A Framework to Learn with Interpretation

Jayneel Parekh [1], **Pavlo Mozharovskyi** [1], **Florence d'Alché-Buc** [1]
[1] LTCI, Telecom Paris,
Institut Polytechnique de Paris

Abstract

To tackle interpretability in deep learning, we present a novel framework to jointly learn a predictive model and its associated interpretation model. The interpreter provides both local and global interpretability about the predictive model in terms of human-understandable high level attribute functions, with minimal loss of accuracy. This is achieved by a dedicated architecture and well chosen regularization penalties. We seek for a small-size dictionary of high level attribute functions that take as inputs the outputs of selected hidden layers and whose outputs feed a linear classifier. We impose strong conciseness on the activation of attributes with an entropy-based criterion while enforcing fidelity to both inputs and outputs of the predictive model. A detailed pipeline to visualize the learnt features is also developed. Moreover, besides generating interpretable models *by design*, our approach can be specialized to provide *post-hoc* interpretations for a pre-trained neural network. We validate our approach against several state-of-the-art methods on multiple datasets and show its efficacy on both kinds of tasks.

1 Introduction

Interpretability in machine learning systems [17, 38, 46] has recently attracted a large amount of attention. This is due to the increasing adoption of these tools in every area of automated decision-making, including critical domains such as law [25], healthcare [53] or defence. Besides robustness, fairness and safety, it is considered as an essential component to ensure trustworthiness in predictive models that exhibit a growing complexity. Explainability and interpretability are often used as synonyms in the literature, referring to the ability to provide human-understandable insights on the decision process. Throughout this paper, we opt for interpretability as in [16] and leave the term explainability for the ability to provide logical explanations or causal reasoning, both requiring more sophisticated frameworks [18, 20, 48]. To address the long-standing challenge of interpreting models such as deep neural networks [47, 10, 9], two main approaches have been developed in literature: *post-hoc* approaches and "*by design* methods".

Post-hoc approaches [7, 44, 39, 49] generally analyze a pre-trained system locally and attempt to interpret its decisions. "Interpretable by design" [3, 1] methods aim at integrating the interpretability objective into the learning process. They generally modify the structure of predictor function itself or add to the loss function regularizing penalties to enforce interpretability. Both approaches offer different types of advantages and drawbacks. Post-hoc approaches guarantee not affecting the performance of the pre-trained system but are however criticized for computational costs, robustness and faithfulness of interpretations [56, 28, 5]. Interpretable systems by-design on the other hand, although preferred for interpretability, face the challenge of not losing out on performance.

Here, we adopt another angle to learning interpretable models. As a starting point, we consider that prediction (computing $\hat{y}$ the model's output for a given input) and interpretation (giving a human-understandable description of properties of the input that lead to $\hat{y}$) are two distinct but strongly related tasks. On one hand, they do not involve the same criteria for the assessment of their quality and might not be implemented using the same hypothesis space. On the other hand, we wish

Preprint. Under review.

that an interpretable model relies on the components of a predictive model to remain faithful to it. These remarks yield to a novel generic task in machine learning called Supervised Learning with Interpretation (SLI). SLI is the problem of jointly learning a pair of dedicated models, a predictive model and an interpreter model, to provide both interpretability and prediction accuracy. In this work, we present FLINT (Framework to Learn With INTerpretation) as a solution to SLI when the model to interpret is a deep neural network classifier. The interpreter in FLINT implements the idea that a prediction to be understandable by a human should be linearly decomposed in terms of attribute functions that encode high-level concepts as other approaches [4, 19]. However, it enjoys two original key features. First the high-level attribute functions leverage the outputs of chosen hidden layers of the neural network. Second, together with expansion coefficients they are jointly learnt with the neural network to enable local and global interpretations. By local interpretation, we mean a subset of attribute functions whose simultaneous activation leads to the model's prediction, while by global interpretation, we refer to the description of each class in terms of a subset of attribute functions whose activation leads to the class prediction. Learning the pair of models involves the minimization of dedicated losses and penalty terms. In particular, local and global interpretability are enforced by imposing a limited number of attribute functions as well as conciseness and diversity among the activation of these attributes for a given input. Additionally we show that FLINT can be specialized to post-hoc interpretability if a pre-trained deep neural network is available.

Key contributions:

- We present FLINT devoted to Supervised Learning with Interpretation with an original interpreter network architecture based on some hidden layers of the network. The role of the interpreter is to provide local and global interpretability that we express using a novel notion of relevance of concepts.
- We propose a novel entropy and sparsity based criterion for promoting conciseness and diversity in the learnt attribute functions and develop a simple pipeline to visualize the encoded concepts based on previously proposed tools.
- We present extensive experiments on 4 image classification datasets, MNIST, FashionMNIST, CIFAR10, QuickDraw, with a comparison with state-of-the-art approaches and a subjective evaluation study.
- Eventually, a specialization of FLINT to post-hoc interpretability is presented while corresponding numerical results are deferred to supplements.

2 Related Works

We emphasize here more on the methods relying upon a dictionary of high level attributes/concepts, a key feature of our framework. A synthetic view of this review is presented in the supplements to effectively view the connections and differences w.r.t wider literature regarding interpretability.

Post-hoc interpretations. Most works in literature focus on producing *a posteriori* interpretations for pre-trained models via input attribution. They often consider the model as a black-box [44, 39, 8, 32, 15] or in the case of deep neural networks, work with gradients to generate saliency maps for a given input [50, 51, 49, 41]. Very few post-hoc approaches rely on high level concepts. They come under the subclass of concept activation vector (CAV)-based approaches. TCAV [27] proposed to utilize human-annotated examples to represent concepts in terms of activations of a pre-trained neural network. The sensitivity of prediction to these concepts is estimated to offer an explanation. ACE [19] attempts to automate the human-annotation process by super-pixel segmentation and clustering these segments based on their perceptual similarity where each cluster represents a concept. ConceptSHAP [55] introduces the idea of "completeness" in ACE's framework. The CAV-based approaches already strongly differ from us in context of problem as they only consider post-hoc interpretations. TCAV generates candidate concepts using human supervision and not from the network itself. While ACE automates concept discovery, the concepts are less dynamic as by design they are associated to a single class and rely on being represented via spatially connected regions. Moreover, since ACE depends on using a CNN as perceptual similarity metric for image segments (regardless of aspect ratio, scale), it is limited in applicability (experimentally supported in supplement Sec. S.2).

Interpretable neural networks by design. Most works from this class learn a single model by either modifying the architecture [3], the loss functions [57, 14], or both [6, 37, 4, 12]. Hendricks et al. [23] proposed a system that jointly learns a predictor and a module that can generate *textual* explanations. GAME [36] shapes the learning problem as a co-operative game between predictor and

interpreter. However, it learns a separate local interpreter for each sample rather than a single model. The above methods do not utilize high-level concepts for interpretation and offer local interpretations, with the exception of neural additive models [2], which are currently only suitable for tabular data. Self Explaining Neural Networks (SENN) [4] presented a generalized linear model wherein coefficients are also modelled as a function of input. The linear structure is to emphasize interpretability. SENN imposes a gradient-based penalty to learn coefficients stably and other constraints to learn human understandable features. Unlike SENN, to avoid trade-off between accuracy and interpretability in FLINT, we allow the predictor to be an unrestrained neural network and jointly learn the interpreter. Interpretations are generated at a local and global level using a novel notion of relevance of attributes. Moreover, FLINT can be specialized for generating post-hoc interpretations of pre-trained networks.
Known dictionary of concepts. Some recent works have focused on different ways of utilizing a known dictionary of concepts for interpretability [26], by transforming the latent space to align with the concepts [14] or by adding user intervention as an additional feature to improve interactivity [30].

3 Learning a classifier and its interpreter with FLINT

We introduce a novel generic task called *Supervised Learning with Interpretation* (SLI). Denoting $\mathcal{X}$ the input space, and $\mathcal{Y}$ the output space, we assume that the training set $\mathcal{S} = \{(x_i, y_i)_{i=1}^N\}$ is composed of n independent realizations of a pair of random variables (X, Y) defined over $\mathcal{X} \times \mathcal{Y}$. SLI refers to the idea that the **interpretation** task differs from the **prediction** task and must be taken over by a dedicated model that depends on the predictive model to be interpreted. Let us call $\mathcal{F}$ the space of predictive models from $\mathcal{X}$ to $\mathcal{Y}$. For a given model $f \in \mathcal{F}$, we denote $\mathcal{G}_f$ the family of models $g_f : \mathcal{X} \to \mathcal{Y}$, that depend on f and are devoted to its interpretation. For sake of simplicity, an interpreter $g_f \in \mathcal{G}_f$ is denoted g, omitting the dependency on f. With these assumptions, the empirical loss of supervised learning is revisited to include explicitly an interpretability objective besides the prediction loss yielding to the following definition.

Supervised Learning with Interpretation (SLI):

$$\textbf{Problem 1}\text{: } \arg \min_{f \in \mathcal{F}, g \in \mathcal{G}_f} \mathcal{L}_{pred}(f, \mathcal{S}) + \mathcal{L}_{int}(f, g, \mathcal{S}),$$

where $\mathcal{L}_{pred}(f, \mathcal{S})$ denotes a loss term related to prediction error and $\mathcal{L}_{int}(f, g, \mathcal{S})$ measures the ability of g to provide interpretations of predictions by f.

Remark 1 *A good example of SLI is provided by the visual explanation generation method introduced by Hendricks et al. [23] which jointly learns to predict a class label and a textual explanation.*

The goal of this paper is to address Supervised Learning with Interpretation when the hypothesis space $\mathcal{F}$ is instantiated to deep neural networks and the task at hand is multi-class classification. We present a novel and general framework, called Framework to Learn with INTerpretation (FLINT) that relies on (i) a specific architecture for the interpreter model which leverages some hidden layers of the neural network network to be interpreted, (ii) notions of local and global interpretation and (iii) corresponding penalties in the loss function.

3.1 Design of FLINT

All along the paper, we take $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \{y \in \{0,1\}^C, \sum_{j=1}^C y^j = 1\}$, the set of C one-hot encoding vectors of dimension C. We set $\mathcal{F}$ to the class of deep neural networks with l hidden layers of respective dimension $d_1, \ldots, d_l$. Each element $f : \mathcal{X} \to \mathcal{Y}$ of $\mathcal{F}$ satisfies: $f = f_{l+1} \circ f_l \circ ... \circ f_1$ where $f_k : \mathbb{R}^{d_{k-1}} \to \mathbb{R}^{d_k}$, $d_0 = d, d_{l+1} = C, k = 1, ..., l+1$ is the function implemented by layer k. A network f in $\mathcal{F}$ is completely identified by its generic parameter θ_f. As for the interpreter model $g \in \mathcal{G}_f$, we propose the following original architecture which exploits the outputs of chosen hidden layers of f. Denote $\mathcal{I} = \{i_1, i_2, ..., i_T\} \subset \{1, \ldots, l\}$ the set of indices specifying the intermediate layers of network f to be accessed and chosen for the representation of input. We define $D = \sum_{t=1}^T d_{i_t}$. Typically these layers are selected from the latter layers of the network f. The concatenated vector of all intermediate outputs for an input sample x is denoted as $f_{\mathcal{I}}(x) \in \mathbb{R}^D$. Given f a network to be interpreted and a positive integer $J \in \mathbb{N}^*$, an **interpreter network** g computes the composition of a dictionary of attribute functions $\Phi : \mathcal{X} \to \mathbb{R}^J$ and an interpretable function $h : \mathbb{R}^J \to \mathcal{Y}$.

$$\forall x \in \mathcal{X}, g(x) = h \circ \Phi(x), \tag{1}$$

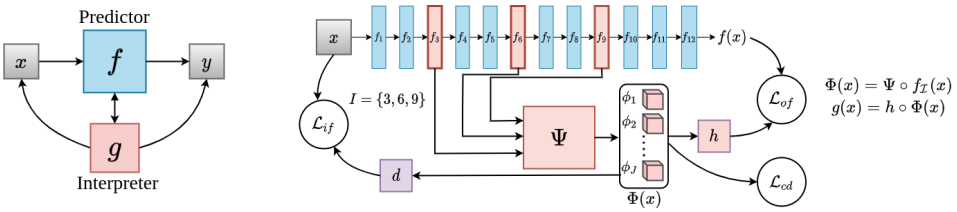


Figure 1: **(Left)** General view of FLINT. **(Right)** Instantiation of FLINT on a deep architecture.

In this work, we take: $h(\Phi(x)) := \text{softmax}(W^T \Phi(x))$ but other models like decision trees could be eligible. The **attribute dictionary** is composed of functions $\phi_j : \mathcal{X} \to \mathbb{R}^+, j = 1, \dots J$ whose non-negative images $\phi_j(x)$ can be interpreted as the activation of some high level attribute, i.e. a "concept" over $\mathcal{X}$. A key originality of the model lies in the fact that the attribute functions ϕ_j (referred to as attribute for simplicity) leverage the outputs of hidden layers of f specified by $\mathcal{I}$:

$$\forall j \in \{1, \dots, J\}, \phi_j(x) = \psi_j \circ f_{\mathcal{I}}(x) \tag{2}$$

where each $\psi_j : \mathbb{R}^D \to \mathbb{R}^+$ operates on the accessed hidden layers. Here, the set of functions $\psi_j, j = 1, \dots J$ is defined to form a shallow network Ψ (around 3 layers) whose output is $\Psi(f_{\mathcal{I}}(x)) = \Phi(x)$ (example architecture in Fig. 1). Interestingly, ϕ_j are defined over $\mathcal{X}$ and as a consequence can be interpreted in the input space which is the most meaningful for the user (see Sec. 4). For sake of simplicity, we denote $\Theta_g = (\theta_\Psi, \theta_h)$ the specific parameters of this model, while the parameters devoted to the computation of $f_{\mathcal{I}}(x)$ are shared with f.

