Towards Explainable Deep Neural Networks (xDNN)

Plamen Angelov[a], Eduardo Soares[a]

[a]*School of Computing and Communications, LIRA Research Centre, Lancaster University, Lancaster, LA1 4WA, UK*
E-mail: p.angelov@lancaster.ac.uk; e.almeidasoares@lancaster.ac.uk

Abstract

In this paper, we propose an elegant solution that is directly addressing the bottlenecks of the traditional deep learning approaches and offers an explainable internal architecture that can outperform the existing methods, requires very little computational resources (no need for GPUs) and short training times (in the order of seconds). The proposed approach, xDNN is using prototypes. Prototypes are actual training data samples (images), which are local peaks of the empirical data distribution called *typicality* as well as of the data density. This generative model is identified in a closed form and equates to the pdf but is derived automatically and entirely from the training data with no user- or problem-specific thresholds, parameters or intervention. The proposed xDNN offers a new deep learning architecture that combines reasoning and learning in a synergy. It is non-iterative and non-parametric, which explains its efficiency in terms of time and computational resources. From the user perspective, the proposed approach is clearly understandable to human users. We tested it on challenging problems as the classification of different lighting conditions for driving scenes (iROADS), object detection (Caltech-256, and Caltech-101), and SARS-CoV-2 identification via computed tomography scan (COVID CT-scans dataset). xDNN outperforms the other methods including deep learning in terms of accuracy, time to train and offers an explainable classifier.

Keywords:

Explainable AI, Interpretability, Prototype-based Models, Deep-Learning.

1. Introduction

Deep learning has demonstrated ability to achieve highly accurate results in different application domains such as speech recognition (Xiong et al., 2018), image recognition (He et al., 2016), and language translation (LeCun et al., 2015) and other complex problems (Goodfellow et al., 2016). It attracted the attention of media and the wider public (Sejnowski, 2018). It has also proven to be very valuable and efficient in automating the usually laborious and sometimes controversial pre-processing stage of feature extraction. The main criticism towards deep learning is usually related to its 'black-box' nature and requirements for huge amount of labeled data, computational resources (GPU accelerators as a standard), long times (hours) of training, high power and energy requirements (Rudin, 2019). Indeed, a traditional deep learning (e.g. convolutional neural network) algorithm involves hundreds of millions of weights/coefficients/parameters that require iterative optimization procedures. In addition, these hundreds of millions of parameters are abstract and detached from the physical nature of the problem being modelled. However, the automated way to extract them is very attractive in high throughput applications of complex problems like image processing where the human expertise may simply be not available or very expensive.

Feature extraction is an important pre-processing stage, which defines the data space and may influence the level of accuracy the end result provides. Therefore, we consider this very useful property of the traditional deep learning and step on it combined with another important recent result in the deep learning domain, namely, the transfer learning. This concept postulates that knowledge in the form of a model architecture learned in one context can be re-used and useful in another context (Hu et al., 2015). Transfer learning helps to considerably reduce the amount of time used for training. Moreover, it also may help to improve the accuracy of the models (Zhuang et al., 2015).

Stepping on the two main achievements of the deep learning - top accuracy combined with an automatic approach for feature extraction for complex prob-

lems, such as image classification, we try to address its deficiencies such as the lack of explainability (Rudin, 2019), computational burden, power and energy resources required, ability to self-adapt and evolve (Soares and Angelov, 2019). Interpretability and explainability are extremely important for high stake applications, such as autonomous cars, medical or court decisions, etc. For example, it is extremely important to know the reasons why a car took some action, especially if this car is involved in an accident (Doshi-Velez and Kim, 2017).

The state-of-the-art classifiers offer a choice between higher explainability for the price of lower accuracy or vice versa (Figure 1). Before deep learning (Schmidhuber, 2015), machine-learning and pattern-recognition required substantial domain expertise to model a feature extractor that could transform the raw data into a feature vector which defines the data space within which the learning subsystem could detect or classify data patterns (LeCun et al., 2015). Deep learning offers new way to extract abstract features automatically. Moreover, pre-trained structures can be reused for different tasks through the transfer learning technique (Hu et al., 2015). Transfer learning helps to considerably reduce the amount of time used for training, moreover, it also may help to improve the accuracy of the models (Zhuang et al., 2015). In this paper, we propose a new approach, xDNN that offers both, high level of explainability combined with the top accuracy.

The proposed approach, xDNN offers a new deep learning architecture that combines reasoning and learning in a synergy. It is based on prototypes and the data density (Angelov and Gu, 2019) as well as *typicality* - an empirically derived pdf (Angelov et al., 2017). It is non-iterative and non-parametric, which explains its efficiency in terms of time and computational resources. From the user perspective, the proposed approach is clearly understandable to human users. We tested it on some well-known benchmark data sets such as iRoads (Rezaei and Terauchi, 2013) and Caltech-256 (Griffin et al., 2007) and xDNN outperforms the other methods including deep learning in terms of accuracy, time to train, moreover, offers an explainable classifier. In fact, the result on the very hard Caltech-256 problem (which has 257 classes) represents a world

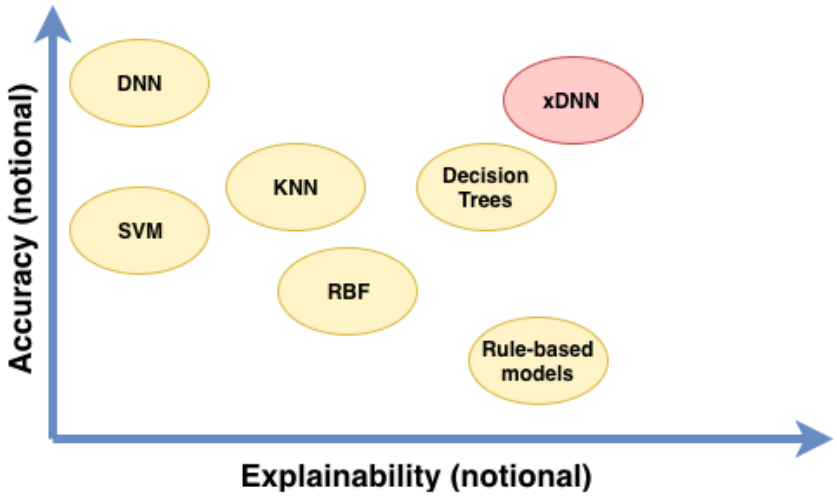

Figure 1: Trade-off between accuracy and explainability.

record (He et al., 2015).

