Quantifying the knowledge in Deep Neural Networks: an overview

Ioanna Valsamara, Ioannis Mademlis, Ioannis Pitas

Abstract

Deep Neural Networks (DNNs) have proven to be extremely effective at learning a wide range of tasks. Due to their complexity and frequently inexplicable internal state, DNNs are difficult to analyze: their black-box nature makes it challenging for humans to comprehend their internal behavior. Several attempts to interpret their operation have been made during the last decade, but analyzing deep neural models from the perspective of the knowledge encoded in their layers is a very promising research direction, which has barely been touched upon. Such a research approach could provide a more accurate insight into a DNN model, its internal state, learning progress, and knowledge storage capabilities. The purpose of this survey is two-fold: a) to review the concept of DNN knowledge quantification and highlight it as an important near-future challenge, as well as b) to provide a brief account of the scant existing methods attempting to actually quantify DNN knowledge. Although a few such algorithms have been proposed, this is an emerging topic still under investigation.

Keywords

Deep Neural Networks, knowledge quantification, knowledge metrics, model interpretability

I. Introduction

Deep Neural Networks (DNNs) are among the most widely used and accurate machine learning models, showing remarkable performance in a variety of tasks [1], [2], [3], [4], [5], [6], [7], [8], [9], [10]. Their efficiency depends directly on the knowledge they manage to encode in their hidden layers during their training process. Thus, explaining and quantifying the knowledge encoded inside a trained DNN has become an important research direction.

Since DNNs were from the start conceived as learning machines [11], early research attempted to exploit statistical learning theory [12], in order to analyze neural network model capacity. Thus, measures such as the Vapnik-Chervonenkis (VC) dimension [13] were used to evaluate DNNs capabilities. VC-dimension is a broad model complexity measure, directly linked to its generalization performance [14], that indicates model learning capacity. The VC-dimension of linear DNNs has been thoroughly investigated in [15], while it was used to explore their generalization capability in [14], [16], [17] and their model complexity in [18], [19], [20]. DNN depth and its impact on DNN model performance has also been investigated by looking into the DNN generalization bound, which has been derived using the VC-dimension [21].

DNNs are typically employed on traditional machine learning tasks such as classification or regression. However, in general, classical statistical learning theory is difficult to employ for measuring the classification capacity of non-linear DNNs. It may lead to loose estimates that are not sufficient to explain DNN model generalization capabilities [22]. Although such metrics could shed some light on the DNN theoretical properties in terms of training data and network architecture, they leave many unanswered questions regarding the *knowledge* encoded in a trained DNN. In fact, DNN architecture affects its classification/regression performance and the amount and quality of the knowledge it learns from a given training dataset.

Moving beyond the narrowly technical realm of machine learning, knowledge, in general, is notoriously difficult to be defined; its existing definitions are far from quantifiable. Essentially, knowledge is composed of mental constructions like reasoning on concepts. In addition to text, its semantic content can also be transmitted and stored in visual and audio formats. The conventional way of representing knowledge is with textual form, where information is expressed in the form of sentences, propositions, or theorems. Additional concept semantics can also be incorporated to produce multimodal knowledge representation forms. However, text-based representation remains the most prevalent approach, utilizing logical propositions/rules that are true. Linguistically and technically, this is the most straightforward way to represent knowledge and quantify it by roughly counting the total number of written sentences that currently exist in human knowledge. While this number is enormous, it is not infinite. The English language has approximately 500,000 words and 15 to 20 words per sentence (on average). Therefore, the maximal number of simple propositions is $0.5^{10} \times 10^{60}$ [23]. Both true and (many more) false or utterly incomprehensible statements are included in this number, which is huge but finite. When focusing on the set of logical propositions that are true, it can be considerably reduced.

From a machine learning perspective, knowledge may generally be described as a function (learned mapping) $\mathbf{y} = \mathbf{f}(\mathbf{x}; \boldsymbol{\theta})$ from input vectors $\mathbf{x}$ to output vectors $\mathbf{y}$ [24]. However, this is a vague definition. Machine learning relies on training data and on humans to configure, launch, and analyze ML systems and their results. DNNs can learn concepts through training, e.g., object recognition. Although some of these concepts can be abstract, most learned concepts are very material-related and can be used in solving everyday real-world problems. This paper argues that, particularly for DNNs, their knowledge quantification should aim to measure the amount of knowledge encoded inside the entire model $\mathbf{y} = \mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$, its intermediate/hidden layers or its individual neurons. This would provide a new perspective on analyzing DNNs, diagnosing problems, and interpreting the superior performance of deep learning, by revealing the internal behavior of a DNN through understanding and estimating the related information flow during the training process. DNN interpretability is an important research area since the prevalent black-box view of Deep Learning creates difficulties in analyzing and, thus, optimizing neural models. Without a practical grasp of how knowledge is encoded inside a trained DNN, and without being able to measure its quantity, it is difficult to cope with any model restrictions and enhance their performance. For deep learning to mature, a much better understanding of encoded knowledge is required. This knowledge is related to several aspects of DNN operation, such as: a) patterns discovered in the input data samples themselves, b) DNN model generalization ability, and c) the patterns extracted separately by each hidden DNN layer, as a function of its input feature map.

Knowledge metrics would allow for detailed comparisons across the various layers of a single DNN, or across the corresponding layers of different DNN models. Such metrics would potentially reveal new ways to enrich network capabilities while providing a new perspective for analyzing DNN performance in various tasks. Finally, measuring DNN knowledge also provides an insightful understanding regarding the nature and inner mechanisms of relevant DNN training and knowledge transfer algorithms. Such an understanding can be used to diagnose, evaluate and optimize the said algorithms, like knowledge distillation [25], deep compression via pruning [26], adversarial attacks [27], object segmentation, or even incremental learning ones.

However, up to now, little effort has been expended on DNN knowledge quantification. Instead, most previous research revolves around visualizing the activation patterns of hidden layers of Convolutional Neural Networks (CNNs) during the inference stage, in order to provide qualitative explanations of their stored knowledge. Indeed, the exact configuration of the CNN convolutional layer activation patterns for a specific input image depends on the knowledge encoded in the trained CNN model parameters, $e.g.$, their convolution kernels. Alternatively, the detection and display of input image patterns, that have the potential to strongly activate a neuron in a specific CNN layer, could be used to infer that a form of knowledge has been encoded therein about these input image patterns. Typically, early CNN filters detect simple visual patterns, like image edges or texture elements, and subsequent CNN convolution kernels encode semantic knowledge about input image objects and object regions. Gradient-based approaches [28] measure the importance of the intermediate-layer activations using the output gradients w.r.t. the input image. Convolutional feature maps can be visualized as images [29]. Alternative approaches visualize the input image pixels attribution/importance/saliency, e.g., by measuring the influence of each input pixel on the final CNN output [30], [31]. *Gradient-weighted Class Activation Mapping* (Grad-CAM) produces visual explanations using the gradients of target patterns into the final convolutional layer. A localization map is produced which highlights important input image regions [32]. Grad-CAM++ is a generalization of Grad-CAM that also provides visual explanations of CNN models [33]. In order to quantify the interpretability of latent CNN representations, the compatibility between individual CNN hidden units and a set of semantic patterns is evaluated in [34]. Interactive visualization tools are designed to specifically visualize DNN architecture and knowledge in [35], [36].

More relevant to this paper is a class of rather recent methods, aiming to *quantitatively* explain the capabilities and characteristics of trained DNNs, such as the relative importance of their various internal representations, their generalization ability or the information flow within them. An approach relying on learning to extract the most informative features for each given data sample, in the form of intermediate DNN layer activation maps, is proposed in [37] for DNN model interpretation. The instance-wise feature selection process exploits the maximization of the mutual information between selected image features and the DNN output. However, it does not quantify knowledge as such but only highlights important input image features.

Mutual information has been used as a metric to estimate the structure of internal representations and the information flow inside a trained DNN [38]. Several other quantitative metrics aiming to explain DNN capabilities have also been proposed lately. To evaluate DNN robustness, *Cross Lipschitz Extreme Value for nEtwork Robustness* (CLEVER) is defined in [39], while Fourier analysis is employed to analyze DNN training with stochastic gradient-based methods and explain their generalization capabilities [40].

DNN knowledge quantification is directly related to the model's generalization capabilities. This claim is based on the underlying intuition that *a lower/higher ability for successful generalization (at the test stage) implies that less/more knowledge was encoded in the model parameters during training*. Thus, the metric of *DNN stiffness* was proposed in order to diagnose and characterize model generalization capabilities [41]. Higher stiffness indicates that a network is learning more generalizable features. In a similar vein, the relationship between trained *DNN sensitivity* and generalization are explored in [42]. Sensitivity is defined as a trained DNN classifier's capability to distinguish between similar inputs belonging to different classes. The contribution of various layers on the generalization ability of DNNs is studied in [43]. Based on the results, a complexity measure is proposed in [44] to analyze DNN generalization power.

Under the assumption that the knowledge stored in an intermediate DNN hidden layer is the set of visual patterns encoded by its neural activations (features, or representations), different stored knowledge can be identified for each input test data sample. Using this approach, *Centered Kernel Alignment* (CKA), closely connected to *Canonical Correlation Analysis* (CCA), has been used to compare such representations from different DNNs and measure their similarities [45]. Quantifiable knowledge consistency between different DNNs has been investigated to diagnose feature representations [46]. A task-agnostic method is proposed to disentangle feature components, which represent the consistent knowledge, from the raw intermediate-layer representations of each DNN. In both [45] and [46], the DNNs can have identical (but differently trained) or different architectures.

Another perspective to explain DNNs is by quantifying the *interactions* between input variables/attributes that are internally transformed by DNNs [47], [48], [49]. The various attributes of the input test vector/data sample do not typically carry significant information individually. In contrast, the DNN infers conclusions based on their correlations and interactions. The strongly interacting ones are considered to form a *prototype feature* that the DNN has previously memorized during training. Multivariate *interaction significance* is defined based on the *Shapley value*, aiming to explain DNNs and interpret their operation [48]. In the specific case of Natural Language Processing (NLP), *interactions* among latent word representations encoded within a DNN are extracted and quantified. A tree structure is generated to organize them hierarchically [47]. In such scenarios, interactions between word representations are proposed as a generic tool to objectively interpret the DNN inference process.

