Self-Explaining Neural Networks

David Alvarez-Melis dalvmel@mit.edu
Tommi S. Jaakkola tommi@csail.mit.edu

Abstract

Most recent work on interpretability for complex machine learning models has focused on estimating *a posteriori* explanations for particular predictions, but less attention has been devoted to designing the models themselves to be interpretable. We approach this problem by characterizing what makes archetypical "interpretable" models—such as linear classifiers—this way, and then deriving a set of principles that more complex models should satisfy to be interpretable. We progressively generalize various aspects of the simple prediction models to yield a rich class of complex models that are *self-explaining*: i.e., they provide human-understandable explanations for their predictions as an intrinsic part of their operation. The resulting model class in flexible, easy to train, and encompasses various existing interpretable models as particular cases. Experimental results in applications ranging from digit classification to cancer detection show that our framework offers a promising direction for reconciling model complexity and interpretability.

1 Introduction

State-of-the-art machine learning methods provide strong predictive power but generally lack interpretability. This limits adoption of such methods in decision-critical domains such as medical or legal. Interpretability also often serves as a step in satisfying other criteria such as fairness, privacy, or causality [Doshi-Velez and Kim, 2017]. Our goal in this paper is to recast models in a manner that makes them "self-explaining" without sacrificing predictive power.

Linear regression models or decision trees ofter serve as prime examples of what we mean by interpretable models. They are, however, too simple for most modern machine learning applications where high modeling capacity is often necessary for competitive performance. Much of the recent work on interpretability has therefore focused on producing *a posteriori* explanations for complex deep learning approaches, where the explanation is provided locally, around a given example. The interpretation is derived on the basis of limited access to the inner workings of the model. For example, explanations may be derived from gradients or running the model in reverse [Bach et al., 2015, Selvaraju et al., 2017] or obtained through oracle queries by estimating simpler surrogate models to capture the observed local input-output behavior [Alvarez-Melis and Jaakkola, 2017, Lundberg and Lee, 2017, Ribeiro et al., 2016]. A number of technical challenges arises already in this setting, from the definition of locality (e.g., in case of structured examples) to identifiability issues [Li et al., 2018]. Offering an explanation of this kind also invariably necessitates additional computation beyond the forward prediction, ranging from a backward pass Bach et al. [2015] to running a full-fledged optimization subroutine [Yosinski et al., 2015].

External a posteriori explanations may be necessary if the model has already been trained or offered only as a black-box system after the fact. However, when we can influence model building, we would ideally have the models set up from the start in a manner that they automatically offer human-interpretable explanations of their predictions. Such "self-explaining" models would facilitate verification and error analysis, and they could be more easily guided by and integrated with the available domain knowledge. Despite the advantages, relatively little work has been devoted to self-explaining models. One approach to integrating explanations into the model while preserving accuracy is to require a compact selection of the (textual) input as an intermediate, explanatory step. In this vein, in the context on text classification, Lei et al. [2016] specify two neural networks, trained cooperatively, where one selects a small portion of the text as an explanation while the other makes predictions only based on the selected text. An alternative approach by Li et al. [2018] relies on a small, learnable set of prototypes to make neural network predictions interpretable, by having all downstream computations depend on the input only through a vector of distances to all the prototypes.

In this paper we offer a different approach to self-explaining models. We commence with a simple a priori model – here a linear predictor – and generalize it to make it more powerful without loosing the key ingredients of how to interpret it. In this sense, in contrast to prevailing approaches, we do not constrain complex models to make them interpretable but rather generalize simple models in a manner that preserves their mechanism of interpretation. Linear models, for example, are built from simple concepts (features) and how relevant they are (coefficients). This combination of features and their coefficients can be generalized substantially without destroying the basic interpretation. The coefficients, for instance, can be made input dependent, produced by a complex deep learning architecture. The only constraint is that the coefficients should vary slowly enough around each input so that the overall model, around each specific input, remains interpretable as a linear model. We also generalize and learn interpretable features to accompany learned functional coefficients.

The paper is structured as follows. In Section 2 we take the example of linear classifiers to build intuition on desirable properties of self explaining models. We successively generalize this class of models while maintaining interpretability. In Section 3 we then take these intuitions and formalize a class of self-explaining complex models. Section 4 discusses the case where the basis concept functions are to be learnt in conjunction with the prediction model. Next, in Section 5 we present experimental results portraying various configurations of our general framework, ranging from digit classification to cancer detection applications. Through various metrics of relevance and stability, we show that the explanations provided by our models are coherent and faithful to its operation. Furthermore, we show that this additional functionality comes a at a minimal cost in performance, often even helping our self-explaining models outperform similar non-interpretable architectures due to the additional regularization provided by our robust training objectives.

2 Interpretability: linear and beyond

To motivate our approach, we start with a simple linear regression model and successively generalize it towards the class of self-explaining models. For input features $x_1, \ldots, x_n \in \mathbb{R}$, and associated parameters $\theta_0, \ldots, \theta_n \in \mathbb{R}$ the linear regression model is given by

$$f(x) = \sum_i^n \theta_i x_i + \theta_0 \tag{1}$$

For simplicity, we will write the model as $f(x) = \theta^T x$, omitting the explicit bias term. We maintain that the linear model is interpretable for three specific reasons: 1) input features (x_i's) are clearly anchored with the available observations, e.g., arising from empirical measurements; 2) each parameter θ_i provides a quantitative positive/negative contribution of the corresponding feature x_i to the predicted value; and 3) the aggregation of feature specific terms $\theta_i x_i$ is additive thus not further conflating the feature-by-feature interpretation of impact. We offer several generalizations in the following subsections and discuss how this mechanism of interpretation is preserved.