The remainder of this paper is organized as follows: The next section introduces a brief literature review. The proposed explainable deep learning approach is presented in Section III. The data employed in the analysis is presented in Section IV, and the results are presented in Section V. The discussion is presented in the last section of this paper.

2. Brief Literature Review

Deep Neural Networks have often been designed purely for accuracy. The decisions made by these networks are at best interpreted by *post hoc* techniques (Li et al., 2018) or not interpreted at all. That is, the first step is the selection of the network architecture by the human and the attempt to interpret the trained model and the learned high-level features follows. Therefore, the *post hoc* interpretability analysis requires a separate modeling effort (Saralajew et al., 2018) and is an approximation rather than a deep explanation of the cause-effect relations and reasoning. One of the problems with *post hoc* approach is that

the explanations can change for different models used. In other words, it is easy to create multiple conflicting yet convincing explanations for how the network would classify a single object.

Prototypes-based classifiers are a reasoning process that do not consider *post hoc* analysis (Biehl et al., 2016). They rely on the similarity (proximity in the feature space) of a data sample to a given prototype (Biehl et al., 2016, 2013). Different works have different meanings for the word "prototype" (Biehl et al., 2016, 2013, Saralajew et al., 2018), in our case we consider prototypes to be the most representative data samples of the training set (the data samples which have local peaks of the density (Angelov and Gu, 2019)). In other cases, a prototype can be considered as a convex combination of several observations, and not necessarily required to be close to any data sample of the training set or even to be feasible (Oyedotun and Khashman, 2017, Liu et al., 2018).

Our work is closely aligned with other prototype classification techniques in machine learning. Prototype classification is a classical form of case-based reasoning (Li et al., 2018); however, as (Li et al., 2018) uses neural networks, the distance measure between prototypes and observations is measured in a latent space. (Li et al., 2018) uses an auto encoder to create a latent low-dimensional space, and distances to prototypes are computed in that latent space. Other works also use Euclidean distance calculation can be expressed in terms of convolution operations in the neural network sense (Nebel et al., 2017, Biehl et al., 2013). This and the computation of the Euclidean distance in terms of a dot product are essential steps towards efficient computational schemes for prototype-based neural network layers.

In contrast, the proposed method uses local densities and global multivariate generative distributions based on an empirically derived form of the probability distribution function (Angelov and Gu, 2019). Furthermore, differently from other prototype-based classifiers, the presented method is non-iterative and non-parametric as it is using recursive calculations and no search procedures. Moreover, the proposed algorithm can learn continuously without full re-training.

3. Explainable Deep Neural Network

3.1. Architecture and Training of the proposed xDNN

The proposed explainable deep neural network (xDNN) classifier is formed of several layers with a very clear semantic and functional meaning. In addition to the internal clarity and transparency it also offers a very clear from the user point of view set of prototype-based $IF...THEN$ rules. Prototypes are selected data samples (images) that the user can easily view, understand and appreciate the similarity to other validation images. xDNN offers a synergy between the statistical learning and reasoning bringing both together. In most of the other approaches there is a dichotomy and preference of one over the other. We advocate and demonstrate that both, learning and reasoning can work together in a synergy and produce very impressive results. Indeed, the proposed xDNN method outperforms all published results (Rezaei and Terauchi, 2013, He et al., 2015, Angelov and Gu, 2018) in terms of accuracy. Moreover, in terms of time for training, computational simplicity, low power and energy required it is also far ahead. The proposed approach can be described as a feedforward neural network which has an incremental learning algorithm that autonomously self-develops and evolves its structure adding new prototypes to reflect the possibly changing (dynamically evolving) data pattern (Soares and Angelov, 2019). As shown in Figure 3, xDNN is composed of the following layers–

1. Features layer;
2. Density layer;
3. Typicality layer;
4. Prototypes layer;
5. *MegaClouds* layer;

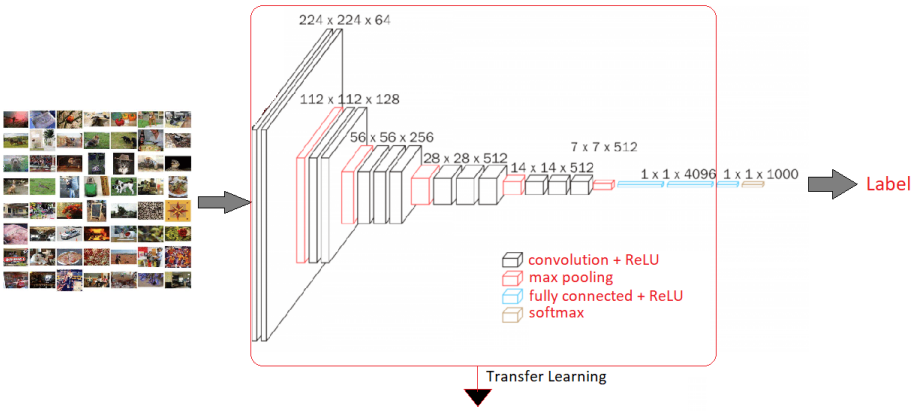

Figure 2: Pre-training a traditional deep neural network (weights of the network are being optimized/trained). Using the transfer learning concept this architecture with the weights are used as feature extractor (the last fully connected layer is considered as a feature vector). Adapted from (Simonyan and Zisserman, 2014).

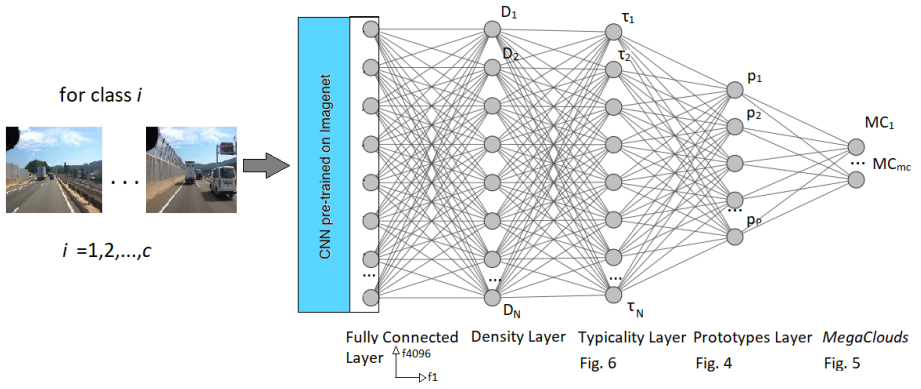

Figure 3: xDNN training architecture (per class).

