NeuCEPT: Learn Neural Networks' Mechanism via Critical Neurons with Precision Guarantee

Minh N. Vu
University of Florida
Gainesville, Florida, USA
minhvu@ufl.edu

Truc D. Nguyen
University of Florida
Gainesville, Florida, USA
truc.nguyen@ufl.edu

My T. Thai
University of Florida
Gainesville, Florida, USA
mythai@cise.ufl.edu

***Abstract*—Despite recent studies on understanding deep neural networks (DNNs), there exists numerous questions on how DNNs generate their predictions. Especially, given similar predictions on different inputs, are the underlying mechanisms generating those predictions the same? In this work, we propose NeuCEPT, a method to identify critical neurons that are important to the model's local predictions and learn their underlying mechanisms. We first formulate a critical neurons identification problem as maximizing a sequence of mutual-information objectives and provide a theoretical framework to efficiently solve for critical neurons while keeping the precision under control. NeuCEPT next heuristically learns different model's mechanisms in an unsupervised manner. Our experiments and case studies show that neurons identified by NeuCEPT not only have strong influence on the model's predictions but also hold meaningful information about model's mechanisms.**

***Index Terms*—Explainable machine learning, Markov chain, Mutual-information objective.**

I. Introduction

Significant efforts have been dedicated to improve the interpretability of modern neural networks, leading to several advancements [1], [2]; however, few works have been conducted to characterize local predictions of the neural networks based on the internal forwarding computation of the model (see Sect. II). In this paper, we focus on investigating different mechanisms learnt by the neural networks to generate predictions (see Fig. 1 as an example). Intuitively, the mechanism of a prediction is the forwarding process producing the prediction in the examined model (see definition in Sect. IV). Our hypothesis is that predictions of the same class label can be generated by different mechanisms which can be captured and characterized by activation of some specific neurons, called *critical neurons*. Analyzing the activation of those neurons can help identify the model's mechanisms and shed light on how the model works.

Following are key reasons motivating our study. First, the identification of critical neurons can serve as an initial model's examination for further study of the model's dynamics. Second, critical neurons allow us to characterize model's predictions based on how they are generated by the model. Each set of similar predictions can be studied and analyzed for downstream tasks such as performance [3] and trust evaluation [4]. Finally, identifying critical neurons provides a new dimension on how we explain the predictions compared to local attribution explanation methods. We will demonstrate the case studies of these motivations in Sect. VI. For now, Fig. 2 provides a concrete example motivating this study. Here, we have 2 LeNet classifying *even* or *odd* digit on the MNIST dataset. While the outputs and local explanations hardly show any differences between the two models, the evidences based on some specific neurons suggest otherwise. This example shows how an explanation at neuron-level can be beneficial.

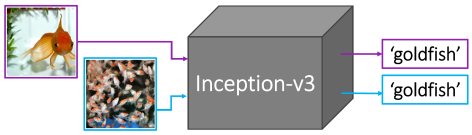


Fig. 1: Two images have the same predictions *goldfish* generated by Inception-v3. One is a *single goldfish* while the other is a *shoal of fish*. Are the mechanisms behind the two predictions the same?

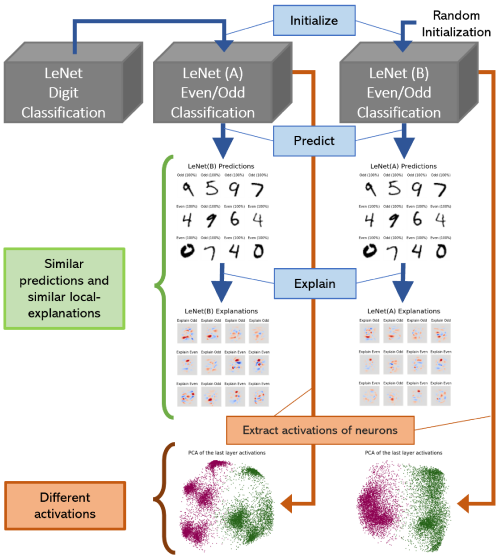


Fig. 2: LeNet(A) is initialized by a LeNet pretrained on digit classification task and LeNet(B) is initialized randomly. While the *even/odd* predictions and the explanations provide little information differentiating the two models, extracting and visualizing the activation of some neurons at the their last layers reveal that LeNet(A) groups inputs into more distinctive clusters with the same ground-truth digit-labels of the dataset. More details of the experiments are in Sect. V-A

We propose NeuCEPT - a method to learn neural network's mechanism via critical neurons with precision guarantee

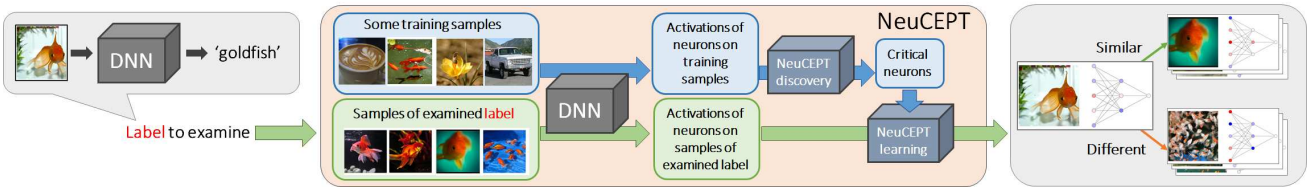


Fig. 3: Overall architecture of NeuCEPT. Given a prediction of a class of interest, NeuCEPT collects the model's inputs from that class. Then, by forwarding them through the DNN, NeuCEPT obtains the activation and solves for the set of critical neurons at some layers. Finally, mechanisms are learnt via unsupervised learning on those neurons' activation.

(Fig. 3). The innovation of NeuCEPT lies in its two main components: NeuCEPT-discovery and NeuCEPT-learning. Followings are the contributions and organization of this manuscript:

- **Critical Neurons Identification with NeuCEPT-discovery (Sect. III).** We introduce a layer-by-layer mutual information objective to discover the *critical neurons* of model's predictions. Intuitively, the critical neurons of a layer is the set of neurons determining the activation of the critical neurons in the sub-sequence layer, which eventually determines the predictions. To solve for these sets of critical neurons efficiently, we develop a pairwise-parallel approximation algorithm with theoretical precision guarantee.
- **Interpretation and NeuCEPT-learning (Sect. IV).** We provide information-theoretic interpretation of the critical neurons and elaborate how learning the mechanism on top of critical neurons can result in better claims on DNNs' mechanisms in term of explainability power and non-redundancy. We propose an unsupervised learning algorithm, called NeuCEPT-learning, to carry out that task.
- **Prior-knowledge Training and Experimental Results (Sect. V).** We propose a new testing approach, the prior-knowledge training, to experimentally evaluate the claims on DNNs' predicting mechanism. Our rigorous experiments on MNIST and CIFAR-10 show that NeuCEPT consistently detects the embedded mechanisms in the models. Ablation study is also conducted extensively.
- **Case studies (Sect. VI).** We demonstrate the advantages of NeuCEPT and the identification of critical neurons in some practical tasks, including: the study of model's linear separability [5], the discovery of some interesting neurons and behaviors of predictions of Inception-v3 [6] and the examination of unreliable predictions in CheXNet [7].