2.1 Generalized coefficients

We can substantially enrich the linear model while keeping the overall structure of (3), if we permit the coefficients themselves to depend on the input x. Specifically, we define (offset function omitted)

$$f(x) = \theta(x)^T x \tag{2}$$

and choose θ from a complex model class Θ, realized for example via deep neural networks. Without further constraints the model is nearly as powerful as any deep neural network. However, in order to maintain interpretability, at least locally, we must ensure that for close inputs x and x' in $\mathbb{R}^n$, $\theta(x)$ and $\theta(x')$ should not differ significantly. To make this more precise, we can, for example, regularize the model in such a manner that $\nabla_x f(x) \approx \theta(x_0)$ for all x in a neighborhood of x_0. In other words, the model functions locally, around each x_0, as a linear model with a vector of coefficients given by $\theta(x_0)$. The individual values $\theta(x_0)_i$ act as and are interpretable as coefficients of a linear model with respect to the final prediction, but adapt dynamically to the input, albeit varying slower than x. We will discuss specific regularizers so as to keep this interpretation in Section 3.

2.2 Beyond raw features – feature basis

Typical interpretable models tend to consider each variable (one feature or one pixel) as the fundamental unit which explanations consist of. However, pixels are rarely the basic units used in human image understanding; instead, humans would rely on strokes and other higher order features. We adopt and refer to these higher order features as *interpretable basis concepts*, and use them in place of raw features in our models. Formally, let $\mathcal{Z} \subset \mathbb{R}^k$ be some space of interpretable atoms, and define $h(x) : \mathcal{X} \to \mathcal{Z}$, i.e., h maps raw inputs to interpretable concepts. Naturally, k should be small so as to keep the explanations easily digestable. Alternatives for $h_i(x)$ include

1. Subset aggregates of the input (e.g., with $h(x) = Ax$ for a boolean mask matrix A).

2. Predefined, pre-grounded feature extractors (e.g. from expert knowledge), such as specific filters for image processing.

3. Prototype based concepts, e.g. $h_i(x) = \|x - z_i\|$ for some $z_i \in \mathcal{X}$ [Li et al., 2018], or cast in terms of similarities to prototypes for stability

4. Learnt representations with specific constraints to ensure groundedness

The generality of basis functions allows for significant flexibility. The extended model is now given by

$$f(x) = \theta(x)^T h(x) = \sum_{i=1}^{K} \theta(x)_i h(x)_i \tag{3}$$

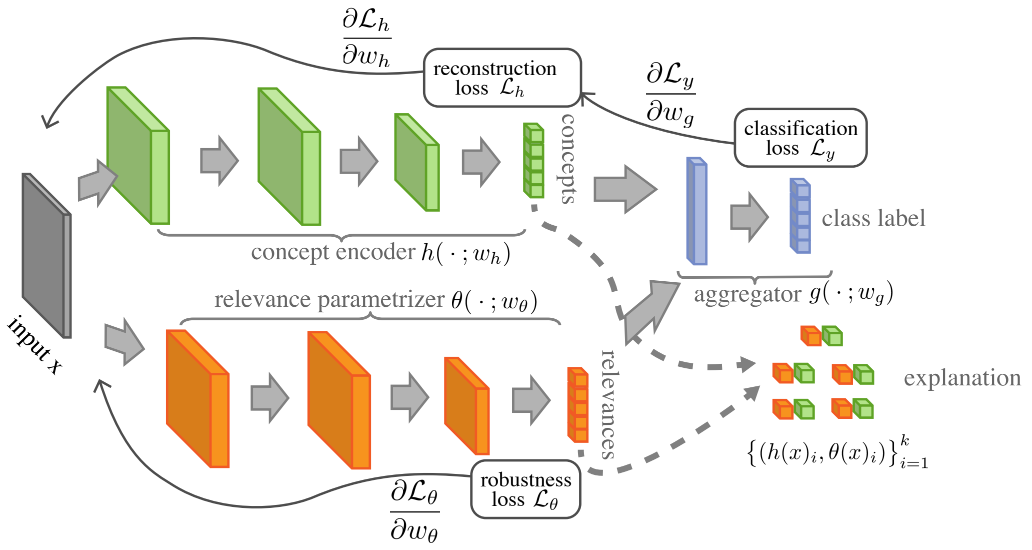
Since each $h(x)_i$ remains a scalar, it can still be interpreted as the degree to which a particular feature is present. $\theta(x)_i$ in turn, with constraints similar to those discussed above, remains interpretable as a local coefficient. Note that the notion of locality must now take into account how the concepts rather than inputs vary since the model is interpreted as being linear in the concepts rather than x.