The majority of these quantitative measures are employed for interpreting DNN operation and are only loosely connected with the issue of DNN knowledge encoding. In most cases (not all), there is a lack of clear correlation between such measures and the knowledge stored in the DNN model layers. The naive alternative of conventional DNN classification accuracy evaluation on a specific test dataset cannot quantify neither the internal DNN state nor its representations, thus, it only provides limited hints about its generalization capabilities.

Knowledge quantification metrics would allow for more precise conclusions regarding what a DNN has learned during its training process. Although no commonly accepted definition of DNN knowledge has been proposed yet, several metrics falling under this general area have been presented in recent years and are discussed in the following Sections. The most obvious and naive choice is to simply measure the accuracy of DNN predictions on a known test set. Besides this, two types of more advanced methods have emerged: a) *information-theoretic metrics*, which leverage an individual DNN layer's information to quantify the knowledge it encodes, and b) *knowledge points metrics*, that quantify the knowledge points stored in a trained DNN.

The remainder of this paper is organized in the following manner. Formal notations and definitions are summarized in Section II. Knowledge quantification metrics are presented in Section III. A discussion follows in Section IV, while conclusions are drawn in Section V.

II. Notation and definitions

Deep Neural Networks (DNNs) are neural networks with a count of layers (*depth*) that is greater than or equal to 3: $L \geq 3$. Typically, DNNs have a large number of hidden layers ($L >> 3$). In the context of this paper, the knowledge quantification of *Convolutional Neural Networks* is examined. They can be composed of *convolutional layers*, possibly followed by *fully connected layers*. For a layer l with an activation function $f^l(\cdot)$, the *convolution or fully connected layer neural activation patterns* are respectively defined as:

$$\mathbf{a}^{(l)} = f^l(\mathbf{w}^{(l)} * \mathbf{a}^{(l-1)} + b^{(l)}), \tag{1}$$

or as:

$$\mathbf{a}^{(l)} = f^l(\mathbf{w}^{(l)T}\mathbf{a}^{(l-1)} + b^{(l)}), \tag{2}$$

where $\mathbf{w}^{(l)}$ are the l-th layer convolution kernels, in the case of a convolutional layer, or the l-th layer's synaptic weights, in the case of a fully connected layer respectively. $b^{(l)}$ are the biases. Given a set of classes $\mathcal{C} = \{\mathcal{C}_k, k = 1, .., m\}$ and an input sample $\mathbf{x} \in \mathbb{R}^N$, DNN models predicts a *class* label vector for each input sample $\mathbf{x}_i$.

Definition 1. *Knowledge quantification* is the process of quantifying the amount of knowledge encoded within a trained DNN. It requires an accurate DNN knowledge definition. The methods described in this survey quantify knowledge as a) the DNN success rate on a test set, b) the information encoded in a layer or c) as knowledge points.

Definition 2. *Knowledge points* are the DNN input patterns (units) whose information is discarded much less than the information of other input patterns. Thus, they are more heavily exploited in DNN prediction. In previous literature, they are also referred to as *visual concepts* or *discriminative visual concepts* and are often defined by human annotations [34]. The amount of knowledge encoded by a DNN can be measured as the number of learned knowledge points that are encoded in its intermediate layers. For example, in a multiclass image classification setting containing a class named "Dog", the visual concepts of a "dog tail" or a "dog head" can be regarded as different knowledge points. Classification accuracy is likely high if the encoded knowledge points are sufficiently discriminative with regard to the supported data classes.

An *input unit* is defined as a variable (or a set of variables) in the input sample, $e.g$, each word in an input sentence, a pixel, or an image region in an input image as illustrated in Fig. 1. These input units, corresponding to specific object parts, are memorized during training as feature maps produced by a DNN layer and are subsequently utilized for prediction during inference.

The information of certain units (e.g., background image pixels, in the case of whole-image classification) is largely discarded, while the information of other units is retained. These input units without significant information discarding are considered to be knowledge points since they are supposed to encode information important for inferring predictions. Contrarily, it is widely assumed that input units with significant information-discarding will have little impact on the prediction. As a result, measuring the amount of information discarding allows for precise knowledge points quantification. Overall, knowledge points can be categorized as task-relevant or task-irrelevant.

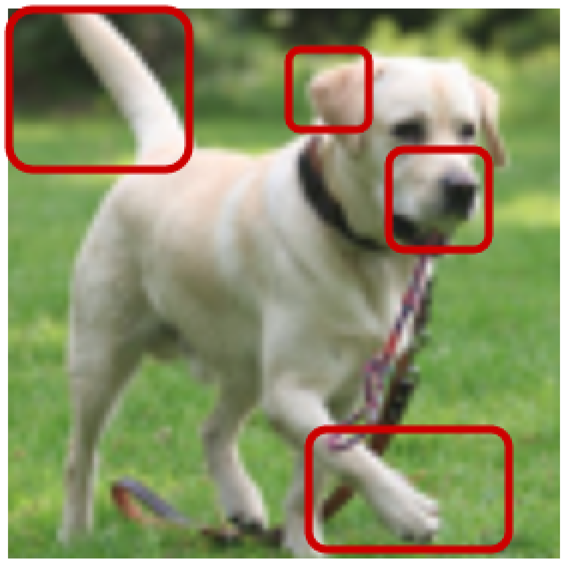

Fig. 1: Possible knowledge points visualized as image regions on the foreground of an STL-10 image.

Definition 3. Forward propagation is regarded as the layer-wise discarding of input information by information-bottleneck theory [50]. In early layers, the majority of the input information is used to compute network features. However, in late layers, only relevant attribute-level (e.g., pixel-level) information is retained as features. Thus, information from input units that is irrelevant to the inference is discarded. As a result, the late layer features will encode information that is highly significant to inference predictions. When a DNN extracts the feature from a particular intermediate layer, the quantity of input information that is discarded can be used to measure the amount of information encoded in that layer. The *Entropy of Input Information* that is contained in the feature of an intermediate layer l, quantifies the amount of information discard during the forward propagation [51].

For a DNN trained on a classification task and an object instance $\mathbf{x}_c \in \mathbb{R}^N$ where N is the number of the input units, let $\mathbf{f}^* = \mathbf{f}(\mathbf{x}_c)$ denote the feature of an intermediate network layer and $\mathbf{y}^*$ the network prediction for $\mathbf{x}_c$. An *input unit* is defined as a variable (or a set of variables) in the input sample ($e.g$ the embedding of a word in the input sentence or a pixel). The DNN is considered to satisfy Lipschitz constraint $||\mathbf{y}' - \mathbf{y}^*|| \leq \kappa||\mathbf{f}(\mathbf{x}'_c) - \mathbf{f}^*||$, where $\mathbf{y}'$ denotes the network prediction for a sample $\mathbf{x}'_c$. The Lipschitz constant κ can be computed as the maximum norm of the gradient within a small range of features, indicating that if the low-dimensional manifold of features $w.r.t.$ the input $\mathbf{x}_c$ is weakly perturbed within a small range $\{\mathbf{f}(\mathbf{x}_c)' \Big| \, ||\mathbf{f}(\mathbf{x}'_c) - \mathbf{f}^*||^2 \leq \tau\}$, where τ is a positive scalar, the DNN output is also perturbed within a small range $||\mathbf{y}' - \mathbf{y}^*|| \leq \kappa \cdot \tau$. In particular, the weakly perturbed features, represent the same object instance. Perturbations $\Delta\mathbf{x}_c$ are added to the input sample $\mathbf{x}_c$ to approximate the feature manifold. Thus, samples $\mathbf{x}'_c = \mathbf{x}_c + \Delta\mathbf{x}_c$ subject to $||\mathbf{f}(\mathbf{x}'_c) - \mathbf{f}^*||^2 \leq \tau$ are generated. $\|\mathbf{f}(\mathbf{x}'_c) - \mathbf{f}^*\|^2 \leq \tau$ denotes a small range of features representing the concept of the object. The conditional entropy $H(\mathcal{X}'_c)$ is defined by the formulation:

$$H(\mathcal{X}'_c) \triangleq - \sum_{\mathbf{x}'_c \in \mathcal{X}'_c} p(\mathbf{x}'_c) log p(\mathbf{x}'_c). \tag{3}$$

It quantifies the uncertainty of the input when the feature represents the same object instance $||\mathbf{y}' - \mathbf{y}^*|| \leq \kappa \cdot \tau$. Thus, $H(\mathcal{X}'_c)$ measures the amount of information that may be discarded without influencing the inference prediction. Generally, the conditional entropy $H(\mathcal{X}'_c)$, where $\mathcal{X}'_c$ denotes a set of inputs that correspond to the concept of a specific object instance, is used to formulate information discarding, given the intermediate-layer feature $\mathbf{f}^* = \mathbf{f}(\mathbf{x}_c)$. $H(\mathcal{X}'{}_c)$ is defined on a set $\mathcal{X}'_c$ consisting of elements $\mathbf{x}'_c$ such that $\|\mathbf{f}(\mathbf{x}'_c) - \mathbf{f}^*\|^2 \leq \tau$.

Assuming that $\Delta\mathbf{x}_c$ is an $i.i.d.$ Gaussian noise, $\mathbf{x}'_c = \mathbf{x}_c + \Delta\mathbf{x}_c \sim \mathcal{N}(\mathbf{x}_c, \Sigma(\boldsymbol{\sigma}) = diag(\sigma_1^2, ..., \sigma_n^2))$ where σ_i denotes the variance of the perturbation for the i-th input unit. Assuming that $\mathbf{x}'_c$ follows a Gaussian distribution and that the covariance matrix is a diagonal one $\mathbf{\Sigma} = diag(\sigma_1^2, ..., \sigma_n^2)$, the entropy of the input $H(\mathcal{X}'_c)$ can be decomposed into *attribute-level (e.g., pixel-level) entropies*:

$$H(\mathcal{X}'_c) = \sum_{i=1}^{N} H_i, \tag{4}$$

where N is the number of input units and H_i is attribute-wise information discarding. H_i is defined as:

$$H_i = log\sigma_i + a, \tag{5}$$

where $a = \frac{1}{2}log(2\pi e)$ is a constant. Large values of H_i indicate that the information of the i-th unit is significantly discarded during forward propagation.

Definition 4. *Coherency* implies that a method needs to enable fair layer-wise comparisons and fair comparisons between different neural networks, indicating that a knowledge quantification metric should be invariant to network configurations.