1. **Features layer**: (Defines the data space)

   The Feature Layer is the first phase of the proposed xDNN method. This layer is in charge of extracting global features vector from the images. This first layer can be formed by more traditional 'handcrafted' meth-

ods such as GIST (Solmaz et al., 2013) or HoG (Mizuno et al., 2012). Alternatively, it can be formed by the fully connected layer (FCL) of the pre-trained convolutional neural network approaches such as AlexNet (Krizhevsky et al., 2012), VGG–VD–16 (Simonyan and Zisserman, 2014), and Inception (Szegedy et al., 2015), residual neural networks such as Resnet (He et al., 2016) or Inception-Resnet (Szegedy et al., 2017), etc. Using pre-trained deep neural network approach allows automatic extraction of more abstract and discriminative high-level features. In this paper, pre-trained VGG–VD–16 DCNN is employed for feature extraction. According to (Ren et al., 2016), VGG–VD–16 has a simple structure and it can achieve a better performance in comparison with other pre-trained deep neural networks. The first fully connected layer from VGG–VD–16 provides a 1×4096 dimensional vector.

a) The values are then standardized using the following equation (1):

$$\widehat{x}_{i,j} = \frac{x_{i,j} - \mu(x_{i,j})}{\sigma(x_{i,j})} \tag{1}$$

where $\widehat{x}$ denotes a standardized features vector x of the image I (x are the values provided by the FCL), $i = 1, 2, ..., N$ denotes the time stamp or the ID of the image, $j = 1, 2, ..., n$ refers to the number of features of the given x in our case $n = 4096$.

b) The standardized values are normalised to bring them to the range [0;1]:

$$\bar{x}_{i,j} = \frac{\widehat{x}_{i,j} - \min\limits_{i}(\widehat{x}_{i,j})}{\max\limits_{i}(\widehat{x}_{i,j}) - \min\limits_{i}(\widehat{x}_{i,j})} \tag{2}$$

where $\bar{x}$ denotes the normalized value of the features vector. For clarity in the rest of the paper we will use x instead of $\bar{x}$.

Initialization:

Meta-parameters for the xDNN are initialized with the first observed data sample (image). The proposed algorithm works per class; therefore, all

the calculations are done for each class separately.

$$P \leftarrow 1; \quad \mu \leftarrow x_i; \tag{3}$$

where μ denotes the global mean of data samples of the given class. P is the total number of the identified prototypes from the observed data samples (images).

Each class C is initialized by the first data sample of that class:

$$\begin{gathered} \mathrm{C}_1 \leftarrow x_1; \quad p_1 \leftarrow x_1; \\ Support_1 \leftarrow 1; \quad r_1 \leftarrow r^*; \quad \hat{I}_1 \leftarrow I_1 \end{gathered} \tag{4}$$

where, p_1 is the vector of features that describe the prototype $\hat{I}$ of the C_1; $\hat{I}$ is the identified prototype; $Support_1$ is the corresponding support (number of members) associated with this prototype; r_1 is the corresponding radius of the area of influence of C_1.

In this paper, we use $r^* = \sqrt{2 - 2cos(30^o)}$ same as (Angelov and Gu, 2019); the rationale is that two vectors for which the angle between them is less than $\pi/6$ or 30^o are pointing in close/similar directions d. That is, we consider that two feature vectors can be considered to be similar if the angle between them is smaller than 30 degrees. Note that r^* is data derived, not a problem- or user- specific parameter. In fact, it can be defined without *prior* knowledge of the specific problem or data through the following equation (5).

$$d(x_i, p_i) = \left\| \frac{x_i}{\|x_i\|} - \frac{p_i}{\|p_i\|} \right\|. \tag{5}$$

2. **Density layer**:

The density layer defines the mutual proximity of the images in the data space defined by the features from the previous layer. The data density, if use Euclidean form of distance, has a Cauchy form (15) (Angelov and Gu, 2019):

$$D(x_i) = \frac{1}{1 + \frac{||x_i - \mu_N||^2}{||\sigma||_N^2}}, \tag{6}$$

where D is the density, μ is the global mean, and σ is the variance. The reason it is Cauchy is not arbitrary (Angelov and Gu, 2019). It can be demonstrated theoretically that if Euclidean or Mahalanobis type of distances in the feature space are considered, the data density reduces to Cauchy type as referred in equation (15). Density can also be updated online (Angelov, 2012):

$$D(x_i) = \frac{1}{1 + ||x_i - \mu_i||^2 + \sum_i - ||\mu_i||^2}. \tag{7}$$

where μ_i and the scalar product, $\sum_i$ can be updated recursively as follows:

$$\mu_i = \frac{i-1}{i}\mu_{i-1} + \frac{1}{i}x_i, \tag{8}$$

$$\sum_i = \frac{i-1}{i}\sum_{i-1} + \frac{1}{i}||x_i||^2 \quad \sum_1 = ||x_1||^2. \tag{9}$$

Data samples (images) that are closer to the global mean have higher density values. Therefore, the value of the data density indicates how strongly a particular data sample is influenced by other data samples in the data space due to their mutual proximity.

3. **Typicality layer**:

   Typicality is is an empirically derived form of probability distribution function (pdf). *Typicality* τ is given by the equation (10). The value of τ even at the point $x = p_i$ is much less than 1; the integral of $\int_{-\infty}^{\infty} \tau dx = 1$ (Angelov and Gu, 2019).

$$\tau(x_i) = \frac{\sum_{i=1}^{c} Support_i D(x_i)}{\sum_{i=1}^{c} Support_i \int_{-\infty}^{\infty} D(x_i) dx} \tag{10}$$

4. **Prototypes layer**:

   The prototypes identification layer is the core of the proposed xDNN classifier. This layer is responsible to provide the clearly explainable model.