2.3 Beyond simple aggregation

The final generalization we propose considers the method by which the weighted units of explanation, i.e. the elements $\theta(x)_i h(x)_i$ are aggregated. Although the sum is a natural way to do so, we can achieve a more flexible class of functions by considering more general aggregation functions $g(z_1, \dots, z_k)$. Naturally, in order for this function to preserve the desired interpretation of $\theta(x)$ in relation to $h(x)$, it should: i) be permutation invariant, so as to eliminate higher order uninterpretable effects caused by the relative position of the arguments, (ii) isolate the effect of individual $h(x)_i$'s in the output (e.g., avoiding multiplicative interactions between them), and (iii) preserve the sign and relative magnitude of the impact of the relevance values $\theta(x)_i$. We formalize these intuitive desiderata in the next section.

2.4 Multidimensional outputs

For a multidimensional output space $\mathcal{Y} \subset \mathbb{R}^m$, with $m > 1$, we can naturally extend (3) by considering $\theta_i : \mathcal{X} \to \mathbb{R}^m$, so that $\theta_i(x) \in \mathbb{R}^m$ is a vector corresponding to the relevance of concept i with respect to each of the m output dimensions. Although in the case of classification, we will most likely be interested in the scores corresponding to the maximizing class, i.e., $\theta_i(x)_{\hat{y}}$ for $\hat{y} = \operatorname{argmax}_y f(x)_y$, a representation of θ of the same dimension as the output allows us to understand individual effects of the concepts on each of the output classes.



$\frac{\partial \mathcal{L}_h}{\partial w_h}$
reconstruction loss $\mathcal{L}_h$
$\frac{\partial \mathcal{L}_y}{\partial w_g}$
classification loss $\mathcal{L}_y$
concepts
class label
concept encoder $h(\cdot\,; w_h)$
relevance parametrizer $\theta(\cdot\,; w_\theta)$
aggregator $g(\cdot\,; w_g)$
input x
explanation
relevances
$\{(h(x)_i, \theta(x)_i)\}_{i=1}^k$
robustness loss $\mathcal{L}_\theta$
$\frac{\partial \mathcal{L}_\theta}{\partial w_\theta}$

Figure 1: The proposed architecture consists of three components: a concept encoder (green) that transforms the input into a small set of interpretable basis features; an input-dependent parametrizer (orange) that generates relevance scores; and an aggregation function that combines them. Concepts and their relevance parameters are used by the aggregation function to produce the final label prediction. The robustness loss on the parametrizer encourages f to behave locally as a linear function on $h(x)$ with parameters $\theta(x)$, allowing for immediate interpretation of both both concepts and relevances.

3 Self-explaining models

In this section we formalize the class of models that were obtained through subsequent generalization of the simple linear predictor in the previous section. We begin by formalizing the properties we wish to impose on θ in order for it to act as coefficients of a linear model on the basis concepts $h(x)$. The intuitive notion of slow-variation discussed in Section 2.2 suggests using a Lipschitz-type condition bounding $\|f(x) - f(y)\|$ with $L\|h(x) - h(y)\|$ for some constant L. We emphasize however that this is not exactly Lipschitz continuity since it bounds the variation of f with respect to a different—and indirect—measure of change, provided by the geometry induced implicitly by g on $\mathcal{X}$. For lack of existing terminology, we refer to such condition as *difference-bounding* with respect to another function.

Definition 3.1. *We say that a function $f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}^m$ is* ***difference-bounded*** *by $g : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}^k$ if there exists $L \in \mathbb{R}$ such that $\|f(x) - f(y)\| \leq L\|g(x) - g(y)\|$ for every $x, y \in \mathcal{X}$.*

Note, however, that imposing such a global condition might be undesirable in practice. The data arising in machine learning applications often lies on low dimensional manifolds of irregular shape. Requiring the function's adaptability to be uniformly bounded throughout the space of inputs might result in excessive and unnecessary constraining. Furthermore, in our case we are interested in having θ be consistent *for neighboring inputs*. Thus, we seek instead a *local* notion of stability. Analogous to the local Lipschitz condition, we propose a pointwise, neighborhood-based version of Definition 3.1:

Definition 3.2. *$f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}^m$ is **locally difference bounded** by $g : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}^k$ if for every x_0 there exist $\delta > 0$ and $L \in \mathbb{R}$ such that $\|x - x_0\| < \delta$ implies $\|f(x) - f(x_0)\| \leq L\|g(x) - g(x_0)\|$.*

Note that, in contrast to Definition (3.1), this second notion of stability allows L (and δ) to depend on x_0, that is, the "Lipschitz" constant can vary throughout the space. With this, we are ready to define the class of functions which form the basis of our approach.

Definition 3.3. *Let $x \in \mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ be the input and output spaces. We say that $f : \mathcal{X} \to \mathcal{Y}$ is a **self-explaining prediction model** if it has the form*

$$f(x) = g\big(\theta_1(x)h_1(x), \ldots, \theta_k(x)h_k(x)\big) \tag{4}$$

where:

P1) g is monotone and completely additively separable

P2) For every $z_i := \theta_i(x)h_i(x)$, g satisfies $\frac{\partial g}{\partial z_i} \geq 0$

P3) θ is locally difference bounded by h

P4) $h_i(x)$ is some interpretable representation of x

P5) k is small.

In that case, for a given input x, we define the explanation of $f(x)$ to be the set $\mathcal{E}_f(x) \equiv \{(h_i(x), \theta_i(x))\}_{i=1}^k$ of basis concepts and their influence scores.