3.2 Interpretation in FLINT

The interpreter being defined, we need to specify its expected role and corresponding interpretability objective. In FLINT, interpretation is seen as an additional task besides prediction. We are interested by two kinds of interpretation, one at the global level that helps to understand which attribute functions are useful to predict a class and the other at the local level, that indicates which attribute functions are involved in prediction of a specific sample. As a preamble, note that, to interpret a local prediction $f(x)$, we require that the interpreter output $g(x)$ matches $f(x)$. When the two models disagree, we provide a way to analyze the conflictual data and possibly raise an issue about the confidence on the prediction $f(x)$ (see Supplementary Sec. S.2). To define local and global interpretation, we rely on the notion of relevance of an attribute.

Given an interpreter with parameter $\Theta_g = (\theta_\Psi, \theta_h)$ and some input x, the **relevance score** of an attribute ϕ_j is defined regarding the prediction $g(x) = f(x) = \hat{y}$. Denoting $\hat{y} \in \mathcal{Y}$ the index of the predicted class and $w_{j,\hat{y}} \in W$ the coefficient associated to this class, the contribution of attribute ϕ_j to unnormalized score of class $\hat{y}$ is $\alpha_{j,\hat{y},x} = \phi_j(x).w_{j,\hat{y}}$. The relevance score is computed by normalizing contribution α as $r_{j,x} = \frac{\alpha_{j,\hat{y},x}}{\max_i |\alpha_{i,\hat{y},x}|}$. An attribute ϕ_j is considered as relevant for a local prediction if it is both activated and effectively used in the linear (logistic) model. The notion of relevance of an attribute for a sample is extended to its "overall" importance in the prediction of any class c. This can be done by simply averaging relevance scores from local interpretations over a random subset or whole of the training set $\mathcal{S}$, where predicted class is c. Thus, we have: $r_{j,c} = \frac{1}{|\mathcal{S}_c|} \sum_{x \in \mathcal{S}_c} r_{j,x}$, $\mathcal{S}_c = \{x \in \mathcal{S} | \hat{y} = c\}$. Now, we can introduce the notions of local and global interpretations that the interpreter will provide.

Definition 1 (Global and Local Interpretation) *For a prediction network f, the* **global interpretation** *$G(g, f)$ provided by an interpreter g, is the set of class-attribute pairs (c, ϕ_j) such that their global relevance $r_{j,c}$ is greater than some threshold $1/\tau, \tau > 1$. A* **local interpretation** *for a sample x provided by an interpreter g of f denoted $L(x, g, f)$ is the set of attribute functions ϕ_j with local relevance score $r_{j,x}$ greater than some threshold $1/\tau, \tau > 1$.*

It is important to note that these definitions do not prejudge the quality of local and global interpretations. Next, we convert desirable properties of the interpreter into specific loss functions.

3.3 Learning by imposing interpretability properties

Although converting desirable interpretability properties into losses is shared by several by-design approaches [11, 39], there is no consensus on these properties. We propose below a minimal set of penalties which are suitable for the proposed architecture and sufficient to provide relevant interpretations.

Fidelity to Output. The output of the interpreter $g(x)$ should be "close" to $f(x)$ for any x. This can be imposed through a cross-entropy loss:

$$\mathcal{L}_{of}(f, g, \mathcal{S}) = -\sum_{x \in \mathcal{S}} h(\Psi(f_{\mathcal{I}}(x)))^T \log(f(x))$$

Conciseness and Diversity of Interpretations. For any given sample x, we wish to get a *small* number of attributes in its associated local interpretation. This property of *conciseness* should make the interpretation easier to understand due to fewer attributes to be analyzed and promote the "high-level" character in the encoded concepts. However, to encourage better use of available attributes we also expect activation of multiple attributes across many randomly selected samples. We refer to this property as *diversity*. This is also important to avoid the case of attribute functions being learnt as class exclusive (for eg. reshuffled version of class logits). To enforce these conditions we utilize notion of entropy defined for real vectors proposed by Jain et al [24] to solve problem of efficient image search. For a real-valued vector v, the entropy is defined as $\mathcal{E}(v) = -\sum_i p_i \log(p_i)$, $p_i = \exp(v_i)/(\sum_i \exp(v_i))$.

Conciseness is promoted by minimizing $\mathcal{E}(\Psi(f_{\mathcal{I}}(x)))$ and diversity is promoted by maximizing entropy of average $\Psi(f_{\mathcal{I}}(x))$ over a mini-batch. Note that this can be seen as encouraging the interpreter to find a sparse and diverse coding of $f_{\mathcal{I}}(x)$ using the function Ψ. Since entropy-based losses have inherent normalization, they do not constrain the magnitude of the attribute activation. This often leads to poor optimization. Thus, we also minimize the ℓ_1 norm $\|\Psi(f_{\mathcal{I}}(x))\|_1$ (with hyperparameter η) to avoid it. Note that ℓ_1-regularization is a common tool to encourage sparsity and thus conciseness, however we show in the experiments that entropy provides a more effective way.

$$\mathcal{L}_{cd}(f, g, \mathcal{S}) = -\mathcal{E}(\bar{\Phi}_{\mathcal{S}}) + \sum_{x \in \mathcal{S}} \mathcal{E}(\Psi(f_{\mathcal{I}}(x))) + \sum_{x \in \mathcal{S}} \eta \|\Psi(f_{\mathcal{I}}(x))\|_1 \quad \text{with} \quad \bar{\Phi}_{\mathcal{S}} = \frac{1}{|\mathcal{S}|} \sum_{x \in \mathcal{S}} \Psi(f_{\mathcal{I}}(x))$$

Fidelity to Input. To encourage encoding high-level patterns related to input in $\Phi(x)$, we use a decoder network $d : \mathbb{R}^J \rightarrow \mathcal{X}$ that takes as input the dictionary of attributes $\Psi(f_{\mathcal{I}}(x))$ and reconstructs x. A similar penalty has previously been applied by [4].

$$\mathcal{L}_{if}(f, g, d, \mathcal{S}) = \sum_{x \in \mathcal{S}} (d(\Psi(f_{\mathcal{I}}(x))) - x)^2$$

Note that one can modify $\mathcal{L}_{if}$ with other reconstruction losses as well (such as ℓ_1-reconstruction).

Given the proposed loss terms, the loss for interpretability writes as follows:

$$\mathcal{L}_{int}(f, g, d, \mathcal{S}) = \beta \mathcal{L}_{of}(f, g, \mathcal{S}) + \gamma \mathcal{L}_{if}(f, g, d, \mathcal{S}) + \delta \mathcal{L}_{cd}(f, g, \mathcal{S})$$

where β, γ, δ are non-negative hyperparameters. The total loss to be minimized $\mathcal{L} = \mathcal{L}_{pred} + \mathcal{L}_{int}$, where the prediction loss, $\mathcal{L}_{pred}$, is the well-known cross-entropy loss.

Let us denote $\Theta = (\theta_f, \theta_d, \theta_\Psi, \theta_h)$ the parameters of these networks. Learning the models f, Ψ, h and d boils down to learning Θ. In practice, introducing all the losses at once often leads to very poor optimization. Thus, we follow the procedure described in Alg. 1. We train the networks with $\mathcal{L}_{pred}, \mathcal{L}_{if}$ for the first two epochs and gain a reasonable level of accuracy. From the third epoch we introduce $\mathcal{L}_{of}$ and from the fourth epoch we introduce $\mathcal{L}_{cd}$ loss.

4 Understanding encoded concepts in FLINT

Once the predictor and interpreter are jointly learnt, interpretation can be given at the global and local levels as in Def. 1. A key component to grasp the interpretations is to understand the concept encoded by each individual attribute function ϕ_j, previously defined in Eq. 2. In this work, we focus on image classification and propose to represent an encoded concept as a set of visual patterns in the **input space** which highly activate ϕ_j. We present a pipeline to generate visualizations for global and local interpretation by adapting various previously proposed tools [4, 40].

Algorithm 1 Learning algorithm for FLINT

Input: $\mathcal{S}$ & parameters $\Theta = (\theta_f, \theta_d, \theta_\Psi, \theta_h)$ & hyperparameters: $\beta_0, \gamma_0, \delta_0, \eta_0$ & number of batches B & number of training epochs N_{epoch}.
Random initialization of parameter Θ_0
$\Theta_1 \leftarrow$ Train $(\mathcal{S}, \Theta_0, \beta = 0, \gamma_0, \delta = 0, \eta = 0, B, 2)$ {% Trains 2 epochs with $\mathcal{L}_{pred}, \mathcal{L}_{if}$}
$\Theta_2 \leftarrow$ Train $(\mathcal{S}, \Theta_1, \beta = \beta_0, \gamma_0, \delta = 0, \eta = 0, B, 1)$ {% Trains 1 epoch with $\mathcal{L}_{pred}, \mathcal{L}_{if}, \mathcal{L}_{of}$}
$\hat{\Theta} \leftarrow$ Train $(\mathcal{S}, \Theta_2, \beta_0, \gamma_0, \delta_0, \eta_0, B, N_{epoch} - 3)$ {% Trains with all losses}
Output: $\hat{\Theta} = (\hat{\theta}_f, \hat{\theta}_d, \hat{\theta}_\Psi, \hat{\theta}_h)$

Algorithm 2 Visualization of global interpretation

Input: (class,attribute):(c, ϕ_j) & subset size:l & training set:$\mathcal{S}_n$ & AM+PI params:$(\lambda_\phi, \lambda_{tv}, \lambda_{bo})$
$\mathcal{S}_c = \{x | (x, c) \in \mathcal{S}_n\}$
MAS$(c, \phi_j, l) \leftarrow \arg\max_{\mathcal{M} \subset \mathcal{S}_c, |\mathcal{M}|=l} \sum_{x_i \in \mathcal{M}} \phi_j(x)$
FOR $x_k \in$ MAS(c, ϕ_j, l)
$x^k_{vis} \leftarrow$ AM+PI$(x_k, \lambda_\phi, \lambda_{tv}, \lambda_{bo})$
ENDFOR
Output:$\{x^1_{vis}, \ldots, x^l_{vis}\}$, MAS$(c, \phi_j, l)$

Visualization of global interpretation. Given any class-attribute pair (c, ϕ_j) in the global interpretation $G(g, f)$, we first select a small subset of training samples from class c that maximally activate ϕ_j. This set of samples is referred to as maximum activating samples and denoted MAS(c, ϕ_j, l) where l is the size of the subset (chosen as 3 in the experiments). Although, MAS reveal some information about the encoded concept, it might not be apparent what aspect of these samples causes activation of ϕ_j. We thus propose further analyzing each element in MAS through tools that enhance the detected concept. This results in a much better understanding. The primary tool we employ is a modified version of activation maximization [40], which we refer to as *activation maximization with partial initialization* (AM+PI).

Given a maximum activating sample $x' \in$ MAS(c, ϕ_j, l), the key idea behind AM+PI is to synthesize appropriate input via optimization, that maximally activates ϕ_j. We thus optimize a common activation maximization objective [40]: $\arg\max_x \lambda_\phi \phi_j(x) - \lambda_{tv}\text{TV}(x) - \lambda_{bo}\text{Bo}(x)$, where TV$(.)$, Bo$(.)$ are regularization terms. However, we initialize the procedure by low-intensity version of sample x'. This makes the optimization easier with the detected concept weakly present in the input. This also allows the optimization to "fill" the input to enhance the encoded concept. As an output, we obtain a map adapted to x', that strongly activates ϕ_j. Complete details of the AM+PI procedure are given in supplementary (Sec. S.2). Visualization of a class-attribute pair is summarized in Alg. 2. Alternative useful tools are discussed in the supplementary (Sec. S.2).

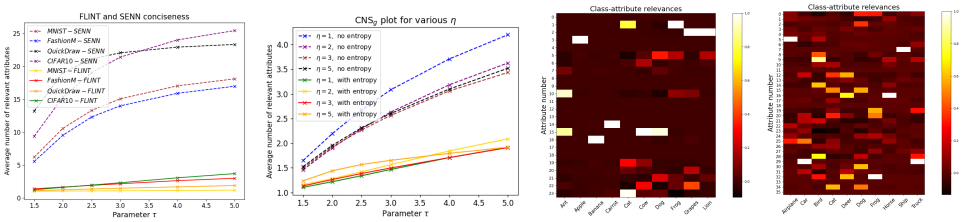
Local analysis. Given any test sample x_0, one can determine its local interpretation $L(x_0, f, g)$, the set of relevant attribute functions accordingly to Def. 1. To visualize a relevant attribute $\phi_j \in L(x_0, f, g)$, we can repeat the AM+PI procedure with initialization using low-intensity version of x_0 to enhance concept detected by ϕ_j in x_0. Note that the understanding built about any attribute function ϕ_j via global analysis, although not essential, can still be helpful to understand the generated AM+PI maps during local analysis, as these maps are generally similar.