The xDNN classifier is free from *prior* assumptions about the data distribution type, as well as the random or deterministic nature of the data. In contrast, it empirically extracts the distribution from the data samples (images) bottom up (Angelov and Gu, 2019). The prototypes are independent from each other. Therefore, one can change the structure by adding a new prototype without influencing the other already existing prototypes. In other words, the proposed xDNN is highly parallelizable and suitable for evolving form of application where new prototypes may be added (if the data pattern requires this). The proposed xDNN method is trained per class forming a set of prototypes per class. Therefore, all the calculations are done for each class separately. Prototypes are the local peaks of the data density (and *typicality*) identified in the previous layers/ stages of the algorithm from the images of the corresponding class based on their feature vectors. The prototypes can be used to form linguistic logical $IF...THEN$ rules of the following form:

R_c: IF $(I \sim \hat{I}_P)$ THEN $(class\ c)$

where $\sim$ stands for similarity, it also can be seen as a fuzzy degree of membership; p is the identified prototype; P is the number of identified prototypes; c is the class $c = 1, 2, ..., C$, I denotes an image.

One rule per prototype can be formed. All rules per class can be combined together using logical OR, also known as disjunction or S-norm:

R_c: IF $(I \sim \hat{I}_1)$ OR $(I \sim \hat{I}_2)$ OR ... OR $(I \sim \hat{I}_P)$ THEN $(class\ c)$

Figure 4 illustrates the area of influence of the identified prototypes. These areas around the identified prototypes are called *data clouds* (Angelov and Gu, 2019). Thus, each prototype defines a *data cloud*.

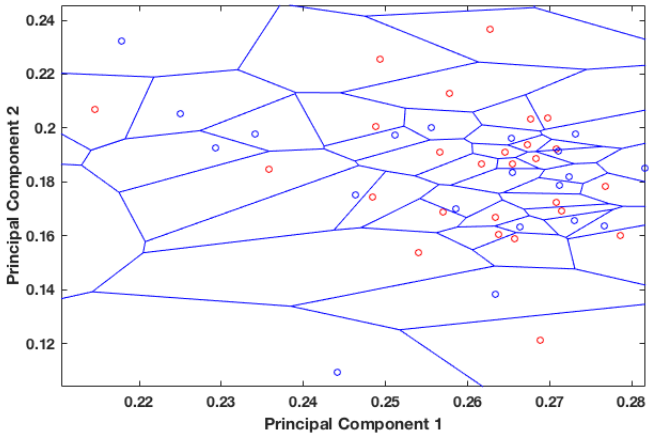

Figure 4: Identified prototypes – Voronoi Tesselation.

We call all data points associated with a prototype *data clouds*, because their shape is not regular (e.g., hyper-spherical, hyper-ellipsoidal, etc.) and the prototype is not necessarily the statistical and geometric mean , but actual image (Angelov and Gu, 2019). The algorithm absorbs the new data samples one by one by assigning then to the nearest (in the feature space) prototype:

$$j^* = \underset{j=1,2,...,P}{\operatorname{argmin}} (||x_i - p_j||^2) \tag{11}$$

In case, the following condition (Angelov and Gu, 2019) is met:

$$\begin{gathered} IF\ (D(x_i) \geq \max_{j=1,2,...,P} D(p_j)) \\ OR \quad (D(x_i) \leq \min_{j=1,2,...,P} D(p_j)) \\ THEN\ (add\ a\ new\ data\ cloud\ (P \leftarrow P+1)) \end{gathered} \tag{12}$$

It means that x_i is out of the influence area of p_j. Therefore, the vector of features x_i becomes a new prototype of a new *data cloud* with meta-

parameters initialized by equation (13). Add a new *data cloud*:

$$P \leftarrow P+1; \quad \mathrm{C}_P \leftarrow x_i; p_P \leftarrow I_i; \quad Support_P \leftarrow 1; \\ r_P \leftarrow r_o; \hat{I}_P \leftarrow I_i; \tag{13}$$

Otherwise, *data cloud* parameters are updated online by equation (14). It has to be stressed that all calculations per *data cloud* are performed on the basis of data points associated with a certain *data cloud* only (i. e. locally, not globally, on the basis of all data points).

$$C_{j^*} \leftarrow C_{j^*}+1; \\ p_{j^*} \leftarrow \frac{Support_{j^*}}{Support_{j^*}+1} p_{j^*} + \frac{Support_{j^*}}{Support_{j^*}+1} x_i; \\ Support_{j^*} \leftarrow Support_{j^*}+1; \\ r^2_{j^*} \leftarrow \frac{r^2_{j^*} + (1-||p_{j^*}||^2)}{2}. \tag{14}$$

The xDNN learning procedure can be summarized by the following algorithm.

xDNN: Learning Procedure

1: Read the first feature vector sample x_i representing the image I_i of the class c;
2: Set $i \leftarrow 1; n \leftarrow 1; P_1 \leftarrow 1; p_1 \leftarrow x_i; \mu \leftarrow x_1; Support \leftarrow 1; r_1 \leftarrow r_0; \hat{I}_1 \leftarrow I_1$;
3: **FOR** $i = 2$, ...
4: Read x_i;
5: Calculate $D(x_i)$ and $D(p_j)$ $(j = 1, 2, ..., P)$ according to equation (9);
6: **IF** equation (12) holds
7: Create rule according to equation (13);
8: **ELSE**
9: Search for p_j according to equation (11);
10: Update rule according to equation (14);

11: **END**

12: **END**

5. ***MegaClouds* layer**:

In the *MegaClouds* layer the *clouds* formed by the prototypes in the previous layer are merged if the neighbouring prototypes have the same class label. In other words, they are merged if they belong to the same class. *MegaClouds* are used to facilitate the human interpretability. Figure 5 illustrates the formation of the *MegaClouds*.

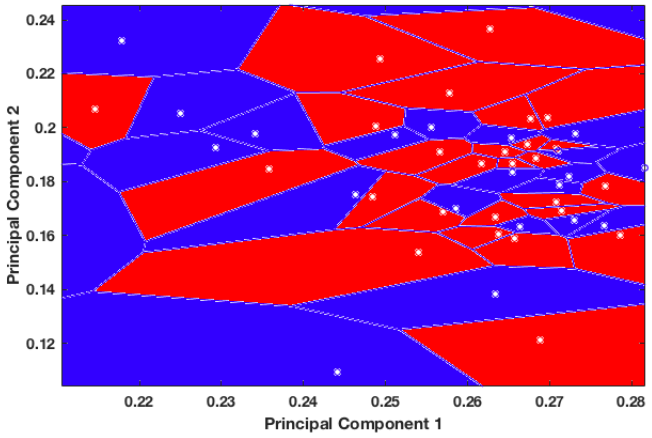

Figure 5: *MegaClouds* – Voronoi Tesselation.

