

Fig. 7. Adding classification loss ($\alpha < 1$) yields worse reconstructions for the last conv. layer. When focusing exclusively on classification loss reconstructions are not human recognizable.

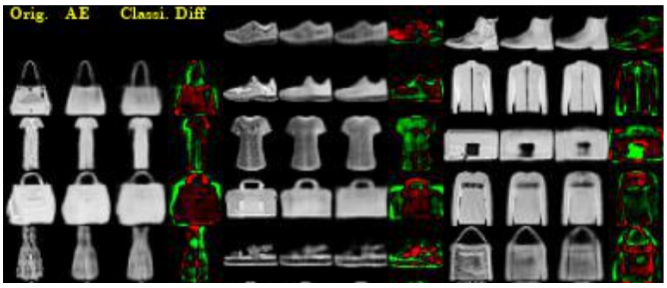

Fig. 8. Comparison of original inputs and reconstructions using the last conv. layer of the encoder in Table I without any training of the classifier in *ClaDec*.

diversity for high values of α. This behavior might be seen as a means to reconstruct a compromise between the sample that yields minimal classification loss and a sample that is true to the input. It suggests that areas of the reconstruction of *ClaDec* that are similar to the original input are also similar to the "prototype" that minimizes classification loss. That is, the network can recognize them well, whereas areas that are strongly modified, resemble parts that seem non-aligned with what the network regards as a "prototype".

3) Explaining AE and the Impact of Classifier Performance: Figure 8 shows reconstructions if the classifier is not trained at all. Interestingly they exhibit fairly good quality. We explain this for the quantitative assessment. Furthermore, comparing reconstructions of the reference AE with those of *ClaDec* for an untrained classifier might be used to assess the relevance of training the encoder of an AE itself, ie. "How does training an encoder (of an AE architecture) impact reconstructions (compared to a random encoder)?" Based on Figure 8 one might conclude that a trained encoder does lead to encodings that allow to better reconstruct original inputs than when using an untrained encoder utilizing randomly initialized layers. While sharpness is generally comparable for both reconstructions, there are several examples, where shapes of objects are altered or some details are missing, if an encoder is not-trained.

B. Quantitative Evaluation

1) Varying Explanation Layers: Table II shows two key messages: First, the reconstruction loss is lower for the *RefAE* than for *ClaDec*. This is expected since the *RefAE* model is optimized entirely towards minimal reconstruction loss of the original inputs. Second, the classification (evaluation) accuracy is higher, when training the evaluation classifier CL_{Eval} using reconstructions from *ClaDec* than from *RefAE*. This behavior is not obvious, since the reconstructions from *ClaDec* are poorer according to the reconstruction loss. That is, they contain less information about the original input than those from *RefAE*. However, it seems that the right information is encoded using a better suited representation. Aside from these two key observations there are a set of other noteworthy behaviors: The reconstruction loss increases the more encoder layers are used. The impact is significantly stronger for *ClaDec*. The difference between *RefAE* and *ClaDec* increases the lower the layer to explain is. This is not surprising, since lower layers are known to be fairly general, ie. in transfer learning lower layers are the most applicable to work well for varying input data. There is a particularly strong increase for the last layer, this is also no surprise, since the last layer consists of logits, meaning just 10 dimensions (1 per class). The classification accuracy for the evaluation classifier improves the more layers are used as encoder, ie. of the classifier that should be explained. The opposite holds for *RefAE*. This confirms that the *RefAE* focuses on the wrong information, whereas the classifier trained towards the task focuses on the right information and encodes it well.

2) Fidelity and Interpretability Tradeoff: Table III shows that evaluation accuracy increases the more emphasis is put on classification loss. This means that reconstructed inputs are stronger influenced by the model to explain, ie. they are more truthful to the model. They are also more amendable to classification, which is shown by a higher accuracy of the evaluation classifier. We also experimented with replacing the classification loss with a loss capturing the reconstruction error of the layer activation $L(X)$. This gives meaningful results, but the reconstructions perform worse on downstream tasks.