Clearly, the linear predictors that served as starting point for Definition 3.3 fall within this class of functions. Various other function classes are contained in it, as stated in the following Lemma, which we state without proof due to its simplicity.

Lemma 3.4. *The following classes of models are self-explaining:*

(i) Linear predictors (c.f. (3))

(ii) Generalized linear models

(iii) Nearest-neighbor classifiers

Naturally, the richness of this function class depends predominantly on the complexity of the functions $h(\cdot)$ and $\theta(\cdot)$. Thus, the true power of the models described in Definition 3.3 comes when $\theta(\cdot)$ (and potentially $h(\cdot)$) are realized by architectures with large modeling capacity, such as deep neural networks. This allows f to model a much richer class of functions. When at least $\theta(x)$ is realized with a neural network, we refer to f as a self-explaining neural network (SENN). If g depends on its arguments in a continuous way, f can be trained end-to-end with back-propagation. Since our aim is maintain model richness even in the case where the h_i are chosen to be trivial input feature indicators, we rely predominantly on θ for modeling capacity, furnishing it with larger, higher-capacity architectures.

It remains to discuss how the properties (P1)-(P5) in Definition 3.3 are to be enforced. The first two depend entirely on the choice of aggregating function g. Besides the trivial function $g(z_1, \dots, z_k) = \sum_i z_i$, other additive functions of the form $g(z_1, \dots, z_k) = \sum_i A_i z_i$ where A_i's values are constrained to be positive. On the other hand, the last two conditions in Definition 3.3 are subjective in the sense that they will depend on the application: what and how many basis "concepts" are adequate should be informed by the problem and goal at hand.

The only condition in Definition 3.3 that warrants further discussion is (P3): the stability of θ with respect to h. For this, let us consider what the function f would look like if the θ_i's were indeed (constant) parameters. Looking at f as a function of h, that is, $f(x) = g(h(x))$, let $z = h(x)$. Using the chain rule we see that its derivative with respect to x would have the form

$$\nabla_x f = \nabla_z f \cdot J_x h \tag{5}$$

where J_x denotes the Jacobian of h (with respect to x). At a given point x_0, we want $\theta(x_0)$ to behave as the derivative of f with respect to the concept vector $h(x)$ around x_0. Thus, we would like $\theta(x_0) \approx J_{h(x_0)}$. Although this is hard to enforce directly, we can instead use (5) and plug in this *ansatz* to obtain a proxy condition

$$\mathcal{L}_\theta(f) := \|\nabla_x f(x) - \theta(x)^T J_x h(x)\| \approx 0 \tag{6}$$

All three terms in $\mathcal{L}_\theta(f)$ can be computed, and when using differentiable architectures $h(\cdot)$ and $\theta(\cdot)$, we can obtains gradients with respect to (6) through automatic differentiation and thus use it as a regularization term in the optimization objective. With this, we obtain a gradient-regularized objective of the form

$$\mathcal{L}_y(f(x), y) + \lambda \mathcal{L}_\theta(f) \tag{7}$$

where $\mathcal{L}_y(f(x), y)$ is the classification loss and λ a parameter that trades off performance against stability—and therefore, interpretability— of $\theta(x)$.

4 Learning interpretable basis concepts

Raw inputs are the natural basis for interpretability when the input is low-dimensional and individual features are meaningful. Yet, even in that case they might lead to brittle, noisy explanations. For example, explanations of computer vision systems based on raw pixels are prone to imperceptible artifacts in the data, often hard to analyze coherently and not robust to simple transformations such as constant shifts [Kindermans et al., 2017].

To avoid some of these shortcomings, we can instead operate higher level features. In the context of images, we might be interested in the effect of textures or shapes—rather than single pixels—on predictions. For example, in medical image processing higher-level visual aspects such as tissue ruggedness, irregularity or elongation are strong predictors of cancerous tumors, and are among the first aspects that doctors look for when diagnosing, so it seems natural that such aspects be the "units" of explanation.

Ideally, these basis concepts would be informed by expert knowledge, such as the doctor-provided features mentioned above. However, in cases where such prior knowledge is not available, we will need to learn the basis concepts. Interpretable concept learning is a challenging task in its own right [Kim et al., 2017], and as other aspects of interpretability, remains ill-defined. What is right level of abstraction for interpretability? What criteria should be imposed on them? Here again, we take a systematic approach by stating desirable conditions, and the proposing computational approaches to achieve them. A minimal set of desiderata for a basis of interpretable concepts is:

i) **Fidelity**: the representation of x via concepts $h(x)$ should preserve relevant information of it

ii) **Diversity**: concepts should be "non-overlapping" and express different aspects of the input, with each input represented in terms of only a few basis concepts.

iii) **Grounding**: each concept should have an immediate interpretation on human-understandable aspects

Here, we propose to enforce these conditions on the set of basis concepts in our self-explaining models by: (i) training h as an autoencoder, (ii) enforcing diversity through sparsity and (iii) providing interpretation on the concepts by prototyping (e.g., by providing a small set of training examples that maximally activate each concept). Learning of h is done end-to-end in conjunction with the rest of the model. If we denote by $\hat{h}(\,\cdot\,) : \mathbb{R}^l \to \mathbb{R}^n$ the decoder associated with h, and $\hat{x} := \hat{h}(h(x))$ the reconstruction of x, we use an additional penalty $\mathcal{L}_h(x, \hat{x})$ on the objective, yielding:

$$\mathcal{L}_y(f(x), y) + \lambda \mathcal{L}_\theta(f) + \xi \mathcal{L}_h(x, \tilde{x}) \tag{8}$$

This is the training objective that we use in all the experiments in the next section.