5 Numerical Experiments for FLINT

Datasets and Networks. We consider 4 datasets for experiments, MNIST [35], FashionMNIST [54], CIFAR-10 [31], and a subset of QuickDraw dataset [21]. Our experiments include 2 kinds of architectures for predictor f: (i) LeNet-based [34] network for MNIST, FashionMNIST, and (ii) ResNet18-based [22] network for QuickDraw, CIFAR. We select one intermediate layer for LeNet based network and two for ResNet based networks, from the last few convolutional layers as they are expected to capture higher-level features. We set the number of attributes $J = 25$ for MNIST, FashionMNIST, $J = 24$ QuickDraw and $J = 36$ for CIFAR. Further details about the QuickDraw subset, precise architecture, hyperparameter choices (with reasons for choice of hidden layers, number of attributes) and optimization details are available in supplementary (Sec. S.2)

	Accuracy (in %)					Fidelity (in %)		
	BASE-f	SENN	PrototypeDNN	FLINT-f	FLINT-g	LIME	VIBI	FLINT-g
MNIST	98.9±0.1	98.4±0.1	**99.2**	98.9±0.2	98.3±0.2	95.6±0.4	96.6±0.7	**98.7±0.1**
FashionMNIST	90.4±0.1	84.2±0.3	90.0	**90.5±0.2**	86.8±0.4	67.3±1.3	88.4±0.3	**91.5±0.1**
CIFAR10	84.7±0.3	77.8±0.7	–	**84.5±0.2**	84.0±0.4	31.5±0.9	65.5±0.3	**93.2±0.2**
QuickDraw	85.3±0.2	85.5±0.4	–	**85.7±0.3**	85.4±0.1	76.3±0.1	78.6±0.4	**90.8±0.4**

Table 1: Results for accuracy (in %) and fidelity to FLINT-f on different datasets. BASE-f is system trained with just accuracy loss. FLINT-f, FLINT-g denote the predictor and interpreter trained in our framework. Mean and standard deviation of 4 runs for each system are reported



(a) (b) (c)

Figure 2: (a) Conciseness comparison of FLINT and SENN. (b) Effect of entropy losses on conciseness of ResNet for QuickDraw for various ℓ_1-regularization levels. (c) Global class-attribute relevances $r_{j,c}$ for QuickDraw (Left) and CIFAR10 (Right). 24 class-attribute pairs for QuickDraw and 32 pairs for CIFAR10 have relevance $r_{j,c} > 0.2$.

5.1 Quantitative evaluation of FLINT

We evaluate and compare our model with other state-of-the-art systems regarding accuracy and interpretability. The evaluation metrics for interpretability [16] are defined to measure the effectiveness of the losses proposed in Sec. 3.3. Our primary method for comparison, wherever applicable, is SENN, as it is an interpretable network by design with same units for interpretation as FLINT. Other baselines include PrototypeDNN [37] for predictive performance, LIME [44] and VIBI [8] for fidelity of interpretations. Details of their implementation are in supplementary (Sec. S.2).

Predictive performance of FLINT. There are two goals to validate related to predictor trained with FLINT (denoted FLINT-f), (i) Jointly training f with g and backpropagating loss term $\mathcal{L}_{int}$ does not negatively impact performance, and (ii) The achieved performance is comparable with other similar interpretable by-design models. For the former we compare the accuracy of FLINT-f with same predictor architecture trained just with $\mathcal{L}_{pred}$ (denoted by BASE-f). For the latter goal we compare accuracy of FLINT-f with accuracy of SENN and another interpretable network by design PrototypeDNN [37] that does not use input attribution for interpretations. Note that PrototypeDNN requires non-trivial changes to the model for running on more complex datasets, CIFAR10 and QuickDraw. To avoid any unfair comparison we skip these results. The accuracies are reported in Tab. 1. They indicate that training f within FLINT does not result in any significant accuracy loss on any dataset. Also, FLINT is competitive with other interpretable by-design models.

Fidelity of Interpreter. The fraction of samples where prediction of a model and its interpreter agree, i.e predict the same class, is referred to as *fidelity*. It is a commonly used metric to measure how well an interpreter approximates a model [8, 33]. Note that, typically, for interpretable by design models, fidelity cannot be measured as they only consider a single model. However, to validate that the interpreter trained with FLINT (denoted as FLINT-g) achieves a reasonable level of agreement with FLINT-f, we benchmark its fidelity against a state-of-the-art black-box explainer VIBI [8] and a traditional method LIME [44]. The results for this are provided in Tab. 1 (last three columns). FLINT-g consistently achieves higher fidelity. Even though it is not a fair comparison as other systems are black-box explainers and FLINT-g accesses intermediate layers, they clearly show that FLINT-g demonstrates high fidelity to FLINT-f.

Conciseness of interpretations. We evaluate conciseness by measuring the average number of *important* attributes in generated interpretations. For a given sample x, it can be computed as number

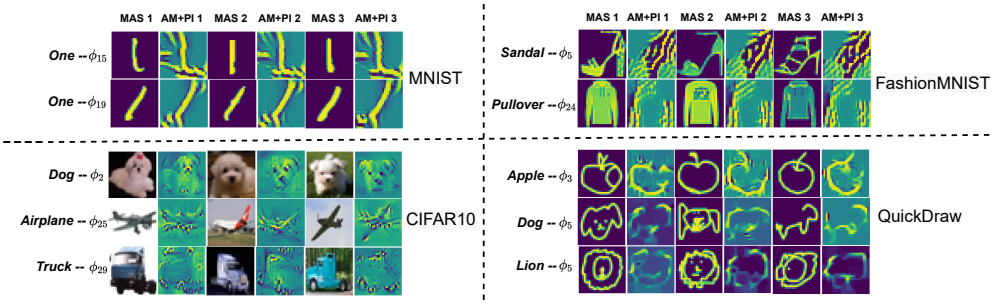


Figure 3: Example class-attribute pair analysis on all datasets, with global relevance $r_{j,c} > 0.2$. Each row contains 3 MAS with corresponding AM+PI outputs

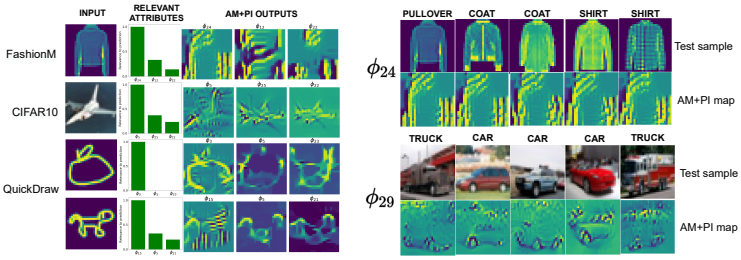


Figure 4: **(Left)** Local interpretations for test samples. Top 3 attributes with corresponding AM+PI output are shown. True labels for inputs are: Pullover, Airplane, Apple, Dog. **(Right)** Examples of attribute functions detecting same part across various test samples. For each sample, their relevance is greater than 0.8. True labels of samples indicated above them.

of attributes ϕ_j with $r_{j,x}$ greater than a threshold $1/\tau, \tau > 1$, i.e. $\text{CNS}_{g,x} = |\{j : |r_{j,x}| > 1/\tau\}|$. For different thresholds $1/\tau$, we compute the mean of $\text{CNS}_{g,x}$ over test data to estimate conciseness of g, CNS_g. Lower conciseness indicates need to analyze a lower number of attributes on an average. SENN is the only other system for which this curve can be computed. We thus compare the conciseness of SENN with FLINT on all four datasets. Fig. 2a depicts the same. It can be easily observed that FLINT produces lot more concise interpretations compared to SENN. Moreover, SENN even ends up with majority of concepts being considered relevant for lower thresholds (higher τ).

Entropy vs ℓ_1 regularization. We validate the effectiveness of entropy losses by computing conciseness curve at various levels of ℓ_1 regularization strength, with and without entropy, for ResNet with QuickDraw. This is reported in Fig. 2b. The figure confirms that using the entropy-based loss is more effective in inducing conciseness of explanations compared to using just ℓ_1-regularization, with the difference being close to use of 1 attribute less when entropy losses are employed.

Importance of attributes. Additional experiments evaluating meaningfulness of attributes by shuffling them and observing the effect (for FLINT and SENN) are given in supplementary (Sec. S.2).

5.2 Qualitative analysis

Global interpretation. Fig. 2c depicts the generated global relevances $r_{j,c}$ for all class-attribute pairs on QuickDraw and CIFAR. Each class-attribute pair with 'high' relevance needs to be analyzed as part of global analysis. Some example class-attribute pairs, with high relevance, are visualized in Fig. 3. For each pair we select MAS of size 3 and also show their AM+PI outputs. As mentioned before, simply analyzing MAS reveals useful information about the encoded concept. For instance, based on MAS, ϕ_{15}, ϕ_{19} on MNIST, relevant for class 'One', clearly seem to activate for vertical and diagonal strokes respectively. However, AM+PI outputs give deeper insights about the concept by revealing more clearly what parts of input activate an attribute function. For eg., while MAS indicate that ϕ_5 on FashionMNIST activates for heels (one type of 'Sandal'), ϕ_2 on CIFAR10 activates for white dogs, it is not clear what part the attribute focuses on. AM+PI outputs indicate that ϕ_2 focuses on the area around eyes and nose (the most enhanced regions), ϕ_5 primarily detects a thin diagonal stroke of the heel surrounded by empty space. AM+PI outputs generally become even more important for attributes relevant for multiple classes. One such example is the function ϕ_5 on QuickDraw,

relevant for both 'Dog' and 'Lion'. It activates for very similar set of strokes for all samples, as indicated by AM+PI maps. For 'Dog' this corresponds to ears and mouth and for 'Lion' it corresponds to the mane. Other such attribute functions in the figure include ϕ_{24} on FashionMNIST, relevant for 'Pullover', 'Coat' and 'Shirt' which detects long sleeves and ϕ_{29} on CIFAR10, relevant for 'Trucks', 'Cars' and primarily detects wheels and parts of upper body. Further visualizations including those of other relevant classes for ϕ_{24}, ϕ_{29} and global relevances are available in supplementary (Sec. S.2).

Local interpretation. Fig. 4 (left) displays the local interpretation visualizations for test samples. f and g both predict the true class in all the cases. We show the top 3 relevant attributes to the prediction with their relevances and their corresponding AM+PI outputs. Based on the AM+PI outputs it can be observed that the attribute functions generally activate for patterns corresponding to the same concept as inferred during global analysis. This can be easily seen for attribute functions present in both Fig. 3, 4 (left). This is further illustrated by Fig. 4 (right) where we illustrate AM+PI outputs for two attributes from Fig. 3. These functions are relevant for more than one class and detect the same concept across various test samples, namely long sleeves for ϕ_{24} and primarily wheels for ϕ_{29}.

5.3 Subjective evaluation

We conducted a *survey based subjective evaluation* with QuickDraw dataset for FLINT with 20 respondents. We selected 10 attributes, covering 17 class-attribute pairs from the QuickDraw dataset. For each attribute we present the respondent with our visualizations (3 MAS and AM+PI outputs) for each of its relevant classes along with a textual description. We ask them if the description meaningfully associates to patterns in the AM+PI outputs. They indicate level of agreement with choices: Strongly Agree (SA), Agree (A), Disagree (D), Strongly Disagree (SD), Don't Know (DK). Descriptions were manually generated by our understanding of encoded concept for each attribute. 40% incorrect descriptions were carefully included to ensure informed responses. These were forcefully related to the classes shown to make them harder to identify. **Results** – for correct descriptions: 77.5% – SA/A, 10.0% – DK, 12.5% – D/SD. For incorrect descriptions: 83.7% – D/SD, 7.5% – DK, 8.8% – SA/A. These results clearly indicate that concepts encoded in FLINT's learnt attributes are understandable to humans. Survey details are given in supplementary (Sec. S.2).

6 Specialization of FLINT to post-hoc interpretability

While interpretability by design is the primary goal of FLINT, it can be specialized to provide a *post-hoc* interpretation when a classifier $\hat{f}$ is already available. The **Post-hoc interpretation learning** (see for instance [44]) comes as a special case of SLI and is defined as follows. Given a classifier $\hat{f} \in \mathcal{F}$ and a training set $\mathcal{S}$, the goal is to build an interpreter of $\hat{f}$ by solving:

$$\textbf{Problem 2}\text{: } \arg\min_{g \in \mathcal{G}_{\hat{f}}} \mathcal{L}_{int}(\hat{f}, g, \mathcal{S}).$$

With FLINT, we have $g(x) = h \circ \Phi(x)$ and $\Phi(x) = \Psi \circ \hat{f}_{\mathcal{I}}(x)$ for a given set of accessible hidden layers $\mathcal{I}$ and a attribute dictionary size J. Learning can be performed by specializing Alg. 1 with slight modification of replacing Θ as $\Theta = (\theta_\Psi, \theta_h, \theta_d)$ while $\theta_{\hat{f}}$ is fixed and eliminating $\mathcal{L}_{pred}$ from training loss $\mathcal{L}$.

Experimental results for post-hoc FLINT: We validate this ability of our framework by interpreting fixed models trained only for accuracy, i.e, BASE-f models from section 5.1. Even after not tuning the internal layers of f, the system is still able to generate high-fidelity and meaningful interpretations. Fidelity comparisons against VIBI, class-attribute pair visualizations and experimental details are available in supplementary (Sec. S.3).

7 Conclusion

FLINT is a novel framework for learning a predictor network and its interpreter network with dedicated losses. A potential attractive use consists in retaining only the interpreter model as the final interpretable prediction model of reduced complexity. Further works will investigate this direction and the enforcement of additional constraints on attribute functions to encourage invariance under various transformations. Eventually FLINT can be extended to other tasks or modalities other than images in particular by adapting the design of attributes and the pipeline to understand them.

References

[1] Tameem Adel, Zoubin Ghahramani, and Adrian Weller. Discovering interpretable representations for both deep generative and discriminative models. In *International Conference on Machine Learning*, pages 50–59. PMLR, 2018.

[2] Rishabh Agarwal, Nicholas Frosst, Xuezhou Zhang, Rich Caruana, and Geoffrey Hinton. Neural additive models: Interpretable machine learning with neural nets. *arXiv preprint arXiv:2004.13912*, 2020.

[3] Maruan Al-Shedivat, Avinava Dubey, and Eric Xing. Contextual explanation networks. *arXiv preprint arXiv:1705.10301*, 2017.

[4] David Alvarez-Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 7775–7784, 2018.

[5] David Alvarez-Melis and Tommi S Jaakkola. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*, 2018.

[6] Plamen Angelov and Eduardo Soares. Towards explainable deep neural networks (xdnn). *Neural Networks*, 130:185–194, 2020.