Rules in the *MegaClouds* layer have the following format:

R_c: IF $(x \sim MC_1)$ OR $(x \sim MC_2)$ OR ... OR $(x \sim MC_{mc})$ THEN $(class\ c)$

where MC are the *MegaClouds*, or the areas formed from the merging of the *clouds*, and mc is the number of identified *MegaClouds*. Multimodal *typicality*, τ, can also be used to illustrate the *MegaClouds* as illustrated by Figure 6.

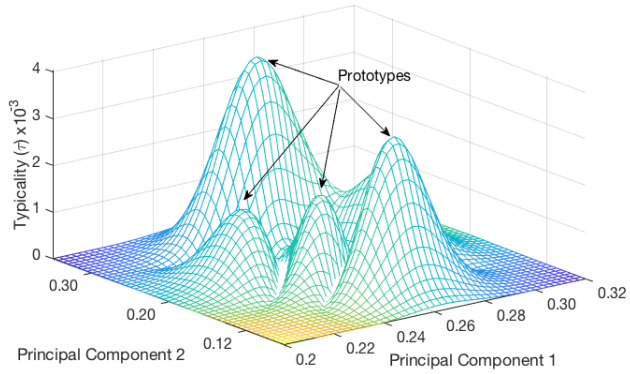

Figure 6: *Typicality* for the iRoads dataset.

3.2. Architecture and Validation of the proposed xDNN

Architecture for the validation process of the proposed xDNN method is illustrated by Figure 7.

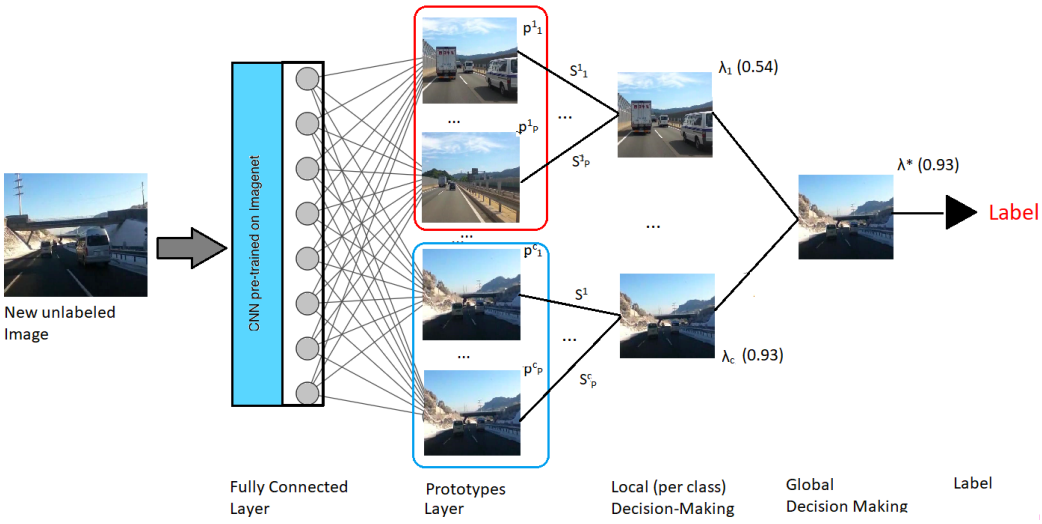

Figure 7: Architecture for the validation process of the proposed xDNN.

The validation process of xDNN is composed of the following layers:

1. Features layer;
2. Similarity layer (density);
3. Local decision-making.
4. Global decision-making.

Which is detailed described as following:

1. **Features layer**:
   Similarly to the features layer described in the training process.
2. **Prototypes layer**:
   In this layer the degrees of similarity to the nearest prototypes (per class) are extracted for each unlabeled (new/validation) data sample/image I_i defined as follows:

$$S(x, p_i) = \frac{1}{1 + \frac{||x-p_i||^2}{||\sigma||_N^2}}, \tag{15}$$

   where S denotes the similarity degree.
3. **Local (per class) decision-making layer**:
   Local (per class) decision-making is calculated based on the 'winner-takes-all' principle and can be obtained by:

$$\lambda_c = \underset{j=1,2,\ldots,P}{max}(S_j), \tag{16}$$

4. **Global decision-making layer**: The global decision-making layer is in charge of forming the decision by assigning labels to the validation images based on the degree of similarity of the prototypes obtained by the prototype identification layer as illustrated by Figure 7 and determining the winning class.

$$\lambda_c^* = \underset{c=1,2,\ldots,C}{max}(\lambda_c), \tag{17}$$

   In order to determine the overall degree of satisfaction, the maximum of the local, per class winners is applied.

The label is obtained by the following equation (18):

$$label = \underset{c=1,2,...,C}{argmax}\,(\lambda_c^*), \tag{18}$$

4. Experimental Data

We validated our proposed approach, xDNN using several complex, well-known image classification benchmark datasets (iRoads, Calltech-256, Calltech-101) as well as we propose our own dataset for SARS-CoV-2 identification.

4.1. iRoads dataset

The iROADS dataset (Rezaei and Terauchi, 2013) was considered in the analysis first. The dataset contains 4,656 image frames recorded from moving vehicles on a diverse set of road scenes, recorded in day, night, under various weather and lighting conditions, as described below:

- Daylight - 903 images
- Night - 1050 images
- Rainy day - 1049 images
- Rainy night - 431 images
- Snowy - 569 images
- Sun strokes - 307 images
- Tunnel - 347 images

4.2. Caltech-256

Caletch-256 has 30,607 images divided into 257 object categories (one of which is the background) (Griffin et al., 2007).

4.3. Caltech-101

Caletch-101 is divided into 102 object categories (one of which is the background) (Fei-Fei et al., 2004).

4.4. COVID-CT dataset

COVID-CT dataset contains 275 computed tomography scans positive for COVID-19 (Zhao et al., 2020).

4.5. Performance Evaluation

We used the following metrics for classification evaluation:

$$ACC(\%) = \frac{TP + TN}{TP + FP + TN + FN} \times 100, \tag{19}$$

Precision:

$$Precision(\%) = \frac{TP}{TP + FP} \times 100, \tag{20}$$

Recall:

$$Recall(\%) = \frac{TP}{TP + FN} \times 100, \tag{21}$$

$F1$ Score:

$$F1\ Score(\%) = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100, \tag{22}$$

where TP, FP, TN, FN denote true and false, negative and positive respectively.