We close this section by discussing approaches to achieve desideratum (iii), i.e., the grounding of $h(x)$. The most obvious solution consists of representing each concept by the elements in the training data that maximize their value, that is, we can represent concept i through the set

$$X^i = \underset{\hat{X} \subseteq X, |\hat{X}| = l}{\operatorname{argmax}} \sum_{x \in \hat{X}} h(x)_i \tag{9}$$

where l is small, e.g. 5 or 10. A slightly different approaches would be to optimize over the input space to find the (synthetic) input that maximally activates each concept (and does not activate others). E.g. by solving for every i $\operatorname{argmax}_{x \in \mathcal{X}} h_i(x) - \sum_{j \neq i} h_j(x)$. Alternatively, when available, one might want to represent concepts via their learnt weights—e.g., by looking at the filters associated with each concept in a CNN-based $h(\,\cdot\,)$. In our experiments, we use approach (9), leaving the exploration of the two other approaches for future work.

5 Experimental Results

5.1 Data and Model Details

5.1.1 Datasets

MNIST. We use the original MNIST dataset with standard mean and variance normalization, using 10% of the training split for validation.

Breast Cancer. Our second benchmark is the popular Breast Cancer Wisconsin (Diagnostic) Dataset from the UCI machine learning repository [Lichman, 2013]: a binary classification task with tissue imaging measurements and an associated cancer diagnosis label. In contains 569 examples and 32 continuous attributes. We preprocess the data by scaling and use $(80\%, 10\%, 10\%)$ train, validation and test splits.

COMPAS Scores. The COMPAS Recidivism Risk Score dataset released by Propublica. It is a publicly available[1] dataset consisting of criminal *recidivism* ("relapse") risk scores produced by COMPAS, a proprietary algorithm by a private company (Northpointe) that is currently used in the Criminal Justice System to aid in bail granting decisions. Propublica's study showing racial biased scores sparked a flurry of interest in the COMPAS algorithm both in the media and in the fairness in machine learning community [Grgic-Hlaca et al., 2018, Zafar et al., 2017]. We preprocess the data by rescaling the ordinal variable `Number_of_priors` to the range $[0, 1]$. The data contains several inconsistent examples, so we filter out examples whose label differs from a strong (80%) majority of other identical examples.

5.1.2 Architectures

The architectures used in each task are summarized below, where CL/FC stand for convolutional and fully-connected layers, respectively. Note that in every case we use more complex architectures for the parametrizer than the concept encoder.

	COMPAS/Cancer	MNIST
$h(\cdot)$	$h(x) = x$	2×CL→ FC(16,10)
$\theta(\cdot)$	$\text{FC}(10, 5, 5, 1)$	LeNet+DropOut
$g(\cdot)$	sum	sum

In all cases, we train using the Adam optimizer with initial learning rate $l = 2 \times 10^{-4}$ and, whenever learning $h(\cdot)$, sparsity strength parameter $\xi = 2 \times 10^{-5}$.

5.2 Evaluation Criteria

The notion of interpretability is notorious for eluding easy quantification [Doshi-Velez and Kim, 2017]. Here, however, the motivation in Section 2 produced a set of desiderata according to which we can validate our models. Throughout this section, we base the evaluation on four main criteria:

(i) **Performance**: *How well do our models perform at task of interest compared to their non-modular, non interpretable counterparts?*

[1] `github.com/propublica/compas-analysis/`

In addition, we evaluate *interpretability* through:

(ii) **Consistency**: *Does $\theta(x)$ really behave as relevance?*

(iii) **Stability**: *How consistent are the explanations for similar/neighboring examples?*

(iv) **Intelligibility**: *How understandable are the explanations? Do they provide useful insights?*

Below, we discuss methods for evaluating (ii) and (iii). Naturally, we can quantify (i) with standard accuracy metrics, while for (iv) we provide several examples of explanations for qualitative assessment.

