Explainable-by-Design Deep Learning

Eduardo Almeida Soares, BSc, MSc
School of Computing and Communications
Lancaster University

A thesis submitted for the degree of
Doctor of Philosophy

August, 2022

I dedicate this thesis to my beloved spouse Sarah Biaso, my precious daughter Giovanna T. B. Soares, my uncle Anderson de Almeida (*in memoriam*), and my father José Paulino Soares (*in memoriam*).

Declaration

I declare that the work presented in this thesis is, to the best of my knowledge and belief, original and my own work. The material has not been submitted, either in whole or in part, for a degree at this, or any other university.
Eduardo Almeida Soares

Explainable-by-Design Deep Learning
Eduardo Almeida Soares, BSc, MSc.
School of Computing and Communications, Lancaster University
A thesis submitted for the degree of *Doctor of Philosophy.* August, 2022

Abstract

Machine learning, and more specifically, deep learning, have attracted the attention of media and the broader public in the last decade due to its potential to revolutionize industries, public services, and society. Deep learning achieved or even surpassed human experts' performance in terms of accuracy for different challenging problems such as image recognition, speech, and language translation. However, deep learning models are often characterized as a "black box" as these models are composed of many millions of parameters, which are extremely difficult to interpret by specialists. Complex "black box" models can easily fool users unable to inspect the algorithm's decision, which can lead to dangerous or catastrophic events. Therefore, auditable explainable AI approaches are crucial for developing safe systems, complying with regulations, and accepting this new technology within society. This thesis tries to answer the following research question: Is it possible to provide an approach that has a performance compared to a Deep Learning and the same time has a transparent structure (non-black box)? To this end, it introduces a novel framework of explainable-by-design Deep Learning architectures that offers transparency and high accuracy, helping humans understand why a particular machine decision has been reached and whether or not it is trustworthy. Moreover, the proposed prototype-based framework has a flexible structure that allows the unsupervised detection of new classes and situations. The approaches proposed in thesis have been applied to multiple use cases, including image classification, fairness, deep recursive learning interpretation, and novelty detection.

Acknowledgements

First and foremost I am extremely grateful to my supervisor, Professor Plamen Angelov, for his invaluable advice, continuous support, and patience during my PhD study. His experience and advises have encouraged me all the time in my academic research and daily life.

I would like to acknowledge Ford Motor Co. for the sponsorship, in special, Dr. Dimitar Filev, Dr. Marcos Gerardo, Dr. Subramanya Nageshrao, and Dr. Bruno Costa for all the kind support and guidance during this project.

I would like to thank Dr. Hossein Rahmani and Dr. Antonios Gouglidis, for their thoughts and contributions during the appraisal panels, and I would also like to thank Lancaster University for all the support during these years.

In addition, I would like to thank my beloved spouse Sarah Biaso, my daughter Giovanna T. B. Soares, and my family, Andrea de Almeida, Maria Aparecida da Silva, Gustavo Soares, and José Maria Soares for their invaluable support during these years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter outlines the research motivation and the summary of the research contributions and methodology. This chapter is organised as follows. Section 1.1 presents the research motivation. The aims and objectives of thesis are given by section 1.2. Methodology is given by section 1.3. The research contribution is presented on section 1.4, and the chapter is finished by thesis outline (section 1.5).

1.1 Motivation

Machine learning, more specifically deep learning (LeCun et al., 2015), attracted the attention of media and wider public (Sejnowski, 2018) in the last decade due their potential to revolutionize industries (Dean et al., 2018), public services (Li et al., 2018a), and society (Chen and Lin, 2014). Deep learning achieved or even surpassed specialists performance in terms of accuracy for different challenging problems such as lung cancer detection (Ardila et al., 2019) and voice recognition (Assael et al., 2016).

Deep learnings have also proven efficient in automating the pre-processing stage of feature extraction (Dara and Tumma, 2018), which can be human laborious. Despite the great success of deep learnings, the main criticisms towards them is related to their 'black-box' nature (Rudin, 2019) and requirements for huge amount of labeled data, computational resources, and long times of training (Marcus, 2018). In fact, traditional deep learning approaches can have millions of parameters that are extremely difficult to interpret as they are abstract and detached from the physical nature of the problem being modelled (Castelvecchi, 2016).

Interpretability and explainability are extremely important for high stake applications, such as autonomous cars, medical or court decisions, etc (Doshi-Velez and Kim, 2017). The absence of transparent and auditable models can produce severe consequences; there have been cases of accidents involving autonomous vehicles (Favarò et al., 2017), unfair decisions over specific groups on problems such as recidivism prediction (Dressel and Farid, 2018), excluded individuals into credit markets (Johnson et al., 2019), and bias in hiring processes (Raub, 2018).

Due the critical aspect of black-box models, the right of explanations is extremely important for the acceptance of artificial intelligence within society, but also for regulatory purposes. In 2019, the US Congress passed the Algorithmic Accountability Act (MacCarthy, 2019) and the EU enshrined the right for an explanation to the consumer (Goodman and Flaxman, 2017; Pedreschi et al., 2019).

Given the circumstances, there has been an exponential increase in interest towards "Explainable AI". However, the majority of the work in this field refers to *post hoc* approaches, where a surrogate model is created to explain the black box model (Rudin and Radin, 2019). However, this can be controversial because *post hoc* explanations are often not reliable as different approaches can provide different explanations for the same model (Angelov and Soares, 2020b; Rudin, 2019).

Therefore, explainable-by-nature algorithms are necessary to provide the desired transparency and trustworthy to AI systems. The focus of this thesis is to provide a novel set of explainable-by-design deep learning architectures that combines reasoning and learning in a synergy. The new methods presented in this thesis offers:

- Prototype-based structures that favors explainable-by-design algorithms;
- Highly transparent and interpretable models derived from data;
- Ability to detect unseen/new data patterns autonomously and learn from them;
- No user- or problem- specific algorithmic parameter necessity.

The research work and methods presented in this thesis have been developed for the project "Transparent Deep Learning Classifier of Driving Scenarios able to Identify and Learn from Unseen Situations" funded by Ford Motor Co. (Palo Alto - USA). The new approaches presented in this thesis are an attractive alternative for state-of-the-art methods.

1.2 Aims and Objectives

This research thesis is focused on the proposal of new explainable machine learning algorithms that allow users to inspect the algorithm's decision through the provision of different levels of explanations. The algorithms presented in this thesis are tested on different benchmark and real applications, including real cases provided by Ford Motor Co.. This thesis has as objectives to present:

1. Theoretical concepts that justify the explainable methods proposed in this research;

2. Algorithms conception and implementation;
3. Testing of the proposed algorithms on different real and benchmark problems.

A theorical research is presented to demonstrate the mathematical and analytical description of the proposed methodologies. Then, the algorithm implementation demonstrates the practical feasibility of the theoretical concepts.

The last part, is constituted by the application and testing of the implemented theoretical concepts on benchmark and real case problems in order to evaluate the efficacy of the proposed approaches.

1.3 Methodology

In order to answer the following research question: "Is it possible to provide an approach that has high performance compared to other machine learning algorithms and the same time has a transparent structure?", different explainable-by-design algorithms are proposed and their potential effectiveness are tested on different real and benchmark applications.

The base algorithm proposed in this thesis, explainable-by-design deep learning (xDNN), is presented on Chapter 3. xDNN is a prototype-based algorithm that uses the similarity concept in order to make its classifications. Therefore, it presents a transparent flexible structure that can be very in interesting in many applications. Experiments were designed to reveal interesting insights in terms of numerical results and interpretability of the proposed method. Furthermore, as described in the research contributions section, the base algorithm is extended to approach problems of different natures.

Therefore, this research thesis adopts an experimental research methodology that uses an iterative process to improve and refine the base algorithm for problems of different natures, as the interpretation of deep neural networks and also novelty detection. The research presented in this thesis is also quantitative as it considers real-world and benchmark data to assess the efficiency of the proposed explainable deep learning model and its variants. The quantitative analysis allows the proposed approaches to be compared with techniques used in state-of-the art literature.

1.4 Research Contributions

This research work focuses on the proposal of novel explainable-by-design deep learning systems. The following main contributions have been achieved during the research:

1. The base method proposed in this thesis is a prototype-based deep explainable architecture, namely xDNN, that can outperform the existing methods and also present a transparent decision structure. It requires very little computational resources and short training times. The proposed method has been tested on benchmarking and real datasets, including a proprietary COVID-19 CT-scans dataset. Our contribution in this area has been published in (Angelov and Soares, 2020b), (Soares et al., 2020b), and (Soares et al., 2020a).

2. A novel prototype-based method and network architecture for deep learning which extends the basic method with the proposal of a decision tree-based inference and synthetic data to balance the classes. This approach have been tested with various challenging benchmark datasets and have demonstrated high results in terms of accuracy, even surpassing traditional deep learning approaches. This contribution has been published in (Angelov and Soares, 2020a).

3. A novel model based on prototypes that maps the continuous multidimensional state space characterizing vehicle positions and velocities to a discrete set of actions in longitudinal and lateral direction. The model is obtained through the approximation of a customized version of the Double Deep Q-Network (DDQN) learning algorithm with a set of $IF...THEN$ rules that provide an alternative interpretability model, and is further enhanced by visualizing the rules. This contribution has been published in (Soares et al., 2020c).

4. An extension of the base method that allows unsupervised detection of new classes, and learning from few labeled data samples. The algorithm was tested on different challenging problems, including real adversarial autonomous cars scenarios classification. The method not only presented higher accuracy compared to state-of-the-art algorithms, but, more significantly, the algorithm used information of just a single class at the start of the learning process and few labeled data samples (few-shot learning). Our contribution to this field has been published in (Soares et al., 2019a) and (Angelov and Soares, 2021).

1.5 Thesis Outline

The remainder of the thesis is organised as follows.

Chapter 2 - Background and Related Work: contains the review of the state-of-the-art works that provides the background and the theoretical part for the research presented in the thesis. It includes review of the main approaches and applications in the field of artificial intelligence and explainable AI.

Chapter 3 - eXplainable-by-design Deep Learning: it presents the main concepts, mechanisms, and architecture of the proposed eXplainable-by-design Deep Learning (xDNN) approach.

Chapter 4 - Deep Machine Reasoning: it presents a prototype-based explainable deep learning with a decision tree inference and balanced amount of prototypes per class. The method uses a decision tree to determine the winning class label, and an internal mechanism to balance the classes by synthesising data around the prototypes.

Chapter 5 - Explaining Deep Learning Through Rules: it presents an explainable approach to redesign a Deep Reinforcement Learning (DRL) model into a set of $IF...THEN$ rules. The method is designed to provide an approximation (mimic) of the DRL model with similar performance.

Chapter 6 - Detecting and Learning from Unknown: it presents an extremely weakly supervised approach (xClass) which is able to learn new patterns from the data using just a single class and using a very small set of labeled data samples in its training phase.

Chapter 7 - Conclusion and Future Work: it presents a summary of the thesis and provides directions for further work.

Chapter 2

Background and Related Work

In this chapter, a review of the main concepts discussed in this thesis is introduced. The review includes the description of the most relevant work and applications of such concepts. Moreover, this chapter is also dedicated to present the shortcomings of current approaches contrasting with the concept of explainable AI.

2.1 Artificial Intelligence and Machine Learning

Artificial intelligence (AI) and Machine Learning (ML) are rapidly transforming every aspect of society as they reshape industries, public services, and society. The recent advances on AI, allowed such technique to achieve or even surpass human levels of performance for a range of problems, from six-player poker (Blair and Saffidine, 2019) to complex problems such as voice generation and recognition (Oord et al., 2016), and radiology (Hosny et al., 2018).

These recent advances in AI and ML research are intimately linked with the availability of massive amounts of data, and modern, powerful computational hardware (Alpaydin, 2016).

However, traditional ML techniques are limited by their design to deal with natural data in their raw form (Zhou et al., 2017). Therefore, a careful feature engineering process and considerable domain expertise is considered to construct an efficient machine learning system that could detect or classify patterns in the input (O'Mahony et al., 2019).

A great challenge in many real-world machine learning applications is that many elements can influence every single piece of data we are able to observe. As stated by Goodfellow et al. (2016), individual pixels of the image of a red car can be very close to black during the night. Moreover, the shape of the car also depends on the viewing angle.

Many of these variation elements that can influence the data can be identified only by a complex understanding of the data itself. Deep learning solves this problem by introducing representations that are expressed in terms of other, simpler representations. In this sense, deep learning system can represent the concept of an image of a person by combining simpler concepts, such as corners and contours, which are in turn defined in terms of edges.

The next session presents the concepts, as well as, the main applications of the main AI approaches.

2.2 Machine Learning Methods for Classification

In machine learning, classification is the task of categorising sets of data into different classes (Mohri et al., 2018). Classification methods are generally based on the concepts of supervised or semi-supervised. In this section, the main approaches used for the classification task are reviewed. It includes the following methods:

- K-nearest neighbour (k-NN),
- Support Vector Machine (SVM),
- Decision Tree (DT),
- Random Forest (RF).

2.2.1 K-nearest neighbour (k-NN)

The K-nearest neighbour (k-NN) algorithm is a non-parametric methods that does not make any *prior* assumptions about the dataset (Taunk et al., 2019). k-NN, is known for its simplicity and effectiveness (Kramer, 2013). The k-NN algorithm makes its classification decision of its k nearest neighbours based on the voting mechanism as follows (Cover and Hart, 1967):

$$y(x_0) = \underset{x \in \{x\}_K}{argmin}(d(x_0, x)) \tag{2.1}$$

where y is the predicted label of data sample x_0, and $x_0 = \underset{x \in \{x\}_K}{min}(d(x_0, x))$, and d is a distance metric. Fig. 2.1 illustrates the k-NN classification decision.

The k-NN classifier has demonstrated satisfactory results in different areas including medical data classification (Xing and Bei, 2019), traffic incident detection (Xiao, 2019), facial expression classification (Dino and Abdulrazzaq, 2019), etc.. However, the k-NN algorithm is highly dependent on the best choice of k to obtain high performance.

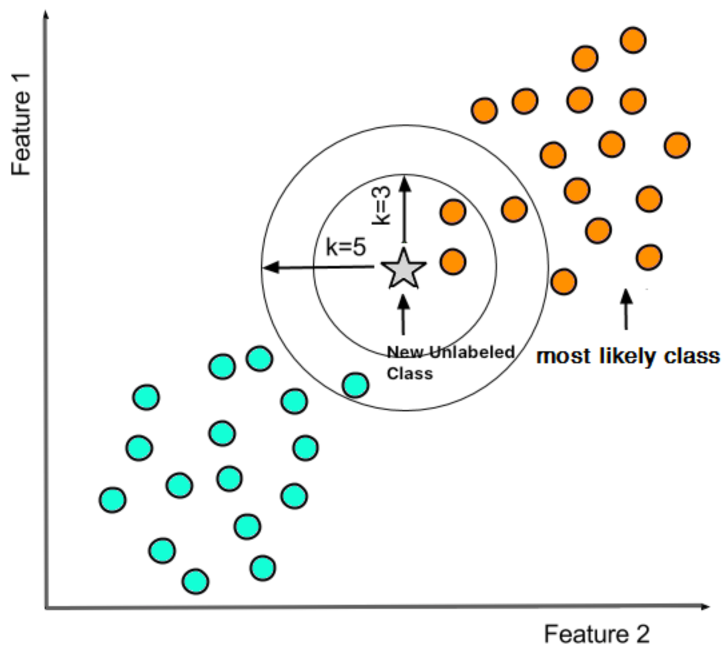


Figure 2.1: k-NN algorithm decision.

Moreover, the k-NN algorithm is also high sensitive to outliers, unbalanced datasets, and other data problems that can direct impact the performance of the algorithm (Prasatha et al., 2017).