Definition 5. *Generality* refers to the fact that a method should have strong connections to existing mathematical theories [52] and should be defined without regard for model architectures or tasks [53].

III. Knowledge Quantification Metrics

We still lack the mathematical tools to accurately evaluate the knowledge encoded in the DNN layers and their learning capacity. Knowledge quantification would allow for a more precise interpretation and evaluation of DNN training performance. Of course, assuming a classification setting, DNN knowledge can be naively quantified post-training based on the DNN test set predictions:

$$k = \sum_{i=1}^{m} \frac{P_i}{m}, \tag{6}$$

where P_i is the DNN success rate for a class C_i. The success rates for the classes are measured based on a test set as $\{(P_i, C_i), i = 1, ..., m\}$. The normalized success rates are defined as follows:

$$P_{iN} = \frac{P_i}{km}. \tag{7}$$

In order to evaluate and quantify the DNN knowledge, the Entropy E of the normalized success rate is defined as:

$$E = -\sum_{i=1}^{N} P_{iN} log P_{iN}. \tag{8}$$

Despite the fact that this approach is a clear and reasonable solution to the problem of knowledge quantification in classification tasks, it necessitates test inference. It is entirely dependent on the test set chosen. Other methods, which are not test-specific and are more generic, need to be developed.

Existing approaches moving in this direction allow us to examine the intermediate DNN layers and the kind or the amount of information they encode. These approaches are described in the following subsections.

A. Information-theoretic metrics

A group of DNN knowledge quantification methods measures a trained DNN layer's information. The underlying assumption is that the amount of this information is proportional to the knowledge this layer encodes, thus it may act as a proxy for the latter one. The information flow, between the layers and specifically between each pair of layers, inside a fully-connected feed-forward DNN can be seen in Fig. 2.

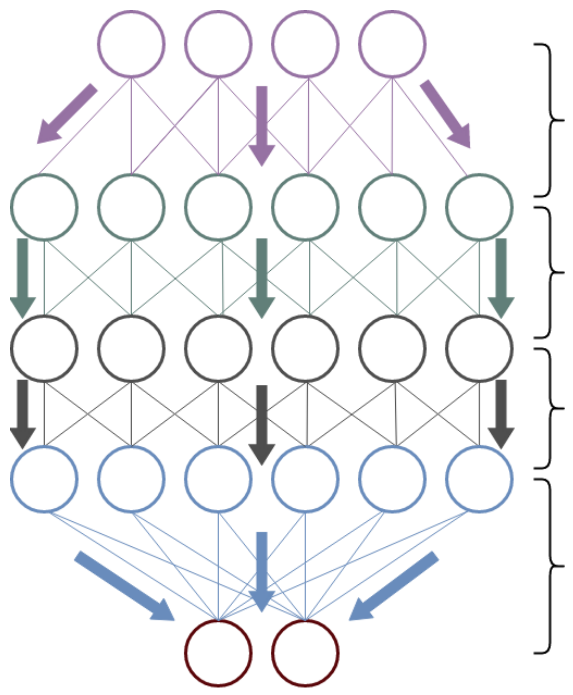

Fig. 2: The information flow through the DNN layers. Information-theoretic metrics utilize layer-wise information measurements in order to quantify the encoded DNN knowledge.

1) : **Information-theoretic metrics for general applications**.
Entropy has been metric widely utilized in recently developed approaches for information quantification. Earlier DNN explainability methods, such as perturbation-based methods [31] or gradient-based approaches [28], depend mainly on a specific network design. Unlike such methods, conditional input entropy ensures fair comparisons between different DNN layers or network architectures. Recent algorithms leverage entropy to quantify knowledge encoded in an intermediate DNN layer [51], [53]. A relevant method attempts to determine how much input information was discarded when the DNN extracts the layer features, when its activations were computed during inference [51]. The intuition is that non-essential input information is discarded and only task-relevant information (i.e., the layer's knowledge) is retained in the layer's output features. DNN knowledge is quantified by defining two entropy-based metrics, namely the *Strict Information Discarding* and the *Reconstruction Uncertainty*. The entropy of the information imported in a layer is measured in order to quantify the degree of information discarding (and, thus, knowledge), during forward propagation. This way, a layer's knowledge representations are defined as the amount of input information that has been passed to its output.

Strict Information Discarding (SID) measures the amount of input information that is discarded during the computation of a feature. DNNs usually selectively discard certain input attributes (e.g., pixels, in the case of the first layer of a CNN that analyzes raw images), and thus, an amount of input information, to compute intermediate-layer features. The discarded attributes are either task-irrelevant or inessential, due to information redundancy. On the other hand, *Reconstruction Uncertainty* (RU) measures the amount of input information discarding that can be recovered by other attributes. The information encoded in the input attributes discarded during intermediate-layer feature extraction can be recovered due to information redundancy.

Information discarding in a DNN layer is modeled as the entropy of input information. Let $\mathbf{x}' \in \mathbb{R}^N$ be the input data point and $\mathbf{f}^* = \mathbf{f}(\mathbf{x})$ the intermediate-layer feature. $\mathbf{x}' \in \mathcal{X}'$ represents either the input $\mathbf{x}$ or the reconstructed input $\widehat{\mathbf{x}} = \mathbf{g}(\mathbf{f})$, where $\mathbf{g}$ is the function reconstructing $\mathbf{x}$ from its features $\mathbf{f}^* = \mathbf{f}(\mathbf{x})$. SID can be evaluated using Eq. (3) for the raw input $\mathbf{x}$. The reconstructed input $\widehat{\mathbf{x}}$ provides the value of RU as detailed subsequently.

As mentioned in Section II, if $\mathbf{x}'$ follows a Gaussian distribution and has a diagonal covariance matrix, the entropy of the Gaussian distribution, and thus, SID, is given by Eq. (4) and Eq. (5):

$$H(\mathcal{X}') = \sum_{i=1}^{N} H_i(\sigma_i), \tag{9}$$

where

$$H_i(\sigma_i) = log\sigma_i + a. \tag{10}$$

Strict Information Discarding (SID) can be used as another metric to evaluate the DNN feature extraction efficiency for a classification task. Generally, a DNN should discard background information and retain foreground information. *Concentration of information discarding* is proposed in [51] for the specific case of image recognition. For an input image $\mathbf{x}$ containing both the target object and some background, let Λ denote the bounding box of the target and $\mathbf{x}_i$ represent target attributes within Λ. *Concentration of information discarding* or simply *Concentration* $\mathbb{C}$ is defined as follows:

$$\mathbb{C} = \mathbb{E}_{\mathbf{x}_i \notin \Lambda}\{H_i(\sigma_i)\} - \mathbb{E}_{\mathbf{x}_i \in \Lambda}\{H_i(\sigma_i)\}. \tag{11}$$

It measures the relative background information discarding w.r.t. foreground information discarding. Concentration indicates the efficiency of neural feature extraction.

Similarly to SID, RU is defined as follows:

$$H(\widehat{\mathcal{X}'}) \triangleq - \sum_{\widehat{\mathbf{x}'}=\mathbf{g}(f)} p(\widehat{\mathbf{x}'}) logp(\widehat{\mathbf{x}'}), \tag{12}$$

where $\widehat{\mathcal{X}}$ is the set of features reconstructed using intermediate-layer features. Moreover, RU can be decomposed into each attribute as:

$$H(\widehat{\mathcal{X}_c}) = \sum_{i=1}^{N} \widehat{H}_i(\sigma), \tag{13}$$

$$\widehat{H}_i(\sigma) = log\widehat{\sigma}_i + a \tag{14}$$

where σ denotes elements on the diagonal of $\mathbf{\Sigma}$. $\widehat{H}_i(\sigma)$ is the attribute-level (pixel-level for images) RU for the $i-th$ input attribute.

An estimator for conditional *Geometric Mutual Information* (GMI) [54] is used in [26] to evaluate the amount of mutual information exchanged between the DNN neurons. This can indicate which neurons contribute the majority of the information that is passed to the next layer. Thus, it reveals the amount of knowledge encoded in the DNN neurons. *Geometric Mutual Information* (GMI) is used as a non-parametric, geometric, dependency measure between pairs of multivariate random variables in [26]. The proposed dependency estimator is constructed using a minimal spanning tree and is a function of the Friedman–Rafsky multivariate test statistic [55]. For the marginal and joint distributions $f_{\mathbf{X}}$, $f_{\mathbf{Y}}$, and $f_{\mathbf{XY}}$ of random vectors $\mathbf{X} \in \mathbb{R}^{dx}$, $\mathbf{Y} \in \mathbb{R}^{dy}$ respectively where dx and dy are positive integers, and for the parameters $p \in (0, 1)$ and $q = 1 - p$, the *Geometric Mutual Information* (GMI) between $\mathbf{X}$ and $\mathbf{Y}$, in [54], is defined by:

$$I_p(\mathbf{X};\mathbf{Y}) = \frac{1}{4pq}\Big[\iint \frac{pf_{\mathbf{XY}}(x,y) - qf_{\mathbf{X}}(y)f_{\mathbf{Y}}(y))^2}{pf_{\mathbf{XY}(x,y)} + qf_{\mathbf{X}}(y)f_{\mathbf{Y}}(y))}dxdy - (p-q)^2\Big]. \tag{15}$$

GMI satisfies similar properties to other mutual information definitions, such as Shannon and Rényi mutual information. As in [56], another form of GMI is given by:

$$I_p(\mathbf{X};\mathbf{Y}) = 1 - A_p(\mathbf{X};\mathbf{Y}). \tag{16}$$

For the specific case $p = q = 1/2$ in the, GMI is given by:

$$I_p(\mathbf{X};\mathbf{Y}) = 1 - 2\iint \frac{f_{\mathbf{XY}}(x,y)f_{\mathbf{X}}(x)f_{\mathbf{Y}}(y)}{f_{\mathbf{XY}}(x,y) + f_{\mathbf{X}}(x)f_{\mathbf{Y}}(y)}dxdy. \tag{17}$$