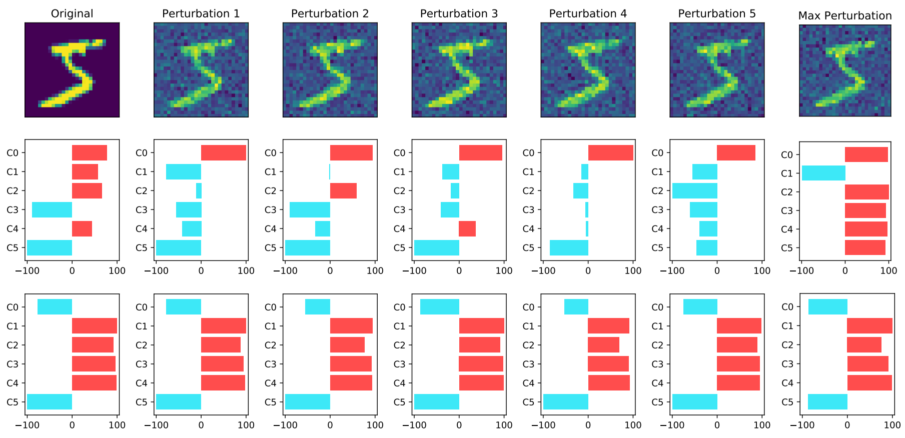
5.2.1 Consistency

Assessing whether the value of θ_i does indeed correspond to the importance of concept $h(x)_i$ requires a reference "true" notion of influence to compare against. A common approach to quantifying influence in the interpretability literature relies on *removing* aspects of the input and observing the effect on the output. Depending on the structure of the model and the type of input data, the notion of *removal* might be ambiguous, and thus previous work often relies on heuristic choices, e.g., substituting input pixels by random noise, white or averaged pixels. Although we could use a similar approach when h is chosen to be the identity (i.e., the concepts are the inputs), in the case were the concepts are learnt it is not clear what these perturbations would correspond to. Instead, we leverage the additive structure of our model, which allows us to *remove* inputs by setting their coefficient to zero. This intervention has the effect of predicting with that concept "missing". We can then use the change in probability of a particular class (e.g., the one with largest probability) to quantify the importance of that concept towards that class. Formally, let $y^{(i \to 0)}$ for $i \in \{1, \dots, k\}$ represent the output of the model when the coefficient θ_i is set to 0 in the forward pass. We quantify the *actual* relevance of concept h_i as the drop in probability of the predicted class: $y_i - y^{(i \to 0)}$. Although there is no reason why the magnitude of these values should be the same as the θ_i's, we would expect their order to be similar if they are indeed behaving as relevance scores, that is, their relative ordering should be preserved. A similar approach has been used by Samek et al. [2017] and Arras et al. [2017].

5.2.2 Stability

As argued in Section 3, we seek a function θ that varies slowly with respect to h, in the sense of Definition 3.2. Thus, we can quantify this stability by estimating, for a given input x and neighborhood size ϵ, the bound L of that definition. Then, we can estimate the overall stability of our method by analyzing the distribution of bounds L_i for every test example x_i. The most straight-forward way to estimate this quantity is

$$\hat{L}_i = \operatorname*{argmax}_{x_j \in B_\epsilon(x_i)} \frac{\|\theta(x_i) - \theta(x_j)\|_2}{\|h(x_i) - h(x_j)\|_2} \qquad (10)$$



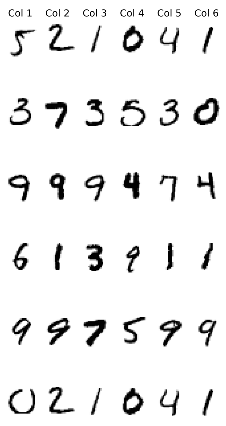


Figure 2: **Left**: The effect of gradient-regularization on explanation stability. The unregularized version (second row) produces highly variable, sometimes contradictory, explanations for slight perturbations of the same input. The regularized version ($\lambda = 2 \times 10^{-4}$) provides substantially more robust explanations. **Right**: Prototypes for the 6 learnt concepts (rows).

To compute this, we leverage the fact that our models are fully differentiable and expressed via computational graphs. In similar spirit to the work on adversarial examples (e.g., [Goodfellow et al., 2015]), we seek to optimize inputs in an adversarial manner. Here, however, we have full access to the model so we can simply use direct automatic differentiation and back-propagation to optimize (10) for the maximizing argument x_j. Note that this type of evaluation is not available for post-hoc explanation frameworks. In our experiments, we compute $\hat{L}_i$ by minimizing a Lagrangian relaxation (11) through backpropagation.

This *continuous* notion of local stability might not be suitable for models with discrete inputs or those where adversarial perturbations are overly restrictive (e.g., when the true data manifold has regions of flatness in some dimensions). In such cases, we can instead define a (weaker) empirical notion of stability based on finite sample neighbors. Let $X = \{x_i\}_{i=1}^n$ denote a sample of points. For any $x \in X$, we define its ϵ-neighborhood within Xt o be $\mathcal{N}_\epsilon(x) = \{x' \in X \mid \|x - x'\| \leq \epsilon\}$. The notion of interest is then

$$\hat{L}_i = \underset{x_j \in \mathcal{N}_\epsilon(x_i)}{\operatorname{argmax}} \frac{\|\theta(x_i) - \theta(x_j)\|_2}{\|h(x_i) - h(x_j)\|_2} \tag{11}$$

Computation of this quantity, unlike (11), is trivial since it operates only over a finite sample of points. To obtain an objective measure of stability, we compute this quantity on the test set.

5.3 MNIST

We observed that any reasonable choice of parameters in our model leads to very low test prediction error ($< 3\%$). Since these variations are within the margin of error, we focus here instead on the evaluation of explanations for this dataset. We first evaluate SENN models trained on the MNIST dataset in terms of consistency (c.f. Section 5.2.2).