2.2.2 Support Vector Machine (SVM)

Firstly introduced by Cortes and Vapnik (1995), the Support Vector Machine (SVM) is a supervised algorithm that maximizes a mathematical function of a given dataset. In this sense, the SVM algorithm selects from the training samples a set of features that the classification is similar to the division of the entire collection of data (Noble, 2006). SVM has been successfully applied to wide range of problems as credit card fraud identification (Bhattacharyya et al., 2011), human action recognition (Schuldt et al., 2004), and sentiment classification (Moraes et al., 2013). The SVM method is composed by four basic concepts: (i) hyperplane separation, (ii) maximum-margin hyperplane, (iii) soft margin and (iv) the kernel function.

2.2.2.1 Hyperplane separation

If two classes are linearly separable, it is possible to obtain a straight line that separates these two classes (Noble, 2006). This procedure can be mathematically extrapolated to higher dimensions. In a high-dimensional space is a straight line is a hyperplane. These points are located on a hyperplane that satisfies the following conditions:

$$x^T w + b = 0 \tag{2.2}$$

where x is a data sample, e w is a vector that is perpendicular to the separating hyperplane, and b denotes the bias value.

2.2.2.2 Maximum-magin hyperplane

The concept of hyperplanes to classify data samples in a high-dimensional space is not unique to the SVM. However, the method used by SVM to define the hyperplane is what differs this algorithm from others (Suthaharan, 2016). SVM uses the maximum-margin hyperplane technique to select the hyperplane (Bhavsar and Panchal, 2012; Noble, 2006). The maximum-margin hyperplane selects the hyperplane based on the maximum distance between it and the nearest data point as follows Cortes and Vapnik (1995):

$$\begin{aligned} x^T w + b &\geq +1 \; if \; y_1 = +1 \\ x^T w + b &\leq -1 \; if \; y_1 = -1 \end{aligned} \tag{2.3}$$

where the maximum margin can be found by minimising the parameter $||w||$. The correct selection of the maximum-margin hyperplane determines the success of performance of the SVM algorithm. The correct selection of it maximises the algorithm's ability of correctly classification of unseen data samples. Fig 2.2 illustrates the maximum-margin generated by the SVM algorithm.

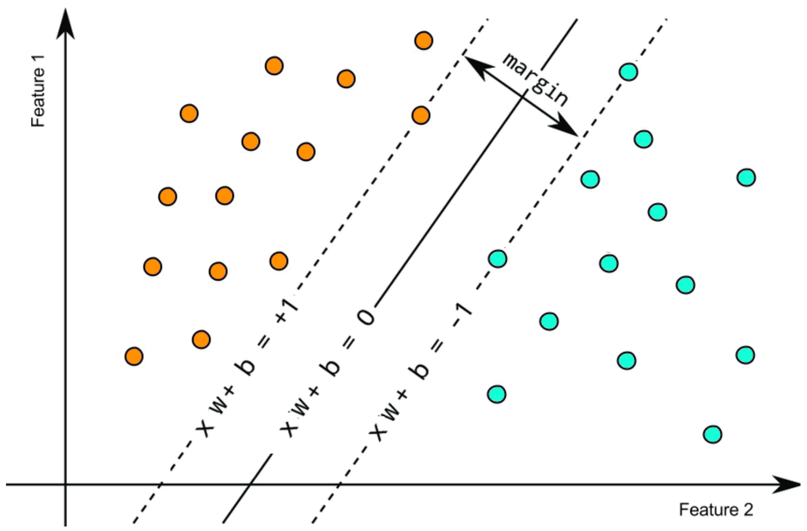


Figure 2.2: SVM maximum-margin illustration.

2.2.2.3 Soft-magin

Real datasets not always are linearly separable; therefore, soft margins are introduces a way to handle such cases (Noble, 2006). The SVM introduces a soft margin through slack variables $\phi_i (i = 1, 2, ..., K)$ where:

$$\begin{aligned} x_i^T w + b \geq 1 - \phi_i \ if \ y_1 = 1 \\ x_i^T w + b \leq 1 + \phi_i \ if \ y_1 = -1 \end{aligned} \tag{2.4}$$

where $\phi_i \geq 0 \ \forall_i$. This property allows SVM to deal with errors by allowing anomalies to be on the wrong side of the separating hyperplane. In other words, the soft-magin property allows some data points to push themselves through the separating hyperplane margin without affecting the algorithm's performance (Pradhan, 2012).

2.2.2.4 Kernel function

To find the maximum-margin separating hyperplane from nonlinear separable points is not possible (Noble, 2006). Even if soft margin (ibid.) is considered, this problem persists. Therefore, kernel functions can be applied as solution for this problem. Kernel functions adds an extra dimension to th-data space projecting it to higher dimensional space (Leslie et al., 2001). Kernel functions includes:

a)Polynomial:

$$K(x_i, x_j) = (x_i^T x_j + 1)^p \tag{2.5}$$

b) Hyperbolic tangent:

$$K(x_i, x_j) = tahn(kx_i^T x_j - \delta) \tag{2.6}$$

and others as Radial basis function (RBF) (Tan et al., 2011), Typicality and Eccentricity Data Analytics (TEDA) (Kangin and Angelov, 2015), Gaussian (Keerthi and Lin, 2003), etc..

Despite the great success of SVM on different applications, the SVM algorithms still presenting some drawbacks i) the necessity of prior knowledge for choosing the right kernel function (Jebara, 2004), ii) its performance may drop significantly on big data problems (Demidova et al., 2016), iii) performance may suffer on multi-class problems (Duan and Keerthi, 2005).

2.2.3 Decision Tree

Decision tree (DT) classifiers are supervised learning algorithms that maps input vectors in the data space to the output labels through an organised tree structure (Safavian and Landgrebe, 1991). The goal of the algorithm is the learning of simple decision rules inferred from prior training data. Fig. 2.3 illustrates a decision tree structure.

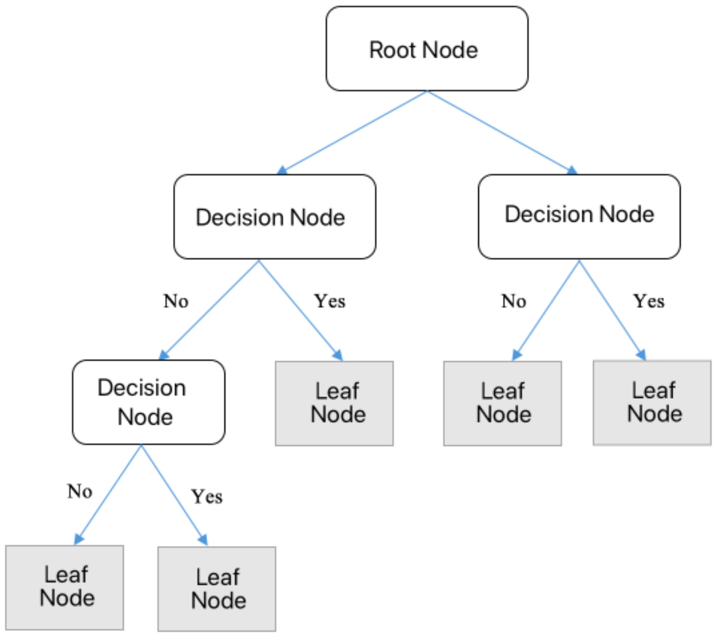


Figure 2.3: Decision tree structure.

The decision tree is composed by a recursive partitioning mechanism that splits the data into subsets until a decision criteria is reached (Swain and Hauska, 1977). Although the simple structure of decision trees, they have demonstrated excellent result on different problems as environmental applications (Lu and Ma, 2020), medical applications (Yoo et al., 2020), and others (Song and Ying, 2015). Decision trees may have performance problems when dealing with complex problems due their simple structure (Bramer, 2007), moreover, these are high sensitive to changes on data, as a simple change on the data can modify the whole structure of the tree (Lee and Siau, 2001).

2.2.4 Random Forest

Random forest (RF) is a supervised ensemble learning method for classification (Liaw, Wiener, et al., 2002). It extends the concept of decision tree to a multiple instances of decision trees (Oshiro et al., 2012). The random forest algorithm uses the bagging and feature randomness mechanism to build each individual tree that will create an uncorrelated forest of trees which is more accurate than one individual tree. Fig. 2.4 illustrates a generic random forest representation.

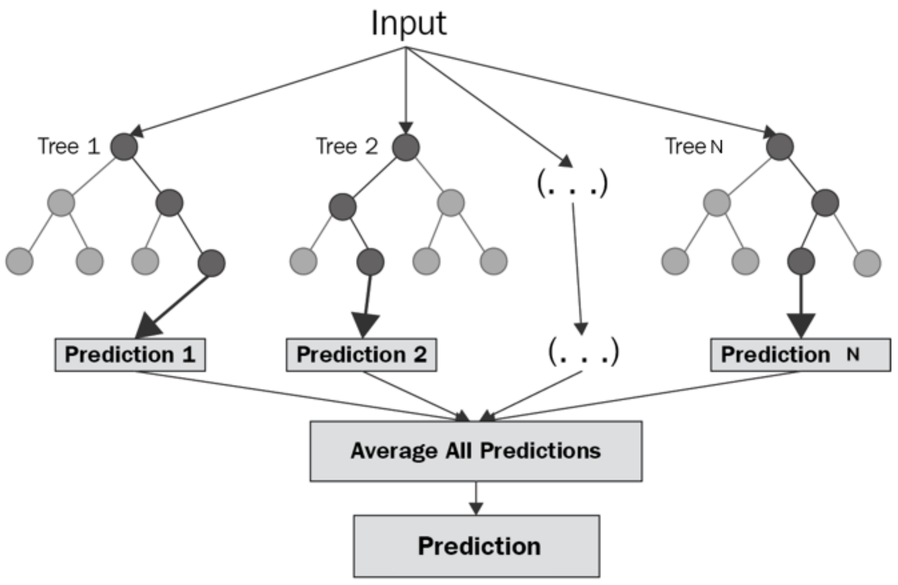


Figure 2.4: Random forest generic structure.

Random forest classifiers have been successfully applied to diverse problems as remote sensing (Pal, 2005), biomedicine (Azar et al., 2014), security (Alam and Vuong, 2013), and others (Zakariah et al., 2014). Gradient boosting algorithms as Catboost

(Dorogush et al., 2018), and XGBoost (Chen and Guestrin, 2016) also uses the concept of ensemble of decision trees. Random forest classifiers cannot be effective for real time problems due to its large number of trees that can make the model slow (Kulkarni and Sinha, 2012).

2.3 Fuzzy Sets and Rules

Firstly introduced by (Zadeh, 1965), fuzzy sets extend the concept of sets by assigning a value to each element of the reference set, which represents its degree of membership in the fuzzy set. Membership values correspond to the degree of similarity of a given element to an associated fuzzy set. The concept of fuzzy sets allows the management of the uncertainty carried by elements (Zadeh, 1983). In this section, a brief definition of the main fuzzy rules are provided, including: Zadeh-Mamdani type (Mamdani and Assilian, 1975), Takagi-Sugeno type (Takagi and Sugeno, 1985) and AnYa type (Angelov and Yager, 2011).

2.3.1 Fuzzy Rules

A Zadeh-Mamdani fuzzy rule type has the following format:

$$\begin{aligned} &IF\ (x_1\ is\ A_1)\ AND\ (x_2\ is\ A_2)\ AND\ ...\ AND\ (x_M\ is\ A_M) \\ &\qquad THEN\ y = L \end{aligned} \tag{2.7}$$

where, $x = [x_1, x_2, ..., x_M]^T$, A is reference value for the i-th fuzzy set, y is the output of the rule, and L a label. On the other hand, a Takagi-Sugeno fuzzy rule can be expressed in the following way:

$$\begin{aligned} &IF\ (x_1\ is\ A_1)\ AND\ (x_2\ is\ A_2)\ AND\ ...\ AND\ (x_M\ is\ A_M) \\ &\qquad THEN\ y = a_0 + a_1 * x_1 + a_2 * x_2 +\ ...\ + a_M * x_M \end{aligned} \tag{2.8}$$

where a is the $(M+1) \times 1$ dimensional parameterised vector of the i-th fuzzy rule for linear regression.

The Zadeh-Mamdani fuzzy rules and Takagi-Sugeno type fuzzy rules have the same antecedent part (IF). However, the consequent part ($THEN$) are different. While Zadeh-Mamdani has its consequent part a label, the Takagi-Sugeno rule has a linear regression.

Differently from the other two types (Zadeh-Mamdani and Takagi-Sugeno), the AnYa fuzzy rule removes the need membership functions definition per variable(ibid.). In addition, AnYa also eliminates the necessity for logical connectives such as AND/OR

to aggregate the scalar variables. Furthermore, AnYa uses data density (a parameter-free Cauchy type kernel) to derive the activation level of each rule. An AnYa fuzzy rule can be expressed in the following way:

$$IF\ (x \sim \pi)\ \ THEN\ y = L \tag{2.9}$$

where $\sim$ denotes the similarity degree which can also be seen as a fuzzy degree of membership (Angelov and Yager, 2011), π denotes a prototype for the i-th fuzzy rule. Similarly to the Takagi-Sugeno rule, AnYa fuzzy rule can also assume a 1^{st} type as expressed by:

$$IF\ (x \sim \pi)\ \ THEN\ y = a_0 + a_1 * x_1 + a_2 * x_2 +\ \ ...\ \ + a_M * x_M \tag{2.10}$$

Table 2.1 compares the three different types of fuzzy rules presented in this section:

Table 2.1: Fuzzy rules differences

Type	Antecedent(IF)	Consequent ($THEN$)	De-fuzzification
Zadeh-Mamdani	parameterised fuzzy sets	parameterised fuzzy sets	Central of gravity
Takagi-Sugeno	parameterised fuzzy sets	Functional	Fuzzily weighted sum
AnYa	Prototypes	parameterised fuzzy sets Functional	Winner-takes-all Fuzzily weighted sum

2.4 Artificial Neural Networks

Firstly introduced by McCulloch and Pitts (1943), artificial neural networks (ANN) have been conceived as generalizations of mathematical models of biological nervous systems. After, (Rosenblatt, 1958) introduced the Perceptron algorithm that allowed machines to perform simulated human behavior. The Perceptron architecture is illustrated in Fig. 2.5.
where,

$$output = w_1x_1 + ... + w_mx_m = \sum_{j=1}^{m} x_jw_j \tag{2.11}$$

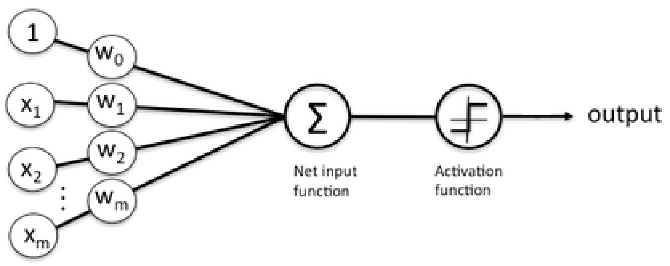


Figure 2.5: Simple Perceptron architecture.

where x are the inputs of the network, w are weights, and y is the output of the network. In this case the activation function f is given by:

$$f(y) = \begin{cases} 1 & \text{if } y \geq 0 \\ -1 & otherwise \end{cases} \tag{2.12}$$

In the context of multilayer neural networks different activation functions, such as ReLu, have been considered.

The basic processing element of a neural networks is named neuron. In this sense, on a artificial neural network the effects of the synapses are represented by connection weights (see Fig 2.5). The neuron impulse is computed as the weighted sum of the input signals (eq. 2.11), and transformed by the activation function. The adjustment of the weights influences the learning capability of the artificial neuron.