3) Impact of Classifier Performance: Table IV shows for *ClaDec* that classifiers that are trained longer and, therefore, achieve higher validation accuracy also lead to better accuracy for the evaluation classifier. While this is expected, the dependence of reconstruction loss on the number of training epochs is more intricate. It is lowest without any training, increases quickly and then steadily decreases again. This pattern is highly statistically significant, ie. we conducted t-tests to verify that means between subsequent rows are different, yielding p-values below 0.01. When taking a closer look, it is not so surprising that an untrained network, ie. using random weights, achieves lower reconstruction loss than the trained classifier.

Layer	(Rec.) Loss *ClaDec*	(Rec.) Loss *RefAE*	Δ	Acc Eval *ClaDec*	Acc Eval *RefAE*	Δ
-4	$3.61_{\pm 0.087}$	$2.62_{\pm 0.075}$	$0.99_{\pm 0.101}$	$0.893_{\pm 0.004}$	$0.893_{\pm 0.003}$	$-0.0_{\pm 0.007}$
-3	$3.627_{\pm 0.081}$	$2.637_{\pm 0.036}$	$0.99_{\pm 0.093}$	$0.893_{\pm 0.004}$	$0.892_{\pm 0.003}$	$0.001_{\pm 0.005}$
-2	$6.081_{\pm 0.069}$	$3.341_{\pm 0.062}$	$2.74_{\pm 0.097}$	$0.895_{\pm 0.002}$	$0.889_{\pm 0.004}$	$0.006_{\pm 0.004}$
-1	$28.066_{\pm 1.009}$	$7.281_{\pm 0.218}$	$20.785_{\pm 0.817}$	$0.904_{\pm 0.005}$	$0.845_{\pm 0.006}$	$0.059_{\pm 0.008}$

TABLE II

EXPLAINING DIFFERENT LAYERS: *ClaDec* HAS LARGER RECONSTRUCTION LOSS BUT THE EVALUATION CLASSIFIER ON RECONSTRUCTIONS FROM *ClaDec* ACHIEVES HIGHER ACCURACY.

α	Total Loss *ClaDec*	Rec Loss	Classifier Loss	Acc Eval *ClaDec*
0	$6.081_{\pm 0.069}$	$6.081_{\pm 0.069}$	$0_{\pm 0}$	$0.895_{\pm 0.002}$
0.9	$0.684_{\pm 0.021}$	$6.808_{\pm 0.206}$	$-6.124_{\pm 0.185}$	$0.909_{\pm 0.005}$
0.99	$0.105_{\pm 0.009}$	$10.207_{\pm 0.664}$	$-10.102_{\pm 0.656}$	$0.913_{\pm 0.003}$
0.999	$0.021_{\pm 0.001}$	$18.789_{\pm 0.902}$	$-18.767_{\pm 0.902}$	$0.913_{\pm 0.004}$
1	$0.002_{\pm 0.001}$	$234.767_{\pm 52.631}$	$-234.765_{\pm 52.631}$	$0.911_{\pm 0.003}$