For random variables $\mathbf{X} \in \mathbb{R}^{dx}$, $\mathbf{Y} \in \mathbb{R}^{dy}$ and $\mathbf{Z} \in \mathbb{R}^{dz}$ with conditional density $f_{\mathbf{XY|Z}}$, the form of this measure is defined as [54]:

$$I_p(\mathbf{X};\mathbf{Y}|\mathbf{Z}) = E_{\mathbf{Z}}[\widetilde{I}_p(f_{\mathbf{XY|Z}})] \tag{18}$$

and

$$\widetilde{I}_p(f_{\mathbf{XY|Z}}) = 1 - 2\int\int \frac{f_{\mathbf{XY|Z}}(x,y|z)f_{\mathbf{X|Z}}(x|z)f_{\mathbf{Y|Z}}(y|z)}{f_{\mathbf{XY|Z}}(x,y|z) + f_{\mathbf{X|Z}}(x|z)f_{\mathbf{Y|Z}}(y|z)}dxdy. \tag{19}$$

To estimate $I_p(\mathbf{X};\mathbf{Y}|\mathbf{Z})$ for m samples drawn from $f(x,y,z)$,the data samples are split into two subsets S_1 and S_2 in [26]. The Nearest Neighbour Bootstrap algorithm [57] can be used to generate independent samples from $\mathcal{S}_2$ to create another set named $\bar{\mathcal{S}}_2$. Then, $\mathcal{S}_1$ and $\bar{\mathcal{S}}_2$ are connected into the concatenated data set: $\mathcal{S} \triangleq \mathcal{S}_1 \cup \bar{\mathcal{S}}_2$. A Minimum Spanning Tree (MST) is constructed on $\mathcal{S}$. Friedman-Rafsky statistic [55] R_m is defined as the number of MST edges linking dichotomous points. It is the number of edges connecting points of $\mathcal{S}_1$ and $\bar{\mathcal{S}}_2$. Thus, the estimate for $I(X;Y|Z)$ is given by:

$$I(X;Y|Z) = 1 - \left(\frac{\mathrm{R}_m}{m}\right). \tag{20}$$

During the inference stage of a trained network, the conditional GMI estimator is calculated on each subsampled set of activations of each node pair at consecutive layers to measure their dependency. It evaluates which neurons pass the majority of the information through layers and thus, encode a larger amount of knowledge inside the neural network.

For a DNN with L layers, GMI is computed as the dependency ρ between the neurons in every consecutive pair of layers. This dependency reflects the relationship between neurons in the context of all the contributions from the preceding layer. Among the chosen pair of consecutive layers, the activation pattern for a given node in the layer l, when layer l is closer to the input and layer $l+1$ is closer to the output, is computed as:

$$F^{(l)}(\mathbf{x}) = f(\mathbf{w}\mathbf{x} + b), \tag{21}$$

where $\mathbf{x} \in \mathbb{R}^{m\times d}$ is the input vector layer l, m is the total number of samples, d the feature dimension. $f()$ is the activation function and $\mathbf{w} \in \mathbb{R}^{N^{(l)}}$ and b are the weight vector and bias. GMI computed between the activations from the selected neuron i from layer $l+1$ and the activations from the neuron j from layer l, $(F_i^{(l+1)}, F_j^{(l)})$, given the existence of every other possible neuron in layer l, is defined as in Eq. (18):

$$\rho(F_i^{(l+1)}, F_j^{(l)}) \triangleq I(F_i^{(l+1)}, F_j^{(l)}|\overline{F_j^{(l)}}) \tag{22}$$

where $\overline{F_j^{(l)}}$ is the vector of all the neuron outputs of the layer l except $F_j^{(l)}$. Larger values of GMI indicate that the neuron passes a larger amount of information from layer l to layer $l+1$, thus, it is assumed to encode a larger amount of knowledge.

2) : **Information-theoretic metrics for Natural Language Processing (NLP)**.
For the specific application of Natural Language Processing (NLP), a unified information-based measure is defined in [53], in order to quantify information and the way it is passed from one DNN layer to the next one. It can provide quantitative explanations about the intermediate deep NLP layers, by quantifying the amount of information of an input word that is encoded in an intermediate DNN layer. This method satisfies generality and provides reliable results. In order to define such a measure, a deep NLP neural network with L intermediate layers is modeled by a function $\mathbf{f}(\mathbf{x})$ where $\mathbf{x}$ is an input sentence of a dataset

$\mathcal{X}$ consisting input sentences. Every sentence is a concatenation of the vectorized embedding of each word $\mathbf{x} = [\mathbf{x}_1^T, \mathbf{x}_2^T, ..., \mathbf{x}_n^T]^T \in \mathcal{X}$ consisting of elements $\mathbf{x}_i$ that denote the i-th word.

The knowledge, or more specifically, the amount of information of an input word encoded in a deep NLP intermediate layer is quantified by input information discarding when the DNN extracted the feature map of this layer. The Deep NLP $\mathbf{f}$ model can be constructed by layers of RNNs, self-attention layers (as in Transformers [58]), or any other type of layers. The output of each intermediate layer is a series of hidden states denoted as $\mathbf{s} = \Phi(\mathbf{x}_i)$ where $\Phi_i, i = 1, ..., L$ is the function of each specific layer. In order to quantify the information of the $\mathbf{x}_i$ that is used to compute the hidden state of the DNN layers, the information at different levels is first defined: the corpus level, the sentence level, and the word level. At the corpus level, knowledge explanations regard the entire sentence space. The random variable $\mathbf{S}$ denotes a hidden state. The information of $\mathcal{X}$ encoded by a hidden state $\mathbf{S}$ is measured by the *mutual information*:

$$MI(\mathcal{X}; \mathbf{S}) = H(\mathcal{X}) - H(\mathcal{X}|\mathbf{S}), \tag{23}$$

where $H(\cdot)$ represents the entropy and $H(\mathcal{X}|\mathbf{S})$ the amount of the information discarding of the hidden state. The decomposition of $H(\mathcal{X}|\mathbf{S})$ at sentence level is given by:

$$H(\mathcal{X}|\mathbf{S}) = \int_{\mathbf{s}\in\mathbf{S}} p(\mathbf{s})H(\mathcal{X}|\mathbf{s})d\mathbf{s}. \tag{24}$$

At sentence level, the information of a sentence $\mathbf{x}$ discarded by a hidden state of an intermediate layer $\mathbf{s} = \Phi(\mathbf{x})$ is quantified as the entropy of the input:

$$H(\mathcal{X}|\mathbf{s}) = -\int_{\mathbf{x}'\in\mathcal{X}} p(\mathbf{x}'|\mathbf{s}) logp(\mathbf{x}'|\mathbf{s})d\mathbf{x}', \tag{25}$$

where the entropy $H(\mathcal{X}|\mathbf{s})$ reflects how much information from sentence $\mathbf{x}$ is discarded by $\mathbf{s}$ during forward propagation. $H(\mathcal{X}|\mathbf{s})$ reaches the minimum value, if and only $p(\mathbf{x}'|\Phi(\mathbf{x})) << p(\mathbf{x}|\Phi)$ where $\mathbf{x}' \neq \mathbf{x}$. Thus, $\Phi(\mathbf{x}') \neq \Phi(\mathbf{x})$ for each elements $\mathbf{x}' \neq \mathbf{x}$, which indicates that all information from $\mathbf{x}$ is utilized. If only a small percentage of the information from $\mathbf{x}$ is used, $p(\mathbf{x}'|\mathbf{s})$ will be more evenly distributed. Thus, the entropy $H(\mathcal{X}|\mathbf{s})$ will have a larger value. At the word level, independence between the input words is assumed. Thus, for a random variable $\mathbf{X}_i$ of the i-th word of the input data, $H(\mathcal{X}|\mathbf{s}) = \sum_i(\mathbf{X}_i|\mathbf{s})$ and

$$H(\mathbf{X}_i|\mathbf{s}) = -\int_{\mathbf{x}_i'\in\mathbf{X}_i} p(\mathbf{x}_i'|\mathbf{s}) logp(\mathbf{x}_i'|\mathbf{s})d\mathbf{x}_i'. \tag{26}$$

3) : **Information-theoretic metrics for computer vision**.

The CNN convolutional layers employ 1D, 2D or 3D convolutions for analyzing 1D signals, images, and videos, respectively. These convolutions are essentially 1D/2D/3D FIR filters. In order to interpret DNNs and their knowledge, recent methods aim to quantify the information contribution of such convolution kernels to the visual pattern (concept) encoding and prediction. Thus, the encoded knowledge in a network or even, of an individual neuron could be evaluated. The visual patterns (concepts) that contribute more, encode more information and thus, knowledge. The so-called *Net2Vec* framework is proposed in [59] in order to examine how CNN filters encode visual patterns. They are aligned with filter activations by learned concept embeddings that are employed to weight filter activations, in order to execute concept interference. A class weight is learned for every concept in the dataset, for each class to be recognized. The weights that arise are interpreted as class embeddings and provide information about how visual patterns and classes are encoded. Images may contain any number of different visual patterns, indexed by $\mathbf{v}$. For an image dataset $\mathcal{X}$, $\mathbf{x} \in \mathcal{X}$ denotes the probe images that contain the visual pattern $\mathbf{v}$. To determine which filter k in layer l best segments pattern $\mathbf{v}$, the IoU score is computed as in [59] as:

$$IoU_{set}(\mathbf{v}; M_k, s) = \frac{\Sigma_{\mathbf{x}\in\mathcal{X}_{s,\mathbf{v}}}|M_k(\mathbf{x}) \cap L_{\mathbf{v}}(\mathbf{x})|}{\Sigma_{\mathbf{x}\in\mathcal{X}_{s,\mathbf{v}}}|M_k(\mathbf{x}) \cup L_c(\mathbf{x})|}. \tag{27}$$

It computes the Intersection over Union (Jakkard index) metric between the ground-truth segmentation masks $L_{\mathbf{v}}$ and the binary segmentation masks M_k produced by the filter. $|\cdot|$ denotes the cardinality of a set and $M_k(\mathbf{x}) \cap L_{\mathbf{v}}(\mathbf{x})$ are the intersections for every $(k, \mathbf{v})$ pair. The sets are merged for all images in the subset $\mathcal{X}_{s,\mathbf{v}}$ of the data, where $s \in \{train, val\}$. The best filter is then selected on the training set as:

$$k^*(\mathbf{v}) = argmax_k IoUset(\mathbf{v}; M_k, train). \tag{28}$$

The validation IoU score, $IoUset(\mathbf{v}; M_k^*, val)$ is reported. Single-filter performance is frequently highly linearly correlated with learned weights vector $\mathbf{w}$ [59]. Therefore, individual filter performance indicates its weight in a linear filter combination. A filter set IoU score is correlated with its associated weight value passed through a ReLU defined as: $max(w_k, 0)$. The contribution of a single filter k to the pattern $\mathbf{v}$ encoding can be quantified by:

$$r = \frac{IoUset(\mathbf{v}; M_k^*, val)}{IoUset(\mathbf{v}; M(\cdot; \mathbf{w}), val)}, \tag{29}$$

being the ratio of the validation score IoU ($IoUset(\mathbf{v}; M_k^*, val)$) to the validation score IoU ($IoUset(\mathbf{v}; M(\cdot; \mathbf{w}), val)$). The numerator is the intersection over union difference between the ground-truth segmentation masks $L_{\mathbf{v}}$ and the binary segmentation mask M_{k^*} produced by the single filter k^*. The denominator is the intersection over union difference between the ground-truth segmentation masks $L_{\mathbf{v}}$ and the segmentation masks $M(\cdot; \mathbf{x})$ produced by the filters K, where $\mathbf{w} \in \mathbb{R}^K$ is learned weights vector of $\mathbf{v}$ and K is the total number of filters in a layer.