II. Related Work

Although the problem of identifying DNN's local mechanism is not well-studied, the related researches of interpretability and neural debugging received a lot of attention in recent years. Regarding explanation methods, there are two major approaches: local and global [1]. On the other hand, our study also relies on a new statistical tool for variable selection, called Model-X Knockoffs. We now briefly discuss those researches.

Local explanation methods either search for an interpretable surrogate model to capture the local behavior of the DNN [8], [9], [10] or carefully back-propagate to attribute contribution scores [11], [12], [13]. The focus of these methods is on identifying features and, technically can be extended to, neurons; however, it is unclear how they can be leveraged to identify model's mechanism. Specifically, highly attributed neurons do not necessarily imply the capability to identify mechanisms (see examples in Fig. 13).

Global explanation methods focus on explaining the model's behavior globally. Some notable techniques are based on decision trees [14], decision sets [15] and recursive-partitioning [16]. Unfortunately, there has been no well-established connection between the explained global dynamics and the model's local predictions; hence, the question regarding local mechanism is still unanswered. Among global methods, we find [17] to be the most related research to our work. The method relies on distillation technique to train another model with sparser data forwarding paths. Even though the paths can be used to partially reveal the model's mechanisms, there is no guarantee that the extracted mechanisms are faithful to the original model since they are from the distilled model.

Model-X Knockoffs [18] is a new statistical tool investigating the relationship between a large set of explanatory variables and a response T. It considers a very general conditional model, where T can depend in an arbitrary fashion on the variables' covariates $R = (R_1, \cdots, R_p)$. From a set of hundreds or thousands variables, Model-X Knockoffs can identify a smaller subset potentially explaining the response while rigorously controlling the the False-discovery-rate (FDR), which will be defined in more details (Eq. (4)).

Specifically, for each sample of R, Model-X Knockoffs generates a knockoff copy $\tilde{R}$ satisfying $Y \perp\!\!\!\perp \tilde{R}|R$ and the *pairwise exchangeable* property [18]. Then, the importance measure U_j and $\tilde{U}_j$ are computed for each R_j and $\tilde{R}_j$, respectively. After that, the statistics $W_j = U_j - \tilde{U}_j$ is evaluated for each feature. A large positive value of W_j implies evidence against the hypothesis that the j^{th} feature is not in the Markov blanket of T (see Sect. III for more details of the Markov blanket). The work [18] has shown that exactly controlling the FDR below the nominal level q can be obtained by selecting $\hat{\mathcal{R}} = \{j : W_j \geq \tau_q\}$, with

$$\tau_q = \min\left\{t > 0 : \frac{1 + |\{j : W_j \leq t\}|}{|\{j : W_j \geq t\}|} \leq q\right\}.$$

In this paper, we use $(.,.)$ to denote the input-response pair for Model-X Knockoffs as a general solver for the Markov blanket, e.g. (R, T) for the formulation above.

III. Critical Neurons Identification with NeuCEPT-discovery

Although modern DNNs contain from thousands to millions neurons, only a small portion of neurons contributes mostly to the predictions [19], [20]. We denote such neurons as *critical neurons of the predictions* or *critical neurons* for short. Identifying critical neurons not only reduces the complexity of mechanism's discovery (Sect. IV) but also offers more compact explanations for the predictions. Unfortunately, due to the sequential structure of DNNs, identifying critical neurons is a daunting task, from formulating a suitable objective function to solving the problem and interpreting those neurons' activation.

A. Problem Formulation

We consider DNNs in classification problems with the forwarding function $y = f(x)$, where $y \in \mathbb{R}^m$ is a logit and $x \in \mathbb{R}^n$ is an input. The neural network has a sequential structure with L layers, each layer has k_l neurons ($l = 1, ..., L$). The activation of neurons at any layers on a given input can be computed by forwarding the model. We denote this computation as $z_l = f_l(x)$ where $f_l : \mathbb{R}^n \to \mathbb{R}^{k_l}$. Then $z = [z_0, ..., z_L]$ is the activation of all model's neurons on that input. We use capital letters to refer to random variables, i.e. Z_l is the random variable representing the activation of neurons at layer l. The superscript notation refers to the random variables associated with a subset of neurons. For instance, given a subset $\mathcal{S}$ of neurons at layer l, $Z_l^{\mathcal{S}}$ is the random activation of the neurons in $\mathcal{S}$ at layer l. Due to the forwarding structure of the model, the activation of neurons at a given layer depends only on the activation of neurons at the previous layer, i.e. $Z_l \perp\!\!\!\perp Z_j | Z_{l-1}, \forall j = 0, ..., l-2$, where $\perp\!\!\!\perp$ denotes the independent relationship. Thus, we have the Markov chain:

$$X = Z_0 \to Z_1 \to \cdots \to Z_L. \tag{1}$$

Given a prediction, a layer l and the corresponding random activation Z_l, we would like to identify a subset of the critical neurons at that layer, i.e. the subset of neurons containing the most information on the prediction of interest. From an information-theoretic approach, we formalize the notion of criticality using mutual-information (MI) and formulate the critical neurons identification (CNI) problem as:

$$\mathcal{S}_l = \text{argmax}_{\mathcal{S} \subseteq \mathcal{N}_l} I\left(Z_l^{\mathcal{S}}; Z_{l+1}^{\mathcal{S}_{l+1}}\right), \text{ s.t. } \mathcal{S} \in \mathcal{C}, \tag{2}$$

where $\mathcal{N}_l$ is the set of neuron at layer l, I is the joint mutual-information function [21] and $\mathcal{S} \in \mathcal{C}$ represents some complexity constraints for compact solutions. Intuitively, at each layer, we search for the neurons holding the most information on the neurons solved in the next layer. By bounding the first optimization at the last layer L to maximize $I\left(Z_{L-1}^{\mathcal{S}}; Y\right)$, where $Y = Z_L^{\{o\}}$ and o is the neuron associated with the prediction's class, we enforce the sub-sequence optimizations at the earlier layers to search for the neurons holding the most information on the examined prediction.

B. Solutions with Precision Guarantee

As CNI is in NP-hard [1], we introduce NeuCEPT-discovery to approximate CNI with precision guarantee. As an abstract level, NeuCEPT-discovery considers each pair of a layer's activation-output of the examined DNN as an input-response pair and conducts critical neuron selection on them. Different from the sequential formulation of CNI (2), NeuCEPT-discovery is executed in a pair-wise manner and, consequently, can be implemented efficiently. Our theoretical results guarantee that the modification to pair-wise optimization still achieves the specified precision level.

Using the Markov blanket. Given a random variable/response T, we denote $\mathcal{M}_l(T) \subseteq \mathcal{N}_l$ as the smallest set of neurons at layer l such that, conditionally on the variables in that set - $Z_l^{\mathcal{M}_l(T)}$, T is independent of all other variables at layer l. In the studies of graphical models, the set $\mathcal{M}_l(T)$ is commonly addressed as the Markov blanket (MB)[2] of T. We just make a slight modification by restricting the set of variables to a certain layer of the model. Under very mild conditions about the joint distribution of T and Z_l, the MB is well defined and unique [24]. We follow researchers in the field, assume these conditions [25] and proceed from there.