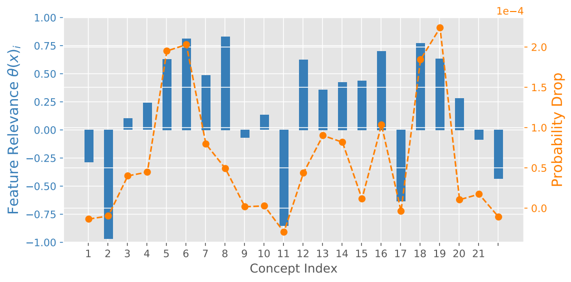


Figure 3: Explanation consistency for a SENN predictor on MNIST with 20 concepts and regularization $\lambda = 1 \times 10^{-4}$.

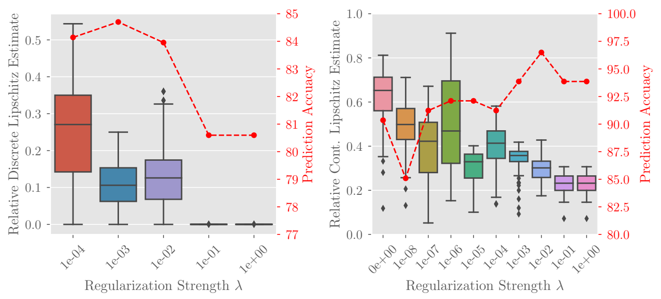


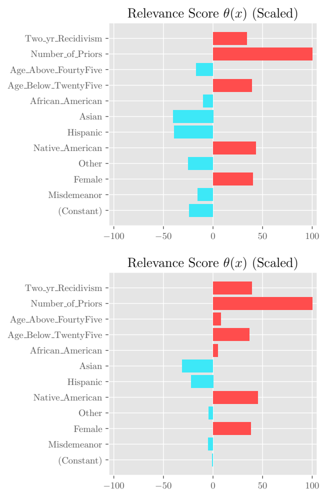
Figure 4: Effect of gradient regularization on prediction performance and local consistency (as per (11) and (10), respectively) on the COMPAS (left) and Cancer (right) datasets.

The relevance scores and concept importance as measured by probability drop for a single prediction is shown in Figure 3. We compute these profiles over the full test set and measure their discrepancy using label ranking loss. The improvement in mean ranking loss (from 26% to 54%) obtained by adding regularization shows that the proposed gradient penalty indeed enforces faithfulness of the $\theta(x)_i$'s as relevance scores.

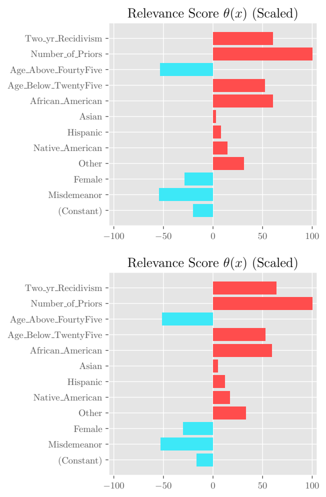
Next, we investigate the stability of the relevance scores. Figure 2 shows a randomly selected input and the explanations produced by two version of our model for 5 random and one adversarial perturbation (the maximizer of (10)).

5.4 COMPAS predictions

With default parameters, our SENNmodel achieves an accuracy of 82.02% on the test set, compared to 78.54% for a baseline logistic classification model. The relatively low performance of both methods is due to the problem of inconsistent examples mentioned above. Since most of the variables in this dataset are discrete, and continuous perturbations are not really meaningful, we use the discrete version of the local difference-bounding condition (11) to quantify consistency. We compare this quantity and the task accuracy for various regularization parameters in Figure 5. As expected, larger values of λ are correlated with lower variation bounds, at the expense of slightly worse prediction performance. Next, we qualitatively evaluate the consistency of the two model's explanations for similar examples on the test set. Following the inspiration of Propublica's original study (racial



(a) Unregularized



(b) Gradient-regularized

Figure 5: Prediction explanation for two individuals differing in only one variable (`African_American`) in the COMPAS dataset. The method trained with gradient regularization (column b) yields more consistent explanations.

bias), we choose two examples which have identical values for all variables except the (boolean) `African_American`. The explanations produced by the two models are shown in Figure 5. Confirming the quantitative results, the unregularized models' relevance scores $\theta(x)$ have little coherence for these two almost identical inputs.

5.5 Cancer

As with previous experiments, our models are able to achieve competitive performance on this dataset for most parameter configurations. Since the input features are continuous, in this case we use 10 to evaluate the local consistency around each point in the test set. The tradeoff between stability and prediction accuracy through the choice of regularization parameter λ is shown in Figure 4 (right). Somewhat surprisingly, in this case we observe an important boost in performance brought by the gradient penalty, likely caused by the additional regularization it imposes.

6 Related Work

Interpretability methods for neural networks. Various methods for computing relevances of inputs on predictions of neural networks exist, such as Sensitivity Analysis [Simonyan et al., 2014], Visualization with Deconvolutional Networks [Zeiler and Fergus, 2014], Layer-wise Relevance Propagation [Bach et al., 2015], to name a few. For additional information on these approaches, we refer the reader to excellent survey by Montavon et al. [2017]. These methods have in common that they do not modify existing architectures, instead relying on a-posteriori computations to reverse-engineer importance values or sensitivities of inputs. Our approach differs both in what it considers the *unit* of explanation (general concepts, not necessarily raw inputs) and how it uses it, with our model intrinsically relying on the relevance scores it produces to make predictions, obviating the need for additional computation. More related to our approach is the method of Lei et al. [2016], who propose a neural network architecture for text classification which "justifies" its predictions by means of selecting relevant tokens in the input text. This approach differs from our method in that the downstream processing of this *interpretable* representation is a complex neural network, so the method is transparent as to *what* aspect of the input it uses for prediction, but not *how* it uses them.