TABLE III

ADDING CLASSIFICATION LOSS ($\alpha > 0$) YIELDS WORSE RECONSTRUCTIONS, BUT HIGHER EVALUATION ACCURACY

Training Epochs of Classifier (to be explained)	Acc Classifier (to be explained)	(Rec.) Loss *ClaDec*	(Rec.) Loss *RefAE*	Δ	Acc Eval *ClaDec*	Acc Eval *RefAE*	Δ
0	$0.1_{\pm 0.0}$	$5.445_{\pm 0.164}$	$3.333_{\pm 0.04}$	$2.112_{\pm 0.159}$	$0.85_{\pm 0.003}$	$0.886_{\pm 0.004}$	$-0.036_{\pm 0.005}$
1	$0.506_{\pm 0.012}$	$6.417_{\pm 0.152}$	$3.3_{\pm 0.038}$	$3.118_{\pm 0.156}$	$0.88_{\pm 0.002}$	$0.888_{\pm 0.003}$	$-0.007_{\pm 0.004}$
4	$0.885_{\pm 0.003}$	$6.608_{\pm 0.079}$	$3.299_{\pm 0.049}$	$3.309_{\pm 0.088}$	$0.893_{\pm 0.004}$	$0.893_{\pm 0.004}$	$-0.0_{\pm 0.005}$
16	$0.902_{\pm 0.003}$	$6.233_{\pm 0.145}$	$3.334_{\pm 0.062}$	$2.898_{\pm 0.116}$	$0.896_{\pm 0.005}$	$0.891_{\pm 0.003}$	$0.005_{\pm 0.005}$
64	$0.904_{\pm 0.003}$	$6.081_{\pm 0.069}$	$3.341_{\pm 0.062}$	$2.74_{\pm 0.097}$	$0.895_{\pm 0.002}$	$0.889_{\pm 0.004}$	$0.006_{\pm 0.004}$

TABLE IV

IMPACT OF CLASSIFIER ACCURACY (MODULATED THROUGH TRAINING EPOCHS): EVALUATION ACCURACY INCREASES WITH HIGHER CLASSIFIER ACCURACY AS EXPECTED, BEHAVIOR OF REC.LOSS FOLLOWS AN INVERTED U SHAPE.

First, it should be noted that the reconstruction loss using random weights is significantly higher as for the reference architecture, where the encoder is optimized. Second, it is known from extreme learning, eg. [15], that encoders with randomly chosen weights can yield good results, if just the decoder is optimized. More generally, this phenomena might be traced back to the behavior of random projections formulated in the JohnsonLindenstrauss lemma, saying that random projections yield good dimensionality reduction properties. The theorem is commonly used for dimensionality reduction in many contexts, eg. [16]. Training of the classifier, seems to destroy some of the desirable properties of random initialization by focusing on information needed for classification (but not for reconstruction) – as motivated theoretically (see Figure 2). The reconstruction improves with more training, indicating that the initial encodings are noisy. But the reconstruction loss seems to converge (with increased training) towards a higher loss than for random initialization.

C. Evaluation on MNIST and TinyImageNet

All datasets exhibited similar behavior, ie. the findings for Fashion-MNIST from Sections VI-B and VI-A can be replicated. See Figure 9 for some reconstructions. For TinyImageNet analyzing explanations is more cognitively demanding, since there are more classes, classes exhibit more diversity and reconstructions are of somewhat worse quality.

VII. RELATED WORK

There exists a vast amount of explainability methods [1], [2]. We discuss approaches that allow to visualize single features as well as to understand particular decisions summarized in Figure 10. We categorize into methods that synthesize inputs (like ours and [3]) and methods that rely on saliency maps [17] based on perturbation [4], [18] or gradients [19], [20]. Saliency maps commonly show feature importance of inputs, whereas synthesized inputs might (also) show higher level representations encoded in the network. Perturbation-based methods include occlusion of parts of the inputs [18] and investigating the impact on output probabilities of specific classes. Linear proxy models such as LIME [4] perform local approximations of a black-box model using simple linear models by also assessing modified inputs. Unfortunately, LIME and similar models are generally less suited for complex data such as images. Saliency maps [17] highlight parts of the inputs that contributed to the decision. They commonly employ gradients in one form or another, eg. integrated gradients as for GradCAM [19], gradients $\times$ inputs as for Layer-Wise Relevance Propagation (LRP) [20]. Some of explainability methods have been under scrutiny for even failing simple sanity checks [6] and being sensitive to factors not contributing to model predictions [7] or adversarial perturbations [8]. We anticipate that our work is less sensitive to targeted, hard to notice perturbations [8] as well as translations or factors not impacting decisions [7], since we rely on encodings of the classifier. Thus, explanations only change if these encodings change, which they should. Even disregarding potential deficiencies mentioned in [6]–[8], explanations are still a long way from "perfect". That is, many explanations contain fairly little information on model behavior or representation. Essentially, many methods just state how relevant an input feature is. This does not provide insights into how (input) information is actually processed and how it is encoded in the network. For example, consider Figure 11 taken from [21]. For those methods that show gradients (or a function of the gradients), one primarily sees how (infinitely small) changes would impact the output. We argue that gradient-based methods result in explanations that are difficult to understand and only provide a very narrow