B. *Knowledge points metrics*

The knowledge of an intermediate DNN layer is defined as the set of knowledge points that are encoded by the neural activations/features of this layer as can be seen in Fig. 3. Consequently, the number of learned knowledge points is proportional to the amount of encoded DNN knowledge. In recent studies, various methods for quantifying the knowledge points encoded in DNN intermediate layers have been developed.

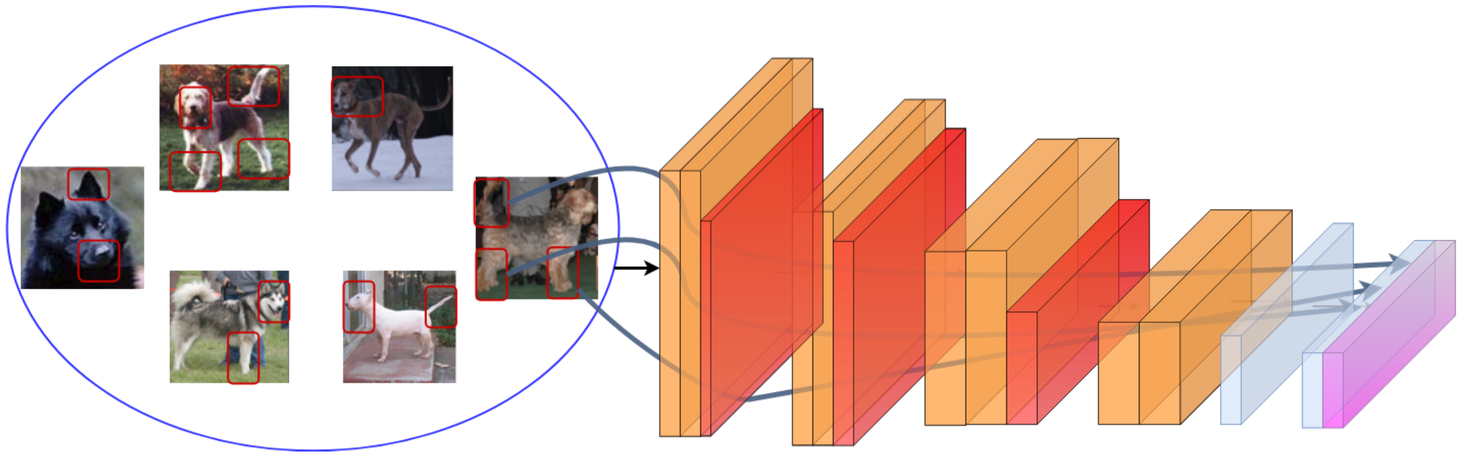

Fig. 3: The emergence of possible knowledge points encoded by the DNN visualized as image regions on the foreground of STL-10 images.

A relevant knowledge quantification approach based on entropy is proposed in [52]. The amount of knowledge of each DNN layer is measured as knowledge points, i.e., input units, whose information is regarded as important for decision-making since it is discarded much less than the information of other input units. For this specific research, the information in each input unit discarded by the DNN is used to define and quantify knowledge points not explicitly defined or labeled by human annotation, namely *Dark Matters* [52]. The knowledge points encoded in the layers of a DNN for classification are quantified, and their quality is evaluated, based on information theory. DNN forward propagation is considered as a gradual, layer-wise process of discarding the input unit information.

The amount of information discarding of each input unit is formulated as the entropy $H(\mathcal{X}_c)$, where $\mathcal{X}_c$ denotes a set of inputs $\mathbf{x}_c$ corresponding to the concept of a specific object instance. This entropy quantifies the amount of input information that can be discarded without affecting the inference of the object. As mentioned in Subsection II, assuming that $\mathbf{x}_c$ follows a Gaussian distribution, $H(\mathcal{X}_c)$ can be decomposed into attribute-level entropies as follows:

$$H(\mathcal{X}_c) = \sum_{i=1}^{N} H_i = \sum_{i=1}^{N} \big(log\sigma_i + \frac{1}{2}log(2\pi e)\big), \tag{30}$$

where N is the number of input units. Attribute-level entropies H_i measure the attribute-wise information discarding in the intermediate DNN layer. High attribute-wise entropies H_i indicate that the information of the i-th unit is significantly discarded by the DNN during forward propagation.

The attribute-wise information of each input unit discarded by the DNN is used to define and quantify knowledge points. For trained DNN and an input sample $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}$ denotes the set of input samples, and the attribute-wise information discarding H_i $w.r.t.$ the feature of a certain network layer $\mathbf{f}^* = \mathbf{f}(\mathbf{x})$, the amount of knowledge encoded is measured by how many knowledge points it encodes. The number of knowledge points encoded on the background $N_b(\mathbf{x})$ and the number of knowledge points encoded on the foreground $N_f(\mathbf{x})$ are defined as follows:

$$N_b(\mathbf{x}) = \sum_{i \in \Lambda_b(\mathbf{x})} I(\overline{H} - H_i > b) \tag{31}$$

$$N_f(\mathbf{x}) = \sum_{i \in \Lambda_f(\mathbf{x})} I(\overline{H} - H_i > b), \tag{32}$$

where $I(\cdot)$ is an indicator function:

$$I(x) = \begin{cases} 1, & \text{if x is true} \\ 0, & \text{else} \end{cases}$$

and $\Lambda_b(\mathbf{x})$ and $\Lambda_f(\mathbf{x})$ denote the background and the foreground input unit sets respectively. The knowledge points are measured/evaluated for a convolutional or a fully-connected (FC) DNN layer. The average entropy value of the background units:

$$\overline{H} = \frac{1}{|\Lambda_b|} \sum_{i=1}^{|\Lambda_b|} H_i \tag{33}$$

is the baseline entropy to determine knowledge points. Thus, input units with entropy H_i significantly lower than $\overline{H}$, ($H_i < \overline{H} - b$), represent valid knowledge points. b is the threshold to determine the knowledge points. The coherency of the method allows the metric to ensure fair comparisons between DNN layers and between different DNNs.

Knowledge quantification using knowledge points also involves a number of additional metrics [52]. The metric λ, which assesses the quality of knowledge points by determining whether the majority of them or not is localized in the foreground of the input data samples, is defined as follows:

$$\lambda = E_{\mathbf{x} \in \mathcal{X}}[N_f(\mathbf{x})/(N_f(\mathbf{x}) + (N_b(\mathbf{x}))] \tag{34}$$

where $\mathcal{X}$ is a set of input samples and $\mathbf{x} \in \mathcal{X}$ an input sample. The larger the value of λ is, the more reliable the DNN feature representation. A well-trained model should be able to encode a large number of knowledge points on the foreground and a small number on the background.

Two additional metrics concern DNN knowledge acquisition in the classification tasks. Their goal is to discover how fast a DNN learns different object classes and whether multiple knowledge points are learned within similar epochs, i.e., whether different knowledge points are learned simultaneously. Firstly, a *weight distance* is defined:

$$d = \sum_{k=1}^{\widehat{m}} \frac{\|\mathbf{w}_k - \mathbf{w}_{k-1}\|}{\mathbf{w}_0}, \tag{35}$$

$\mathbf{w}_0$ and $\mathbf{w}_k$ are the initial parameter vector and the one learned at epoch k and $\widehat{m} = \arg max_k N_{f_k}(\mathcal{X})$ is the training epoch number at which the model learns the most task-relevant, or foreground, knowledge points. Weight distance d measures the progress of learning at the $\widehat{m} - th$ epoch, by quantifying the update rate of the weight vector $\mathbf{w}_k$. Its first and second-order statistics ($\bar{d}_k$ and $\sigma^2_{d_k}$) over different input data samples can be measured, in order to quantify the degree to which a DNN learns multiple knowledge points simultaneously:

$$\bar{d}_k = \frac{1}{N_I} \sum_{i=1}^{N_I} \sum_{k=1}^{\widehat{m}} \frac{\|\mathbf{w}_k - \mathbf{w}_{k-1}\|}{\|\mathbf{w}_0\|} \tag{36}$$

$$\sigma^2_{d_k} = \frac{1}{N_I} \sum_{i=1}^{N_I} \left(\sum_{k=1}^{\widehat{m}} \frac{\|\mathbf{w}_k - \mathbf{w}_{k-1}\|}{\|\mathbf{w}_0\|} - \bar{d}_k \right)^2, \tag{37}$$

where N_I is the total number of training samples. According to information-bottleneck theory [50], a DNN tends to learn many knowledge points during early training epochs and then starts discarding the task-irrelevant knowledge points later. Considering that epoch $\widehat{m}$ encodes the richest foreground knowledge points, $\bar{d}_k$ represents the average weight distance, at which the DNN obtains the most foreground visual concepts and measures whether a DNN learns knowledge points quickly. $\sigma^2_{d_k}$ describes the variance of the weight distance across different input samples and evaluates whether a DNN learns knowledge points of various input samples at similar speeds. Small $\bar{d}_k$ and σ_{d_k} values suggest that the DNN can learn many different knowledge points fast and simultaneously.