We have $\mathcal{S}_l = \mathcal{M}_l\left(Z_{l+1}^{\mathcal{S}_{l+1}}\right)$, the MB at layer l of $Z_{l+1}^{\mathcal{S}_{l+1}}$, achieves the maximum of the objective (2) since the MB contains all information about $Z_{l+1}^{\mathcal{S}_{l+1}}$. Using the MB, we have a straight approach to solve (2): Given the activation of interest at the last layer $Y = Z_L^{\{o\}}$, we solve for $\mathcal{M}_{L-1}(Y)$ - the MB at layer $L-1$. Then, at layer $L-2$, we find the MB of the variables in $\mathcal{M}_{L-1}(Y)$. The process continues until the first layer is reached. The computation can be described as:

$$\mathcal{S}_{L-1} \leftarrow \mathcal{M}_{L-1}(Y), \quad \mathcal{S}_{l-1} \leftarrow \mathcal{M}_{l-1}\left(Z_l^{\mathcal{S}_l}\right). \tag{3}$$

Algorithm 1: NeuCEPT-discovery.

Input : Samples of model's activation $Z = (Z_1, ..., Z_M)$ at M given layers. Precision thresholds $p = (p_1, ..., p_M)$.

Output : Estimation of critical neurons at all examined layers $\hat{\mathcal{M}}_1, \cdots, \hat{\mathcal{M}}_M$.

$Y \leftarrow Z_{M+1}^{\{o\}}$.
For $l = 1$ to M do:
    $\hat{\mathcal{M}}_l \leftarrow$ estimation of the Markov blanket of Y at layer l with precision control p_l.
Return $\hat{\mathcal{M}}_1, \cdots, \hat{\mathcal{M}}_M$.

Controlling the precision. Directly solving (3) is impractical as the problem is in NP-hard [26]. Additionally, estimating the distribution of $Z_l^{\mathcal{S}_l}$ via sampling is also impractical due to

[1]The CNI can be considered as a general version of the feature-selection problem with mutual-information objective, which is known to be NP-hard [22].

[2]There is an ambiguity between the notions of *Markov-blanket* and *Markov-boundary*. We follow the notion of the *Markov-blanket* defined in [23], which is consistent with [18]. In fact, the Markov-blanket of a random variable T, i.e. $\mathcal{M}(T)$, is the minimum set of variables such that, given realizations of all variables in $\mathcal{M}(T)$, T is conditionally independent from all other variables.

the curse-of-dimensionality. Our key observation to overcome those challenges is that the MB of the model's output variable Y at each layer l is a subset of $\mathcal{S}_l$ (Eq. (3)). As a result, given a solver solving for $\mathcal{M}_l(Y)$ with a precision at least p, the output of that solver is also an approximation of $\mathcal{S}_l$ with the precision at least p. This allows us to solve for $\mathcal{M}_l(Y)$ instead of $\mathcal{S}_l$ and overcome the high-dimensionality of $Z_l^{\mathcal{S}_l}$. We exploit this observation and implement it in the NeuCEPT-discovery step of our algorithm, which is described in Algorithm 1. The proof that NeuCEPT-discovery achieves precision guarantee is based on the following Theorem 1.

Theorem 1. *Suppose we have a solver solving for the MB of a set of random variables and apply that solver to each layer of a neural network as described in equation (3), then the solution returned by the solver at each layer must contain the MB of the neural network's output at that layer, i.e.* $\mathcal{M}_l(Y) \subseteq \mathcal{S}_l,\ \forall l = 0, ..., L-1$.

Proof. For simplicity, we consider the following Markov chain $Z_0 \to Z_1 \to Y$. We now show that $\mathcal{M}_0(Y) \subseteq \mathcal{M}_0\left(Z_1^{\mathcal{M}_1(Y)}\right)$. We have:

- Z_1 determines Y and $Y \perp\!\!\!\perp \left\{Z_1 \setminus Z_1^{\mathcal{M}_1(Y)}\right\} | Z_1^{\mathcal{M}_1(Y)}$ so that $Z_1^{\mathcal{M}_1(Y)}$ determines Y. We can also write this statement as $Z_1^{\mathcal{S}_1}$ determines Y.
- Z_0 determines $Z_1^{\mathcal{M}_1(Y)}$ and
$$Z_1^{\mathcal{M}_1(Y)} \perp\!\!\!\perp \left\{Z_0 \setminus Z_0^{\mathcal{M}_0\left(Z_1^{\mathcal{M}_1(Y)}\right)}\right\} | Z_0^{\mathcal{M}_0\left(Z_1^{\mathcal{M}_1(Y)}\right)}.$$
so that $Z_0^{\mathcal{M}_0\left(Z_1^{\mathcal{M}_1(Y)}\right)}$ determines $Z_1^{\mathcal{M}_1(Y)}$. We can write this statement as $Z_0^{\mathcal{S}_0}$ determines $Z_1^{\mathcal{S}_1}$.
- Combine the two above statements, we have $Z_0^{\mathcal{M}_0\left(Z_1^{\mathcal{M}_1(Y)}\right)}$ determines Y.

On the other hand, we have $\mathcal{M}_1(Y)$ is the smallest subset of neurons at the Z_0 layer that determines Y. Due to the uniqueness of the minimal set that separates Y from the rest of the variables (which is the MB of Y) [24], we have $\mathcal{M}_1(Y) \subseteq \mathcal{M}_1\left(Z_2^{\mathcal{M}_2(Y)}\right)$.

The proof generalizes for the case of L layers Markov chain $Z_0 \to Z_1 \to \cdots \to Z_L$ as the same arguments can be applied to conclude that $Z_l^{\mathcal{S}_l}$ determines $Z_{l+1}^{\mathcal{S}_{l+1}}$. This would lead to the fact that all $Z_l^{\mathcal{S}_l}$ can determine Y; hence, $\mathcal{S}_l$ contains $\mathcal{M}_l(Y)$ due to the uniqueness of the MB [24]. □

We now can formalize and prove the statement that any MB solvers with a precision guarantee p on the input-response pair (Z_l, Y) can be used to solve for the MB of the pair $(Z_l, Z_{l+1}^{\mathcal{S}_{l+1}})$ with a precision at least p in the following Corollary 1:

Corollary 1. *Suppose we have a solver solving for the MB of a random response T with the precision at least p for a given $0 < p < 1$. Let $\hat{\mathcal{M}}_l$ be the output of that solver on the input-response pair (Z_l, Y) defined in procedure (3). Then, $\hat{\mathcal{M}}_l$ also satisfies the precision guarantee p as if we solve for the input-response pair $(Z_l, Z_{l+1}^{\mathcal{S}_{l+1}})$.*

Proof. Denote $q = 1 - p$. Since the precision is one minus the FDR, we can instead prove:

$$\text{FDR} := \mathbb{E}\left[\frac{\#\{j : j \in \hat{\mathcal{M}}_l \setminus \mathcal{S}_l\}}{\#\{j : j \in \hat{\mathcal{M}}_l\}}\right] \leq q. \tag{4}$$

From Theorem 1, we have $\mathcal{M}_l(Y) \subseteq \mathcal{S}_l$ for all $l = 0, \cdots, L-1$. This implies:

$$\begin{aligned} &\hat{\mathcal{M}}_l \setminus \mathcal{S}_l \subseteq \hat{\mathcal{M}}_l \setminus \mathcal{M}_l(Y) \\ \Longrightarrow &\#\{j : j \in \hat{\mathcal{M}}_l \setminus \mathcal{S}_l\} \leq \#\{j : j \in \hat{\mathcal{M}}_l \setminus \mathcal{M}_l(Y)\} \end{aligned} \tag{5}$$

On the other hand, as $\hat{\mathcal{M}}_l$ is the solution of the solver on the input-response pair (Z_l, Y) with FDR less than or equal to q:

$$\mathbb{E}\left[\frac{\#\{j : j \in \hat{\mathcal{M}}_l \setminus \mathcal{M}_l(Y)\}}{\#\{j : j \in \hat{\mathcal{M}}_l\}}\right] \leq q. \tag{6}$$

Combining (5) and (6), we have the Corollary. □

Corollary 1 enables us to exploit any solver with precision control to efficiently solve for procedure (3) with precision guarantee. In our implementation of NeuCEPT-discovery, we use Model-X Knockoffs [18] (discussed in Sect. II).