The area under the curve, AUC, is defined through the TP rate and FN rate.

All the experiments were conducted with MATLAB 2018a using a personal computer with a 1.8 GHz Intel Core i5 processor, 8-GB RAM, and MacOS operating system. The classification experiments were executed using 10-fold cross validation under the same ratio of training-to-testing (90% to 10%) sample sets.

5. Results and Analysis

Computational simulations were performed to assess the accuracy of the proposed explainable deep learning method, xDNN against other state-of-the-art approaches.

5.1. iRoads Dataset

Table 1 shows that the proposed xDNN method provides the best result in terms of classification accuracy as well as time/complexity and simplicity of the model structure (number of parameters/prototypes). The number of model parameters for xDNN (and DRB) is, strictly speaking, zero, because the 2 parameters (mean, μ and standard deviation, σ) per prototype (*data cloud*) are derived from the data and are not algorithmic parameters or user-defined parameters. For kNN method one can argue that the number of parameters is the number of data samples, N. The proposed explainable DNN surpasses in terms of accuracy the state-of-the-art VGG–16 algorithm which is a well-established convolutional deep neural network. Moreover, the proposed xDNN has at its top layer a set of a very small number of *MegaClouds* (27 or, on average, 4 *MegaClouds* per class) which makes it very easy to explain and visualize. For comparison, our earlier version of deep rule-based models, called DRB (Angelov and Gu, 2018) also produced a high accuracy and was trained a bit faster, but ended up with 521 prototypes (on average 75 prototypes per class) (Soares et al., 2019). With xDNN we do generate meaningful $IF...THEN$ rules as well as generate an analytical description of the *typicality* which is the empirically derived pdf in a closed form which lends itself for further analysis and processing.

Table 1: Performance Comparasion: iRoads Dataset

Method	*Accuracy*	Time(s)	# Parameters
xDNN	**99.59%**	4.32	**27**
VGG–16 (He et al., 2016)	99.51 %	836.28	Not reported
DRB (Angelov and Gu, 2019)	99.02%	**2.95**	521
SVM (Suykens and Vandewalle, 1999)	94.17%	5.67	Not reported
KNN (Bishop, 2006)	93.49%	4.43	4656
Naive Bayes (Bishop, 2006)	88.35%	5.31	Not reported

MegaClouds generated by the proposed xDNN model can be visualized in terms of rules as illustrated by the Fig. 10.

IF (I ∼ ) OR

(I ∼ ) OR

OR (I ∼ )

THEN ‘Daylight scene’

Figure 8: xDNN rule generated for the ‘Daylight scene’.

Voronoi tesselation can also be used to visualize the resulting *MegaClouds* as illustrated by Figure 9.

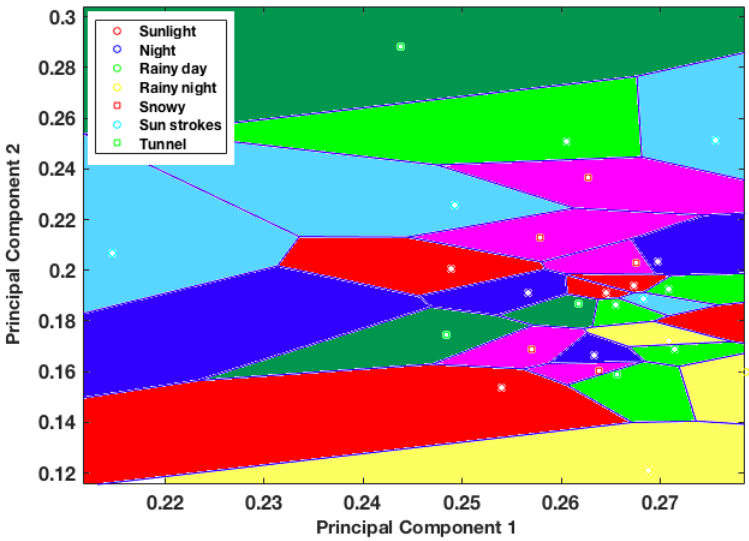

Figure 9: *MegaClouds* for the iRoads dataset.

5.2. Caltech-256 and Caltech-101 Dataset

Results for Caltech-256 are presented in Table 2.

Table 2: Performance Comparasion: Caltech-256 Dataset

Method	*Accuracy*
xDNN	**75.41%**
MSVM (Cao et al., 2019)	70.18%
VGG–16 (He et al., 2016)	73.2%
VGG–19 (He et al., 2016)	70.62 %
ResNet–101 (Simonyan and Zisserman, 2014)	75.14 %
GoogLeNet (Szegedy et al., 2015)	72.42 %
Softmax(7) (Zeiler and Fergus, 2014)	74.2%

Results presented in Table 2 demonstrate that the proposed xDNN approach can obtain highly accurate results compared to state-of-the-art approaches for this complex problem, it is important to highlight that we just compared the proposed approach with DNNs that do not use any trick for image augmentation. The proposed approach offers explainable models which can be visualized in terms of $IF...THEN$ rules. xDNN produced on average 3 *MegaClouds* per class (a total of 721) which are clearly explainable. Rules have the following format:

IF (x ∼ ) OR (x ∼ ) OR (x ∼ )

THEN 'CD'

We also tested the proposed xDNN approach on the Caltech-101 dataset. Results for the Caltech-101 dataset demonstrated on Table 3 showed that the proposed approach could surpass other state-of-the-art approaches in terms of accuracy.

Table 3: Performance Comparison: Caltech-101 Dataset

Method	*Accuracy*
xDNN	**94.31%**
SPP–net (He et al., 2015)	91.44%
ResNet–50 (He et al., 2016)	90.39%
CNN S TUNE-CLS (Chatfield et al., 2014)	88.35%
(Zeiler and Fergus, 2014)	86.5%
VGG–16 (He et al., 2016)	90.32%
KNN (Bishop, 2006)	85.65%
DT (Quinlan, 1986)	54.42%

We compared the proposed xDNN approach with the best published single-label classifiers methods and achieved better result. There are couple of alter-

native methods that report higher results on Caltech problems, but they use additional information such as the context (Leng et al., 2019) or multiple labels (Qian et al., 2019) processes in order to enhance the classification performance, include extra features (labels and descriptions) and this makes the underlying problem different even if the name is still the same (Caltech-101 or Caltech-256). We believe that the comparison has to be in the same playing field using the same amount of information and therefore, we do not report these methods. Apart from them, to the best of our knowledge, there is no better result achieved on Caltech data sets.