The two-stage training process, where task-irrelevant features are discarded at the later epochs, leads to *detours*, namely inconsistent and unstable optimization directions in the model parameters space [52]. A new metric is defined in order to discover if the network obtains knowledge points (hence knowledge), without significant detours. The stability of optimization directions of a DNN is quantified by ρ:

$$\rho = \frac{\|S_M(\mathbf{x})\|}{\| \cup_{j=1}^{M} S_j(\mathbf{x})\|}, \tag{38}$$

where $S_j(\mathbf{x}) = \{i \in \mathbf{x} | H_i < \overline{H} - b\}$, denote a set of foreground knowledge points encoded by the DNN learned after the j-th epoch, $j = 1, 2, ..., M$. Every knowledge point $\mathbf{v} \in S_j(\mathbf{x})$ refers to a specific foreground input sample unit i, which satisfies $H_i < \overline{H} - b$. The numerator represents the number of foreground knowledge points chosen for classification, whereas the denominator denotes the total number of knowledge points that have temporarily been learned during training. Thus, ρ represents the ratio of the encoded knowledge points to all temporarily attempted knowledge points in intermediate epochs. The ratio of the discarded knowledge points is $1 - \rho$. Thus, higher values of ρ indicate a stable network that is optimized with fewer detours during learning.

The discriminative power of intermediate layer feature maps can be visualized, in order to measure the amount of *knowledge points* learned by the model [27]. The intermediate-layer features are divided into feature components representing a specific image region. Knowledge points are defined as discriminant feature components. Quantifying them serves as a proxy for measuring the encoded CNN knowledge. The visualization displays the emergence of intermediate DNN layer visual patterns in a spatiotemporal manner. Initially, the method illustrates how the DNN gradually learns the regional visual patterns at each intermediate DNN layer during training. Any increases in the discriminative power of each individual visual pattern that emerge along the learning process are also visualized.

Based on these visual illustrations, the quality and quantity of intermediate knowledge points encoded in the CNN model layers are measured. All knowledge points encoded in the model as regional patterns with strong discriminative power are counted, while their significance for classification is also measured as follows. For a pre-trained CNN and an input image $\mathbf{x} \in \mathbb{R}^N$, the output feature of a specific DNN layer is denoted by $\mathbf{f} \in \mathbb{R}^d$. Feature $\mathbf{f}$ is considered to follow a radial distribution of massive pseudo-categories (many more than the actual number of semantic classes in the dataset). Each pseudocategory c has a mean direction $\boldsymbol{\mu}_c \quad (c = 1, ..., C)$ in the feature space. $\cos(\mathbf{f}, \boldsymbol{\mu}_c)$ shows the similarity between the feature $\mathbf{f}$ and category c. Specifically, the von Mises-Fisher (vMF) distribution [60] models the radial distribution. The likelihood of each feature $\mathbf{f}$ belonging to category $c \in \{1, ..., C\}$ is assumed to follow a vMF distribution. The discriminative power of the feature $\mathbf{f} \in \mathbb{R}^d$ of each sample $\mathbf{x} \in \mathcal{X}$ is visualized to illustrate the classification confidence of each feature towards distinct classes. The objective is to learn a linear transformation that will project $\mathbf{f}$ to a low dimensionality space. Thus, the projected feature is $\mathbf{g} = \mathbf{M}\mathbf{f} \in \mathbb{R}^{d'}$ where $d' << d$. It preserves the similarity between each feature $\mathbf{f}$ and different class mean vectors. The linear transformation $\mathbf{M}$ uses the projected feature $\mathbf{g}$ for classification and forces the classification based on $\mathbf{g}$ to mimic the classification based on the original feature $\mathbf{f}$. Let $y \in \mathcal{Y} = \{1, ..., C\}$ denote a class. To compute the classification probability $p_M(y|x)$, it is assumed that the distribution $p(\mathbf{g})$ is a mixture model, with each mixture component $p(\mathbf{g}|y)$ following a revised von Mises-Fisher (vMF) distribution. Then $\mathbf{g} = [l_\mathbf{g}, \mathbf{O}_\mathbf{g}]$ where $l_\mathbf{g} = ||\mathbf{g}||_2$ and $\mathbf{O}_\mathbf{g} = \mathbf{g}/l_\mathbf{g}$ denote the magnitude and orientation of $\mathbf{g}$. $p(\mathbf{g})$ is defined as:

$$p(\mathbf{g}) = \sum_y \pi_y . p(l_\mathbf{g}|y) . p_{vMF}(\mathbf{O}_\mathbf{g}|\mu_y, \kappa(l_\mathbf{g})), \tag{39}$$

where π_y denotes the prior propability of the y-th category. The variance parameter $\kappa(l_\mathbf{g})$ is determined based on statistics of all features of the same strength $l_\mathbf{g}$.

The classification probability $p_M(y|x)$ based on the projected feature $\mathbf{g}$ is measured by the posterior probability $p(y|\mathbf{g})$ in the mixture model as:

$$p_M(y|x) = \frac{p(y) \cdot (\mathbf{g}|y)}{p(\mathbf{g})} = \frac{\pi_y \cdot p_{vMF}(O_\mathbf{g}|\mu_y, \kappa(l_\mathbf{g}))}{\Sigma_{(y')}\pi_{(y')} \cdot p_{vMF}(O_\mathbf{g}|\mu_{(y')}, \kappa(l_\mathbf{g}))}. \tag{40}$$

It is assumed that the prior of the magnitude of $\mathbf{g}$ is independent from the pattern class: $p(l_\mathbf{g}|y) = p(l_\mathbf{g})$.

Regional features are projected into a low-dimensional space via a linear transformation $\mathbf{\Lambda}$ as: $\mathbf{h}^{(r)} = \mathbf{\Lambda}\mathbf{f}^{(r)} \in \mathbb{R}^{d'} \quad (d' << K)$. The importance and reliability of the original feature $\mathbf{f}^{(r)}$ are reflected by the projected regional features $\mathbf{h}^{(r)}$. As mentioned above, the orientation of $\mathbf{h}^{(r)}$ represents the reliability of classification vs different classes, while the magnitude $||\mathbf{h}^{(r)}||_2$ represents the feature importance.

Knowledge points are defined as regional features $\mathbf{h}^{(r)}$ that satisfy the condition:

$$max_c p(y = c|\mathbf{h}^{(r)}) > \tau. \tag{41}$$

Thus, they are discriminant enough for any pattern classification. $p(y = c|\mathbf{h}^{(r)})$ is the classification probability similar to Eq.(40). It is defined as follows:

$$p(y = c|\mathbf{h}^{(r)}) = \frac{\pi_c \cdot exp[\kappa(||\mathbf{h}^{(r)}||2) \cdot \cos(\mathbf{h}^{(r)}, \mu_c)]}{\Sigma_c' \pi_c' \cdot exp[\kappa(||\mathbf{h}^{(r)}||2) \cdot \cos(\mathbf{h}^{(r)}, \mu_{c'})]}, \tag{42}$$

where π_c denotes the prior probability of the c-th class. After quantifying the total number of knowledge points, the reliable ones are disentangled as those that satisfy:

$$c^{truth} = argmax_c p(y = c|\mathbf{h}^{(r)}), \tag{43}$$

i.e., those that push classification decisions toward the correct inference. The *ratio of reliable knowledge points* is defined as the ratio of reliable knowledge points number to the total knowledge points number. It can be used as a metric to evaluate the quality of knowledge points.

By juxtaposing [52] and [27], one can see that the first method extracts knowledge points encoded in intermediate DNN layers via attribute-wise information discarding [52]. In contrast, the latter method quantifies knowledge points and their reliability for classification via their discriminative power.

IV. Discussion

The aim of this survey is to identify DNN knowledge quantification methods. The naive approach to achieve this is to test the DNN and utilize its test predictions to directly measure the DNN model success rate on a particular test data set. This intuitive approach obviously does not provide any intuition on DNN structure and its relation to knowledge.

We have identified two types of more advanced DNN knowledge quantification metrics: *information-theoretic* and *knowledge points metrics*. These two types have been detailed in Subsections III-A and III-B, respectively. Fig. 4 depicts the evolution of the knowledge quantification methods studied in this survey. The relevant metrics are summarized in Table I.

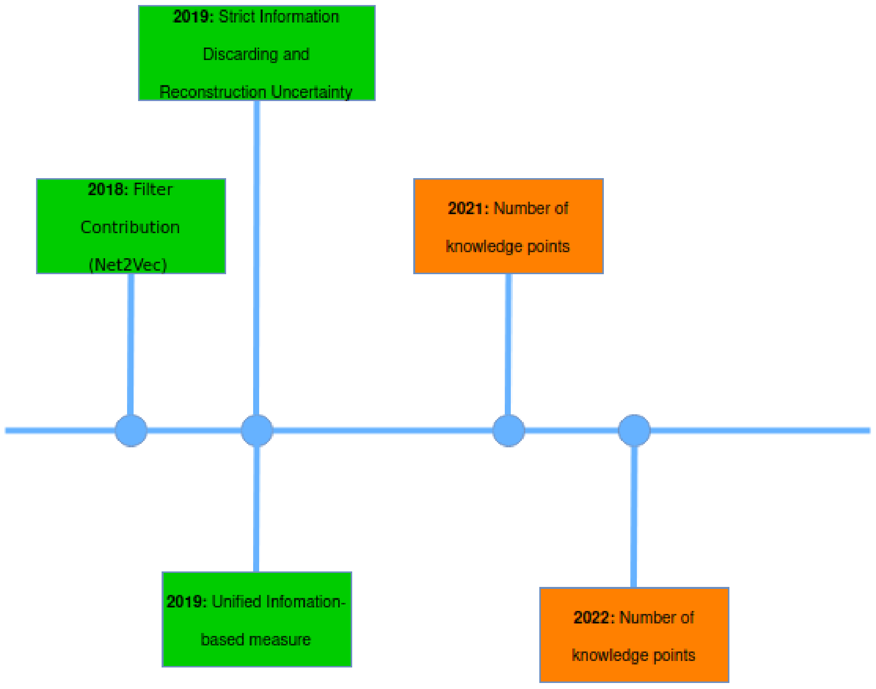
2019: Strict Information Discarding and Reconstruction Uncertainty
2018: Filter Contribution (Net2Vec)
2021: Number of knowledge points
2019: Unified Infomation-based measure
2022: Number of knowledge points

Fig. 4: A timeline of the metrics defined to quantify the knowledge of Deep Neural Networks.