IV. Information-theoretic Interpretation and NeuCEPT-learning of Critical Neurons

The goal of finding critical neurons is to correctly identify the model's mechanisms. Sect. IV-A discusses in more detail about mechanisms and how the MI objective in Eq. (2) is apt for the task. Sect. IV-B describes how NeuCEPT extracts information from critical neurons to identify the model's mechanism.

A. Information-theoretic Interpretation

Mechanism. Previous analysis of DNNs [20] and our examples, e.g. case studies later shown in Figs. 13 and 14, reveal distinctive patterns of neurons' activation shared among some input samples. This similarity suggests they might be processed in the same manner by the model, which is what we call *mechanism*. Similar to how unlabeled data is handled in unsupervised learning, mechanism in this work is modeled as a discrete latent random variable whose realization determines how the predictions are generated. Fig. 4 provides an intuition on the relationship between the neurons' activation and mechanisms under this assumption. Suppose the latent mechanism variable C determines the generation of predictions of the class *goldfish* in the Inception-v3. Different realizations of C, i.e. 0 or 1, result in different patterns in the activation Z. On one hand, these patterns specify how the model predicts, which is the intuitive meaning of mechanism. On the other hand, observing the activation on some neurons, i.e. critical neurons, can be sufficient to determine the realization of C, i.e. the model's underlying mechanism.

Explainability power. An intuitive necessary condition on the selection of critical neurons is that their activation should determine (or significantly reduce the uncertainty of) the mechanism C. We call this condition *explainability power*. To see how the objective (2) fits into this condition, let's

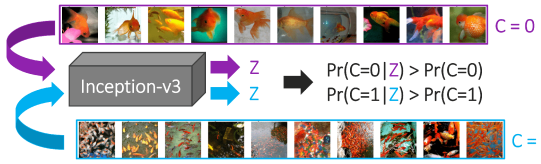


Fig. 4: The probabilistic intuition of the neurons' activation and mechanism: the mechanism determines the activation, and, conversely, observing the activation can reveal the mechanism.

consider its MB solution $\{\mathcal{S}_l\}_{l=0}^{L-1}$. From the definition of the MB, for any set of neurons $\mathcal{R}_l$ at a layer l that is disjoint with $\mathcal{S}_l$, we have $Z_l^{\mathcal{R}_l}$ is independent with $Z_{l+1}^{\mathcal{S}_{l+1}}$ given $Z_l^{\mathcal{S}_l}$. Since $Z_{l+1}^{\mathcal{S}_{l+1}}$ determines $Z_{l+k}^{\mathcal{S}_{l+k}}$ for all $k > 1$, variables in $\mathcal{R}_l$ must also independent with $Z_{l+k}^{\mathcal{S}_{l+k}}$ given $Z_l^{\mathcal{S}_l}$. Thus, knowing $Z_l^{\mathcal{R}_l}$ does not provide any additional information on how the model generates the prediction of interest, i.e. $\{\mathcal{S}_l\}_{l=0}^{L-1}$ is sufficient.

Non-redundancy. Neurons' activation not only determine the mechanisms and the predictions but also contain other information on the data. However, not all information stored in the activation is necessarily used by the model in generating the examined predictions. Thus, another desirable property of the identified neurons is that they must be used by the model in generating the predictions of interest. We call this condition *non-redundancy*. Intuitively, selecting more neurons gives us more explainability power and less non-redundancy.

To demonstrate how our proposed objective (2) meets the notion of non-redundancy, we compare it with another objective aiming to identify the set of globally important neurons derived from the Markov chain (1):

$$\mathcal{S}_l^* = \text{argmax}_{\mathcal{S}\subseteq\mathcal{N}_l} I\left(Z_l^{\mathcal{S}}; Z_{l+1}, ..., Z_L\right), \text{ s.t. } \mathcal{S} \in \mathcal{C}. \quad (7)$$

This objective tells us how much we know about the model's later layers given the activation of the neurons in $\mathcal{S} \subseteq \mathcal{N}_l$. From the lens of information theory, information encoded in $\{\mathcal{S}_l^*\}_{l=0}^{L-1}$ can fully determine all information of the models and satisfies the notion of explainability power. However, for a specific prediction, those information is too excessive since the objective consider all classes equally. Mechanisms discovered on top of them is prone to redundancy. This reason motivates us to use the objective (2). This also demonstrates the importance of the critical neuron discovery step in NeuCEPT.

B. NeuCEPT-learning on Critical Neurons

Given the critical neurons identified by NeuCEPT-discovery, NeuCEPT-learning extracts their activation to identify the model's mechanisms. Since the ground-truth mechanisms are not given, it is natural to consider the mechanism identification/discovery problem as an unsupervised learning task, which has been extensively studied [27].

Algorithm 2 describes how NeuCEPT extracting information from critical neurons. Besides their activation, the algorithm's inputs include a set of compactness parameters limiting the number of representative neurons and an integer K guessing the number of mechanisms. The usage of the compactness parameters is common in many existing explanation methods

Algorithm 2: NeuCEPT-learning.

Input : Critical neurons' activation at M examined layers, denoted as $Z^{\mathcal{S}}$.
The number representative neurons $\{v_i\}_{i=1}^{M}$.
Number of clusters/mechanisms K.

Output : The explained-representations of all inputs and their corresponding mechanisms/clusters.

```
# Constraints enforcement
For l = 1 to M do:
    V_l ← Feature Agglomeration (Z_l^{S_l}) with constraint
          |V_l| ≤ v_l or selecting top v_l neurons.
# Unsupervised learning
g ← Initialize an unsupervised clustering model with K
    components.
Fit g on V = (V_0, ··· , V_{L−1}).
Mechanism C ← g(V).
Return C.
```

for the sake of visualization. The first step of NeuCEPT-learning is to enforce this compactness requirement, i.e. either by selecting the top neurons identified in NeuCEPT-discovery or by feature-agglomerating those neurons into a smaller set of representative neurons. Similar techniques have been used to apply Model-X Knockoffs on real-data with very high-correlated features [18]. Then, an unsupervised learning method is chosen among K-means, Gaussian Mixture, and Agglomerative Clustering to map each input sample to one of the K clusters representing K mechanisms.

We end this section with a demonstration of some NeuCEPT's outputs in analyzing predictions of Inception-v3. The examined layers are the last layers of the *Mixed-5d* and the *Mixed-6e* blocks (Fig. 5). The number of representative neurons are restricted to 5 and 3. Next to each input, we show a graph representing the activation's level (red for high, blue for low) of those representative neurons from the *Mixed-5d* (left) to the *Mixed-6e* (middle). The last dot represents the output neuron (right). NeuCEPT-learning helps us visualize similar activation's patterns among samples of the same mechanism, and differentiate them from another mechanism.