5.3. COVID CT-scan dataset

In this section we report the results obtained by the proposed xDNN classification approach when applied to the COVID CT-scan dataset (Zhao et al., 2020). Results presented in Table 4 compare the proposed algorithm with other state-of-the-art approaches, including traditional *"black-box"* Deep Neural Network, Support Vector Machines, etc.

Table 4: Performance Comparison: COVID CT-scan Dataset

Method \ Metric	Accuracy	Precision	Recall	F1 Score	AUC
xDNN	**88.6%**	89.7%	**88.6%**	**89.2%**	**88.6%**
Baseline (Zhao et al., 2020)	84.7%	**97.0%**	76.2%	85.3%	82.4%
SVM (Suykens and Vandewalle, 1999)	80.5%	84.4%	83.5%	84%	79.7%
KNN (Bishop, 2006)	83.9%	90.4%	82.4%	86.2%	84.3%
AdaBoost (Hastie et al., 2009)	83.9%	87.7%	83.5%	85.5%	84%
Naive Bayes (Bishop, 2006)	70.5%	77%	73.6%	75.3%	69.6%

The proposed xDNN classifier provided better results in terms of accuracy, recall, $F1$ score, and AUC. Moreover, the proposed approach also provided highly interpretable results that may be helpful for specialists (in this case, med-

ical doctors). The proposed classifier identified 30 prototypes for non-COVID and 33 prototypes for COVID partients. Rules generated by the identified prototypes for COVID and non-COVID patients are illustrated by Figures 10 and 11 respectively. The baseline approach Zhao et al. (2020) is a Deep Neural Network approach which is 'black box' (offers no interpretability).

Using the proposed method we extracted form the data linguistic *IF...THEN* rules which involve actual images of both cases (COVID-19 and non-COVID) as illustrated in Figures 10 and 11. Such transparent rules can be used in the decision-making process for early diagnostics for COVID-19 infection. Rapid detection with high sensitivity of viral infection may allow better control of the viral spread. Early diagnosis of COVID-19 is crucial for the disease treatment and control.

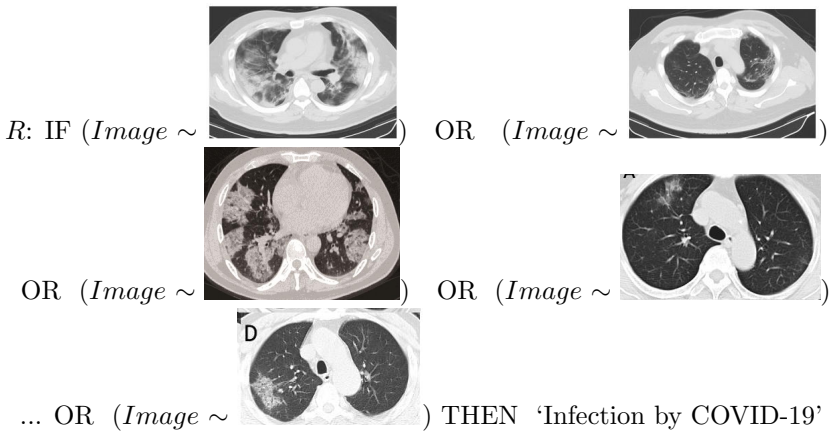

Figure 10: Final rule given by the proposed xDNN classifier for the COVID-19 identification. Differently from 'black box' approaches as deep neural networks, the proposed approach provides highly interpretable rules which can be used by human experts for the early evaluation of patients suspected of SARS-Cov-2 infection.

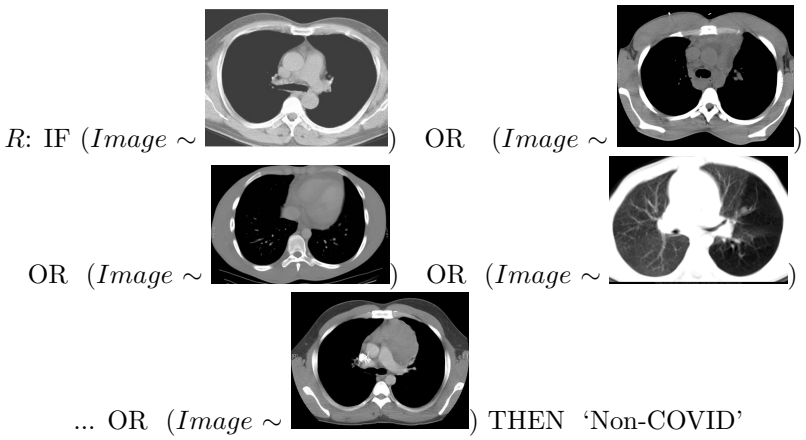

Figure 11: Non-Covid final rule given by the proposed eXplainable Deep Learning classifier.

Figure 12 illustrates the evolving nature of the proposed approach. xDNN is able to continuously learn as new data is presented to it. Therefore, no full re-training is required due to its life-long learning architecture. On the contrary, the baseline approach Zhao et al. (2020) is based on a Deep Neural Network that requires full re-training for any new data sample, which can be very costly in terms of time, computational complexity and requirements for hardware and computer experts. xDNN continuously learns as new training data arrives to the system. It can be observed that with 478 training data samples the proposed approach could obtain better results in terms of accuracy (84.56%) than the baseline approach (84.0%) with 537 training data samplesZhao et al. (2020). The baseline approach is a Deep Neural Network that needs a large number of training data to obtain a high performance in terms of classification accuracy and once trained can not be further improved unless fully re-trained. In contrast, the proposed approach can obtain higher performance using less training data due to its prototype-based nature.