Different forms of ANNs architectures became widely popular including: i) Deep Neural networks (LeCun et al., 2015); ii) Recurrent Neural networks (Lipton et al., 2015); iii) Feedforward neural networks (Svozil et al., 1997); iv) Radial basis functions networks (Orr et al., 1996), etc.

Despite the great success of neural networks on different applications, the main criticisms towards them rely on the fact that they are "black-box", given the fact that it is extremely difficult to interpret their parameters and decisions.

2.5 Neuro-fuzzy Systems

Neuro-fuzzy systems uses the learning capability of neural networks to determine the membership function and fuzzy rules of the fuzzy logic systems (Campos Souza, 2020). Fig 2.6 illustrates a generic architecture for a neuro-fuzzy system.

Neural networks and fuzzy systems can work in synergy to join their advantages. Neural networks has better learning characteristics and the fuzzy systems provides interpretation to the "black-box" systems (Jang, 1993). Given these advantages,

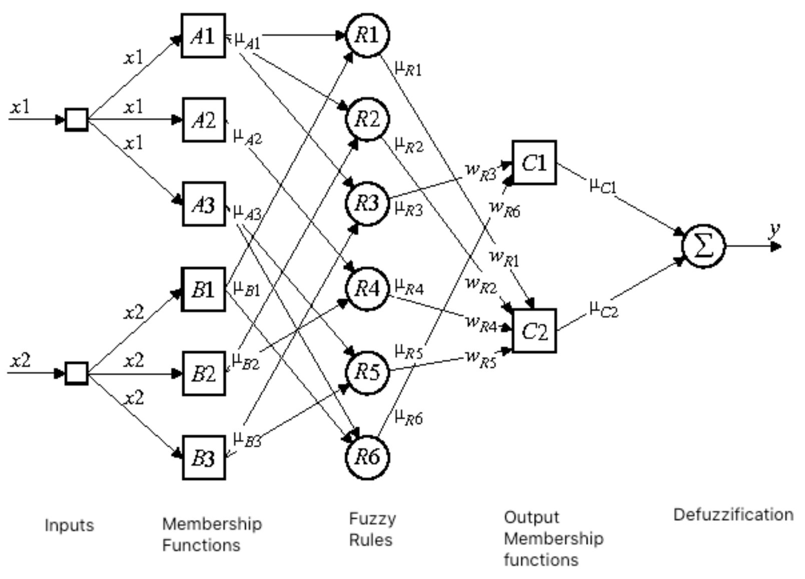


Figure 2.6: A generic architecture for Neuro-fuzzy.

different models have been designed proposed in this direction (Figueiredo and Gomide, 1999). Models have been proposed for different tasks such as control (Leite et al., 2013), dynamic time series prediction (Angelov and Filev, 2004; Kasabov and Song, 2002), missing data (Garcia et al., 2019), hurricane track prediction (Soares et al., 2018), and others.

2.6 Deep Learning

Deep learning is responsible for the major advances in the artificial intelligence community in the recent years (Deng and Yu, 2014). The deep learning architecture allows the discovering of complex structures in high-dimensional data (Nielsen, 2015). Therefore, this method has become popular in many domains of science, business and government (LeCun et al., 2015).

Deep learning has beaten human performance in image recognition (Pak and Kim, 2017) and speech recognition (Xiong et al., 2016). Moreover, it has proven to be an excellent approach for complex tasks such as autonomous driving (Grigorescu et al., 2020), drug discovery (Gawehn et al., 2016), and more recently on Covid-19 identification (Ting et al., 2020).

Deep learning approaches belong to the so-called group representation-learning, as they are equipped with systems to automatically discover the representations needed for feature detection (Bengio et al., 2013). Deep learnings are composed of multiple levels of representation, which are obtained by non-linear modules that transforms the representation at one level into a representation at a higher abstract level. Therefore, very complex functions can be learned with enough number of transformations (Zhong et al., 2016).

For example, an image comes to the deep learning system in the form of pixel values, and the learned features in the first layer of representation generally represents the presence or absence of edges at particular orientations and locations in the image. The second layer is responsible to detect patterns and specific disposition of edges. The third layer, arranges the learned patterns into larger clusters that correspond to parts of similiar objects. Fig. 2.7 illustrates the feature identification process aforementioned.

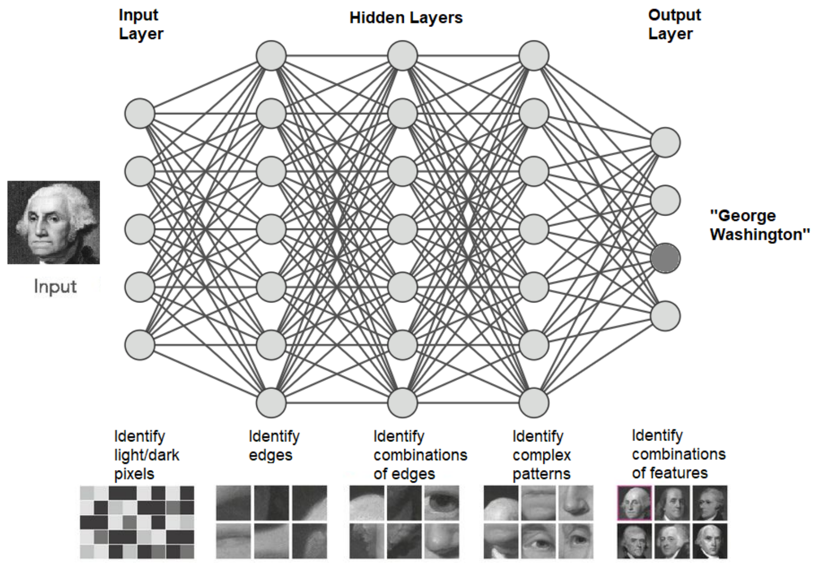

Input Layer
Hidden Layers
Output Layer
Input
"George Washington"
Identify light/dark pixels
Identify edges
Identify combinations of edges
Identify complex patterns
Identify combinations of features

Figure 2.7: Deep learning feature identification process.

The main difference from deep learning to traditional approaches is that the layers of features that compose them are not designed by human experts. Instead, deep learning is composed by a generic learning procedure that learns features from data (data-oriented) (LeCun et al., 2015).

2.6.1 Convolutional Neural Networks

Convolutional neural networks (CNN), are a special case of deep learning designed design specifically to deal with images (ibid.). This allows CNNs to encode encode image-specific features into their architecture (O'Shea and Nash, 2015). CNNs were firstly introduced by Fukushima and Miyake (1982) in 80s, and significantly improved during the 90s by LeCun et al. (1989).

The generic architecture of a CNN is illustrated by Fig. 2.8. According to LeCun et al. (2015), there are four central ideas that CNNs take advantage: local connections, shared weights, pooling and the use of many layers.

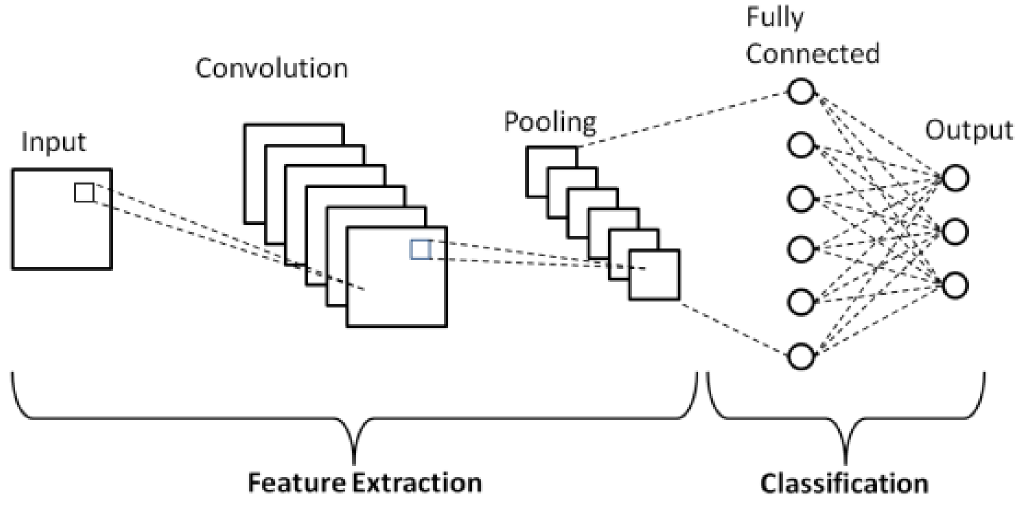


Figure 2.8: CNN generic architecture.

The first stages of a CNN are the convolutional layers and pooling layers.

Convolutional layers are structured in feature maps. Each unit of the convolutional layer is connected to local patches in the feature maps of the previous layer through a set of weights.

All units in a feature map share the same set of weights. Different feature maps in a specific layer use different set of weights. The reason for this design, is that if a pattern can appear in one part of the image, it could also appear in other part. Therefore, this the reason of units at different locations sharing the same weights and detecting the same pattern (Voulodimos et al., 2018).

The function of the pooling layer is to merge semantically similar features into one. It computes the maximum of a local patch of units in one feature map or few feature maps (Schmidhuber, 2015). The convolutional and pooling layers are inspired by the notion of simple cells and complex cells in visual neuroscience (LeCun et al., 2015).

The extremely accurate results provided by CNNs have attracted investments of the so called big-tech companies including Facebook, Microsoft, Google, and Apple. The belief is that this technology will be the next game changer in the near future (Zhang et al., 2020).

2.6.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are powerful dynamic systems that process an input sequence at time (Schuster and Paliwal, 1997). It maintains in their hidden layers a memory that contains historical information about the elements of the sequence (Medsker and Jain, 2001). Then, the outputs of the hidden units are given for different discrete time steps. Fig 2.9 illustrates the operation of a recurrent node unfolded along the input sequence for three steps, where h is the hidden state, x is the input, y is the output of the network, and w are the weights.

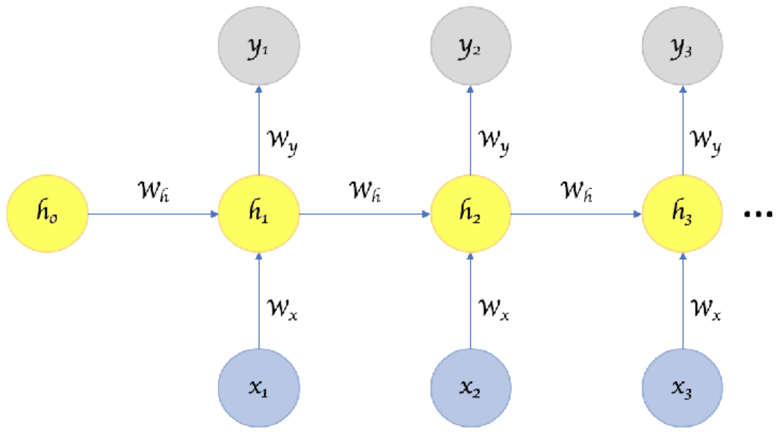


Figure 2.9: RNN generic architecture.

The algorithmic and hardware advances allowed RNNs to be widely used for different tasks (Tarwani and Edem, 2017). RNNs have been proven to be very efficient to deal with natural language processing (Yin et al., 2017), sentiment analysis (Tang et al., 2015), image generation (Gregor et al., 2015), and speech recognition (Graves et al., 2013).

Training RNNs has been proved to be problematic because weights update are mainly based on gradient which can lead to either vanishing or exploding gradient problems (Kanai et al., 2017). Therefore, to overcome this challenging task, Hochreiter and Schmidhuber (1997) introduced the 'Long short-term memory' (LSTM) algorithm. LSTM is a special case of RNN that has three gates: an input, a forget and an output gate. The architecture of LSTM allows changes on a cell state vector that is propagated iteratively to capture long-term dependencies (Smagulova and James, 2019) as illustrated by Fig 2.10.

The LSTM algorithm has a temporal memory mechanism that allows the switch of gates to prevent the gradient vanishing/explosion. For the basic LSTM unit, its external inputs are its previous cell state $c_{(t-1)}$, the previous hidden state $h_{(t-1)}$ and the current input vector $x_{(t)}$. The gates of the LSTM algorithm are calculated as:

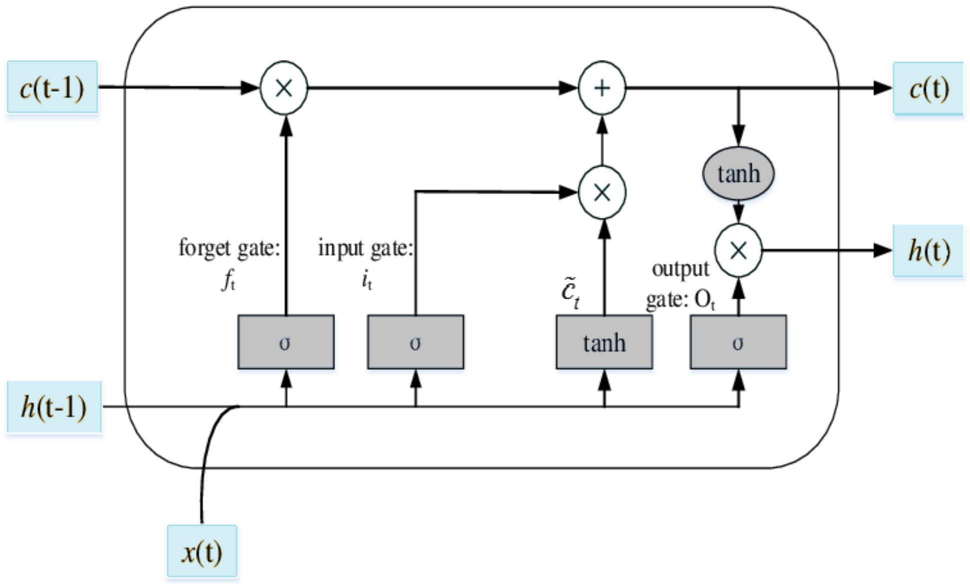


Figure 2.10: LSTM generic architecture.

$$f_{(t)} = \sigma(w_{fx}x_{(t)} + w_{fh}h_{(t-1)} + b_f) \tag{2.13}$$

$$i_{(t)} = \sigma(w_{ix}x_{(t)} + w_{ih}h_{(t-1)} + b_i) \tag{2.14}$$

$$o_{(t)} = \sigma(w_{ox}x_{(t)} + w_{oh}h_{(t-1)} + b_i) \tag{2.15}$$

where f is the forget gate, i is the input gate, o is the output gate, σ is the non-linear activation function, and b is the bias. For LSTM, generally the sigmoid is used as activation function. The intermediate state in the LSTM is given by:

$$\tilde{c}_{(t)} = tanh(w_{cx}x_{(t)} + w_{ch}h_{(t-1)} + c_i) \tag{2.16}$$

where *tildec* is the intermediate state, and *tanh* is the nonlinear *tanh* activation function. The memory cell c, and the hidden state h are updated as follows:

$$gc_{(t)} = f_{(t)} \odot c_{(t)} + i_{(t)} \odot c_{(t)} \tag{2.17}$$

$$h_{(t)} = o_{(t)} \odot tanh(c_{(t)}) \tag{2.18}$$

$\odot$ is used to represent the pointwise multiplication operation for two vectors.

2.6.3 Transformers

Firstly introduced by Vaswani et al. (2017), Transformers are a form of multi-layered deep learning stack that has self-attention mechanism as the principal characteristic. The self-attention mechanism can be viewed as a graph-like inductive bias that connects all tokens in a sequence with a relevance-based pooling operation (Tay et al., 2020). Transformers could obtain high impact results in many fields such as natural language processing (Devlin et al., 2018; Wang et al., 2019b) and image processing (Carion et al., 2020), even surpassing traditional methods as convolutional neural networks and RNNs. The generic architecture of a Transformer model is illustrated by Fig. 2.11.