V. Experiments: Setting and Results

Our experiments focus on evaluating the explainability power and the non-redundancy properties, mentioned in Sect. IV-A (source code [28]). While evaluating the explainability power can be conducted via ablation study, evaluating the non-redundancy is more challenging as we normally do not know the underlying mechanism. To tackle this, we propose a training setup, called *prior-knowledge training*, so that both the non-redundancy and the explainability power can be evaluated.

A. Prior-knowledge training (PKT).

We introduce the PKT so that models with partially known mechanisms can be obtained. Specifically, during the PKT,

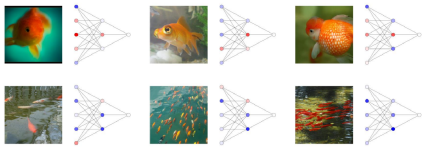


Figure 5: Examples of NeuCEPT's outputs on Inception-v3. Images from each row are from the same cluster learnt by NeuCEPT.

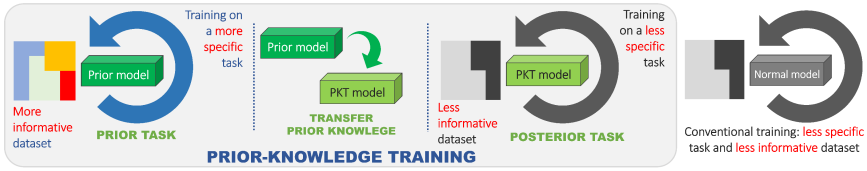


Figure 6: Prior-knowledge training: parameters of the prior-model trained on a more informative task are transferred to the PKT model, which is later trained on a less specific task. The mechanisms of the PKT model is expected to be different from normally-trained models.

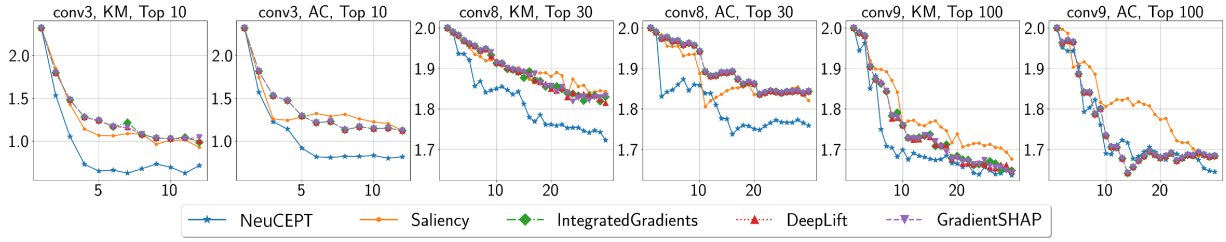


Fig. 7: CE (bits) resulted from K-mean (KM) and Agglomerative clustering (AC) clustering on activation of neurons selected by different methods. The x-axis and y-axis are the number of clusters and the CE, respectively. The first 2 plots are the analysis at layer *conv3* of the LeNet on the class *even* using top-10 neurons. The last 4 plots are of VGG on class *object* with top 30/100 neurons at layer *conv8* and *conv9*.

certain prior-knowledge is injected into a model, called PKT model, before the training of its main task, called *posterior-task*. Due to the injection, those mechanisms are expected to be embedded and used by the PKT model to conduct its posterior-task. With that knowledge on the PKT model, we then can evaluate explaining methods accordingly.

Fig. 6 describes this process in more details. First, a model, called *prior-model*, is trained on a more specific task, called *prior-task*. The prior-task's training data is chosen such that there exists information that is only available to the prior-model (therefore the term *more informative dataset*). To ensure that information is embedded in the prior-model, the prior-task is the task of predicting that information. The weights of the prior-model is then transferred, completely or partially, to the PKT model before its training on the posterior-task. As such, given a PKT model and a conventionally trained model, a good mechanism discovery algorithm should claim that the PKT model relies on some specific information, i.e. the prior-knowledge, to generate its prediction (explainability power) while the conventionally trained one does not (non-redundancy). In fact, given a cluster label c returned by the algorithm and the label y_{prior} of the informative dataset, the algorithm can be evaluated by the clusters' entropy (CE) metric:

$$\sum_c \sum_{y_{prior}} p(y_{prior}, c) \log\left(1/p(y_{prior}|c)\right),$$

where $p(.)$ is the empirical probability. The lower the CE, the more the clusters/mechanisms identified by the algorithm align with the prior-knowledge. Thus, the CE of PKT model resulted from a good mechanism discovery algorithm should be lower than that of a conventionally trained model.

Readers can refer to Fig. 2 for a better understanding of PKT. In that experiment, the prior-task and the posterior-task are the digit classification and the *even/odd* classification, respectively. The models, from left-to-right, are the prior-model, the PKT model and the conventionally trained model.

B. Experimental setting

Hardware. Our experiments are implemented in Python 3.8 and conducted on a single GPU-assisted compute node with a Linux 64-bit operating system. The allocated resources include 32 CPU cores (AMD EPYC 7742 model) with 2 threads per core, 8 GPUs (NVIDIA DGX A100 SuperPod model) with 80GB of memory per GPU and 100GB of RAM.

Dataset and Model. We experiment with LeNet [29] and VGG [30] trained on MNIST [31] and CIFAR10 [32] dataset, respectively. We use the default Pytorch models [33] with slight modification at the last layer to fit for the tasks. Specifically, the output layers of the normal and the PKT model are changed to 2 neurons so that they predict the *even/odd* in MNIST and *animal/object* in CIFAR10. Note that, in training the PKT model on CIFAR10, we freeze the parameters of the first 6^{th} convolutional layers (there are 9 layers) to better maintain the prior-knowledge/mechanisms transferred from the prior model.

Baseline. To our knowledge, there exists no work directly addressing the proposed problem. As such, we adopt state-of-the-art local explanation methods, including Saliency [12], Integrated-Gradients (IG) [13], Deeplift [11] and Gradient-SHAP (G-SHAP) [8], implemented by Captum [34], to identify critical neurons. Specifically, the neurons are selected by their total attribution scores given by the explanation methods on all images of the examined class.

The choice of precision values. There are 3 factors deciding the choices of precision threshold in our experiments: the layer's location, the number of neurons in that layer and the model's complexity. The general intuition is, the deeper the layer, the smaller of number of neurons in that layer and the less complexity the model, the easier the critical neurons can

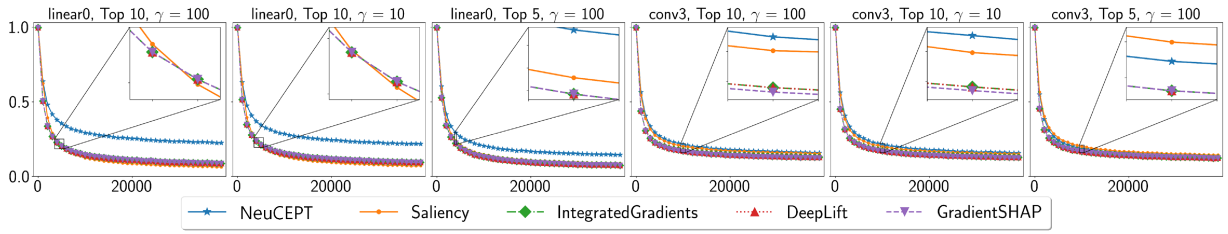


Fig. 8: Ablation tests on LeNet. The x-axis and y-axis are the noise levels and the test accuracy.