Explanations through concepts and prototypes. Li et al. [2018] propose an interpretable neural network architecture based on prototypes. The prediction is based on the similarity of the input to a small set of prototypes, which are learnt during training. Our approach can be understood as generalizing this approach beyond similarities to prototypes into more general interpretable concepts. In addition, our proposed method differs in how these higher-level representation of the inputs are used. More similar in spirit to our approach of explaining by means of learnable interpretable concepts is the work of Kim et al. [2017]. They propose a method for learning human interpretable concepts through additional supervision; their framework relies on having human-annotated inputs with a set of desired concepts, and finding inputs that maximally activate certain concepts in a recognition network. Thus, their approach is focused around training rather than modifying model architectures.

Residual architectures. The form of the proposed architecture is reminiscent of skip-connections Srivastava et al. [2015] and residual networks He et al. [2016], which seek training stability by adding short-cuts between layers of deep neural architectures which by-pass some number of layers. Thus, the motivation for such approaches differs substantially from this work; while skip-connections are sought for their effect on training convergence of deep networks, here the design of the model is inspired by the interpretability of simple models which act directly in inputs.

7 Discussion and future work

Interpretability and performance stand in apparent conflict in machine learning. The results of this work suggest that drawing inspiration from classic notions of interpretability to inform the design of modern complex architectures might hold the key to prove this a false dichotomy. There are various possible extensions beyond the model choices discussed here, particularly in terms of interpretable basis concepts. In terms of applications, the natural next step would be to evaluate interpretable models in more complex domains, such as multi-class image prediction or text analysis.

References

D. Alvarez-Melis and T. S. Jaakkola. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In *Proc. 2017 Conf. Empir. Methods Nat. Lang.*

Process., pages 412–421, 2017. URL https://www.aclweb.org/anthology/D17-1042.

L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek. " What is relevant in a text document?": An interpretable machine learning approach. *PLoS One*, 12(8):e0181142, 2017.

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*, 10(7):1–46, 2015. doi: 10.1371/journal.pone.0130140. URL https://doi.org/10.1371/journal.pone.0130140.

F. Doshi-Velez and B. Kim. Towards a Rigorous Science of Interpretable Machine Learning. *ArXiv e-prints*, (Ml):1–12, 2017.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. In *Int. Conf. Learn. Represent.*, 2015.

N. Grgic-Hlaca, M. B. Zafar, K. P. Gummadi, and A. Weller. Beyond Distributive Fairness in Algorithmic Decision Making: Feature Selection for Procedurally Fair Learning. *AAAI*, 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, pages 770–778, 2016.

B. Kim, J. Gilmer, F. Viegas, U. Erlingsson, and M. Wattenberg. TCAV: Relative concept importance testing with Linear Concept Activation Vectors. 2017. URL http://arxiv.org/abs/1711.11279.

P.-J. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. Schütt, S. Dähne, D. Erhan, and B. Kim. The (Un)reliability of saliency methods. *NIPS Work. Explain. Vis. Deep Learn.*, 2017.

T. Lei, R. Barzilay, and T. Jaakkola. Rationalizing Neural Predictions. In *EMNLP 2016, Proc. 2016 Conf. Empir. Methods Nat. Lang. Process.*, pages 107–117, 2016. URL http://arxiv.org/abs/1606.04155.

O. Li, H. Liu, C. Chen, and C. Rudin. Deep Learning for Case-Based Reasoning through Prototypes: A Neural Network that Explains Its Predictions. In *AAAI*, 2018. URL http://arxiv.org/abs/1710.04806.

M. Lichman. {UCI} Machine Learning Repository, 2013. URL http://archive.ics.uci.edu/ml.

S. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In *Adv. Neural Inf. Process. Syst. 30*, pages 4768—–4777, 2017. URL http://arxiv.org/abs/1705.07874.

G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digit. Signal Process.*, 2017.

M. T. Ribeiro, S. Singh, and C. Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proc. 22Nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pages 1135–1144, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939778. URL http://arxiv.org/abs/1602.04938http://doi.acm.org/10.1145/2939672.2939778.

W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K. R. Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE Trans. Neural Networks Learn. Syst.*, 28(11):2660–2673, 2017. ISSN 21622388. doi: 10.1109/TNNLS.2016.2599820.

R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. URL http://arxiv.org/abs/1610.02391.

K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Int. Conf. Learn. Represent. (Workshop Track)*, 2014.

R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. In *Adv. Neural Inf. Process. Syst.*, pages 2377–2385, 2015.

J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *arXiv Prepr. arXiv1506.06579*, 2015.

M. B. Zafar, I. Valera, M. Rodriguez, K. Gummadi, and A. Weller. From parity to preference-based notions of fairness in classification. In *Adv. Neural Inf. Process. Syst.*, pages 228–238, 2017.

M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Eur. Conf. Comput. Vis.*, pages 818–833. Springer, 2014.