Experiments have demonstrated that the proposed xDNN approach is able to produce highly accurate results surpassing state-of-the-art methods for different challenging datasets. Moreover, xDNN presents highly interpretable results that can be presented in the form of $IF...THEN$ logical rules, Voronoi tessella-

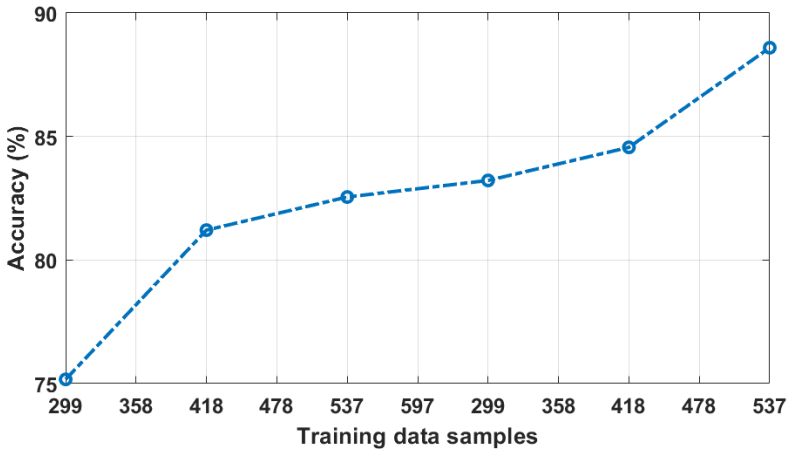

Figure 12: The figure illustrates the evolving nature of the proposed xDNN approach

tions, and/or *typicality* (empirically derived form of pdf) in a closed analytical form allowing further analysis. Because of its recursive, non-iterative and non-parametric form it allows computationally very efficient implementations to be realized.

6. Conclusion

In this paper we propose a new method, explainable deep neural network (xDNN), that is directly addressing the bottlenecks of the traditional deep learning approaches and offers an explainable internal architecture that can outperform the existing methods. The proposed xDNN approach requires very little computational resources (no need for GPUs) and short training times (in the order of seconds). The proposed approach, xDNN is prototype-based. Prototypes are actual training data samples (images), which have local peaks of the empirical data distribution called *typicality* as well as of the data density. This generative model is identified in a closed form and equates to the pdf but is derived automatically and entirely from the training data with no user- or problem-specific thresholds, parameters or intervention. The proposed xDNN offers a new deep learning architecture that combines reasoning and learning in

a synergy. It is non-iterative and non-parametric, which explains its efficiency in terms of time and computational resources. From the user perspective, the proposed approach is clearly understandable to human users. Results for some well-known benchmark data sets such as iRoads, Caltech-256, Caltech-101, and COVID CT-scan show that xDNN outperforms the other methods including state-of-the-art deep learning approaches in terms of accuracy, time to train and offers an explainable classifier. Future research will concentrate on the development of a tree-based architecture, synthetic data generation, and local optimization in order to improve the proposed deep explainable approach.

References

P. Angelov. *Autonomous learning systems: from data streams to knowledge in real-time*. John Wiley & Sons, 2012.

P. P. Angelov and X. Gu. Deep rule-based classifier with human-level performance and characteristics. *Information Sciences*, 463:196–213, 2018.

P. P. Angelov and X. Gu. *Empirical approach to machine learning*. Springer, 2019.

P. P. Angelov, X. Gu, and J. C. Príncipe. A generalized methodology for data analysis. *IEEE transactions on cybernetics*, 48(10):2981–2993, 2017.

M. Biehl, B. Hammer, and T. Villmann. Distance measures for prototype based classification. In *International Workshop on Brain-Inspired Computing*, pages 100–116. Springer, 2013.

M. Biehl, B. Hammer, and T. Villmann. Prototype-based models in machine learning. *Wiley Interdisciplinary Reviews: Cognitive Science*, 7(2):92–111, 2016.

C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

J. Cao, M. Wang, Y. Li, and Q. Zhang. Improved support vector machine classification algorithm based on adaptive feature weight updating in the hadoop cluster environment. *PloS one*, 14(4), 2019.

K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004.

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

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. 2007.

T. Hastie, S. Rosset, J. Zhu, and H. Zou. Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360, 2009.

K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

J. Hu, J. Lu, and Y.-P. Tan. Deep transfer metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 325–333, 2015.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

J. Leng, Y. Liu, and S. Chen. Context-aware attention network for image recognition. *Neural Computing and Applications*, 31(12):9295–9305, 2019.

O. Li, H. Liu, C. Chen, and C. Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

C. Liu, G. Bellec, B. Vogginger, D. Kappel, J. Partzsch, F. Neumärker, S. Höppner, W. Maass, S. B. Furber, R. Legenstein, et al. Memory-efficient deep learning on a spinnaker 2 prototype. *Frontiers in neuroscience*, 12:840, 2018.

K. Mizuno, Y. Terachi, K. Takagi, S. Izumi, H. Kawaguchi, and M. Yoshimoto. Architectural study of hog feature extraction processor for real-time object detection. In *2012 IEEE Workshop on Signal Processing Systems*, pages 197–202. IEEE, 2012.

D. Nebel, M. Kaden, A. Villmann, and T. Villmann. Types of (dis-) similarities and adaptive mixtures thereof for improved classification learning. *Neurocomputing*, 268:42–54, 2017.

O. K. Oyedotun and A. Khashman. Prototype-incorporated emotional neural network. *IEEE transactions on neural networks and learning systems*, 29(8): 3560–3572, 2017.

G. Qian, L. Zhang, and Y. Wang. Single-label and multi-label conceptor classifiers in pre-trained neural networks. *Neural Computing and Applications*, 31 (10):6179–6188, 2019.

J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun. Object detection networks on convolutional feature maps. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1476–1481, 2016.

M. Rezaei and M. Terauchi. Vehicle detection based on multi-feature clues and Dempster-Shafer fusion theory. In *Pacific-Rim Symposium on Image and Video Technology*, pages 60–72. Springer, 2013.

C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

S. Saralajew, L. Holdijk, M. Rees, and T. Villmann. Prototype-based neural network layers: incorporating vector quantization. *arXiv preprint arXiv:1812.01214*, 2018.

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

T. J. Sejnowski. *The deep learning revolution*. MIT Press, 2018.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

E. Soares and P. Angelov. Novelty detection and learning from extremely weak supervision. *arXiv preprint arXiv:1911.00616*, 2019.

E. Soares, P. Angelov, B. Costa, and M. Castro. Actively semi-supervised deep rule-based classifier applied to adverse driving scenarios. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

B. Solmaz, S. M. Assari, and M. Shah. Classifying web videos using a global video descriptor. *Machine vision and applications*, 24(7):1473–1485, 2013.

J. A. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke. The microsoft 2017 conversational speech recognition system. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5934–5938. IEEE, 2018.

M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

J. Zhao, Y. Zhang, X. He, and P. Xie. Covid-ct-dataset: a ct scan dataset about covid-19. *arXiv preprint arXiv:2003.13865*, 2020.

F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He. Supervised representation learning: Transfer learning with deep autoencoders. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.