scope for interpretation. The latter is potentially due to the fact that derivatives, ie. gradients, in general are inherently sensitive to small changes and noise. For example, in deep learning, it is well-known that gradients might zig-zag after every update, which has motivated the use of momentum for stochastic gradient descent. More specifically, consider a saliency map depending on gradients. The map might suggest that increasing the brightness of pixels A and B increases confidence in the predicted class. But how much can we increase A and B? Maybe, A can be increased just a tiny bit (and if it is increased beyond the point confidence actually decreases), while B can be increased a lot. That is, gradients are a very local measure and given recent criticism might not be the best approach [6]–[8] if applied unconstrained. For *ClaDec* (Figure 1), we do not employ gradients originating from the model that should be explained. In the extended model (Figure 3) gradients originating from classification loss are used for training of the decoder. Later, when computing explanations they are not needed.

So far, inputs have only been synthesized to understand individual neurons through activation maximization in an optimization procedure [3]. The idea is to identify inputs that maximize the activation of a given neuron. This is similar to the idea to identify samples in the input that maximize neuron activation. [3] uses a (pre-trained) GAN on natural images relevant to the classification problem. It identifies through optimization the latent code that when fed into the GAN results in a realistic looking image that maximally activates a neuron. As such the generated inputs are constrained by the images used for pre-training. That is, generated images might resemble realistic looking samples from the input or unrealistic samples, eg. in case the optimal latent code corresponds to a region with few samples or the neuron encodes non-natural concepts. Our idea to verify if a "distortion" of realism is due to the decoder might also be beneficial to improve on explanations of [3]. The reconstructions of *ClaDec* are also constrained by the training data as [3]. In our case, poor reconstructions might already manifest during training of the decoder, ie. if a classifier yields representations that do not allow to reconstruct realistic samples. As highlighted in Figure 10 there are additional differences between our method and [3]. Our approach does not require any optimization per instance to be explained (though it could be added). Note, that any approach that aims at explicitly interpreting individual neurons (or representations) such as [3], cannot be easily extended to explain the entire model or layer behavior. It suffers from the problem that networks allowing to distinguish just few classes still have hundreds or even thousands of neurons (per layer). Thus, while explaining neurons is highly important, it is a conceptually different problem from explaining a decision or an entire layer.

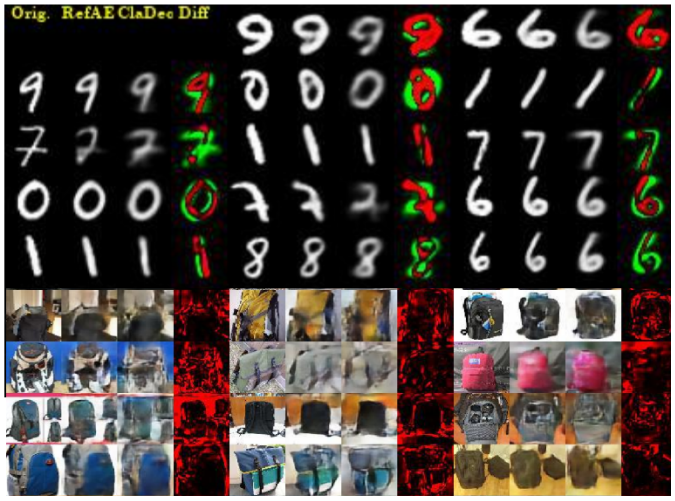

Fig. 9. Comparison of original and reconstructions using the FC layer of the encoder in Table I for MNIST and the last conv. layer for TinyImageNet.