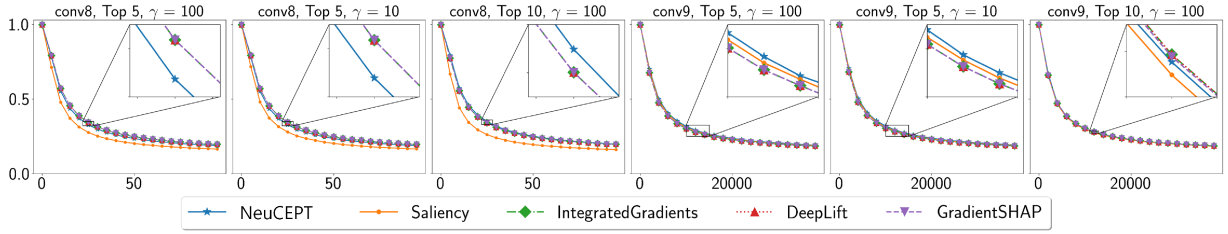


Fig. 9: Ablation tests on VGG. The x-axis and y-axis are the noise levels and the test accuracy, respectively.

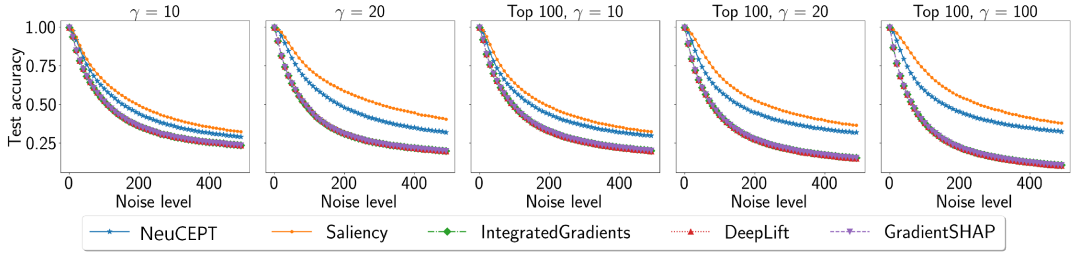


Fig. 10: Ablation results at input layer of LeNet. The two plots on the left show the ablation results with continuous noise, i.e. the noise are added based on the score given by the associated methods. The three figures on the right shows results when top 100 features are selected. γ is an exponential decay parameter determining how noise is distributed among neurons.

be identified. Thus, in such case, we can set a high value of precision and expect good solutions. In experiments for LeNet, the precisions are between 0.9 and 0.98. For CIFAR10, the values are between 0.4 and 0.8 based on the layer and the number of neurons. In Inception-v3 and CheXNet, the values are 0.6 for all layers.

Model-X Knockoffs implementations. One major concern on the usage of Model-X Knockoffs on high-dimensional real-data and, specifically, on neurons' activations. The challenge arises when we need to chose between two or more very highly-correlated features. Note that "this is purely a problem of power and would not affect the Type I error control of knockoffs" [18]. To overcome this, the method clustered features using estimated correlations as a similarity measure. After that, one representative feature is chosen from each cluster. The representatives are then used in the feature-discovery task instead of the original features. Our feature agglomeration step (described in Sect. IV and Algo. 2) is based on this alleviation.

C. Explainability power.

CE metric. High explainability power implies the selected neurons hold more information determining the mechanism. Thus, the CE should be small if the guessing number of clusters K (specified in Sect. IV-B) aligns with the actual number of the mechanisms. In the posterior-tasks of MNIST and CIFAR10, we train LeNet and VGG to classify *even/odd* and *animal/object*, respectively. The actual numbers of the mechanisms in the PKT models are expected to be at least $(5, 5)$ and $(4, 6)$ for each pair of labels in the task. The reason is they are the number of original labels belonging to those categories, i.e. 5 odd digits, 5 even digits, 4 animals and 6 objects.

Fig. 7 shows the CE of the clusters learnt on neurons identified by different explanation methods versus the number of clusters K. MNIST's results clearly show neurons identified by NeuCEPT can differentiate the inputs based on their original labels embedded by prior-knowledge training. Notably, when $K = 5$, NeuCEPT achieves its lowest, which aligns with our expectation on the number of actual mechanisms that the PKT model uses. VGG's experiments show similar result with less distinction in the number of clusters.

Ablation test. Since the mechanism specifies the prediction, critical neurons should have high impact on model's predictions. This impact can be evaluated via ablation test [35], in which noise of different levels and configurations are added to those neurons and the model's accuracy is recorded accordingly. If the neurons have explainability power, protecting them from the noise should maintain the model's accuracy [35], [36].

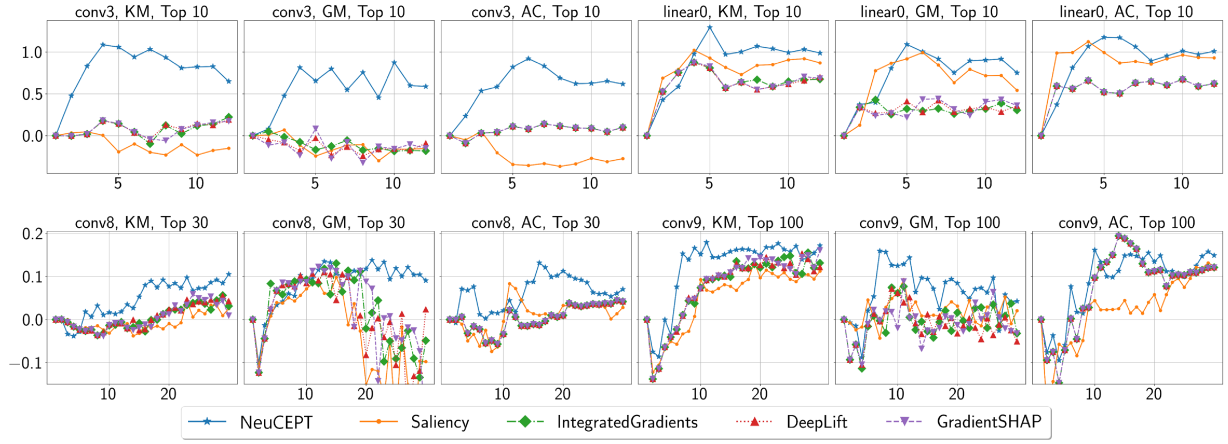


Fig. 11: The CE differences (bits) between the conventionally trained model and the PKT model. Clusters are learnt using k-mean (KM), Gaussian Mixture (GM), and Agglomerative clustering (AC) on activation of neurons chosen by different methods. The x-axis and y-axis are the number of clusters and the CE differences. Top figures show the results of the LeNet at different layers with top 10 important neurons. Bottom are the results of VGG with top 30/100 neurons at convolutional layer 8 and 9.

Fig. 8 and 9 show our ablation tests of neurons identified by different methods. In the figures, *Top* is the number of neurons protected from the noise, determined by the score of the explanation methods. γ is an exponential decay parameter determining how noise is distributed among neurons: the larger the γ, the lesser the noise added to the protected neurons. The tests are conducted at different layers of LeNet and VGG. The results show that neurons identified by NeuCEPT hold higher predictive power among most of the experiments

Fig. 10 shows our adaptation of NeuCEPT to LeNet to select important input features for each class. The purpose of this experiment is to demonstrate that NeuCEPT can adapt to the concerns regarding Model-X Knockoffs on high-dimensional highly-correlated data and unknown covariance [18]. The plots show that the explainability power of features selected by NeuCEPT is comparable with other methods.