[7] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *J. Mach. Learn. Res.*, 11:1803–1831, August 2010. ISSN 1532-4435.

[8] Seojin Bang, Pengtao Xie, Wei Wu, and Eric Xing. Explaining a black-box using deep variational information bottleneck approach. *arXiv preprint arXiv:1902.06918*, 2019.

[9] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82 – 115, 2020.

[10] Valérie Beaudouin, Isabelle Bloch, David Bounie, Stéphan Clémençon, Florence d'Alché-Buc, James Eagan, Winston Maxwell, Pavlo Mozharovskyi, and Jayneel Parekh. Flexible and context-specific AI explainability: A multidisciplinary approach. *CoRR*, abs/2003.07703, 2020.

[11] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8), 2019.

[12] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8928–8939, 2019.

[13] Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pages 883–892. PMLR, 2018.

[14] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782, 2020.

[15] Jonathan Crabbe, Yao Zhang, William Zame, and Mihaela van der Schaar. Learning outside the black-box: The pursuit of interpretable models. In *Advances in Neural Information Processing Systems*, pages 17838–17849, 2020.

[16] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[17] Finale Doshi-Velez and Been Kim. Interpretable machine learning. In *Proceedings of the Thirty-fourth International Conference on Machine Learning*, 2017.

[18] D. Dubois and H. Prade. Possibilistic logic-an overview. *Computational logic*, 9, 2014.

[19] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9277–9286, 2019.

[20] Riccardo Guidotti, Anna Monreale, Fosca Giannotti, Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Factual and counterfactual explanations for black box decision making. *IEEE Intelligent Systems*, 2019.

[21] David Ha and Douglas Eck. A neural representation of sketch drawings. In *International Conference on Learning Representations*, 2018.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[23] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *European Conference on Computer Vision*, pages 3–19. Springer, 2016.

[24] Himalaya Jain, Joaquin Zepeda, Patrick Pérez, and Rémi Gribonval. Subic: A supervised, structured binary code for image search. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 833–842, 2017.

[25] Berk Ustun Jyaming Zeng and Cynthia Rudin. Interpretable classification models for recidivism prediction. *Journal of the Royal Statistical Society, Series A, Statistics in Society*, 180, part 3: 689–722, 2017.

[26] Dmitry Kazhdan, Botty Dimanov, Mateja Jamnik, Pietro Liò, and Adrian Weller. Now you see me (cme): Concept-based model extraction. *arXiv preprint arXiv:2010.13233*, 2020.

[27] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *arXiv preprint arXiv:1711.11279*, 2017.

[28] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. The (un) reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 267–280. Springer, 2019.

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

[30] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning*, pages 5338–5348. PMLR, 2020.

[31] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[32] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 131–138, 2019.

[33] Himabindu Lakkaraju, Nino Arsov, and Osbert Bastani. Robust and stable black box explanations. In *International Conference on Machine Learning*, pages 5628–5638. PMLR, 2020.

[34] Yann LeCun. Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*.

[35] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[36] Guang-He Lee, Wengong Jin, David Alvarez-Melis, and Tommi S Jaakkola. Functional transparency for structured data: a game-theoretic approach. *arXiv preprint arXiv:1902.09737*, 2019.

[37] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[38] Zachary C. Lipton. The mythos of model interpretability. *Commun. ACM*, 61(10):36–43, 2018.

[39] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

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

[41] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[42] Karl Mosler. Depth statistics. In Claudia Becker, Roland Fried, and Sonja Kuhnt, editors, *Robustness and Complex Data Structures: Festschrift in Honour of Ursula Gather*, pages 17–34. Springer. Berlin, 2013.

[43] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[44] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM, 2016.

[45] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[46] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *CoRR*, abs/2103.11251, 2021.

[47] Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller, editors. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, volume 11700 of *Lecture Notes in Computer Science*. Springer, 2019.

[48] Sascha Saralajew, Lars Holdijk, Maike Rees, Ebubekir Asan, and Thomas Villmann. Classification-by-components: Probabilistic modeling of reasoning over a set of components. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2788–2799, 2019.

[49] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.

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

[51] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smooth-grad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[52] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[53] Gregor Stiglic, Primoz Kocbek, Nino Fijacko, Marinka Zitnik, Katrien Verbert, and Leona Cilar. Interpretability of machine learning-based prediction models in healthcare. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 10(5), 2020.

[54] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[55] Chih-Kuan Yeh, Been Kim, Sercan O Arik, Chun-Liang Li, Pradeep Ravikumar, and Tomas Pfister. On concept-based explanations in deep neural networks. *arXiv preprint arXiv:1910.07969*, 2019.

[56] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[57] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8827–8836, 2018.

[58] Yijun Zuo and Robert Serfling. General notions of statistical depth function. *The Annals of Statistics*, 28(2):461–482, 2000.

This supplementary is organized as follows:

- Sec. S.1 contains a synthetic overview of various works in interpretability w.r.t FLINT.
- Sec. S.2 contains details and additional experiments regarding *interpretability by-design* models.
- Sec. S.3 contains details and additional experiments regarding *post-hoc* interpretations generated using FLINT.
- Sec. S.4 discusses the limitations of our proposed method.
- Sec. S.5 discusses the potential negative societal impact.
- Interested readers can also find code attached with the supplementary.

S.1 Overview of related works

To recap the properties of the methods exposed in Sec. 2 (main paper), we provide in Tab. 2 a synthetic view of the major properties of interpretable methods along three aspects. *Type* denotes if the method implements *post-hoc* interpretations for a trained model or interpretable models *by-design*). *Scope* reflects the ability of the approach to provide interpretation of decisions for individual samples (*Local*) or to understand the model as a whole (*Global*). *Means* denotes the units in which the interpretations are generated. Categories include raw input features, a simplified representation of input, logical rules, prototypes, high-level concepts.

System	Means	Type	Scope
LIME, SHAP	Simplified input	Post-hoc	Local+Global
Gradient based	Raw input	Post-hoc	Local
VIBI, L2X	Raw input	Post-hoc	Local
Anchors	Logical rules	Post-hoc	Local
ICNN	Raw input	By-design	Local
CEN, GAME	Simplified input	By-design	Local
PrototypeDNN	Prototypes	By-design	Local
CAV-based	Concepts (External)	Post-hoc	Local+Global
SENN	Concepts (Learnt)	By-design	Local
FLINT	Concepts (Learnt)	Both	Local+Global

Table 2: Various interpretability systems and their properties.

- LIME, SHAP: Local Interpretable Model-agnostic Explanations [44], SHapley Additive exPlanations [39]
- VIBI, L2X: Variational Information Bottleneck for Interpretation [8], Learning to Explain [13]
- ICNN: Interpretable CNN [57]
- CEN, GAME: Contextual Explanation Networks [3], Game-theoretic transparency[36]
- PrototypeDNN: [37]
- Anchors: [45]
- CAV-based: Testing with Concept Activation Vectors (TCAV) [27], Towards Automatic Concept-based Explanations (ACE) [19], ConceptSHAP [55]
- SENN: Self Explaining Neural Networks [4]

S.2 Interpretability by design: Additional information and experiments

We cover all the implementation details in Sec. S.2.1, including network architectures, choice and effect of hyperparameters, optimization procedures, resource consumption. Additional analysis and

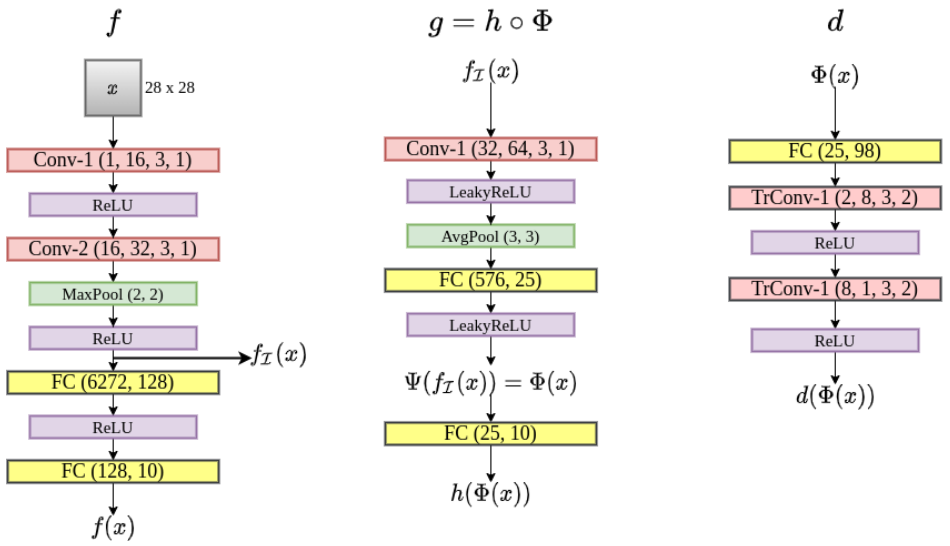


Figure 5: Architecture of networks based on LeNet [34]. Conv (a, b, c, d) and TrConv (a, b, c, d) denote a convolutional, transposed convolutional layer respectively with number of input maps a, number of output maps b, kernel size c $\times$ c and stride size d. FC(a, b) denotes a fully-connected layer with number of input neurons a and output neurons b. MaxPool(a, a) denotes window size a $\times$ a for the max operation. AvgPool(a, a) denotes the output shape a $\times$ a for each input map

visualizations for attributes are available in Sec. S.2.2. We also present other useful tools for analysis in Sec. S.2.3. Baseline implementations are discussed in Sec. S.2.4. Details about the subjective evaluation, including the form link are available in Sec. S.2.5. Note that the experiments with ACE are deferred to Sec. S.3.3.

S.2.1 Implementation details

S.2.1.1 Network architectures

Predictor Fig. 5 and 6 depict the architectures used for experiments with predictor architecture based on LeNet [34] (on MNIST, Fashion-MNIST) and ResNet18 (on CIFAR10, QuickDraw) [22] respectively.

Interpreter The architecture of interpreter $g = h \circ \Phi$ and decoder d for MNIST, FashionMNIST are shown in Fig. 5. Corresponding architectures for QuickDraw are in Fig. 6. For CIFAR-10, the interpreter architecture is almost exactly the same as QuickDraw, with only difference being output layer for $\Phi(x)$, which contains 36 attributes instead of 24. The decoder d also contains corresponding changes to input and output FC layers, with 36 dimensional input in first FC layer and 3072 dimensional output in last FC layer.

The choice of selection of intermediate layers is an interesting part of designing the interpreter. In case of LeNet, we select the output of final convolutional layer. For ResNet, while we tend to select the intermediate layers from the latter convolutional layers, we do not select the last convolutional block (CBlock 8) output. This is mainly because empirically, when selecting the output of CBlock 8, the attributes were trivially learnt, with only one attribute activating for any sample and attributes exclusively activating for a single class. The hyperparameters are much harder to tune to avoid this scenario. Thus we selected two outputs from CBlock 6, CBlock 7 as intermediate layers. The layers in the interpreter itself were chosen fairly straightforwardly with 1-2 conv layers followed by a pooling and fully-connected layer.

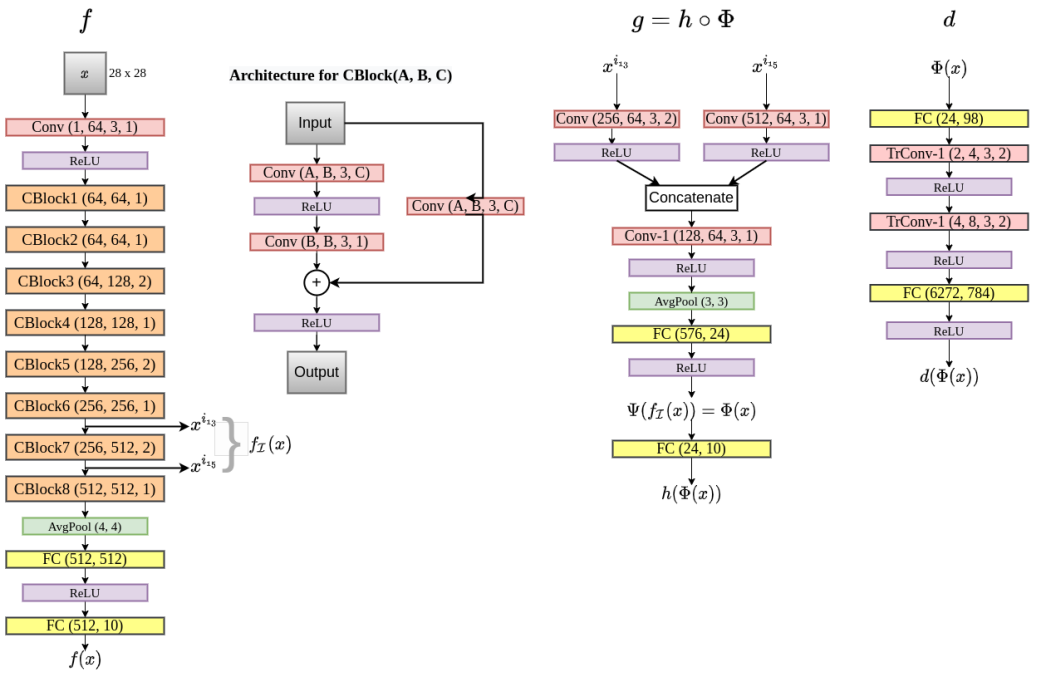


Figure 6: Architecture of networks for experiments on QuickDraw with network based on ResNet [22]. Conv (a, b, c, d) and TrConv (a, b, c, d) denote a convolutional, transposed convolutional layer respectively with number of input maps a, number of output maps b, kernel size c $\times$ c and stride size d. FC(a, b) denotes a fully-connected layer with number of input neurons a and output neurons b. AvgPool(a, a) denotes the output shape a $\times$ a for each input map. Notation for CBlock is explained in the figure.