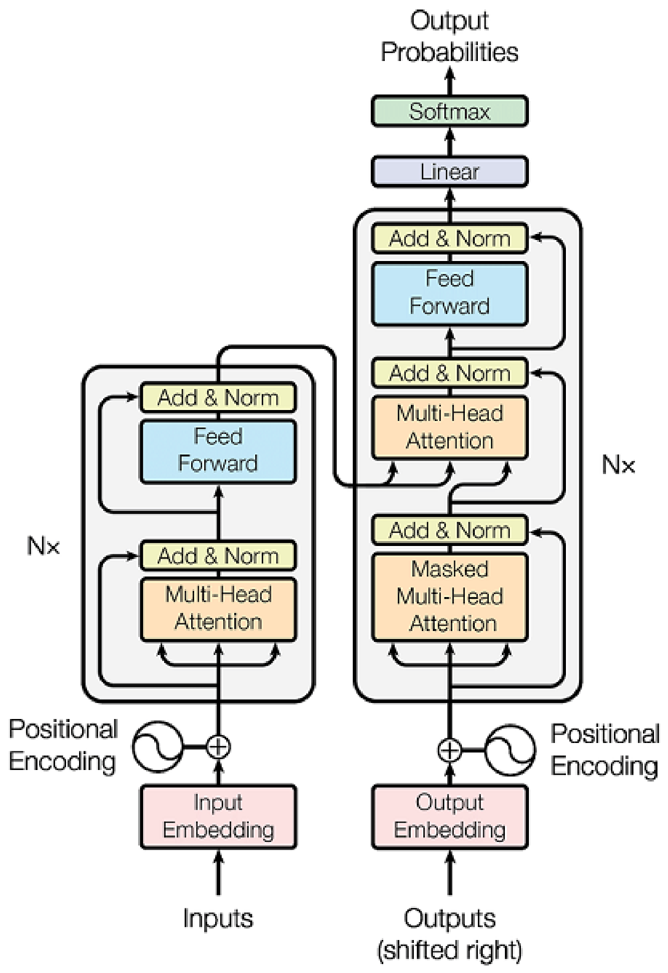


Figure 2.11: Transformers generic architecture (Vaswani et al., 2017).

Transformer blocks are composed by a multi-head self-attention mechanism, a position-wise feed-forward network, layer normalization and residual connectors for both the encoder and decoder (Vaswani et al., 2017) as shown in Fig. 2.11. The input of Transformer model is generally a tensor of shape $\mathbb{R}^B \times \mathbb{R}^R$, where B is the batch size, and N the length of the sequence. The inputs are passed to a multi-headed self-attention module.

The main idea of the self-attention mechanism is the alignment elements in a sequence (Li et al., 2020). During the alignment the algorithm learns from other tokens in the sequence. The attention is given by (Tay et al., 2020):

$$A_h = softmax(\alpha Q_h K_h^T) V_h, \tag{2.19}$$

where, $Q_h = w_q x$, $K_h = w_k x$, and $V_h = w_v x$ are temporal linear transformations applied on the input sequence; w are the weights parameters and x the input of the query, α is a scaling factor that the default values generally is $\sqrt{\frac{1}{d}}$ where d is a dimensional embedding.

It is important to highlight that there are differences in the usage mode of the Transformers models. In this sense, Transformers can be used for mainly in three ways: i) encoder-only (generally for classification tasks); ii) decoder-only (e.g., natural language processing); and iii) encoder-decoder (e.g., object detection and machine translation). Multiple multi-headed self-attention modules are used in the encoder-decoder mode, as well as a cross-attention that allows the decoder to utilize information from the encoder. In the encoder only mode, there is no restriction the self-attention mechanism has to be causal. However, in the encoder-decoder mode, the encoder and encoder-decoder cross attention can be non-causal but the decoder self-attention must be causal (ibid.).

2.7 From black-box to explainable approaches

Despite the success of deep learning and their extraordinary performance in many challenging tasks, the most criticism towards them is often characterized by their "black box" nature (Rai, 2020). Indeed, deep learning models are composed by millions or even a billion of weights/parameters learned from data during the training phase. The link of these huge number of weights with the physicality of the problem also is extremely difficult (Castelvecchi, 2016). Therefore, the explanation of the decisions made by such methods are highly problematic (Rudin, 2019).

Model explanations are essential especially in highly sensitive areas such as healthcare, autonomous driving, and other applications related to human life, rights, finances, and privacy (ibid.). As such applications are becoming more and more usual, the necessity for transparency and explainability is gaining more attention (Gunning, 2017). A search in Google Trends [1] revealed that in the last decade publications using the terms “Deep Learning” and “explainable AI” (XAI) grew significantly. However, while the curve for the term “Deep Learning” is now under saturation, the curve for the term “XAI” is growing exponentially. The rise on the search for the term “explainable AI” starts precisely 3 years ago, which is the same date that the search for the term “Deep Learning” started to saturate as illustrated by Fig 2.12 and 2.14. This is not coincidental and demonstrates that the interest towards XAI is increasing as an attempt to address open research questions (Arrieta et al., 2020) left by traditional deep learning methods.

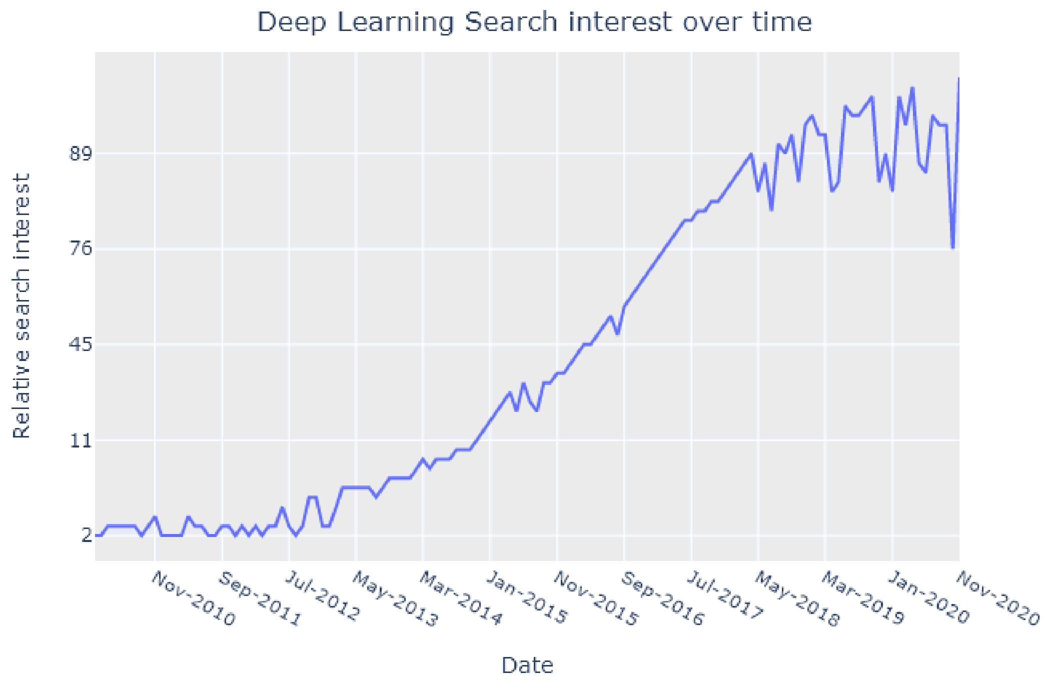


Figure 2.12: Search interest over the time for the term “Deep Learning”.

2.8 Explainable AI

An explainable AI (XAI) is an AI system that the actions and decisions can be easily understood, analyzed, and interpreted by humans (Hagras, 2018). In the last years, as AI human-centric applications have gained more attention, algorithmic decisions become more consequential to individuals and society (Wang et al., 2019a). Therefore,

[1] `https://trends.google.com/trends/`

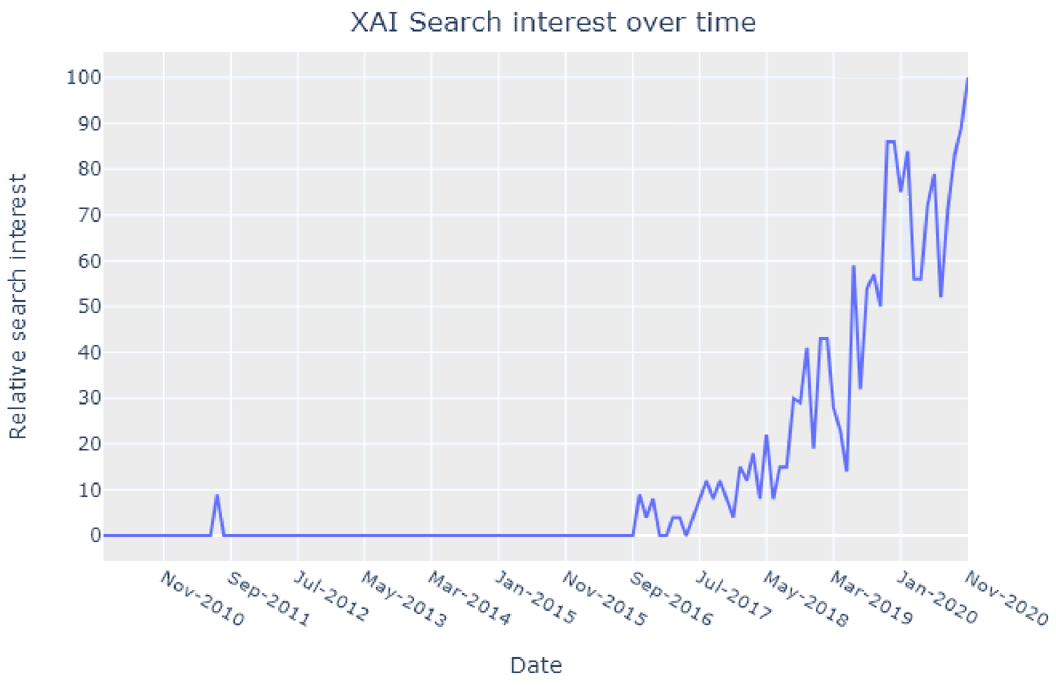


Figure 2.13: Search interest over the time for the term "explainable AI".

explainability and interpretability of AI methods have become an important issue that concerns not only for scientists, but also regulators, and politicians (Holzinger et al., 2018).

Complex and "black box" (Pasquale, 2015; Rudin, 2019) models can easily fool users which are unable to inspect the algorithms decision (Nguyen et al., 2015) and, this can lead to dangerous or even fatal consequences (Stilgoe, 2020). Hence, transparent and explainable AI approaches are critically important for acceptability within society, but also for regulatory purposes (In 2019 the US Congress passed the Algorithmic Accountability Act (MacCarthy, 2019) and the EU enshrined the right for an explanation to the consumer (Goodman and Flaxman, 2017)).

Goodman and Flaxman (ibid.) states that algorithms must "provide appropriate safeguards" including "the right to obtain human intervention to express his or her point of view and to contest the decision.". Therefore, it clearly says that algorithms must provide transparent decision mechanisms that allow humans to investigate and contest if necessary.

The lack of transparency and accountability of AI models already has severe consequences (Rudin, 2019); e.g. people that have been incorrectly denied parole, poor bail decisions leading to the release of dangerous criminals (Wexler, 2017), accidents involving autonomous cars (Gurney, 2013), and non-effective use of valuable resources in criminal justice, medicine, and energy reliability (Rudin, 2019; Varshney and Alemzadeh, 2017).

The actual data-rich environment and powerful hardware allowed solutions that provides highly accurate results from a very large number of abstract, purely numerical parameters (Angelov and Soares, 2020b; Rudin, 2019; Stock and Cisse, 2018), without providing a deep insight into, and understanding of, the underlying dependencies, causalities, and internal model structures.

AI explainability still an open research question in the machine learning field (Bishop, 2006), as illustrated by Fig. 2.14.

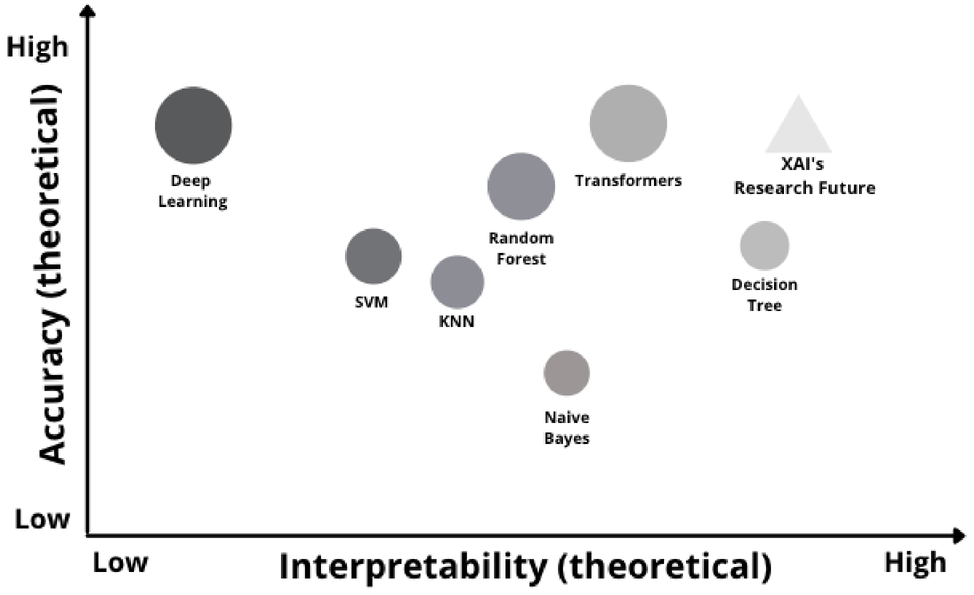


Figure 2.14: XAI's future research.

2.8.1 XAI Classification

There are notable differences among the concepts of interpretability and explainability, however, the misuse of these terms are common in the literature (Arrieta et al., 2020). While the term interpretability refers to a passive characteristic referring to the level of understanding of model for a human expert, the term explainability makes reference to an active characteristic of the model that has the intention of clarification of its internal functions (Adadi and Berrada, 2018).

Here, the most used nomenclature used in XAI communities are summarized:

- **Transparency**: is the opposite of "black-box" or opaque. The model is considered transparent if it has the capability to be understandable by its own (Adadi and Berrada, 2018).
- **Interpretability**: is defined as the ability of provide meaning interpretations that are understandable to human experts (Gilpin et al., 2018).
- **Explainability**: It embraces AI systems that are both accurate and comprehensible to humans (ibid.). Explanation works as an interface between *humans* and AI systems.
- **Understandability** is defined by the characteristic of a model to be comprehend by human expert with the minimum effort to understand its internal structure and processes (Arrieta et al., 2020).

Even though, these terms may sound semantically similar, they offer important and different levels of AI explanations. Below, the taxonomy and ontology of XAI is defined at a high level:

- **Transparent Models**: examples of typical transparent models (Adadi and Berrada, 2018) include traditional methods as the k-Nearest Neighbours (kNN) algorithm (Bishop, 2006), Decision Trees (Rokach and Maimon, 2005), Rule-based Learning (Grosan and Abraham, 2011), Bayesian Network (Friedman et al., 1997). The aforementioned algorithms are transparent by their nature, however, it does not necessarily means that they are explainable by themselves.
- **"Black-box" Models**: Black box models are generally highly accurate, however, they suffer from lack of transparency (Rudin, 2019). This category of models includes Random Forest (Breiman, 2001), Neural Networks (Müller et al., 1995), and Support Vector Machines (Hearst et al., 1998).
- **Model Agnostic**: this type of approach is designed to be flexible and generally applicable (Dieber and Kirrane, 2020). Model agnostic approaches do not rely on a intrinsic architecture, thus, they are based on the input data its outputs.
- **Model Specific**: it is defined by a particular type of approach that takes advantage of model and aims to bring transparency and explanation to it (Langer et al., 2021).
- **Explanation by Simplification**: denoted by surrogate models that are simplification of complex ones in order to improve explainability (Soares et al., 2020c).