D. Non-redundancy.

The following experiments show that NeuCEPT meets the non-redundancy requirement, i.e. it should return a significantly higher CE when examining conventionally trained models, which has no knowledge on the original labels. Fig. 11 plots the differences in the CE between the conventionally trained models and the PKT models on MNIST and CIFAR10. While the differences in CE at certain layers of some other methods fluctuate around 0, indicating there is no difference between the two models, NeuCEPT consistently supports the expectation that only the PKT models recognize the prior-knowledge and the conventionally trained models do not. Note that NeuCEPT differentiates the models simply by observing their activation on critical neurons. This means those critical neurons are indeed used by the model in generating the examined predictions.

E. Running-Time.

Table I shows the average running-time of all methods in searching for the important neurons for a single class. All experiments are on 10000 test samples. A straight comparison among methods is not trivial due to the difference in resource utilization and hyper-parameters selections. Technically, NeuCEPT only needs to run Model-X Knockoffs once; however, we run it 50 times for more stable results. The main takeaway is that NeuCEPT can be run in a reasonable amount of time on moderate-size models such as VGG.

TABLE I: The running-time in seconds of tested methods. For the 4 other explanation methods, we use the Captum library which runs on GPUs. Our NeuCEPT runs entirely on CPU cores and the reported running-time is for one iteration. For more stable results, the experimental results of NeuCEPT are obtained after 50 iterations.

Model	Layer	NeuCEPT	Saliency	IG	DeepLift	G-SHAP
LeNet prior	conv3	3.18	17.35	79.59	51.32	31.42
	linear0	1.93	16.25	79.20	20.12	31.13
LeNet normal	conv3	2.86	16.52	79.95	51.22	31.03
	linear0	2.00	15.47	77.57	19.93	31.08
LeNet PKT	conv3	2.76	16.00	80.46	50.49	31.93
	linear0	1.94	15.53	77.13	20.05	30.77
VGG11 prior	conv8	20.86	167.82	379.16	195.01	355.65
	conv9	22.36	180.6	362.28	183.97	305.65
VGG11 normal	conv8	26.12	307.74	709.38	335.29	708.50
	conv9	23.22	307.87	708.71	309.38	707.38
VGG11 PKT	conv8	17.26	223.24	457.78	263.63	457.09
	conv9	18.80	222.50	457.81	220.53	457.32

VI. Case Studies

This section provides some usages of NeuCEPT in analyzing different interesting aspects of DNNs: the linear separability (studied along with the Linear Probe), the explainability (studying on Inception-v3 model) and the predictions' reliability (studying on CheXNet model).

Linear Probe (LP) [5] are linear classifiers hooking on intermediate layers of DNNs to measure their linear separability. Intuitively, a low loss of the probe in predicting some labels at a layer implies that the activation of that layer is more linearly separable. This metric is important in the analysis of DNNs since it can be used to characterize layers, to debug models, or to monitor training's progress.

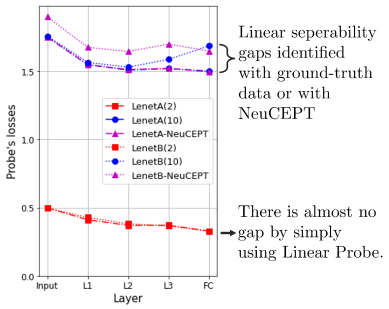
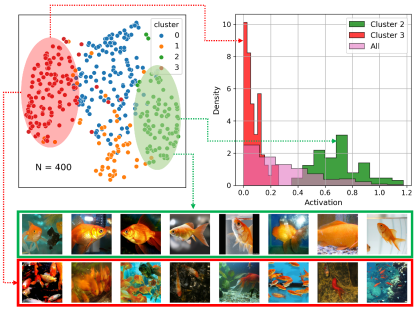
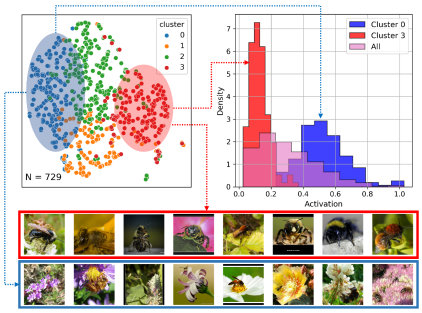


Fig. 12: Using LP on the predicted labels, i.e. *even/odd* (indicated by notation (2)), does not reveal the difference in the linear separability of LeNetA and LeNetB. However, with the aiding of the ground-truth labels or NeuCEPT, LP can detect the linear separability of the two models (indicated by notation (10) and notation NeuCEPT).

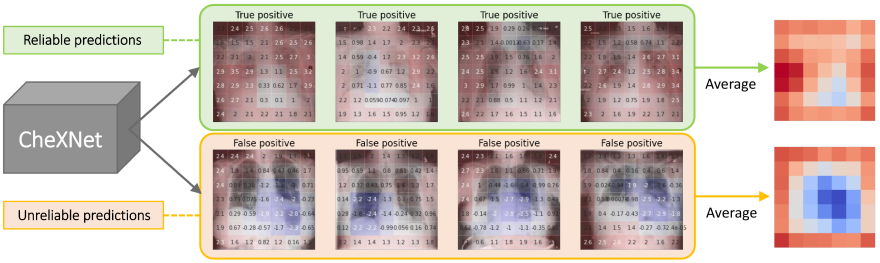


Neuron 322 differentiates predictions of two sub-classes (red/green) of the class *goldfish*.



Neuron 305 differentiates predictions of two sub-classes (blue/red) of the class *bee*.

Fig. 13: Results obtained by applying NeuCEPT with $K = 4$ on classes *goldfish* and *bee* at the *Mixed6e* block of Inception-v3. For each class, we use NeuCEPT to identify a neuron whose activation's histogram of 2 distinctive clusters (among 4) are shown on the top-right of the sub-figures. While these neurons are not among the top-activated neurons of each class, they hold vital information in differentiating the predictions as inputs of one cluster activate the neuron and the others do not. The top-left and the bottom figures plot the low-dimensional representations of the inputs and some images drawn from each cluster for visualization.



Experiment's info	
Model	CheXNet
Examined predictions	*'Pneumonia'*
Total samples	2978
Positive samples	1035
Positive predictions	1035
Model's precision	60.6%
Reliable-subclass precision	91.3% (180 samples)
Unreliable-subclass precision	36.7% (150 samples)

Fig. 14: NeuCEPT discovers a subclass of unreliable predictions with precision of 36.7% (compared to 60.6% on the whole class) in CheXNet. Explanation heat-maps provided by the model also suggest that the unreliable predictions are made by the area at the boundary, not the lung's area of the image (red means high importance score).