Method	Architecture	Modality	Task	Year
Information-theoretic metrics				
Strict Information Discarding [51]	Generic	Generic	Generic	2019
Reconstruction Uncertainty [51]	Generic	Generic	Generic	2019
Filter Contribution (Net2Vec) [59]	CNN	Images	Segmentation	2018
Unified Information-based measure [53]	Generic	Language	Generic	2019
Geometric Mutual Information (GMI) [54]	Generic	Generic	Generic	2021
Knowledge points metrics				
Number of knowledge points [52]	Generic	Generic	Classification	2022
Number of knowledge points [27]	CNN	Images	Classification	2021

TABLE I: Knowledge quantification methods.

As it can be observed in Table I, all methods primarily aim to interpret and define the knowledge encoded in the DNN layers while striving to design appropriate quantification metrics. Information-theoretic methods focus on measuring the DNN layer's information, during inference on input samples, under the assumption that this information is proportional to the encoded/stored knowledge. In particular, Strict Information Discarding (SID) and Reconstruction Uncertainty (RU) [51] are both entropy-based metrics that measure input information discarding by a specific neural layer from different perspectives. Since entropy is a generic mathematical tool, both SID and RU are agnostic to the task and to the network architecture. Therefore, they can be used for comparisons between DNNs trained for different tasks, between different neural architectures executing the same task, as well as between different layers of the same DNN model. However, in most of the relevant literature, such metrics are only applied for whole-image classification tasks. Furthermore, based on the input data sample entropy, the unified information-based measure defined in [53] aims at interpreting deep NLP neural networks by quantifying the amount of information, or the knowledge, of an input word that is encoded in their intermediate layers. More specifically, four broadly used architectures are studied in this research: BERT [61], a Transformer [58], an LSTM, and a CNN. This method provides consistent and interpretable results across different layers of a model and also across different NLP models. The Net2Vec framework introduced in [59] is applied for both image segmentation and classification tasks, in order to investigate how semantic concepts are encoded by CNN convolution kernels. The framework links semantic image concepts with filter activations. Using the IoU score, a metric for measuring the contribution of a CNN convolution kernel to the encoding of an image concept is designed. The conditional geometric mutual information (GMI) can be used to measure the dependency between the neurons of successive DNN layers [26]. For each pair of DNN layers, the conditional GMI is used to compute a model-agnostic and task-agnostic importance score.

Knowledge points metrics aim to quantify the encoded DNN knowledge points and, thus, the encoded knowledge. In [52], knowledge points encoded in the intermediate layers of a classification DNN are quantified based on the entropy of the input data sample: they are defined as the input units whose information is discarded considerably less than that of other input units. It can be applied on multiple different model architectures and on different classification tasks (image classification, 3D point cloud classification, and binary sentiment classification), enabling comparisons between various layers of a single DNN, as well as between various DNN models. In [27] the total amount of knowledge points is measured by their discrimination power. It allows the quantitative and qualitative estimation of knowledge points encoded in different layers, in order to identify the percentage of the reliable ones. It has been implemented in conjunction with various CNN architectures trained for whole-image classification.

Method	**Coherency**		**Generality**
	Fair layer-wise comparisons	**Fair comparisons between different networks**	
Information-theoretic methods			
Strict Information Discarding[51]	Yes	Yes	Yes
Reconstruction Uncertainty[51]	Yes	Yes	Yes
Filter Contribution (Net2Vec)[59]	Yes	Yes	No
Unified Information-based measure[53]	Yes	Yes	Yes
Geometric Mutual Information (GMI)[54]	Yes	No	Yes
Knowledge points based methods			
Number of Knowledge points[52]	Yes	Yes	Yes
Number of Knowledge points[27]	Yes	Yes	No

TABLE II: Coherency and generality comparisons of the knowledge quantification methods.

A reliable method for quantifying the encoded DNN knowledge should meet the coherency and generality criteria mentioned in [51], [53], [52], as overviewed in Table II. SID and RU defined in [51], reflecting the entropy of input data samples, are both generic metrics and thus, ensure coherent evaluations and enable fair comparisons over different layers of the same, or different, DNNs. The unified information-theoretic measure defined in [53], based on entropy and mutual information measurements, provides reliable and coherent evaluations across neurons (or timestamps in NLP), layers, and models, demonstrating how the network structure processes data inputs through its layers. Knowledge points in [52] are also defined based on the pixel-wise information discarding and without strong assumptions on the neural architectures. Thus, this method provides coherent evaluations and enables fair layer-wise comparisons on the interior of a DNN architecture, as well as between different DNNs. The filter contribution measure defined in [59], quantifying how much a filter contributes to a concept encoding based on the IoU score, enables comparisons between different layers and various networks, hence providing also coherent results. The Geometric Mutual Information (GMI) score used in [26] to approximate an importance score of each neuron contribution, enables comparisons between different neurons and layers of the same DNN. However, it cannot compare different trained models, as it is neither agnostic to the task nor to the network architecture. The knowledge points defined in [27] are counted as regional patterns with strong discrimination power and evaluated as reliable or not, yielding coherent evaluations both between the DNN layers and between different DNNs (e.g., between the teacher and the student network for the specific task of knowledge distillation).

Entropy is a generic metric with significant connections to existing information theories [50]. As a result, the generality of the entropy-based metrics listed in Table II, Strict Information Discarding (SID), Reconstruction Uncertainty (RU) [51] and the number of the knowledge points defined in [52], enables fair comparisons between DNNs learned for different tasks and between different DNNs architectures. The unified information-based measure is defined in [53] without much restrictions on tasks and network architectures, since it is based on mutual information which is a fundamental quantity in information theory, ensuring the generality of the metric. The filter contribution measure defined in [59] and the knowledge points defined in [27] are both task-specific metrics and thus, do not allow fair comparisons between different tasks or architectures. The Geometric Mutual Information (GMI) score defined in [26], based on the geometric mutual information measurement, is used to compute a model-agnostic and task-agnostic importance score without hurting the generality of the metric.

It is clearly not generic nor trustworthy enough to employ the trained DNN predictions and directly measure the DNN model success rate at a specific test set when aiming at understanding and quantifying the amount of the encoded DNN knowledge. The methods outlined in this survey appear to be more in line with the issue of knowledge quantification and actually attempt to access, define and measure the encoded DNN knowledge, providing a new perspective on analyzing deep network properties. The information-theoretic metrics measure a trained DNN layer's information assuming that the amount of this information is proportional to the knowledge this layer encodes. This assumption is drawn as an obvious conclusion, given the strong connection between the encoded information and knowledge. However, a specific DNN knowledge definition is needed to directly develop trustworthy knowledge quantification metrics. Knowledge points methods directly define knowledge as the total amount of knowledge points encoded by the DNN and fairly quantify its actual amount. They can, therefore, interpret and diagnose the inner workings of DNNs not only by measuring the number of the encoded knowledge points but moreover, by evaluating their quality and determining whether they are reliable enough for classification. Knowledge points are quantified in input samples from the perspective of pixel-wise entropies [52], or in terms of their discrimination power [27], hence restricting their application exclusively to image classification tasks. On the other hand, since the network may concurrently acquire new knowledge points and discard old ones which are irrelevant to the task at each epoch, DNN learning cannot be accurately divided into a learning phase and a discarding phase. Thus, the division in [52] is just a rough approximation.

A number of the methods summarized above have been designed specifically for deep CNNs. However, the vast majority are generic ones and, thus, applicable to multiple different neural architectures. Still, the majority of the relevant experiments found in the literature have been conducted on image analysis tasks. Furthermore, almost all of the presented methods, which can be employed for knowledge quantification, focus solely on experiments for classification settings. Given the level of maturity other computational tasks have achieved thanks to DNNs (e.g., regression, object detection in images, image segmentation, machine translation, etc.) it is evident that this is still an emerging area in urgent need of additional investigation. Novel metrics need to be precisely defined and alternative approaches to measuring the knowledge of trained DNN knowledge must be discovered.

V. Conclusions

DNN knowledge quantification and interpretability remain a challenge within the machine learning community. This paper argues that precise DNN knowledge quantification measures are necessary to this end. However, although knowledge quantification metrics applied to trained models could potentially greatly facilitate this quest for interpretable DNNs, this is still very much a nascent and emerging area. Only a handful of existing methods attempt to measure the knowledge encoded inside a trained model, almost exclusively for the simple classification setting. Much more research is needed for knowledge in other machine learning tasks $e.g$ in regression. This survey reviews the state-of-the-art in this area and attempts to highlight knowledge quantification as an important emerging research challenge. Despite the identification of certain central trends (such as information-theoretic measures) in the scant relevant literature, further research is urgently needed.

Acknowledgement

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 951911 (AI4Media). This publication reflects only the authors' views. The European Commission is not responsible for any use that may be made of the information it contains.

References

[1] Efstratios Kakaletsis, Charalampos Symeonidis, Maria Tzelepi, Ioannis Mademlis, Anastasios Tefas, Nikos Nikolaidis, and Ioannis Pitas. "Computer Vision for Autonomous UAV Flight Safety: An Overview and a Vision-based Safe Landing Pipeline Example". In: *ACM Computing Surveys (CSUR)* 54.9 (2021), pp. 1–37.

[2] Dionysios Karamouzas, Ioannis Mademlis, and Ioannis Pitas. "Public opinion monitoring through collective semantic analysis of tweets". In: *Social Network Analysis and Mining* 12.1 (2022), pp. 1–21.

[3] Christos Papaioannidis, Ioannis Mademlis, and Ioannis Pitas. "Autonomous UAV safety by visual human crowd detection using multi-task deep neural networks". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 11074–11080.

[4] Christos Papaioannidis, Ioannis Mademlis, and Ioannis Pitas. "Fast CNN-based Single-Person 2D Human Pose Estimation for Autonomous Systems". In: *IEEE Transactions on Circuits and Systems for Video Technology* (2022).

[5] Iason Karakostas, Vasileios Mygdalis, Anastasios Tefas, and Ioannis Pitas. "Occlusion detection and drift-avoidance framework for 2D visual object tracking". In: *Signal Processing: Image Communication* 90 (2021), p. 116011.

[6] Yannis Assael, Thea Sommerschield, Brendan Shillingford, Mahyar Bordbar, John Pavlopoulos, Marita Chatzipanagiotou, Ion Androutsopoulos, Jonathan Prag, and Nando de Freitas. "Restoring and attributing ancient texts using deep neural networks". In: *Nature* 603.7900 (2022), pp. 280–283.