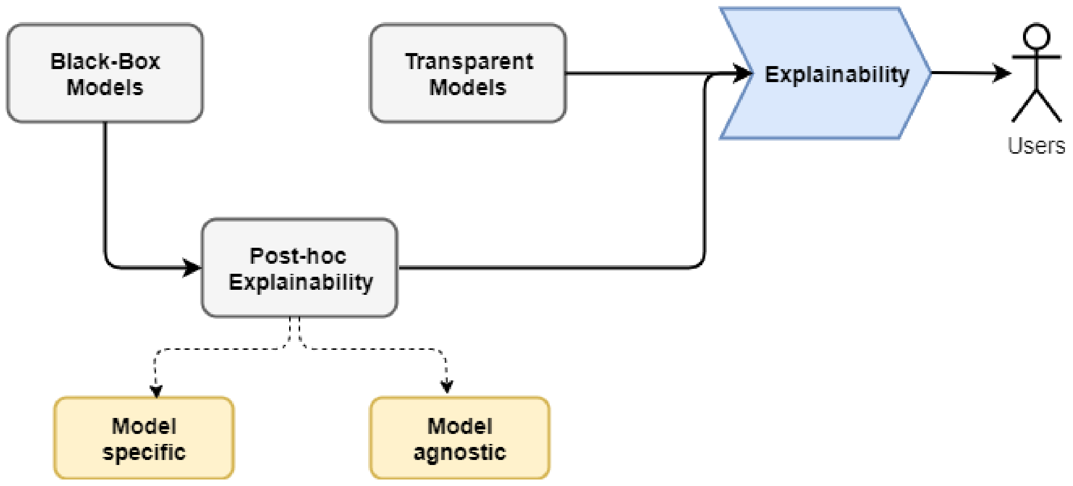


Figure 2.15: The high-level XAI description structure.

- **Explanation by Feature Relevance**: Similarly to simplification, the idea of this approach is to evaluate and rank features that are most influential to the model's decision, for example the SHAP values (Lundberg and Lee, 2017).
- **Visual Explanation**: this type of approach considers data visualization as a form to interpret the model's decision. Generally, are applied to image-based classifiers and detectors, Grad-cam is an example of visual explanation approaches (Selvaraju et al., 2017).
- **Local Explanation**: this type of explanation is based on the matching of data inputs that are similar to the one we are interested in explaining (Ghai et al., 2020).

Fig. 2.15 illustrates the structure of the XAI taxonomy. Transparent models can easily achieve explainabilty, while opaque models require post-hoc approaches to make them explainable. The categories of post-hoc approaches are illustrated accordingly.

Phillips et al. (2020) list the 4 principles (Table 2.2) that defines an explainable AI according to the NIST.

2.8.2 State-of-the-art on XAI methods

Different approaches for explainable AI methods includes: game-theoretic Shapley additive explanations (SHAP) (Chen et al., 2019), local pseudo explanations (LIME) (Dieber and Kirrane, 2020), GradCam (Selvaraju et al., 2017) and its variations, layer-wise feature relevance propagation and attribution (Tritscher et al., 2020), sensitivity analysis (Arrieta et al., 2020), and surrogate models.

Table 2.2: XAI 4 principles summary.

Principle	Definition
Explanation	AI system must supply evidence, support and reasoning for each decision made by the system
Meaningful	Explanation provided by the AI system must be understandable and meaningful to its users
Accuracy	Explanation provided by the AI system must reflect accurately the system's processes
Knowledge Limits	AI systems must identify cases that they were not designed for, and their answers may not be reliable.

2.8.2.1 Features-oriented methods

Introduced by Lundberg and Lee (2017), the SHapely Additive exPlanation (SHAP) values are an unified measure of feature importance for machine learning predictions. The SHAP values provides unique additive feature importance measure that addresses the following properties (Chen et al., 2019):

i) **Local accuracy**: it requires that the explanation model matches the output of a function f for an input x.

ii) **Missingness**: the missingness principle requires that features that are missing in the original input may not have impact to a model if the simplified inputs represent feature presence.

iii) **Consistency**: this property states that if a model changes the features attribution should not decrease.

The SHAP method calculates its Shapley values based on the coalitional game theory (Lundberg and Lee, 2017). In this sense, each feature that composes the dataset act as players in a coalition game. Therefore, the payoff of the game is an additive measure of importance which represents the weighted average contribution of a particular feature within every possible combination of features. A player can also be a group of feature values. Different applications have demonstrated the efficacy if SHAP in explaining model's decision (Mokhtari et al., 2019; Padarian et al., 2020). However, the SHAP values may not be completely transparent if the model is not additive because such predictive models may have non independent pay-off splits. Fig 2.16 illustrates a SHAP plot for house price prediction.

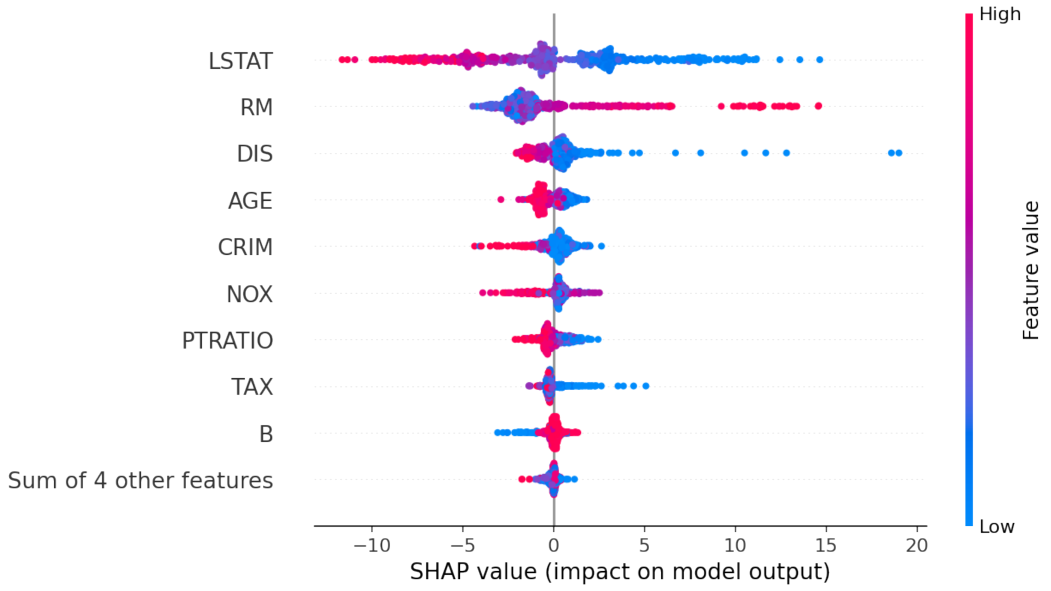


Figure 2.16: Example of SHAP plot (Lundberg and Lee, 2017).

The Fig 2.16 shows the distribution of the impact of each feature on the model's output. Features are ordered by the sum of SHAP value magnitudes over all samples, where the color red represents a high impact, and the color blue denotes a negative impact. In this case, the plot reveals that the feature "LSTAT" has the higher impact in the model´s decision.

Class activation maps (CAMs) are specific to CNNs. CAMs represent the per-class weighted linear sum of visual patterns present at various spatial locations in an image (Zhou et al., 2016). More formally, global average pooling is applied to the final convolutional feature map in a network, before the output layer. These pooled feature maps are then used as the input features to a fully connected layer and output through a loss function. By projecting the weights of the output back to the previous convolutional layer, the areas in the input image with greater influence over the CNNs' decision are highlighted per-class and visible through a heatmap representation. CAMs cannot be applied to pre-trained networks and networks that do not adhere to the specified fully convolutional network architecture. Additionally, spatial information can be lost by the fully connected layer and map scaling. Two generalizations of the base CAM model, Grad-CAM (Selvaraju et al., 2017) and Grad-CAM++ (Chattopadhay et al., 2018), try to further increase the explainability of CNNs.

Gradient-weighted class activation mapping (Grad-CAM) (Selvaraju et al., 2017) generalizes CAM to any arbitrary CNN architecture and without retraining. The gradients for any target class are fed into the final convolutional layer and an importance

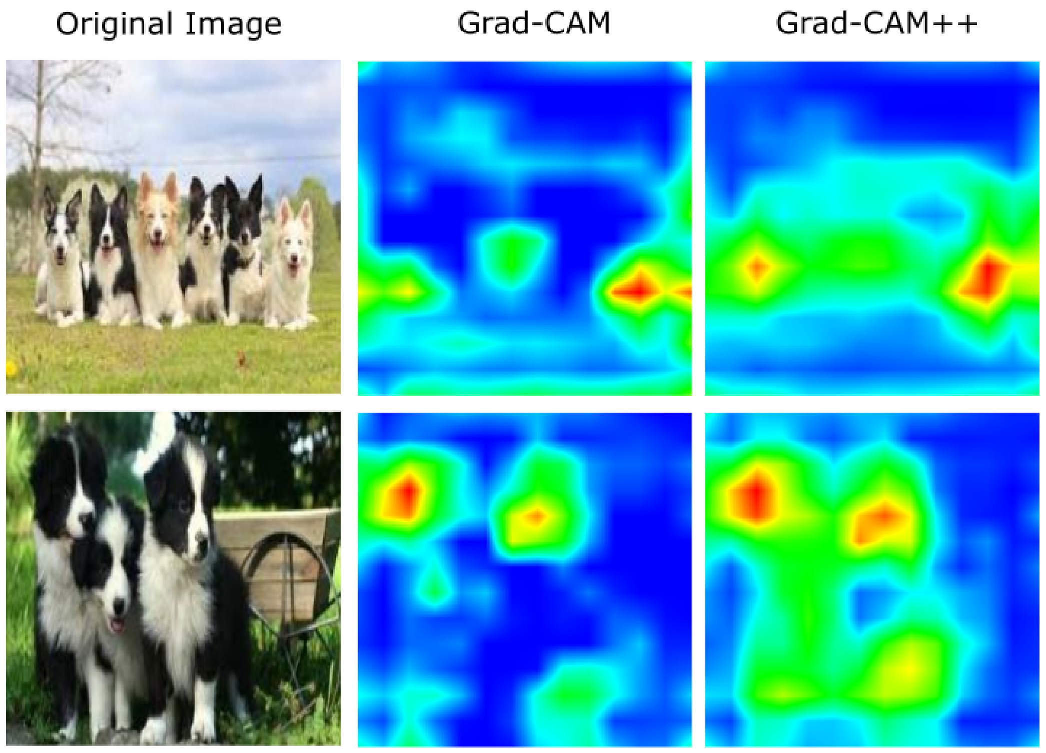


Figure 2.17: Example of Grad-CAM and Grad-CAM++ explanations (Chattopadhay et al., 2018).

score computed in respect to the gradients. As with other methods, a heatmap representation of the Grad-CAM indicates which regions of the input image were most important in the CNN's decisions. However, Grad-CAM produces only coarse-grained visualizations and cannot explain multiple instances of the same object in an image. Grad-Cam++ (Chattopadhay et al., 2018) considers the weighted average of the gradients to overcome these drawbacks. Fig. 2.17 illustrates the difference of explanations provided by Grad-CAM and Grad-CAM++ for a same image.

Fig 2.17 illustrates how *post-hoc* methods for explainability can provide different explanations for a same model. Explainable-by-design approaches avoid this incoherence by providing a unique interpretation of their decision. Therefore, they a clearly audible for users.

Feature oriented methods provide insights into where a decision is taking place in terms of the input, but fall short of a human level explanation of how and why the model came to those decisions. Consequently, a human could not exactly reproduce the explanations rendered by the model.

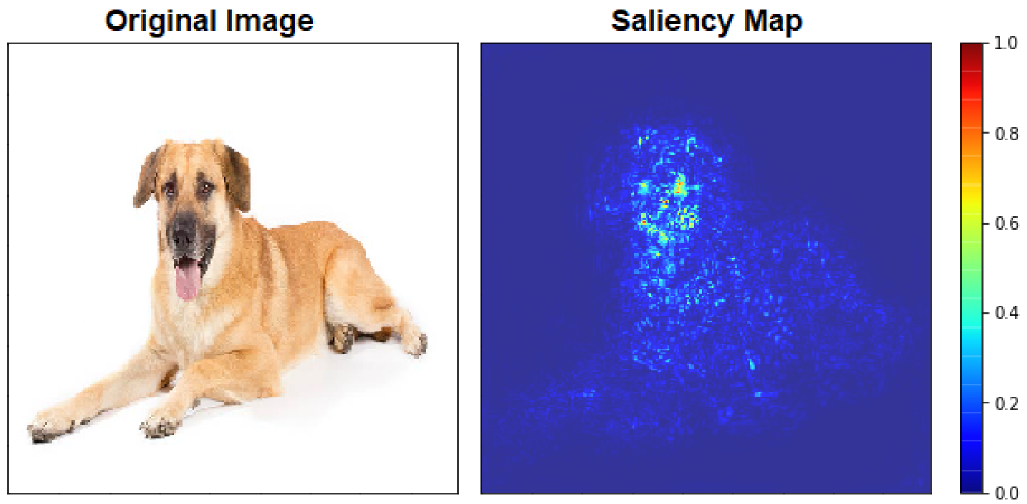


Figure 2.18: Saliency map for dog classification.

2.8.2.2 Global methods

Global Attribution Mappings (GAMs) (Ibrahim et al., 2019) are used for features with precise semantic definitions. In this sense, GAMs are able to explain a neural network's predictions on a global level through weighted conjoined rankings. The advantages of this method is that different sub populations can be taking into consideration through different tuneable granularity thresholds. K-medoids are used to cluster features with similar importances, and each medoid summarizes a different pattern as a global attribution. Therefore, GAMs are pertinent to feature exploration among different sub-populations of samples.

Another well-known global method for explanations is the Gradient-based saliency maps (Simonyan et al., 2013). This technique is responsible to render the absolute value of the gradient of the majority predicted class as a normalized heatmap. The Fig 2.18 illustrates how the saliency map is built for a dog classification. The heatmap indicates the pixels with higher activation during the classification process are close to the dog's snout. Pixels with high activation are highlighted and correspond to areas that are most influential. However, the absolute value means that gradients of neurons with negative input are suppressed when propagating non-linear layers. Therefore, this type of technique is not reliable and fails in passing the real message to users.