S.2.1.2 QuickDraw subset and pre-processing

QuickDraw. We created a subset of QuickDraw from the original dataset [21], by selecting 10000 random images from each of 10 classes: 'Ant', 'Apple', 'Banana', 'Carrot', 'Cat', 'Cow', 'Dog', 'Frog', 'Grapes', 'Lion'. We randomly divide each class into 8000 training and 2000 test images.

Input pre-processing. For MNIST, FashionMNIST and QuickDraw, we use the default images with pixel values in range $[0, 1]$. No data augmentation is performed. For CIFAR-10 we apply the most common mean and standard deviation normalization. The training data is generated by randomly cropping a 32x32x3 image after padding the original images by zeros (size of padding is 2).

S.2.1.3 Effect of number of attributes J

Effect of J We study the effect of choosing small values for number of attributes J (keeping all other hyperparameters same). Tab. 3 tabulates the values of input fidelity loss $\mathcal{L}_{if}$, output fidelity loss $\mathcal{L}_{of}$ on the training data by the end of training for MNIST and the fidelity of g to f on MNIST test data for different J values. Tab. 4 tabulates same values for QuickDraw. The two tables clearly show that using small J can harm the autoencoder and the fidelity of interpreter. Moreover, the system packs more information in each attribute and this makes it hard to understand them, specially for very small J. This is illustrated in Figs. 7 and 8, which depict part of global interpretations generated on MNIST for $J = 4$ (all the parameters take default values). Fig. 7 shows global class-attribute relevances and Fig. 8 shows generated interpretation for a sample attribute ϕ_2. It can be clearly seen that the attributes start encoding concepts for too many classes (high number of bright spots). This also causes their AM+PI outputs to be muddled with two many patterns. This adds a lot of difficulty in understandability of these attributes.

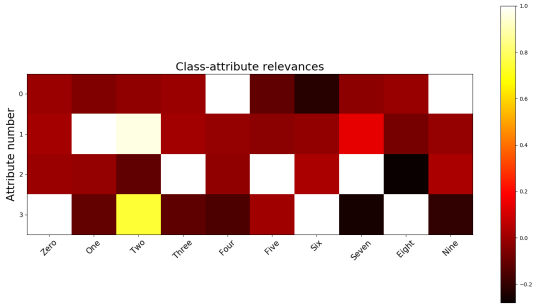


Figure 7: Global class attribute relevances for model with $J = 4$ on MNIST.

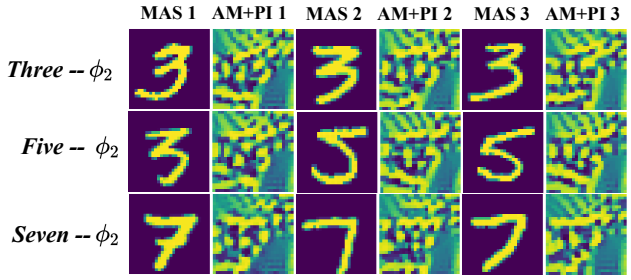


Figure 8: Interpretation for attribute ϕ_2 for model learn on MNIST with $J = 4$.

How to choose the number of attributes Assuming a suitable architecture for decoder d, simply tracking $\mathcal{L}_{if}, \mathcal{L}_{of}$ on training data can help rule out very small values of J as they result in poorly trained decoder and relatively poor fidelity of g. One can also qualitatively analyze the generated explanations from the training data to tune J to a certain extent. Too small values of J can result in attributes encoding concepts for too many classes, which affects negatively their understandability. It is more tricky and subjective to tune J once it becomes large enough so that $\mathcal{L}_{if}, \mathcal{L}_{of}$ are optimized well. The upper threshold of choosing J is subjective and highly affected by how many attributes the user can keep a tab on or what fidelity user considers reasonable enough. It is possible that due to enforcement of conciseness, even for high value of J, only a small subset of attributes are relevant for interpretations. Nevertheless, for high J value, there is a risk of ending up with too many attributes or class-attribute pairs to analyze.

It is important to notice that it is possible to select J from the training set only by using a cross-validation strategy. In practise, it seems reasonable to agree on smallest value of J for which the increase of the cross-validation fidelity estimate drops dramatically, since further increase of J would generate less understandable attributes with very little gain in fidelity.

S.2.1.4 Hyperparameter settings

Tab. 5 reports the setting of our hyperparameters for different datasets. We briefly discuss here our method to tune the different weights.

We varied γ between 0.8 to 20 for all datasets, and stopped at a value for which the $\mathcal{L}_{if}$ loss seemed to optimize well (value dropped by at least 50% compared to the start). For MNIST and FashionMNIST,

	$\mathcal{L}_{if}$ (train)	$\mathcal{L}_{of}$ (train)	Fidelity (test) (%)
$J = 4$	0.058	0.57	87.4
$J = 8$	0.053	0.23	97.5
$J = 25$	0.029	0.16	98.8

Table 3: Effect of J on losses and fidelity for MNIST with LeNet.

	$\mathcal{L}_{if}$ (train)	$\mathcal{L}_{of}$ (train)	Fidelity (test) (%)
$J = 4$	0.094	2.08	19.5
$J = 8$	0.079	1.48	57.6
$J = 24$	0.069	0.34	90.8

Table 4: Effect of J on losses and fidelity for QuickDraw with ResNet.

Variable	MNIST	FashionM	CIFAR10	QuickDraw
N_{epoch} – Number of training epochs	12	12	25	12
β – Weight for $\mathcal{L}_{of}$	0.5	0.5	0.6	0.1
γ – Weight for $\mathcal{L}_{if}$	0.8	0.8	2.0	5.0
δ – Weight for $\mathcal{L}_{cd}$	0.2	0.2	0.2	0.1
η – Relative strength of ℓ_1-regularization	0.5	0.5	1.0	3.0

Table 5: Hyperparameters for FLINT

the first value, 0.8 worked well. For the others, γ needed to be increased so that the autoencoder worked well. Too high γ might result in failed optimization due to exploding gradients.

The variation of β was based on two indicators: (i) The system achieves high fidelity, for eg. at least 90%, so too small β can't be chosen, (ii) For high β, the attributes become class-exclusive with only one attribute activating for a sample and result in high $\mathcal{L}_{if}$. Thus, β was varied to get high fidelity and avoiding second scenario. $\beta = 0.5$ worked well for MNIST, FashionMNIST. For QuickDraw, we needed to decrease β because of second scenario.

The system is fairly robust to choice of δ, η. Too high ℓ_1 regularization results in loss of fidelity (Tab. 6). These values were mostly heuristically chosen, and small changes to them do not cause much difference to training. We kept the effect of entropy low for ResNet because of its very deep architecture and high computational capacity of intermediate layers which can easily sway attributes to be class-exclusive.

S.2.1.5 AM+PI procedure

In our case this optimization problem for an attribute j is:

$$\arg\max_{x} \lambda_\phi \phi_j(x) - \lambda_{tv}\text{TV}(x) - \lambda_{bo}\text{Bo}(x)$$

where $\text{TV}(x)$ denotes total variation of x and $\text{Bo}(x)$ promotes boundedness of x in a range. We fix parameters for AM+PI for MNIST, FashionMNIST, QuickDraw as $\lambda_\phi = 2, \lambda_{tv} = 6, \lambda_{bo} = 10$ and $\lambda_\phi = 2, \lambda_{tv} = 20, \lambda_{bo} = 20$ for CIFAR10. For each sample x_0 to be analyzed, we analyze input for this optimization as $0.3x_0$ for MNIST, FashionMNIST, QuickDraw and as $0.4x_0$ for CIFAR10. For optimization, we use Adam with learning rate 0.05 for 300 iterations, halving learning rate every 50 iterations.

S.2.1.6 Optimization and Runs

The models are trained for 12 epochs on MNIST, FashionMNIST and QuickDrawm and for 25 epochs on CIFAR-10. We use Adam [29] as the optimizer with fixed learning rate 0.0001 and train on a single NVIDIA-Tesla P100 GPU. Implementations are done using PyTorch [43].

	$\eta = 1$	$\eta = 2$	$\eta = 3$	$\eta = 5$
no entropy	92.7	90.4	91.2	84.2
with entropy	91.2	90.7	90.8	82.9

Table 6: Fidelity (in %) variation for η and entropy losses for QuickDraw. $\delta = 0.1$ is fixed

Number of runs: For the accuracy and fidelity results in the main paper, we have reported mean and standard deviation for 4 runs with different seeds for each system. The conciseness results are computed by averaging conciseness of 3 models for each reported system.

S.2.1.7 Resource consumption

Compared to f, Ψ, h and d have fewer parameters. For networks shown in Fig. 5, the LeNet based predictor has around 800,000 trainable parameters, interpreter g contains 70,000 parameters, decoder d contains 3000 parameters. For networks in Fig. 6, ResNet based predictor contains 11 million parameters, interpreter g contains 530,000 parameters, and decoder d contains 4.9 million parameters (almost all of them in the last FC layer). In terms of space, FLINT occupies more storage space according to the decoder, but is still of comparable size to that of only storing predictor.

Training time In terms of training time consumption there is lesser difference when f is a very deep network, due to all networks Ψ, h, d being much shallower (lesser number of layers) than f. For eg. on both CIFAR-10, QuickDraw, FLINT consumes just around 10% more time for training compared to training just the predictor (BASE-f). The difference is more pronounced on with shallower f where Ψ, h, d also have comparable number of layers to f. Training BASE-f on MNIST consumes 50% less time compared to FLINT.

We compare the average training times (for four runs) for SENN and FLINT in Tab. 7. Each model is trained for the same number of epochs, on the same computing machine (1 NVIDIA Tesla P100 GPU). It is clear that SENN requires significantly more time to train. This is primarily because of gradient of output w.r.t input being part of their loss function. Thus the computational graph for a forward pass is twice as big as their model architecture and followed by a backward pass through the bigger graph.

Dataset	SENN	FLINT
MNIST	2311	**518**
FashionMNIST	2333	**519**
CIFAR-10	10210	**1548**
QuickDraw	10548	**1207**

Table 7: Training times for FLINT and SENN (in seconds)

S.2.2 Analysis and additional interpretations

S.2.2.1 Shuffling experiment

By structure, for both FLINT-g and SENN, the output are generated by combining high level attributes and weights. To test how crucial the learnt attributes are to their predictions, we shuffle the attribute values $\Phi(x)$ for each sample x (this corresponds to shuffling $h(x)$ for SENN with their notations). This is an extreme test: we therefore expect an important drop in accuracy. Tab. 8 reports the results for the experiments for our method and SENN. More precisely, we calculate the drop in prediction accuracy of FLINT-g (and SENN), compared to their mean accuracies. For SENN, the very small drop in accuracy indicates its robustness to this shuffling, which highlights the fact that in this model, the activation of a given subset of attributes is not crucial for the prediction. In contrast FLINT-g relies strongly on its attributes for its prediction.

Dataset	SENN	FLINT-g
MNIST	0.5	87.6
FashionMNIST	10.9	76.6
CIFAR-10	17.5	74.4
QuickDraw	0.3	74.9

Table 8: FLINT and SENN accuracy drop for shuffled attributes (in %)

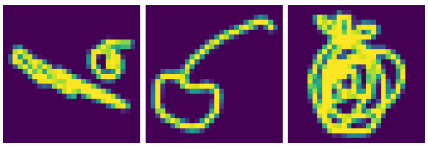


Figure 9: The three 'Apple' class samples classified correctly by f but not by g.

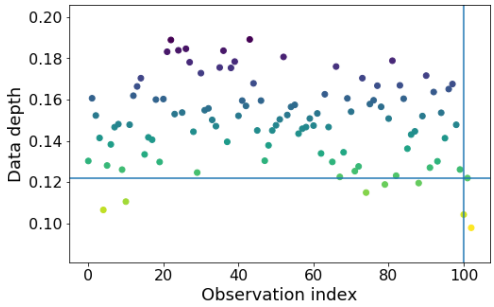


Figure 10: Projection data depth calculated with (3) w.r.t. the 8000 'Apple' training sample for 100 'Apple' test samples and for the three (observation indices 101–103) 'Apple' class samples classified correctly by f but not by g.

S.2.2.2 Disagreeement analysis

In this part, we analyse in detail the "disagreement" between the predictor f and the interpreter g. Note that we already achieve very high fidelity to predictor for all datasets. We limit our analysis to QuickDraw, our dataset with least fidelity. Understanding disagreement can help us improving our framework as well as providing a measure of reliability about predictors output.

For a given sample with disagreement, if the class predicted by f is among the top predicted classes of g, the disagreement is acceptable to some extent as the attributes can still potentially interpret the prediction of f. The worse kind of samples for disagreement are the ones where predicted by f class is not among the top g predicted classes, and even worse are where, in addition to this, f predicts the true label. We thus compute the top-k accuracy (for $k = 2, 3, 4$) on QuickDraw with ResNet, which for the default parameters described in the main paper, achieves a top-2 accuracy of 94.7%, top-3 accuracy 96.9%, and top-4 accuracy 98.2%. Only on 141 (i.e. 0.7%) samples the class predicted by f, same as true class, is not in top-3 predicted by g classes.