The methods in Figure 10 are non-exhaustive, but cover those most relevant to our work. Other ideas include [22]–[25]. [22], [25] allow to investigate how much/ which (high level) concepts are relevant to a specific decision. DeepLift [23] compares activations to a reference and propagates them backwards. Defining the reference is non-trivial and domain specific. [24] estimates the impact of individual training samples. [26] discusses how to explain variational AEs using gradient-based methods. We also propose a method to explain AEs or, more precisely, just the impact of training an encoder in an AE architecture in Section VI-A3.

Auto-encoders: [27] uses a variational auto-encoder for contrastive explanations. They trained an AE on the training data. Given that one should explain why a sample X is of class Y and not of Y', they compute the latent representation of X. Then, they search for the closest sample X' (in latent space) of class Y'. They sample points between X and X' in latent space and reconstruct images based on the code provided by the sampled points. To compute a contrastive explanation the image being classified as class Y' and with latent code closest to X is chosen. From our perspective it can be valuable to replace the latent representation produced by the VAE, with a representation of the classifier. Denoising AEs are well established [28], [29]. They can be used to remove noise from images, reconstruct images and leverage data in an unsupervised manner [28], [29]. Ideas to combine unsupervised learning approaches to remove noise and supervised learning by extending loss functions have already been uttered in the early 90ies [30]. In the context of explanations, [31] used an AE with skip-connections for saliency map predictions.

VIII. Conclusions

Explaining complex deep learning models is difficult as witnessed by multiple works pointing out shortcomings of many explanation methods that often lack sound theoretical grounding and evaluation. We have proposed a theoretically grounded method that allows to synthesize inputs based on representations originating from the model to be explained. It takes into account distortions originating from the reconstruction process and it has been verified using novel sanity checks. We believe that our method might form the basis for

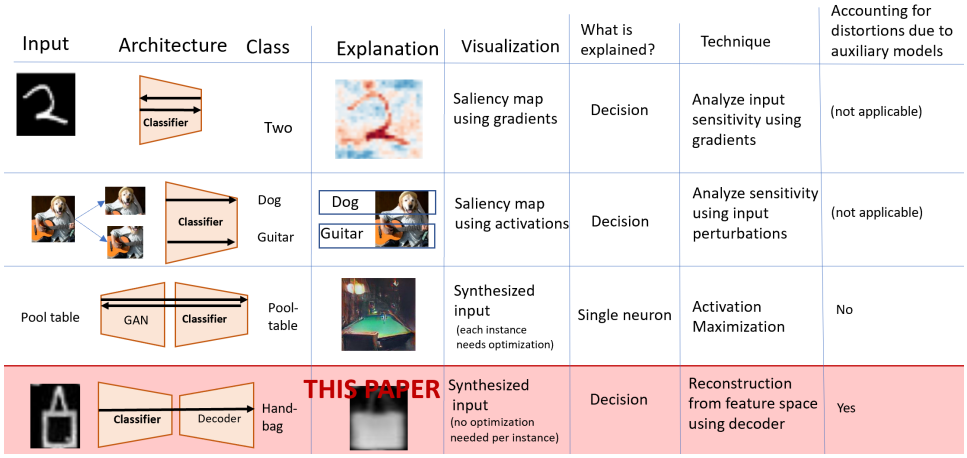

Fig. 10. Method Overview. Figures are from cited papers.

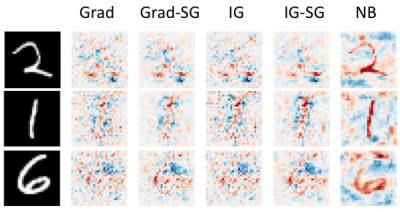

Fig. 11. Saliency maps examples. Taken from [21].

many more methods that further expand and contribute to the field of explainability.