2.8.2.3 Surrogate models

Local Interpretable Model-Agnostic Explanations (LIME) (Dieber and Kirrane, 2020) is a model-agnostic technique to create locally optimized explanations of machine

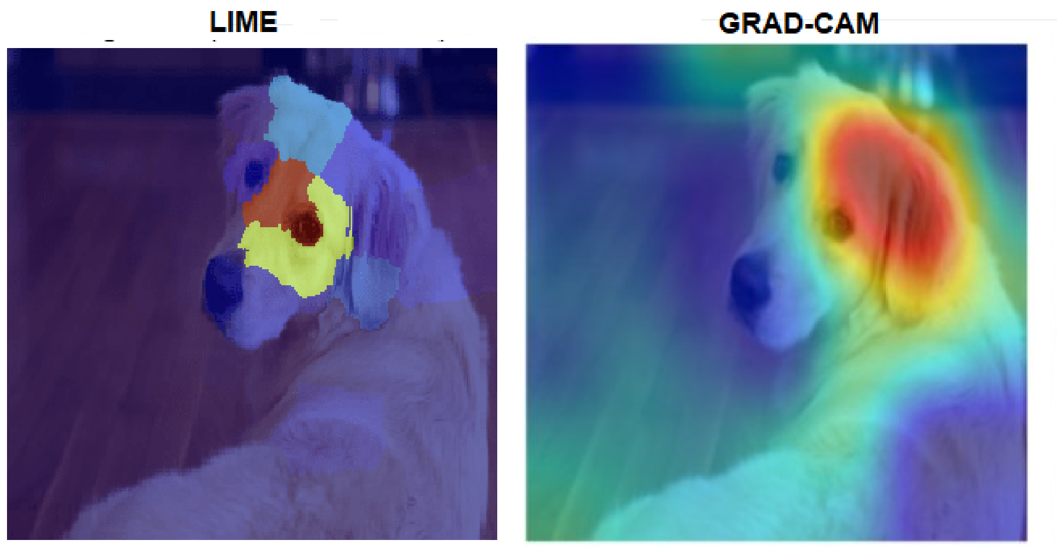


Figure 2.19: Lime and Grad-CAM explanations for labrador retriever.

learning models. LIME trains an interpretable surrogate model to learn the local behavior of a global "black box" model's predictions. For image classification, an input image is divided into patches of contiguous superpixels (i.e. an image object) and a weighted local model is then trained on a new set of permuted instances of the original image (i.e., some superpixels are turned to gray). The intuition is then that by changing aspects of the input data that are human understandable (spatial objects) and learning the differences between those perturbations and the original observations, one can learn what about the input contributed to each class score. However, these explanations are not always informative or reliable at a human level if the parameters that control the perturbations are chosen based solely on heuristics. Fig 2.19 illustrates the difference of explanations provided by LIME and Grad-CAM for a labrador retriever. While LIME, produces its explanations based on local patches or superpixels, GRAD-CAM is based on activations maps.

2.8.2.4 Local, pixel-based methods

The Layer-wise Relevance BackPropagation (LRP) (Bach et al., 2015) method renders a heatmap to provide insights about pixels contributions during the model's prediction. The method is based on predefined propagation rules that provides explanations of neural networks. This method is only valid if the network implements backpropogation. Fig 2.20 illustrates the LRP explanation process using backpropagation where x_i is the image input, x_f is the output function, R_f is the relevance function, and R_i is the

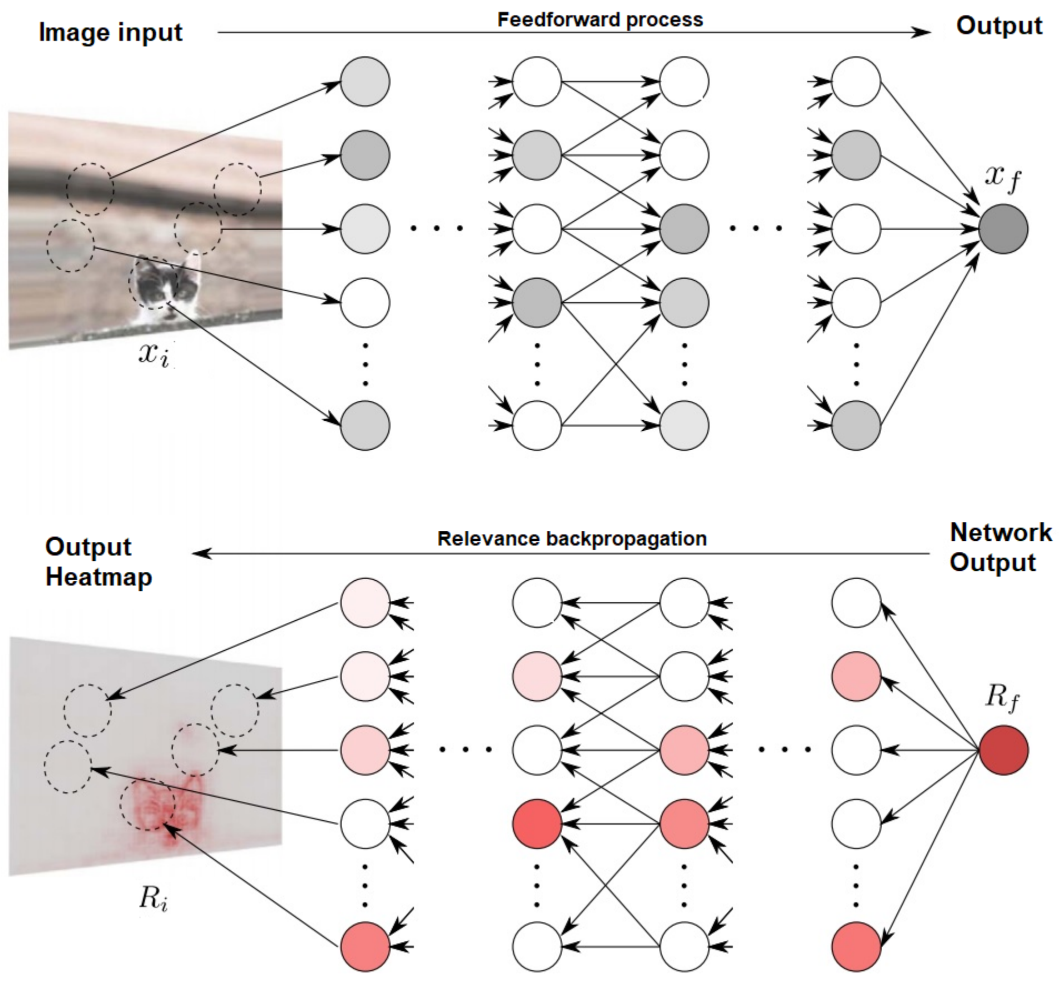


Figure 2.20: LRP explanation process (Bach et al., 2015).

relevance heatmap. Similarly, to LRP, Hinton et al. (2006) introduced the Deep Belief Network to improve the interpretability of traditional neural networks.

2.8.2.5 Anthropomorphic machine learning

Although, the aforementioned methods provide clear advantages in terms of explanations in contrast to pure "black-box" decisions, they rely on *post-hoc* analysis about the features, weights, and other aspects of the data.

This is completely different from the way people make decisions (Angelov and Gu, 2018b). In fact, the aforementioned methods do not answer the fundamental questions of model structure and parameters relating to the nature of the problem and completely ignore reasoning process.

In this thesis, a different approach to explainability is explored. The anthropomorphic approach explored in here investigates how people use similarity to associate new data with previously learned and aggregated prototypes (Bien and Tibshirani, 2011; Bishop, 2006). Indeed, humans compare items (e.g. images, songs, and movies) in their entirety and not per feature or pixel.

2.8.2.6 Discussion

Table 2.3 summarizes some of the different approaches for explainability presented so far. As aforementioned, although most of the model-specific approaches present high level of explanability, they do not provide high accurate results. On the other hand, *post-hoc* approaches try to explain highly accurate systems, however, the explanations provided by these methods are not totally reliable. In this thesis, new approaches that combine accuracy and explanability in synergy are explored in a way reduce the gap for this research field.

Table 2.3: Summary of main XAI techniques.

Method	*Post-Hoc*	Global/Local	Specific/Agnostic
Decision Trees	No	Global	Specific
K-nn	No	Global/Local	Specific
Rule-based models	No	Global/Local	Specific
SHAP	Yes	Global	Agnostic
Activation Maps	Yes	Global	Agnostic
Surrogate Models	Yes	Global/Local	Agnostic
LRP	Yes	Global	Agnostic
Saliency maps	Yes	Global	Agnostic
Prototype-based models	No	Global/Local	Specific

2.9 XAI applications

The frequency and importance of algorithms in applications have led regulators and official bodies to develop policies that provide clearer accountability for algorithmic decision making. One such example is the European Union's General Data Protection Right, which some have interpreted as a "Right to Explanation" (Goodman and Flaxman, 2017). Although the extent of this right is in dispute, the discourse around such topics has reinforced that automated systems must avoid inequality and bias in decisions. Furthermore, they must fulfill the requirements for safety and security in safety-critical tasks. Consequently, there has been a recent explosion of interest in explainable AI models in different areas as depicted below.

2.9.1 Autonomous Vehicles

Autonomous vehicles have to make milliseconds decisions based on challenging drive environments (Schwarting et al., 2018). If an autonomous car has some misclassification problem and acts abnormally the consequences can be dangerous or even fatal. a recent crash (on 18 March 2018) by an autonomous car owned by Uber led to the operator being charged with negligent homicide two and a half years later (Stilgoe, 2020). This was the first known fatality involving a fully autonomous vehicle. Claims are that the vehicle misclassified the person as a plastic bag or tumbleweed carried on the wind (ibid.). An explainable system could clarify the circumstances that lead to such missclassification and eventually prevent it from happening (Shen et al., 2020).

2.9.2 Medicine

In the medical domain there is a growing demand for AI approaches, most notably during the COVID-19 pandemic. However, AI applications must not only perform well in terms of classification metrics, but need also to be trustworthy, transparent, interpretable and explainable, especially for clinical decision-making (Holzinger et al., 2017).

Detection of a disease at its early phase is most of the time critical to the recovery of patients or to prevent the disease from advancing to more severe stages (Tjoa and Guan, 2020). However, explanations for machine decisions are needed to justify their reliability. Therefore, clinicians and practitioners needs caution to use such methods as they require great interpretability and explanability as it is necessary to understand the mechanism underlying the algorithm's decision (Holzinger et al., 2019).

2.9.3 Legal justice

Another example application of XAI is the criminal justice system. In some countries such as the USA automated algorithms are being used to predict where crimes will most likely occur, who is most likely to commit a violent crime, who is likely to fail to appear at their court hearing, and who is likely to re-offend at some point in the future (Dressel and Farid, 2018). One such widely-used criminal risk assessment tool is the Correctional Offender Management Profiling for Alternative Sanctions (COMPAS). Although the data used by COMPAS do not include an individual's race, other aspects of the data may be correlated to race that can lead to racial biases in the predictions.

Therefore, explanations of such critical decisions are necessary to favor fairness and reduce racism during the decisions (Dressel and Farid, 2018).

2.9.4 Defense

As in the medical field, the military applications generally involves human lives. Therefore, explainability in AI decision's is crucial as stated by DARPA (Gunning, 2017). Challenges on the military applications include reliability of autonomous systems for military operations which often leads to ethical and legal dilemmas (Gunning and Aha, 2019).

The efforts for XAI does not limit to the applications above cited. Indeed, it includes applications in others key domains domains as cybersecurity, education, environment, entertainment, and government. Therefore, this justifies the necessity for XAI. Even though, this research line in its beginning and there is research gap to be filled (Adadi and Berrada, 2018).

2.10 Conclusion

This chapter contains the surveys which covers the background that is necessary to understand the scope of the research work presented in this thesis. Traditional machine learning techniques suffer from various problems including: strong prior assumptions, predefined user- and problem- specific parameters, and other *ad-hoc* decisions that may be problematic in large-scale, complex real problems.

Deep learning are the state-of-the-art approaches in the fields of machine learning and computer vision. However, their structures lack transparency as they have million of parameters that are extremely difficult to interpret and relate it to the physical world. Therefore, *post-hoc* methods are used to explain these "black-box" deep learning approaches. However, different *post-hoc* explainable methods may offer different explanations for the same model, which is controversial. Therefore, in the next chapter a new explainable-by-design or anthropomorphic machine learning is presented in order to overcome problems presented in this chapter.

Chapter 3

eXplainable-by-design Deep Learning

In the last decade, deep learning has revolutionized the artificial intelligence research field as it has been proved to be able to obtain highly accurate results in extremely complex problems (Goodfellow et al., 2016; LeCun et al., 2015). It turns out to be very valuable and efficient in the often laborious and sometimes controversial pre-processing stage of automated feature extraction. The main criticisms of deep learning are usually related to its “black box” nature and requirements for large amounts of labeled data, computing resources, and long (hours, in some cases even days) training (Rudin, 2019). In fact, deep learning algorithms involves millions of weights/parameters that are abstract and detached from the physical nature of the modelled problem and very difficult to interpret (Castelvecchi, 2016).

The pre-processing stage of feature extraction is an important task that defines the data space and influences the performance of that a model provides. Therefore, this is an important and useful property of deep learning that can be applied to other machine learning methods. Another important characteristic of deep learning is the possibility of use transfer learning to speed up the training process as illustrated in Fig. (3.1). It possibilities that the knowledge acquired in the form of model architecture for a specific context can be transferred an re-used on another different context (Hu et al., 2015; Tan et al., 2018). The necessary time to train a deep neural network from scratch is reduced considerably through the transfer learning process. Furthermore, it is also beneficial to improve the classification performance of models (Shaha and Pawar, 2018; Zhuang et al., 2015).

Explainable Deep Neural Network (xDNN) uses the two main achievements of deep learning, higher accuracy combined with automatic methods to extract features for complex problems (such as image classification), trying to solve the lack of explainability and other defects, such as the required amount of computational resources, adaptive and self-evolving capabilities. Interpretability and explainability are of supreme importance for high-risk applications, such as self-driving cars, medical treatment, or court decisions(Rudin, 2019).

The xDNN method provides a new deep learning architecture that combines inference and learning in synergy. It is a prototype-based approach which uses data density in its core mechanism (Angelov and Gu, 2019). Besides, xDNN it is non-iterative and non-parametric, improving its efficiency in terms of time and computing resources. From the user's point of view, the approach is clearly understandable to human users.

The remainder of this chapter is organized as follows: The next section introduces a brief literature review. The xDNN approach is presented in Section 3.2. The discussion is presented in the last section of this chapter.

3.1 Brief Literature Review

Deep networks have been purely designed to provide the best results in terms of classification accuracy. The decisions made by such networks are generally interpreted by *post hoc* techniques (Li et al., 2018b) or not interpreted at all. In this sense, the first task for *post hoc* interpretation is the selection of the network architecture and then the attempt to interpret the resulting model through the learned high-level features. Consequently, *post hoc* interpretability analysis requires a separate modeling effort (Saralajew et al., 2018). Moreover, *post hoc* analysis works more as an approximation system than an cause-effect explanation mechanism. Problems with *post hoc* analysis includes non-consistency of explanations as different *post hoc* models can provide different explanations for a same deep learning model. In other words, the creation of multiple conflicting explanations it is facilitated by such methods.

The prototype-based classifier is a reasoning process that does not consider *post hoc* analysis(Biehl et al., 2016).Generally, they are related with the proximity in feature space of a data sample to a prototype (similarity) (Biehl et al., 2013, 2016). Each prototype aims to capture the distribution of a set of data points based on the concept of similarity to the prototype or proximity, which may be affected by (prototype-specific) size and shape parameters (Perner, 2008). In this thesis, a prototype is considered the data sample with higher densities (local density peaks) in the training set (Angelov and Gu, 2019). In other cases, a prototype can be considered as a convex combination of multiple observations, and it does not necessarily require any data samples close to the training set to be even feasible (Liu et al., 2018; Oyedotun and Khashman, 2017).

Classification through prototypes is a well-known form of case-based reasoning (Li et al., 2018b). Li et al. (ibid.) uses neural networks as distance measure between prototypes and data samples. Differently, (ibid.) uses an auto encoder process to establish a low-dimensional space, therefore, the distances to prototypes are computed in the formed latent space. Euclidean distance is also used as it can be expressed in terms of convolution operations in the neural network (Biehl et al., 2013; Nebel et al., 2017). Euclidean distance is essential step towards an efficient architecture for prototype-based neural network layers (Graf et al., 2009).

xDNN uses local densities and global multivariate generative distributions based on an empirically derived form of the probability distribution function for prototype identification (Angelov and Gu, 2019). Differently from other prototype-based classifiers, xDNN is non-iterative and non-parametric as it is using recursive calculations and no search procedures, therefore, it can learn continuously with no necessity for re-training.