We applied LP to LeNet on MNIST dataset (Fig. 2) and plotted the results in Fig. 12. The notations (2), (10) and NeuCEPT refer to the labels that the LPs predict: (2) is the model's labels *even/odd*, (10) is the original digit labels and NeuCEPT is the mechanisms identified by NeuCEPT. From the bottom visualization of Fig. 2, LeNet(A) is expected to be more linearly separable as its point-cloud is divided into more distinctive clusters. However, simply applying LP on the model's labels (notation (2)) shows little difference between the two models. On the other hand, both results of the LPs with the digit labels (notation (10)) and with the unsupervised clusters/mechanisms learnt by NeuCEPT state that LeNet(A) is more linearly separable. Note that the gap between LeNetA(10) and LeNetB(10) cannot be obtained by LP in practice due to the lack of the ground-truth knowledge. This example shows how NeuCEPT can be used to strengthen the results obtained by LP and further the study of linear separability.

Inception-v3. As attribution methods mainly attribute scores to neurons with the most contribution to the prediction, aggregating the resulted scores among inputs of a class naturally gives highly attributing neurons of the class. However, as the activation of those neurons can be highly similar among samples of the class (a trivial example is the output's neuron associated with the class), they hold limited information on how the model processes its inputs differently, hence, do not fit for the task of identifying mechanisms.

Our analysis on the class *goldfish* and *bee* of Inception-v3 [6] (Fig. 13) demonstrates the above claim. Specifically, at the *Mixed6e* block of the model, NeuCEPT identifies a neuron that is not among the top-highly activated of the examined class, but hold valuable information about how the model processes inputs of that class. In fact, for each class, there are two subsets of inputs such that members of one subset activate the neuron while those of the other do not. Simply using activation's level, which can be considered as the simplest form of attribution method, would miss this neuron. Interestingly, there are distinctive visual concepts associated with images belonging to clusters identified by NeuCEPT. In the class *goldfish*, one subset is about *a single fish* while the other is about *a shoal of fishes*. For the class *bee*, it is *a single bee* versus *a single flower* or *a bunch of flowers*. This observation supports the hypothesis that the two subsets of images are indeed processed differently by the model.

CheXNet is a modern DNN which can detect pneumonia from chest X-rays at a level exceeding practicing radiologists [7]. However, the work [37] found out that the model has learned to detect a hospital-specific metal token on the scan to generate some classifications. This finding raises an important question: are there systematical methods to differentiate reliable predictions from unreliable ones.

We partially address that question using NeuCEPT (Fig. 14): we discover a subclass of 150 unreliable *'pneumonia'* predictions with a much lower precision than that on the whole class (2978 samples), i.e. 36.7% compared to 60.6%. We then use the local explanations provided by the model itself to further verify our finding. We observer that the false-positive predictions generally are made using features outside of the lung's area. We provide this example to demonstrate that identifying mechanisms underlying the model's predictions can help evaluate the reliability of individual prediction.

VII. Conclusion

This work aims to learn mechanisms underlying DNNs' predictions to provide a deeper explanation on how the models work. From an information-theoretic viewpoint, the problem is formulated as a sequence of MI maximization, whose solution, called critical neurons, can be solved by our NeuCEPT-discovery with guarantee. We develop NeuCEPT-learning, an algorithm clustering inputs based on their activation on critical neurons, to reveal the model's mechanisms. We further designed a training procedure so that the mechanism discovery task can be evaluated. Our experiments and case studies show that NeuCEPT consistently identifies the underlying mechanisms and reveals interesting behaviors of the DNNs.

Acknowledgement

This work is partially supported by the National Science Foundation under Grant No. FAI-1939725 and SCH-2123809.

References

[1] Z. C. Lipton, "The mythos of model interpretability," *Queue*, vol. 16, no. 3, p. 31–57, Jun. 2018. [Online]. Available: https://doi.org/10.1145/3236386.3241340

[2] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu, "Definitions, methods, and applications in interpretable machine learning," *Proceedings of the National Academy of Sciences*, vol. 116, no. 44, pp. 22 071–22 080, 2019.

[3] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 2009.

[4] J. Wang, X. Jing, Z. Yan, Y. Fu, W. Pedrycz, and L. T. Yang, "A survey on trust evaluation based on machine learning," *ACM Computing Surveys (CSUR)*, vol. 53, 2020.

[5] G. Alain and Y. Bengio, "Understanding intermediate layers using linear classifier probes," *ICLR*, 2017.

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp. 2818–2826.

[7] P. Rajpurkar, J. A. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Y. Ding, A. Bagul, C. Langlotz, K. S. Shpanskaya, M. P. Lungren, and A. Ng, "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning," *ArXiv*, vol. abs/1711.05225, 2017.

[8] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30*, 2017.

[9] M. T. Ribeiro, S. Singh, and C. 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*, 2016.

[10] M. Vu and M. T. Thai, "PGM-Explainer: Probabilistic graphical model explanations for graph neural networks," in *Advances in Neural Information Processing Systems*, 2020.

[11] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[12] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *workshop at International Conference on Learning Representations*, 2014.

[13] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, 2017.

[14] G. Hinton and N. Frosst, "Distilling a neural network into a soft decision tree," 2017. [Online]. Available: https://arxiv.org/pdf/1711.09784.pdf

[15] H. Lakkaraju, S. H. Bach, and J. Leskovec, "Interpretable decision sets: A joint framework for description and prediction," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1675–1684.

[16] C. Yang, A. Rangarajan, and S. Ranka, "Global model interpretation via recursive partitioning," in *IEEE 20th International Conference on High Performance Computing and Communications*, 2018, pp. 1563–1570.

[17] Y. Wang, H. Su, B. Zhang, and X. Hu, "Interpret neural networks by identifying critical data routing paths," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8906–8914.

[18] E. Candes, Y. Fan, L. Janson, and J. Lv, "Panning for gold: Model-x knockoffs for high-dimensional controlled variable selection," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2016.

[19] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA*, 2019.

[20] D. Bau, J. Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba, "Understanding the role of individual units in a deep neural network," *Proceedings of the National Academy of Sciences*, 2020.

[21] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*, USA, 2006.

[22] U. M. Khaire and R. Dhanalakshmi, "Stability of feature selection algorithm: A review," *Journal of King Saud University - Computer and Information Sciences*, 2019.

[23] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[24] J. Pearl and A. Paz, "Confounding equivalence in causal inference," *Journal of Causal Inference*, vol. 2, no. 1, pp. 75–93, 2014.

[25] D. Edwards, *Introduction to Graphical Modelling*. New York, USA: Springer, 2000.

[26] D. Margaritis and S. Thrun, "Bayesian network induction via local neighborhoods," in *Advances in Neural Information Processing Systems 12*, 2000, pp. 505–511.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[28] M. N. Vu, *Source Code for NeuCEPT*, https://github.com/vunhatminh/NeuCEPT.git.

[29] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[30] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pp. 730–734, 2015.

[31] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[32] A. Krizhevsky, "Learning multiple layers of features from tiny images," *University of Toronto*, 05 2012.

[33] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[34] "Captum: A unified and generic model interpretability library for pytorch," *ArXiv*, vol. abs/2009.07896, 2020.

[35] A. M. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[36] M. N. Vu, T. D. Nguyen, N. Phan, R. Gera, and M. T. Thai, "c-eval: A unified metric to evaluate feature-based explanations via perturbation," in *2021 IEEE International Conference on Big Data*, 2021.

[37] J. R. Zech, M. A. Badgeley, M. Liu, A. B. Costa, J. J. Titano, and E. K. Oermann, "Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study," *PLOS Medicine*, vol. 15, no. 11, pp. 1–17, 11 2018.