[7] Spiros Chadoulos, Iordanis Koutsopoulos, and George C Polyzos. "One model fits all: Individualized household energy demand forecasting with a single deep learning model". In: *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*. 2021, pp. 466–474.

[8] Brais Bosquet, Daniel Cores, Lorenzo Seidenari, Víctor M Brea, Manuel Mucientes, and Alberto Del Bimbo. "A full data augmentation pipeline for small object detection based on generative adversarial networks". In: *Pattern Recognition* 133 (2023), p. 108998.

[9] Nikolaos Passalis, Loukia Avramelou, Solon Seficha, Avraam Tsantekidis, Stavros Doropoulos, Giorgos Makris, and Anastasios Tefas. "Multisource financial sentiment analysis for detecting Bitcoin price change indications using deep learning". In: *Neural Computing and Applications* 34.22 (2022), pp. 19441–19452.

[10] Charalampos Symeonidis, Ioannis Mademlis, Ioannis Pitas, and Nikos Nikolaidis. “Neural Attention-driven Non-Maximum Suppression for Person Detection”. In: *IEEE Transactions on Image Processing* (2023). accepted for publication.

[11] Adam S Charles. “Interpreting deep learning: The machine learning Rorschach test?” In: *arXiv preprint arXiv:1806.00148* (2018).

[12] Vladimir N. Vapnik. “An overview of statistical learning theory”. In: *IEEE Transactions on Neural Networks* vol. 10.no. 5 (1999), pp. 988–999.

[13] Vladimir Vapnik, Esther Levin, and Yann Le Cun. “Measuring the VC-dimension of a learning machine”. In: *Neural computation* vol. 6.no. 5 (1994), pp. 851–876.

[14] Runqi Wang, Linlin Yang, Baochang Zhang, Wentao Zhu, David Doermann, and Guodong Guo. “Confidence Dimension for Deep Learning based on Hoeffding Inequality and Relative Evaluation”. In: *arXiv preprint arXiv:2203.09082* (2022).

[15] Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. “Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks”. In: *The Journal of Machine Learning Research* vol. 20.no. 1 (2019), pp. 2285–2301.

[16] Husheng Li. “Analysis on the nonlinear dynamics of deep neural networks: Topological entropy and chaos”. In: *arXiv preprint arXiv:1804.03987* (2018).

[17] Jingwei Zhang, Tongliang Liu, and Dacheng Tao. “An optimal transport view on generalization”. In: *arXiv preprint arXiv:1811.03270* (2018).

[18] Himanshu Pant, Mayank Sharma, Abhimanyu Dubey, Sumit Soman, Suraj Tripathi, Sai Guruju, Nihal Goalla, et al. “Learning Neural Network Classifiers with Low Model Complexity”. In: *arXiv preprint arXiv:1707.09933* (2017).

[19] Mayank Sharma, Aayush Yadav, Sumit Soman, et al. “Effect of Various Regularizers on Model Complexities of Neural Networks in Presence of Input Noise”. In: *aarXiv preprint arXiv:1901.11458* (2019).

[20] Mayank Sharma, Sumit Soman, et al. “Radius-margin bounds for deep neural networks”. In: *rXiv preprint arXiv:1811.01171* (2018).

[21] Shizhao Sun, Wei Chen, Liwei Wang, Xiaoguang Liu, and Tie-Yan Liu. “On the depth of deep neural networks: A theoretical view”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* vol. 30.no. 1 (2016).

[22] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. “Understanding deep learning (still) requires rethinking generalization”. In: *Communications of the ACM* vol. 61.no. 3 (202), pp. 107–115.

[23] Pitas Ioannis. *Artificial Intelligence Science and Society / Part A: Introduction to AI Science and Information Technology.* Pitas Ioannis; 1st edition (October 11, 2022), 2022.

[24] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. “Distilling the Knowledge in a Neural Network”. In: *arXiv preprint arXiv:1503.02531* vol. 2.no. 7 (2015).

[25] Xu Cheng, Zhefan Rao, Yilan Chen, and Quanshi Zhang. “Explaining Knowledge Distillation by Quantifying the Knowledge”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 12925–12935.

[26] Madan Ravi Ganesh, Jason J Corso, and Salimeh Yasaei Sekeh. “MINT: Deep Network Compression via Mutual Information-based Neuron Trimming”. In: *IEEE 2020 25th International Conference on Pattern Recognition (ICPR)* (2021), pp. 8251–8258.

[27] Mingjie Li, Shaobo Wang, and Quanshi Zhang. “Visualizing the Emergence of Intermediate Visual Patterns in DNNs”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 6594–6607.

[28] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”. In: *In Workshop at International Conference on Learning Representations* (2014).

[29] Alexey Dosovitskiy and Thomas Brox. “Inverting Visual Representations with Convolutional Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4829–4837.

[30] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. “Visualizing Deep Neural Network Decisions: Prediction Difference Analysis”. In: *arXiv preprint arXiv:1702.04595* (2017).

[31] Pieter-Jan Kindermans, Kristof T Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. “Learning how to explain neural networks: PatternNet and PatternAttribution”. In: *arXiv preprint arXiv:1705.05598* (2017).

[32] 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* (2017), pp. 618–626.

[33] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. “Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 839–847.

[34] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. “Network Dissection: Quantifying Interpretability of Deep Visual Representations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 6541–6549.

[35] Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. “The Language Interpretability Tool: Extensible, Interactive Visualizations and Analysis for NLP Models”. In: *arXiv preprint arXiv:2008.05122* (2020).

[36] Zijie J. Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das, Fred Hohman, Minsuk Kahng, and Duen Horng Polo Chau. “CNN Explainer: Learning Convolutional Neural Networks with Interactive Visualization”. In: *IEEE Transactions on Visualization and Computer Graphics* vol. 27.no. 2 (2021), pp. 1396–1406.

[37] 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* (2018), pp. 883–892.

[38] Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. “Estimating Information Flow in Deep Neural Networks”. In: *arXiv preprint arXiv:1810.05728* (2018).

[39] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. “Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach”. In: *arXiv preprint arXiv:1801.10578* (2018).

[40] Zhiqin John Xu. “Understanding training and generalization in deep learning by Fourier analysis”. In: *arXiv preprint arXiv:1808.04295* (2018).

[41] Stanislav Fort, Paweł Krzysztof Nowak, Stanislaw Jastrzebski, and Srini Narayanan. “Stiffness: A New Perspective on Generalization in Neural Networks”. In: *arXiv preprint arXiv:1901.09491* (2019).

[42] Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. “Sensitivity and Generalization in Neural Networks: an Empirical Study”. In: *arXiv preprint arXiv:1802.08760* (2018).

[43] Chiyuan Zhang, Samy Bengio, and Yoram Singer. “Are All Layers Created Equal?” In: *arXiv preprint arXiv:1902.01996* (2019).

[44] Niladri S Chatterji, Behnam Neyshabur, and Haniei Sedghi. “The intriguing role of module criticality in the generalization of deep networks”. In: *arXiv preprint arXiv:1912.00528* (2019).

[45] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. “Similarity of Neural Network Representations Revisited”. In: *International Conference on Machine Learning* (2019), pp. 3519–3529.

[46] Ruofan Liang, Tianlin Li, Longfei Li, Jing Wang, and Quanshi Zhang. “Knowledge Consistency between Neural Networks and Beyond”. In: *arXiv preprint arXiv:1908.01581* (2019).

[47] Die Zhang, Huilin Zhou, Hao Zhang, Xiaoyi Bao, Da Huo, Ruizhao Chen, Xu Cheng, Mengyue Wu, and Quanshi Zhang. “Building Interpretable Interaction Trees for Deep NLP Models”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2021).

[48] Hao Zhang, Yichen Xie, Longjie Zheng, Die Zhang, and Quanshi Zhang. “Interpreting Multivariate Shapley Interactions in DNNs”. In: *arXiv preprint arXiv:2010.05045* (2020).

[49] Hao Zhang, Sen Li, Yinchao Ma, Mingjie Li, Yichen Xie, and Quanshi Zhang. “Interpreting and Boosting Dropout from a Game-Theoretic View”. In: *International Conference on Learning Representations* vol. 35.no. 16 (2021), pp. 14328–14337.

[50] Ravid Shwartz-Ziv and Naftali Tishby. “Opening the Black Box of Deep Neural Networks via Information”. In: *arXiv preprint arXiv:1703.00810* (2017).

[51] Haotian Ma, Yinqing Zhang, Fan Zhou, and Quanshi Zhang. “Quantifying Layerwise Information Discarding of Neural Networks”. In: *arXiv preprint arXiv:1906.04109* (2019).

[52] Quanshi Zhang, Xu Cheng, Yilan Chen, and Zhefan Rao. “Quantifying the Knowledge in a DNN to Explain Knowledge Distillation for Classification”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[53] Chaoyu Guan, Xiting Wang, Quanshi Zhang, Runjin Chen, Di He, and Xing Xie. “Towards a Deep and Unified Understanding of Deep Neural Models in NLP”. In: *36th International Conference on Machine Learning* (2019), pp. 2454–2463.

[54] Salimeh Yasaei Sekeh and Alfred O Hero. “Geometric Estimation of Multivariate Dependency”. In: *Entropy* vol.21.no.8 (2019), p. 787.

[55] Jerome H Friedman and Lawrence Cs Rafsky. “Multivariate Generalizations of the Wald-Wolfowitz and Smirnov Two-Sample Tests”. In: *The Annals of Statistics* vol. 7.no. 4 (1979), pp. 697–717.

[56] Visar Berisha and Alfred O Hero. “Empirical non-parametric estimation of the Fisher information”. In: *IEEE Signal Processing Letters* vol. 2.no. 7 (2014), pp. 988–992.

[57] Rajat Sen, Ananda Theertha Suresh, Karthikeyan Shanmugam, Alexandros G Dimakis, and Sanjay Shakkottai. “Model-powered conditional independence test”. In: *Advances in neural information processing systemss* vol. 30 (2007).

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[59] Ruth Fong and Andrea Vedaldi. “Net2Vec: Quantifying and Explaining how Concepts are Encoded by Filters in Deep Neural Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 8730–8738.

[60] Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, Suvrit Sra, and Greg Ridgeway. “Clustering on the Unit Hypersphere using von Mises-Fisher Distributions.” In: *Journal of Machine Learning Research* 6.9 (2005).

[61] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).