3.2 Explainable Deep Neural Network

3.2.1 xDNN Training Architecture

The explainable deep neural network (xDNN) classifier presented in this thesis is composed by different layers with a very clear semantic and functional meaning. xDNN also offers a very clear set of prototype-based $IF...THEN$ rules which favors interpretability. Prototypes are selected data samples that the users can easily view, understand and analyze their similarity to other data samples. xDNN demonstrates that learning and reasoning can work together in a synergy and produce very impressive results.

xDNN can be described as a feedforward neural network. It has an incremental learning algorithm that can develop and evolve its structure autonomously. If necessary, add new prototypes to reflect possible changes (dynamic evolution) of data patterns. As illustrated by Fig. 3.2, th following layers compose xDNN–

1. Features layer;
2. Density layer;
3. Conditional probability layer;
4. Prototype identification layer;
5. *MegaClouds* layer;

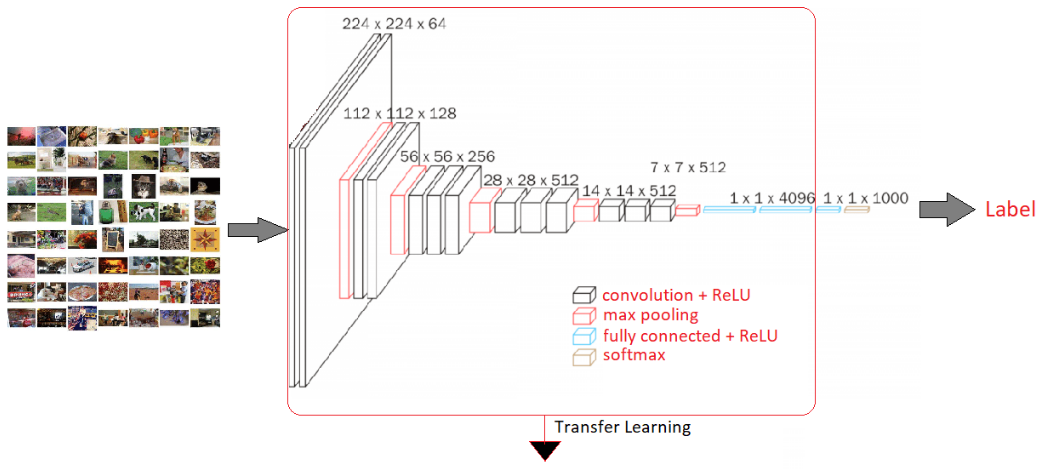


Figure 3.1: Using the transfer learning concept this architecture with the weights can be used as feature extractor (the last fully connected layer is considered as a feature vector). The quality of the features is extremely important on the final result provided by xDNN. Adapted from (Simonyan and Zisserman, 2014).

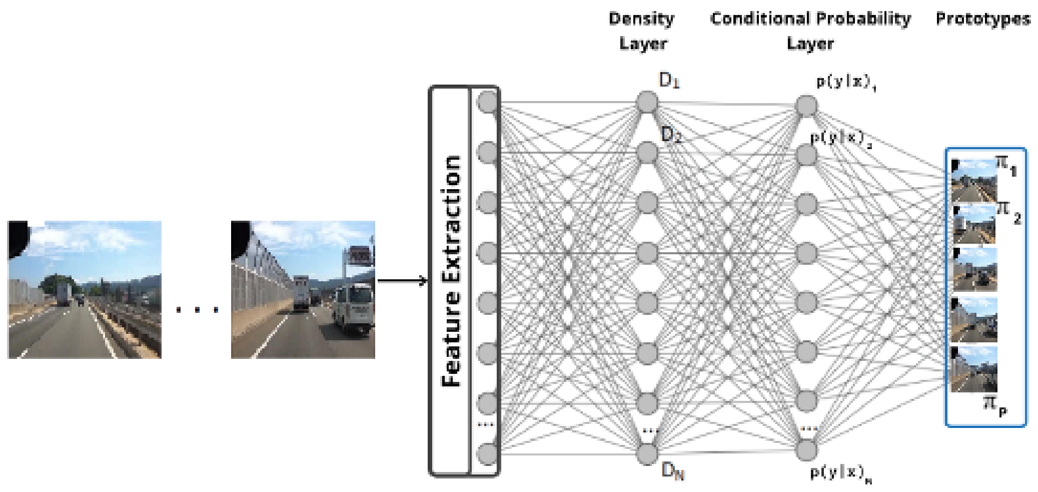


Figure 3.2: xDNN training architecture. The MegaClouds layer is an optional layer used to reduce the dimensionality of the prototypes layer and is not illustrated in this diagram.

3.2.1.1 Features layer

The Feature Layer is in charge of extracting global features vector from images. This layer be formed by the fully connected layer (FCL) of convolutional neural network approaches such as AlexNet (Krizhevsky et al., 2012), VGG–16 (Simonyan and Zisserman, 2014) (see Fig. (3.1), Inception (Szegedy et al., 2015), residual neural networks such as Resnet (He et al., 2016) or Inception-Resnet (Szegedy et al., 2017), etc. Deep neural network approach allows automatic extraction of more abstract and discriminative high-level features. The quality of the extracted features have direct impact on the xDNN accuracy. Therefore, specific problems may require a dedicated training for optimization of the features quality. More traditional 'handcrafted' methods such as GIST (Solmaz et al., 2013) or HoG (Mizuno et al., 2012) can also form this feature layer.

Let us denote the training data set of points by $x = \{x_1, ..., x_N\} \in \mathbb{R}^n$ with corresponding class labels $y_1, ..., y_C \in \{1, ..., C\}$. Here, N is the number of training data samples and n is their number of features (dimensionality); C is the number of classes. xDNN starts by selecting a set of descriptive prototypes $\pi \in P \subset X$ for each class/per class, M_j is the total number of prototypes of class j; $M_j = |P_j|$; $M = \sum_{j=1}^{C} M_j$. Notice that $M_j > 1$ for $\forall j$, i.e. we usually consider more than a single prototype per class.

Prototype-based approaches rely on the fact that they are *explainable-by-design* (Wang et al., 2019a). Therefore, prototypes represent samples of the training data which are easy for users to understand. So, any new data sample, $x \in \mathbb{R}^n$ can be associated with the nearest prototype from the sets $P_1, P_2, ..., P_C$; $P = P_1 \cup P_2 \cup ... \cup P_C$.

$$L(x) = \underset{x \in X}{\operatorname{argmin}} \min_{\pi \in P} d(x, \pi). \tag{3.1}$$

3.2.1.2 Density layer

This layer is composed of neurons whose activation function represent the data density, D, which defines the mutual proximity of the images in the data space defined by the features from the previous layer, formed by a Cauchy function (Angelov and Gu, 2019):

$$D(x) = \frac{1}{1 + \frac{||x-\mu||^2}{||\sigma||^2}}, \tag{3.2}$$

where D is the density, μ is the global mean, and σ is the variance.

In (ibid.) it was demonstrated theoretically that starting from the mutual proximity of the data samples in the data space and using Euclidean (or Mahalanobis) type distance D takes the form of a Cauchy function. Moreover, data density can be updated recursively as detailed in (Angelov, 2012).

Density can also be updated online (ibid.):

$$D(x_i) = \frac{1}{1 + ||x_i - \mu_i||^2 + \sum_i - ||\mu_i||^2}. \tag{3.3}$$

where $i = 1, ..., N$, μ and the scalar product, $\sum$ can be updated recursively as follows:

$$\mu_i = \frac{i-1}{i}\mu_{i-1} + \frac{1}{i}x_i, \tag{3.4}$$

$$\sum_i = \frac{i-1}{i}\sum_{i-1} + \frac{1}{i}||x_i||^2 \quad \sum_1 = ||x_1||^2. \tag{3.5}$$

The value of the data density, D represents the closeness to the mean and is in the range $0 < D \leq 1$ for normalized values. It obtains its maximum (of 1) when $x = \mu$. D is indicative for the centrality of a data sample and its suitability to be a prototype due to its proximity to other data samples.

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.

3.2.1.3 Conditional probability layer

The conditional probability can be estimated from the empirically observed data as described in (Angelov and Gu, 2019). It is also called *typicality* τ. It is given by eq. (3.6), where integral of $\int_{-\infty}^{\infty} p(C|x)dx = 1$ is the same as the pdf (ibid.), but it is multi-modal:

$$p(y|x) = \frac{\sum_{i=1}^{M} N_i D(x)}{\sum_{i=1}^{M} N_i \int_{-\infty}^{\infty} D(x)dx} \tag{3.6}$$

where N_i denotes the number of data samples associated with the $i - th$ *data cloud*, $\sum_{i=1}^{C}$; $N_i = N$.

Notice that since $p(C|x)$ is empirically derived (ibid.) it does not rely on any *prior* assumption about the data distribution type or even about the random or deterministic nature of the data.

3.2.1.4 Prototypes layer

The prototypes identification layer is the core of the proposed xDNN classifier. Prototypes are independent from each other. Therefore, it is possible to change the xDNN structure by adding or removing prototypes without influencing the other existing ones. In other words, the proposed xDNN is highly parallelizable and suitable for self-evolving form of application where new prototypes may be added if required.

The proposed xDNN method is trained per class. So, it forms a set of prototypes per class and all the calculations are done for each class separately. Prototypes are the local peaks of the data density identified in the previous layer of the algorithm as illustrated by Fig. (3.3).

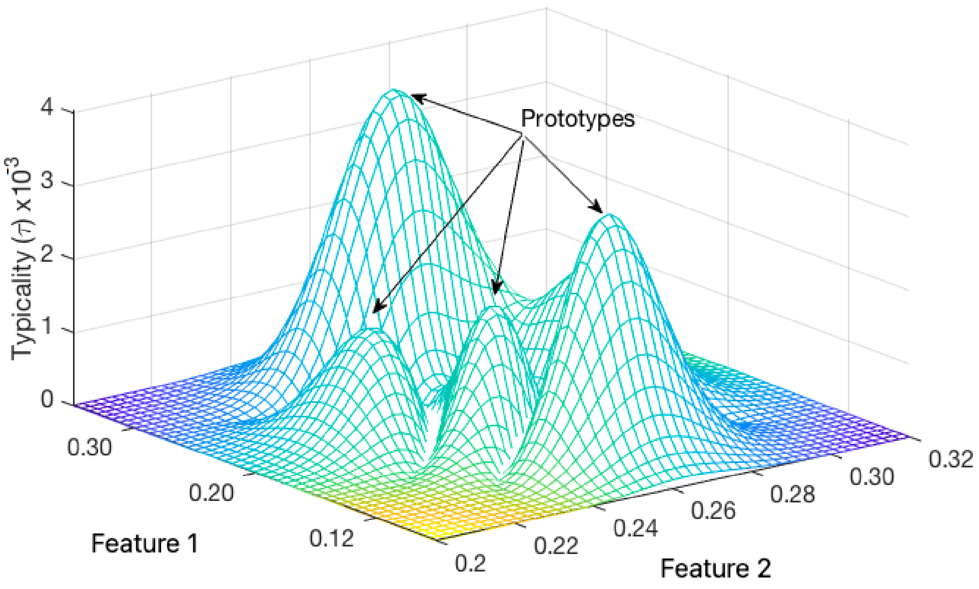


Figure 3.3: Local peaks as identified prototypes.

The xDNN algorithm absorbs the new data samples by assigning them to the nearest prototype:

$$j^* = \underset{i=1,..,N;j=1,..,M}{\operatorname{argmin}} ||x_i - \pi_j||^2 \tag{3.7}$$

Fig. (3.4) 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*.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 (ibid.).

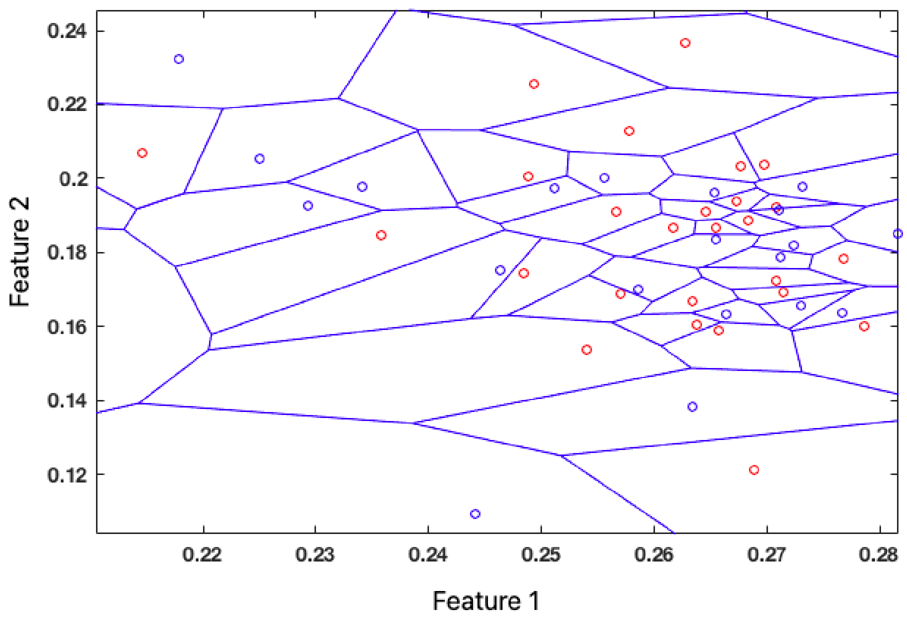


Figure 3.4: Identified prototypes – Voronoi Tesselation.

New prototypes are added to this layer when the following condition is met (Angelov and Gu, 2019):

$$\begin{aligned} &IF\ (D(x) \geq \max_{j=1,..,M} D(\pi_j)) \\ &OR \quad (D(x) \leq \min_{j=1,..,M} D(\pi_j)) \\ &THEN\ (add\ a\ new\ data\ cloud\ (j \leftarrow j+1)) \end{aligned} \tag{3.8}$$

3.2.1.5 Learning Procedure

xDNN learning mechanism is summarised below by the following pseudo-code. The proposed method can work both, in a batch mode as well as on a per sample basis, online.

xDNN: Learning Procedure

```
1: Read the first feature vector sample $x_i$ of class $c$;
2: Standardise and normalise the data as detailed in (Angelov and Soares, 2020b)
3: Set $i \leftarrow 1; j \leftarrow 1; \pi_1 \leftarrow x_i; \mu \leftarrow x_1; N \leftarrow 1$
```

```
4:  FOR $i = 2, ...$
5:      Read $x_i$;
6:     Calculate $D(x_i)$ and $D(\pi_j)$ $(j = 1, 2, ..., M)$ according to eq. (3.2);
7:      IF eq. (3.8) holds
8:        Create new prototype: $j \leftarrow j + 1; \pi_j \leftarrow x_i; N \leftarrow N + 1$
9:      ELSE
10:       Search for the nearest prototype according to eq. (3.7);
11:       Update the nearest prototype as:
          $N \leftarrow N + 1;$
          $\pi_j \leftarrow \frac{N_j}{N_j+1}\pi_j + \frac{N_j}{N_j+1}x_i;$
12:     END
13: END
```

3.2.1.6 *MegaClouds* layer

This is the final layer of the training architecture. Unlike the previous layers it is cross-class. At this layer prototypes from all classes are put together and once this is done all the adjacent *data clouds* that have the same class label are combined into *mega-clouds*, see Fig. (3.5). Notice that the number of *megaclouds*, $i = 1, 2, ..., MG$ is significantly smaller than the number of prototypes, $(MG << M)$ and the interpretability improves significantly. Fig. (3.5) illustrates the formation of the *MegaClouds*.