For eg., for the 'Apple' class (in QuickDraw), there only three disagreement samples for which f delivers correct prediction (plotted in Fig. 9) are not resembling apples at all. We propose an original analysis approach that consists in calculating a *robust centrality measure*—the projection depth—of these three samples as well as of another 100 training samples w.r.t. the 8000 training 'Apple' samples, plotted in Fig. 10. To that purpose, we use the notion of projection depth [58, 42] for a sample $\boldsymbol{x} \in \mathbb{R}^d$ w.r.t. a dataset $\boldsymbol{X}$ which is defined as follows:

$$D(\boldsymbol{x}|\boldsymbol{X}) = \left(1 + \sup_{\boldsymbol{p}\in\mathcal{S}^{d-1}} \frac{|\langle \boldsymbol{p}, \boldsymbol{x}\rangle - \text{med}(\langle \boldsymbol{p}, \boldsymbol{X}\rangle)|}{\text{MAD}(\langle \boldsymbol{p}, \boldsymbol{X}\rangle)}\right)^{-1}, \tag{3}$$

with $\langle \cdot, \cdot \rangle$ denoting scalar product (and thus $\langle \boldsymbol{p}, \boldsymbol{X}\rangle$ being a vector of projection of $\boldsymbol{X}$ on $\boldsymbol{p}$) and med and MAD being the univariate median and the median absolute deviation form the median. Fig. 10 confirms the visual impression that these 3 disagreement samples are outliers (since their depth in the training class is low).

Fig. 11 depicts 26 such cases for 'Cat' class to illustrate their logical dissimilarity. Being a complex model, the ResNet-based predictor f still manages to learn to distinguish these cases (while g does not), but in a way g does not manage at all to explain. Eventually, exploiting disagreement of f and g could be used as a means to measure trustworthiness. Deepening this issue is left for future works.

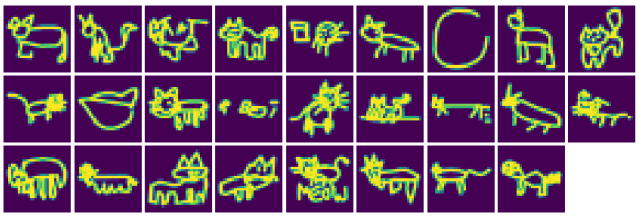


Figure 11: 26 samples from 'Cat' class which are not in top3 f-predicted classes.

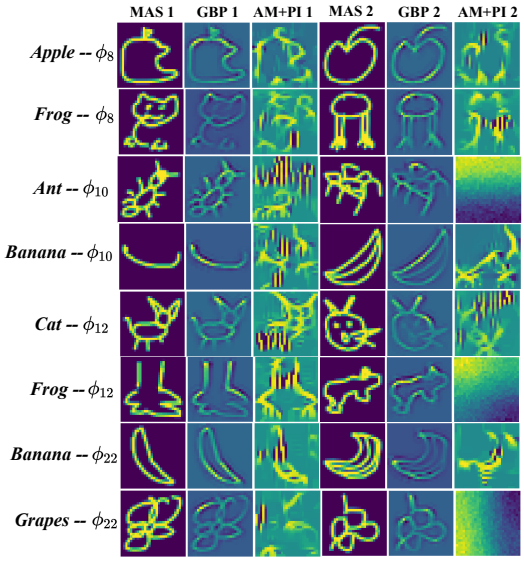


Figure 12: Sample class-attribute pair visualizations learnt without autoencoder loss $\mathcal{L}_{if}$. GBP stands for Guided Backpropagation.

S.2.2.3 Effect of autoencoder loss

Although the effect of $\mathcal{L}_{of}, \mathcal{L}_{cd}$ can be objectively assessed to some extent, the effect of $\mathcal{L}_{if}$ can only be seen subjectively. If the model is trained with $\gamma = 0$, the attributes still demonstrate high overlap, nice conciseness. However, it becomes much harder to understand concepts encoded by them. For majority of attributes, MAS and the outputs of the analysis tools do not show any consistency of detected pattern. Some such attributes are depicted in Fig. 12 Such attributes are present even for the model trained with autoencoder, but are very few. We thus believe that autoencoder loss enforces a consistency in detected patterns for attributes. It does not necessarily guarantee semantic meaningfulness in attributes, however it's still beneficial for improving their understandability.

S.2.2.4 Additional visualizations

For completeness, we show some additional visualizations of global interpretations (relevances, class-attribute pairs) and local interpretations.

Fig. 13 contains global relevances generated for MNIST and FashionMNIST. Global relevances for QuickDraw and CIFAR10 are in main paper.

Figs. 14, 15, 16, 17 show some additional class-attribute pairs and their visualizations for all 4 datasets. Local interpretations on some test samples from these datasets are depicted in Figs. 18, 19, 20, 21.

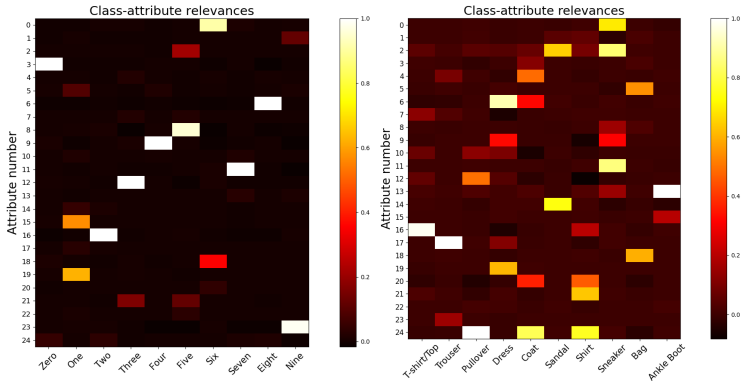


Figure 13: Global class-attribute relevances $r_{j,c}$ for MNIST (Left) and FashionMNIST (Right). 14 class-attribute pairs for MNIST and 26 pairs for FashionMNIST have relevance $r_{j,c} > 0.2$.

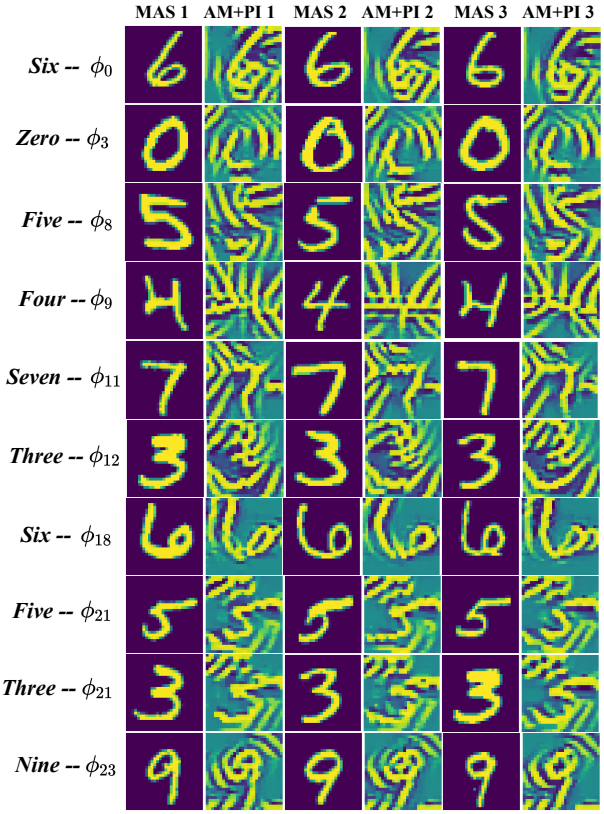


Figure 14: Additional class-attribute visualizations for MNIST. Three MAS and their corresponding AM+PI outputs are shown.

S.2.3 Other tools for analysis

Although we consider AM+PI as the primary tool for analyzing concepts encoded by attributes (for MAS of each class-attribute), other tools can also be helpful in deeper understanding of the attributes. We introduce two such tools:

- *Input attribution*: This is a natural choice to understand an attribute's action for a sample. Any algorithms ranging from black-box local explainers to saliency maps can be employed. These maps are less noisy (compared to AM+PI) and very general choice, applicable to almost all domains.

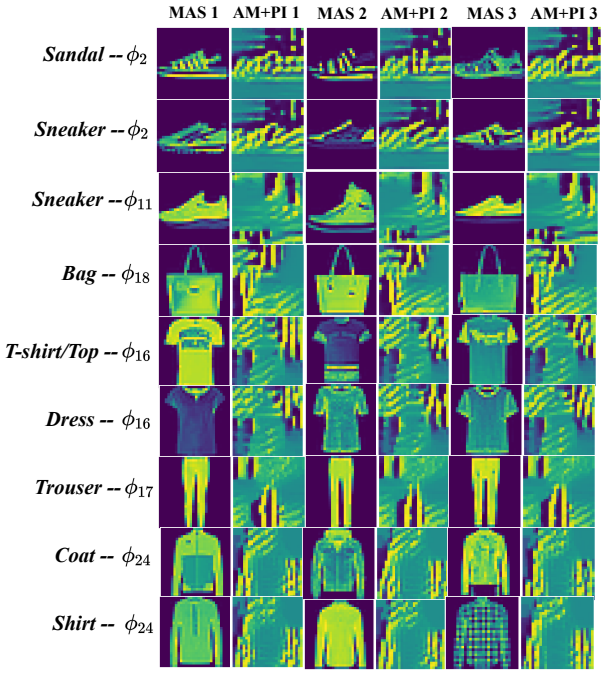


Figure 15: Additional class-attribute visualizations for Fashion-MNIST. Three MAS and their corresponding AM+PI outputs are shown.

- *Decoder*: Since we also train a decoder d that uses the attributes as input. Thus, for an attribute j and x, we can compare the reconstructed samples $d(\Phi(x))$ and $d(\Phi(x)\backslash j)$ where $\Phi(x)\backslash j$ denotes attribute vector with $\phi_j(x) = 0$, i.e., removing the effect of attribute j. While, the above comparison can be helpful in revealing information encoded in attribute j, it is not guaranteed to do so as the attributes can be entangled.

We illustrate the use of these tools for certain example class-attribute pairs on QuickDraw in Fig. 22 and 23. Note that as discussed in the main paper, these tools are not guaranteed to be always insightful, but their use can help in some cases.

Fig. 22 depicts example class-attribute pairs where decoder d contributes in understanding of attributes. The with ϕ_j column denotes the reconstructed sample $d(\Phi(x))$ for the maximum activating sample x under consideration. The without ϕ_j column is the reconstructed sample $d(\Phi(x))\backslash j)$ with the effect of attribute ϕ_j removed for the sample under consideration ($\phi_j(x) = 0$). For eg. ϕ_1, ϕ_{23}, strongly relevant for Cat class, detect similar patterns, primarily related to the face and ears of a cat. The decoder images suggest that ϕ_1 very likely is more responsible for detecting the left ear of cat and ϕ_{23}, the right ear. Similarly analyzing decoder images for ϕ_{22} in the third row reveals that it is likely has a preference for detecting heads present towards the right side of the image. This is certainly not the primary concept ϕ_{22} detects as it mainly detects blotted textures, but it certainly carries information about head location to the decoder.

Fig. 23 depicts example class-attribute pairs where input attribution contributes in understanding of attributes. We use Guided Backpropagation [52] (GBP) as input attribution method for ResNet on QuickDraw. It mainly assists in adding more support to our previously developed understanding of attributes. For example, analyzing ϕ_5 (relevant for Dog, Lion) based on AM+PI outputs suggested that it mainly detects curves similar to dog ears. The GBP output support this understanding as the most salient regions of the map correspond to curves similar to dog ears.

S.2.4 Baseline implementations

We cover the implementation details of various baselines used in this work (Tab 2, 3, 4 from main paper). The accuracy of FLINT-f is compared against BASE-f, PrototypeDNN, SENN. Fidelity of FLINT-g is compared against VIBI and LIME.

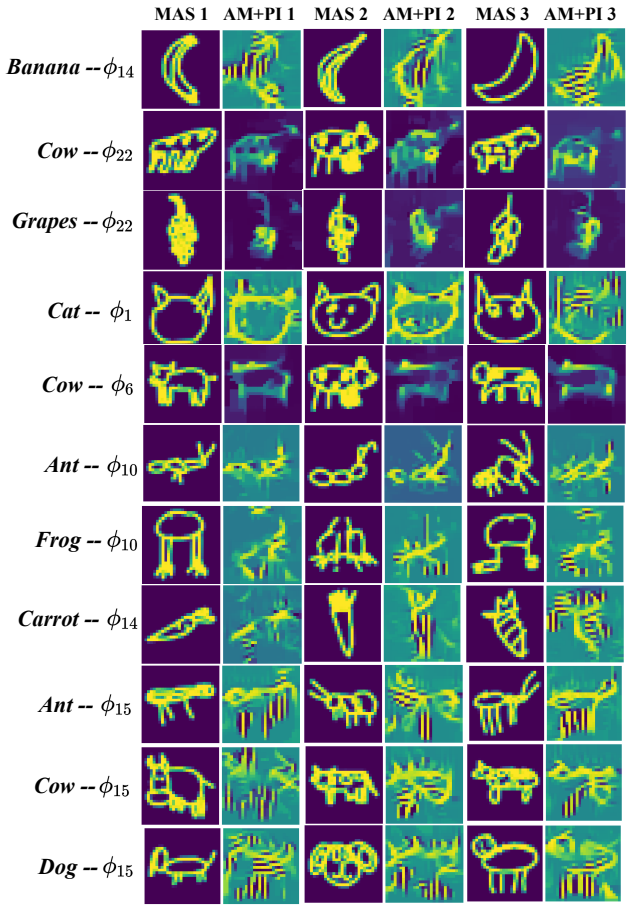


Figure 16: Additional class-attribute visualizations for QuickDraw. Three MAS and their corresponding AM+PI outputs are shown.

BASE-f We compare accuracy of FLINT-f with BASE-f. The BASE-f model has the same architecture as FLINT-f but is trained with $\beta, \gamma, \delta = 0$, that is, only with the loss $\mathcal{L}_{pred}$ and not interpretability loss term. All the experimental settings while training this model are same as FLINT.

PrototypeDNN We directly report the accuracy of PrototypeDNN on MNIST, FashionMNIST (Tab 2 main paper) from the results mentioned in their paper [37]. Note that we do not report any results of PrototypeDNN on CIFAR10 and QuickDraw. This is because for processing more complex images and achieving higher accuracy, one would need to non-trivially modify architecture of their proposed model. Thus to avoid any unfair comparison, we did not report this result. The results of BASE-f and SENN on CIFAR, QuickDraw help validate performance of FLINT-f on QuickDraw.