IX. Acknowledgments

We thank Jeroen van Doorenmalen for valuable discussions.

References

[1] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, 2018.

[2] J. Schneider and J. P. Handali, "Personalized explanation for machine learning: a conceptualization," in *European Conference on Information Systems (ECIS)*, 2019.

[3] 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*, 2016, pp. 3387–3395.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proc. ACM Special Interest Group on Knowledge Discovery and Data Mining*, 2016.

[5] F. Yang, M. Du, and X. Hu, "Evaluating explanation without ground truth in interpretable machine learning," *arXiv preprint arXiv:1907.06831*, 2019.

[6] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," in *Neural Information Processing Systems (NIPS)*, 2018, pp. 9505–9515.

[7] P.-J. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim, "The (un) reliability of saliency methods," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, 2019, pp. 267–280.

[8] A. Ghorbani, A. Abid, and J. Zou, "Interpretation of neural networks is fragile," in *AAAI Conference on Artificial Intelligence*, 2019.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[10] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural networks*, vol. 2, no. 1, pp. 53–58, 1989.

[11] R. Gulati, "Does familiarity breed trust? the implications of repeated ties for contractual choice in alliances," *Academy of management journal*, vol. 38, no. 1, pp. 85–112, 1995.

[12] Y.-H. Chen and S. Barnes, "Initial trust and online buyer behaviour," *Industrial management & data systems*, 2007.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Int. Conference on Learning Representations (ICLR)*, 2014.

[14] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[15] K. Sun, J. Zhang, C. Zhang, and J. Hu, "Generalized extreme learning machine autoencoder and a new deep neural network," *Neurocomputing*, vol. 230, pp. 374–381, 2017.

[16] J. Schneider and M. Vlachos, "Fast parameterless density-based clustering via random projections," in *Proc. of the international conference on Information & Knowledge Management(CIKM)*, 2013.

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

[18] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, 2014.

[19] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618–626.

[20] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one*, vol. 10, no. 7, 2015.

[21] C.-K. Yeh, C.-Y. Hsieh, A. Suggala, D. I. Inouye, and P. K. Ravikumar, "On the (in) fidelity and sensitivity of explanations," in *Advances in Neural Information Processing Systems*, 2019, pp. 10 965–10 976.

[22] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)," *arXiv preprint arXiv:1711.11279*, 2017.

[23] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. of the International Conference on Machine Learning (ICML)*, 2017.

[24] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proc. of Int. Conference on Machine Learning*, 2017.

[25] A. Ghorbani, J. Wexler, J. Y. Zou, and B. Kim, "Towards automatic concept-based explanations," in *Advances in Neural Information Processing Systems*, 2019.

[26] W. Liu, R. Li, M. Zheng, S. Karanam, Z. Wu, B. Bhanu, R. J. Radke, and O. Camps, "Towards visually explaining variational autoencoders," *arXiv preprint arXiv:1911.07389*, 2019.

[27] J. van Doorenmalen and V. Menkovski, "Evaluation of cnn performance in semantically relevant latent spaces," in *Int. Symposium on Intelligent Data Analysis*, 2020.
[28] 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*, vol. 11, no. Dec, pp. 3371–3408, 2010.
[29] B. Du, W. Xiong, J. Wu, L. Zhang, L. Zhang, and D. Tao, "Stacked convolutional denoising auto-encoders for feature representation," *IEEE transactions on cybernetics*, vol. 47, no. 4, pp. 1017–1027, 2016.
[30] G. Deco, W. Finnoff, and H. Zimmermann, "Elimination of overtraining by a mutual information network," in *Int. Conference on Artificial Neural Networks*, 1993.
[31] F. Qi, C. Lin, G. Shi, and H. Li, "A convolutional encoder-decoder network with skip connections for saliency prediction," *IEEE Access*, vol. 7, pp. 60 428–60 438, 2019.