SENN We compare the accuracy as well as conciseness curve for FLINT with Self-Explaining Neural Networks (SENN) [4]. We implemented it with the help of their official implementation available on GitHub [1]. SENN employs a LeNet styled network for MNIST in their paper. We use the same architecture for MNIST and FashionMNIST. For QuickDraw and CIFAR10 we use the VGG based architecture proposed for SENN in their paper to process more complex images. However, to maintain fairness, the number of attributes used in all the experiments for SENN are same as those for FLINT, that is, 25 for MNIST & FashionMNIST, 24 for QuickDraw and 36 for CIFAR10, and also train for the same number of epochs. We use the default choices in their implementation for all hyperparameters and other settings. Another notable point is that although interpretations of SENN are worse than FLINT in conciseness (even when compared non-entropy version of FLINT), the

[1] `https://github.com/dmelis/SENN`

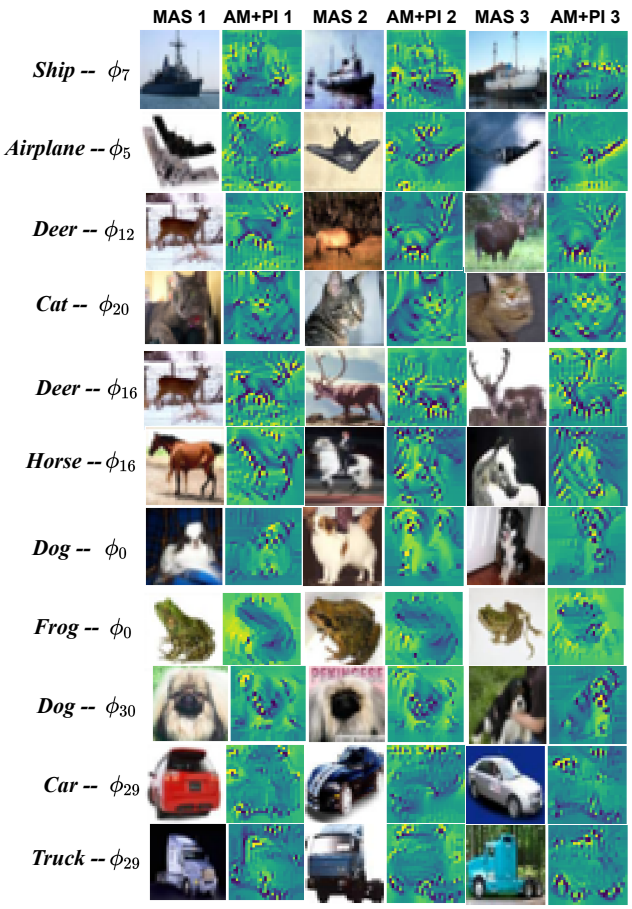


Figure 17: Additional class-attribute visualizations for CIFAR-10. Three MAS and their corresponding AM+PI outputs are shown.

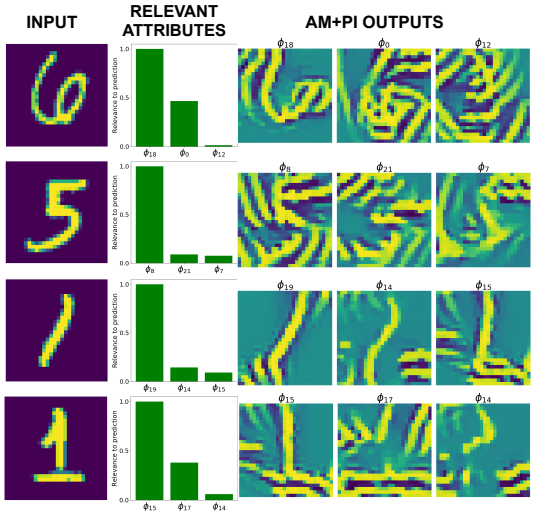


Figure 18: Local interpretations on test samples for MNIST. True labels are: 'Six', 'Five', 'One' and 'One'. Top 3 most relevant attributes and their corresponding AM+PI outputs are shown.

strength of ℓ_1 regularization in SENN is 2.56 times our strength (for identical $\mathcal{L}_{pred}$, i.e, cross-entropy loss with weight 1.0).

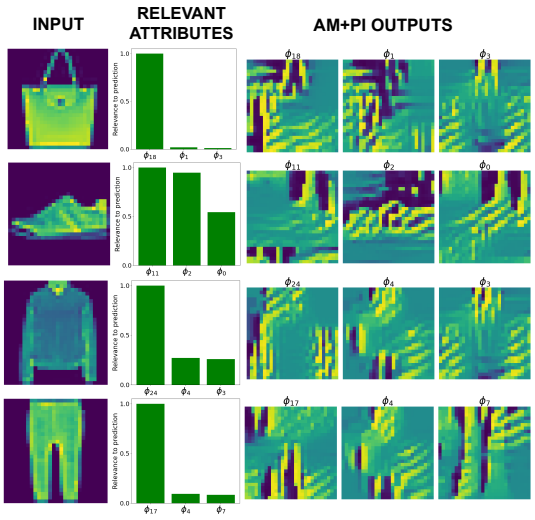


Figure 19: Local interpretations on test samples for Fashion-MNIST. True labels are: 'Bag', 'Sneaker, 'Coat', 'Trousers'. Top 3 most relevant attributes and their corresponding AM+PI outputs are shown.

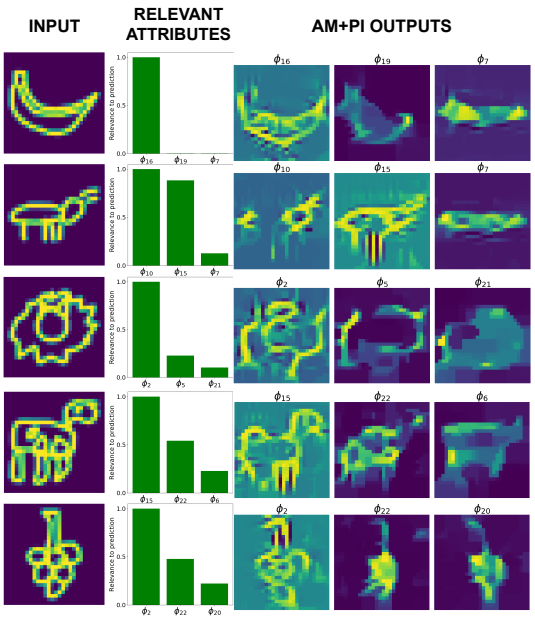


Figure 20: Local interpretations on test samples for QuickDraw. True labels are: 'Banana', 'Ant', 'Lion', 'Cow' and 'Grapes'. Top 3 most relevant attributes and their corresponding AM+PI outputs are shown.

VIBI & LIME We benchmark the fidelity of interpretations of FLINT-g for both by-design and post-hoc interpretation applications against a state-of-the-art black box explainer variational information bottleneck for interpretation (VIBI) [8] and traditional explainer LIME [44]. Note that VIBI also possesses a model approximating the predictor for all samples. Both methods are implemented using the official repository for VIBI [2]. We compute the "*Approximator Fidelity*" metric as described in their paper, for both systems. In the case of VIBI, this metric exactly coincides with our definition of fidelity. We set the hyperparameters to the setting that yielded best fidelity for datasets reported in their paper. For VIBI, chunk size 4×4, number of chunks $k = 20$, for LIME, chunk size 2×2, number of chunks $k = 40$. The other hyperparameters were the default parameters in their code.

[2] `https://github.com/SeojinBang/VIBI`

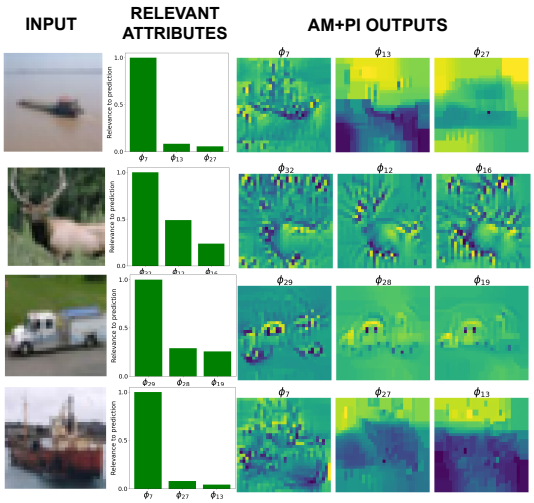


Figure 21: Local interpretations on test samples for CIFAR-10. True labels are: 'Ship', 'Deer', 'Truck' and 'Ship'. Top 3 most relevant attributes and their corresponding AM+PI outputs are shown.

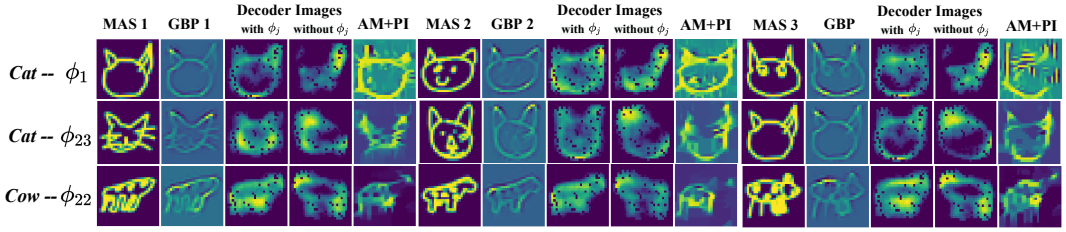


Figure 22: Examples of class-attribute pairs on QuickDraw, where decoder assists in understanding of encoded concept for the attribute.

S.2.5 Subjective evaluation details

The form taken by the participants can be accessed here [3]. 17 of the 20 respondents were in the age range 24-31 and at least 16 had completed a minimum of masters level of education in fields strongly related to computer science, electrical engineering or statistics. The form consists of a description where the participants are briefly explained through an example the various information (class-attribute pair visualizations and textual description) they are shown and the response they are supposed to report for each attribute, which is the level of agreement/disagreement with the statement: "The patterns depicted in AM + PI outputs can be meaningfully associated to the textual description". As mentioned in the main paper, four descriptions (questions #2, #5, #8, #9 in the form) were manually corrupted to better ensure that participants are informed about their responses. The corruption mainly consisted of referring to other parts or concepts regarding the relevant class which are *not* emphasized in the AM+PI outputs.

S.3 Post-hoc interpretations

S.3.1 Implementation details

The network architecture, the optimization procedures and hyperparameters are set to exactly the same values they were for their 'by-design', with one small change, β for CIFAR10 is used as 0.3, and not 0.6, this is because for $\beta = 0.6$, the system was running into scenario discussed in Sec. S.2.1.4, thus β was lowered.

[3] `https://forms.gle/PW6DEPZSmXb46Lnv9`

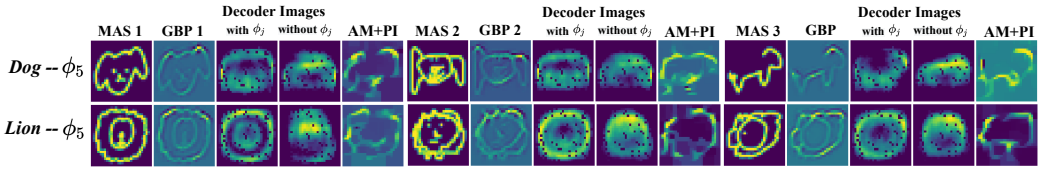


Figure 23: Examples of class-attribute pairs on QuickDraw, where input attribution (GBP) assists in understanding of encoded concept for the attribute. GBP stands for Guided Backpropagation.

Dataset	VIBI	FLINT-g
MNIST	95.8±0.2	**98.6±0.2**
FashionMNIST	88.4±0.2	**92.8±0.3**
CIFAR10	64.2±0.3	**89.1±0.5**
QuickDraw	78.0±0.4	**90.5±0.3**

Table 9: Fidelity for post-hoc interpretations of BASE-f (in %)

Results. Fidelity benchmarked against VIBI is tabulated in Tab. 9 and conciseness curves for post-hoc interpretations are shown in Fig. 24. They clearly indicate that FLINT can yield high fidelity and highly concise *post-hoc* interpretations.

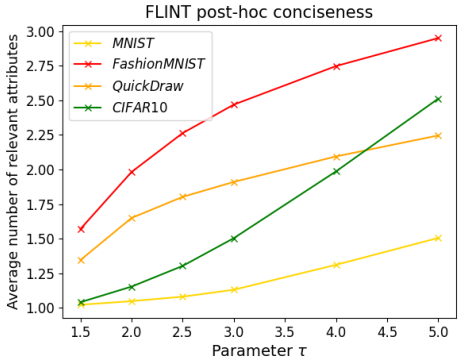


Figure 24: Conciseness curve of post-hoc interpretations generated using FLINT

S.3.2 Additional visualizations

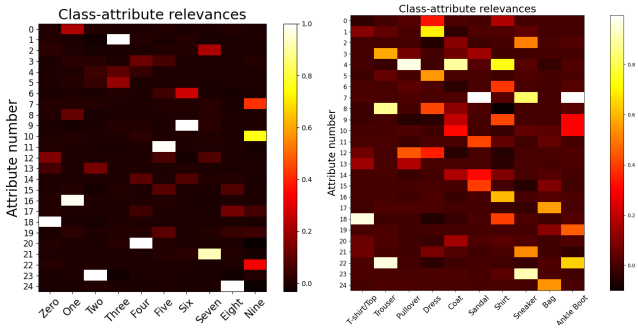


Figure 25: Global class-attribute relevances $r_{j,c}$ for post-hoc interpretations on MNIST (Left) and FashionMNIST (Right). 15 class-attribute pairs for MNIST and 28 pairs for FashionMNIST have relevance $r_{j,c} > 0.2$.

Figs. 25 and 26 contain global relevances for post-hoc interpretations on all four datasets. Figs. 27, 28, 29 and 30, illustrate some additional visualizations of class-attribute pairs on all datasets.

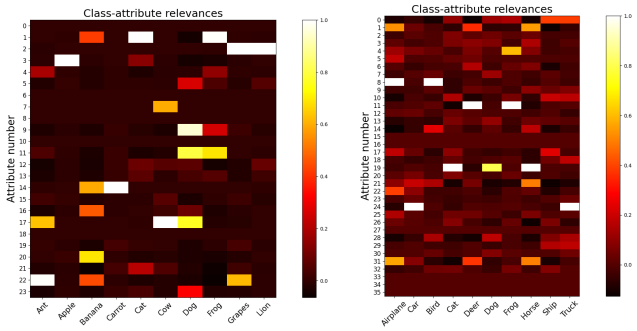


Figure 26: Global class-attribute relevances $r_{j,c}$ for post-hoc interpretations on QuickDraw (Left) and CIFAR10 (Right). 24 class-attribute pairs for QuickDraw and 26 pairs for CIFAR10 have relevance $r_{j,c} > 0.2$.

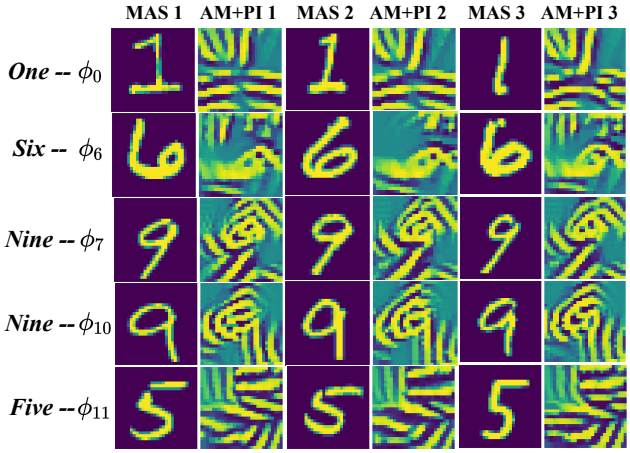


Figure 27: Sample class-attribute visualizations for post-hoc interpretations for MNIST.

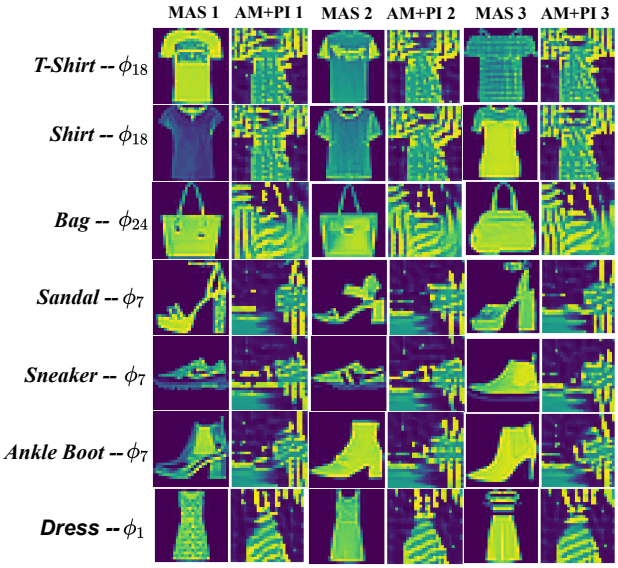


Figure 28: Sample class-attribute visualizations for post-hoc interpretations for Fashion-MNIST

S.3.3 Experiments using ACE

We conducted additional experiments using ACE to interpret trained models from our experiments. The key bottleneck for ACE's application on our datasets and networks is the use of CNN as a

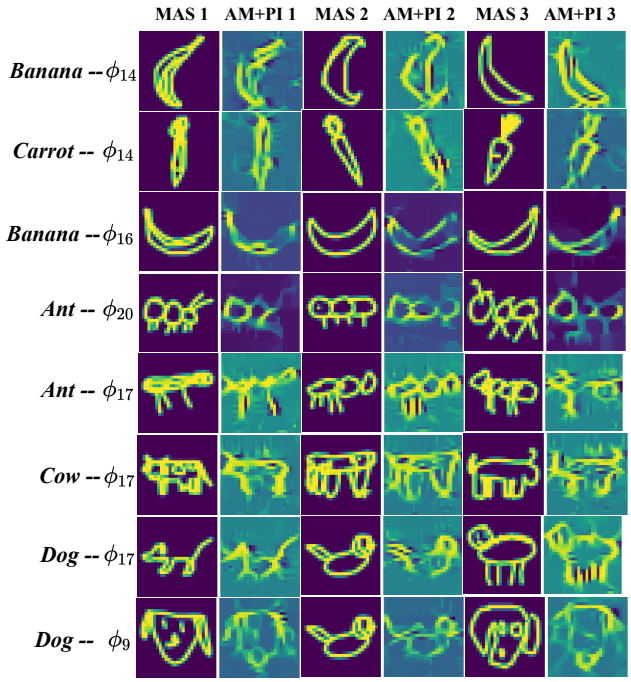


Figure 29: Sample class-attribute visualizations for post-hoc interpretations on QuickDraw

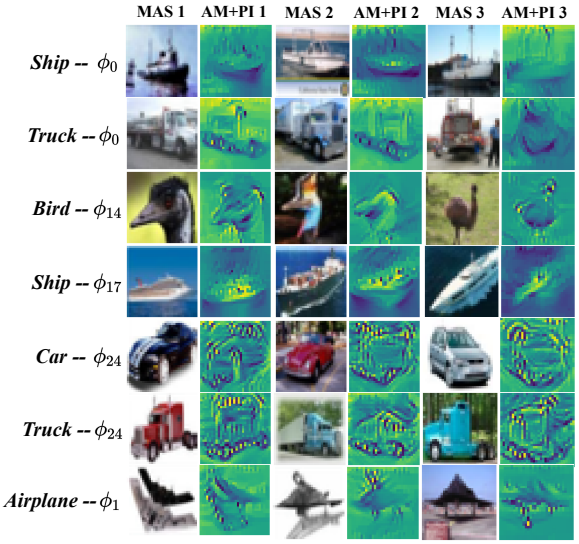


Figure 30: Sample class-attribute visualizations for post-hoc interpretations on CIFAR-10

similarity metric (to automate human annotation) for image segments irrespective of their scale, aspect ratio. This is a specialized property only been empirically shown for specific CNN's trained on ImageNet (as discussed in their paper). The networks trained on our datasets thus very often cluster unrelated segments, resulting in little to no consistency in any extracted concept. To illustrate the above we describe the experimental settings and show extracted concepts for a few classes from QuickDraw and CIFAR-10 on the BASE-f models. The quality of results is the same when interpreting FLINT-f models although we only illustrate interpretations from BASE-f models.

Experimental setting. We utilize the official open-sourced implementation of their method [4]. Due to the smaller sized images we perform segmentation at a single scale. We experimented with different configurations for “number of segments" and “number of clusters/concepts". The number of segments were varied from 3 to 15. For higher values the segments were often too small for concepts to be meaningful. We thus kept the number of segments 5 for each sample. For each class we chose 100 samples. The number of clusters were varied from 5 to 25. Due to the smaller number of segments (compared to original experiments from ACE which used 25), we kept number of clusters at 12. We access the deepest intermediate layer used in experiments with FLINT (shown in Fig. 6).

Results. The top 3 discovered concepts (according to the TCAV scores) are shown in Fig. 31. The segments for any concept on CIFAR show almost no consistency. This is mainly because the second step pf ACE, requiring a CNN's intermediate representations to replace a human subject for measuring the similarity of superpixels/segments, is hard to expect for these networks not trained on ImageNet. Thus, segments capturing background or any random part of the object, completely unrelated, end up clustered together. For QuickDraw, the segmentation algorithm also suffers problems in extracting meaningful segments due to sparse grayscale images. It generally extracts empty spaces or a big chunk of the object itself. This, compounded with the earlier issue about segment similarity results in mostly meaningless concepts. The only slight exception to this is concept 3 for 'Ant' for which two segments capture a single flat blob with small tentacles.

S.4 Limitations

- The current design of attributes and their encoded concept visualization procedure is more suited for classification tasks and image as input modality. Although multiple proposed losses/visualization tools could be generalized to other input modalities (e.g. audio, video, graphs etc.) or other machine learning tasks (regression), it requires work in that direction.
- The set of proposed properties is not exhaustive and can be further improved. It could be desired that attributes encode concepts which are invariant to certain transformations, or focus on specific spatial regions, or are robust to adversarial attacks / specific types of noise or contamination.
- The choice of hidden layers requires some level of experience with neural architectures.

S.5 Potential negative societal impact

Interpretability becoming a frequently raised issue when training and exploiting neural network (NN) architectures, the main expected societal impact of FLINT is improvement of their understandability as well as providing explanations of the decisions made by NNs. Nevertheless, even this intrinsically benevolent machinery can be used for harm when in malicious hands.

Potential misuse can be expected on two different levels: First, if incorrectly trained (e.g., wrong NN design, insufficient number of training examples and/or or training epochs, in particular for FLINT-f), due to lack of knowledge or on purpose, FLINT can provide misleading interpretations. Second, even a well-trained explainable AI can serve evil purpose in hands of a maliciously destined user.

Clearly, the authors expect proper use of the developed FLINT methodology, although direct misuse-protection mechanisms were not developed in this piece of research, not being the initial goal.

[4] `https://github.com/amiratag/ACE`

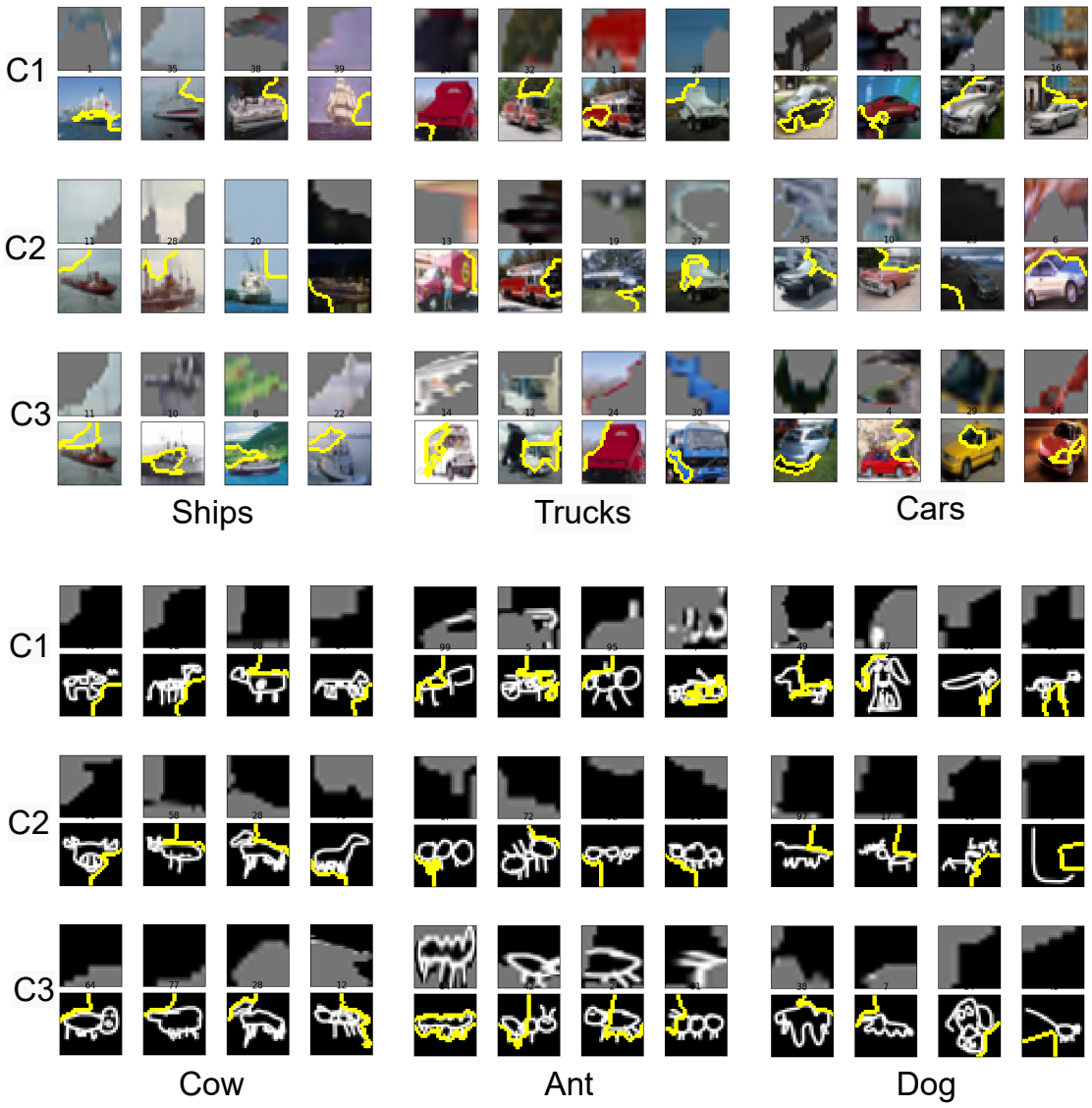


Figure 31: Discovered concepts using ACE for 3 classes on CIFAR-10 (Top) and QuickDraw (Bottom). We show the top 3 concepts according to their TCAV scores. Each concept consists of 4 segments extracted from images of the class. They are shown in 2 rows, the first contains the segments and the second shows where the segment was extracted from.