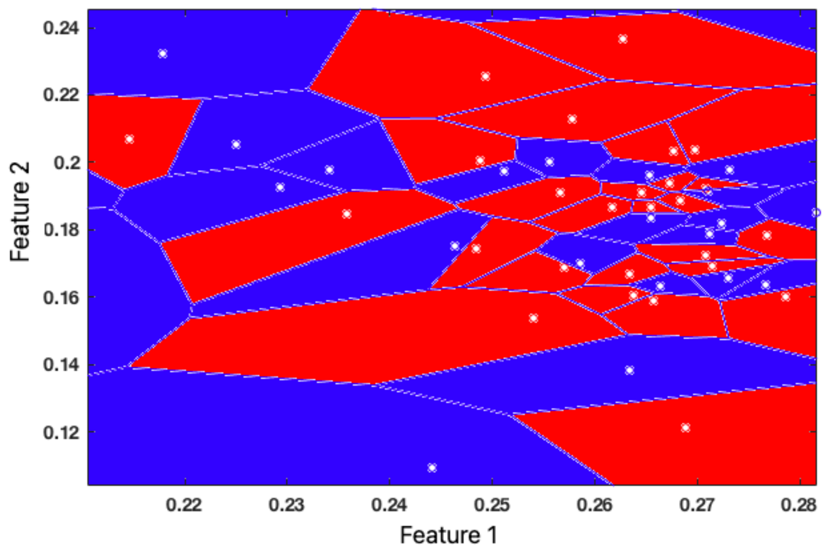


Figure 3.5: *MegaClouds* – Voronoi Tesselation.

3.2.2 xDNN Decision Structure

Architecture for the decision process of the proposed xDNN method is illustrated by Fig. (3.6).

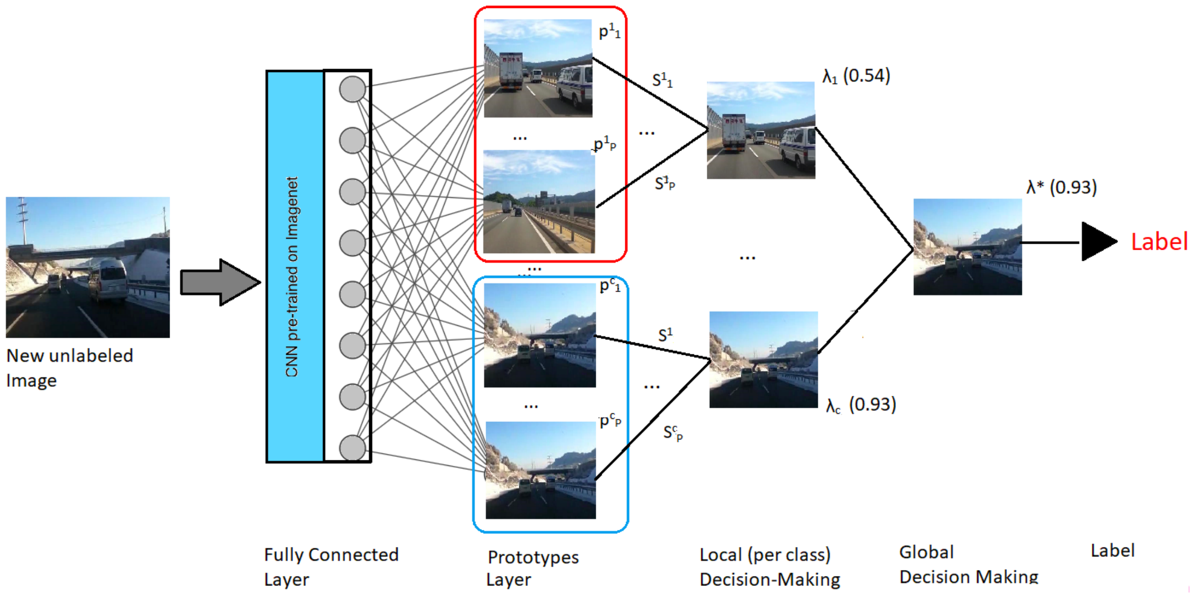


Figure 3.6: Architecture for the validation process of the proposed xDNN.

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:

3.2.2.1 Features layer

Similarly to the features layer described in the training process.

3.2.2.2 Similarity layer

This layer is responsible to calculate the similarly degree between the test sample and the prototypes acquired during the training. Details of the similarity degree calculation are given in the next sub-section.

3.2.2.3 Local decision making layer

The output of this layer is the degree of similarity, S between the unlabeled data sample and the respective prototype. The decision making in xDNN can be represented by a similar deep network architecture in which real images (prototypes) play a key role and the similarity between any new data sample (image) of which the class is to be determined and the prototypes with known class labels are being calculated through a SoftMax-like eq. (3.9).

$$\lambda(Y = x_i|\pi_j) = \frac{S_j}{\sum_{j=1}^{M} S_j}, \tag{3.9}$$

where,

$$S_j = S(x_i, \pi_j) = \frac{1}{1 + \frac{||x_i - \pi_j||^2}{||\sigma_j||^2}}, \tag{3.10}$$

where Y is the $j - th$ test sample. S is the degree of similarity between the unlabeled data sample and the respective prototype.

3.2.2.4 Global decision making layer

The global decision making is responsible to provide the label to the test sample which is obtained by the following eq. (3.11):

$$label = \underset{c=1,2,\ldots,C}{argmax}(\lambda_c^*), \tag{3.11}$$

3.3 Experimental Data

The xDNN approach presented in this chapter is tested with different complex, and well-known image classification benchmark datasets (Calltech-256, Calltech-101) as well as with real-world challenging tasks as iRoads for driving scenario classification, and a proprietary dataset of CT-scans for COVID-19 identification. Tabular datasets as the COMPAS for recidivism risk prediction, and the Physionet heart sound classification were also considered in the analysis.

3.3.1 Caltech-256

Caletch-256 has 30,607 images divided into 257 object categories (one of which is the background) (Griffin et al., 2007). Data was collected from Google Images and manually screened. The minimum number of images per category is 80. Fig. (3.7) illustrates the Caltech-256 dataset.

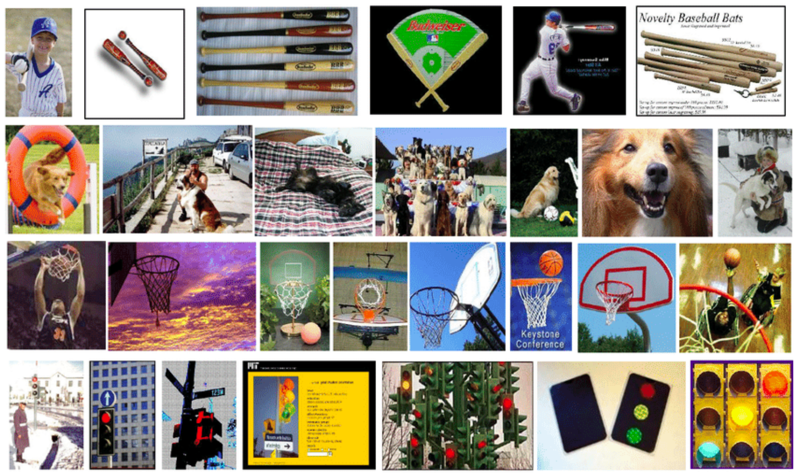


Figure 3.7: Samples of the Caltech-256 (Griffin et al., 2007).

3.3.2 Caltech-101

Caletch-101 is divided into 102 object categories (one of which is the background) (Fei-Fei et al., 2004). For each object category, there are about 40 to 800 images, while most classes have about 50 images. The resolution of the image is roughly about 300×200 pixels. The Caltech-101 dataset is highly unbalanced and is widely used as benchmark data set. Fig (3.8) illustrates the Caltech-101 dataset.

3.3.3 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 as illustrated by Fig. (3.9), recorded in day, night, under various weather and lighting conditions, as described below:

- Daylight - 903 images
- Night - 1050 images

Figure 3.8: Samples of the Caltech-101 (Griffin et al., 2007).

- Rainy day - 1049 images
- Rainy night - 431 images
- Snowy - 569 images
- Sun strokes - 307 images
- Tunnel - 347 images

The dataset is divided into 80% for training purposes and 20% for testing.

3.3.4 COVID CT-scan dataset

Recent findings have observed specific image patterns from computed tomography (CT) for patients infected by SARS-CoV-2 which are distinct form other pulmonary disease. For example, for patients diagnosed with COVID-19 the analysis revealed bilateral lung opacities in 40 out of 41 (98%) chest CTs in infected patients in Wuhan and described lobular and sub-segmental areas of consolidation as the most typical findings (Ai et al., 2020). Other investigators found high rates of ground-glass opacities and consolidation, sometimes with a rounded morphology and peripheral lung distribution (Kong and Agarwal, 2020; Ng et al., 2020). Thoracic radiology evaluation is often key to the evaluation of patients suspected of COVID-19 infection (Shi et al., 2020). Prompt detection and diagnosis of the disease is invaluable in the efforts to ensure timely treatment.

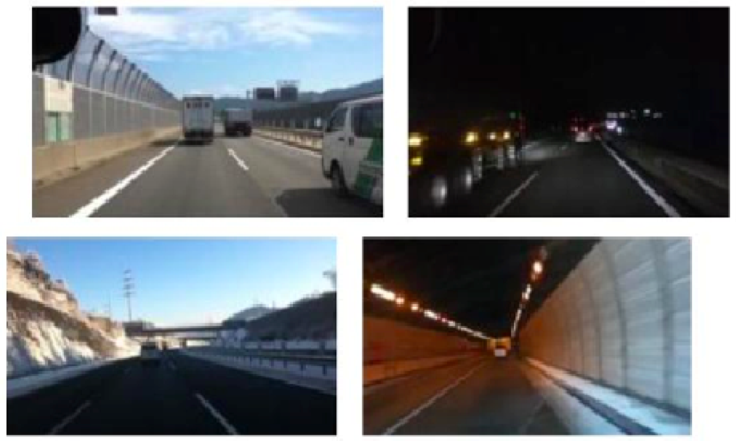


Figure 3.9: Samples of the iRoads dataset (Rezaei and Terauchi, 2013).

In this experiment we consider the COVID CT-scan dataset for Covid-19 identification. This dataset is composed of 2482 CT-scans slices, which is divided between 1252 for patients infected by SARS-CoV-2, and 1230 CT-scans slices for non-infected by SARS-CoV-2 patients, but who presented other pulmonary diseases (Soares et al., 2021). Data was collected from March 15 to April 15 2020 in the Public Hospital of the Government Employees of Sao Paulo - Brazil. The detailed number of patients is illustrated by Table 3.1. Fig. (3.10) illustrates some examples of CT-scans slices for patients infected and non-infected by SARS-CoV-2 that composes the dataset.

Patients	Infected	non-Infected
Male	32	30
Female	28	30
Total	60	60

Table 3.1: This table demonstrates the number of patients considered to compose the dataset. In this case, it was considered data of 60 patients infected by SARS-CoV-2, out of which 32 were male and 28 were female. It was also considered data of 60 patients not infected by SARS-CoV-2, out of which 30 were male and 30 were female.

The inclusion criteria are listed as follows:

- Patients with a positive new coronavirus nucleic acid antibody and confirmed by the RT-PCR test;

- Patients who underwent thin-section CT;
- Age>= 18;
- Presence of lung infection in CT slices;
- Patients who were suspicious of having COVID-19, but tested negative for SARS-CoV-2 nucleic acid antibody.

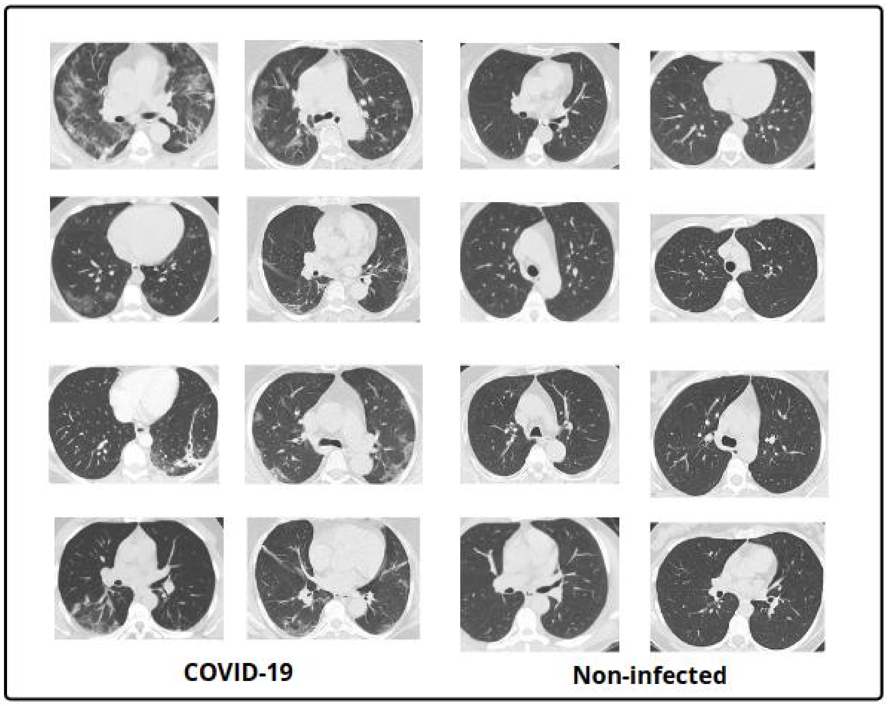


Figure 3.10: The figure illustrates the example of CT-scans slices for different patients infected and non-infected by SARS-CoV-2. The first two columns refers to CT scans slices of patients infected by SARS-CoV-2, and the two last columns refers to CT-scans slices of patients non-infected by this virus.

It is important to highlight that chest CT should only be arranged for individuals with certain clinical features in conjunction with RT-PCR tests.

3.3.4.1 Image acquisition parameters

The median duration from the onset of the illness to CT scan was 5 days, ranging from 1 to 14 days. The CT protocol was as follows: 120 kV; automatic tube current

(180 mA-400 mA); iterative reconstruction; 64 mm detector; rotation time, 0.35 sec; slice thickness, 5 mm; collimation, 0.625 mm; pitch, 1.5; matrix, 512×512; and breath hold at full inspiration. The reconstruction kernel used is set as "lung smooth with a thickness of 1 mm and an interval of 0.8 mm". During reading, the lung window (with window width 1200 HU and window level-600 HU) was used.

3.3.5 COMPAS dataset for Fairness

The Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) risk assessment tool, has been developed in 1998, and since then has been used to assess more than 1 million offenders (Dressel and Farid, 2018).

In a study conducted by ProPublica (ibid.) that analyzed the efficacy of COMPAS on more than 7000 individuals arrested in Broward County, Florida between 2013 and 2014 it was found that the likelihood of a non-recidivating black defendant being assessed as high risk is nearly twice that of white defendants. African-american defendants who did not recidivate were incorrectly predicted to reoffend at a rate of 44.9%. On the other hand, their white counterparts were incorrectly classified with 23.5%. Moreover, white defendants who recidivated were incorrectly predicted as not high risk to reoffend with 47.7%, African-americans who recidivated were incorrectly predicted as not high risk to reoffend with 28.0%. These findings indicate that the COMPAS instrument has considerably higher false positive rates and lower false negative rates for black defendants than for white defendants (ibid.).

3.3.6 Heart sound classification

The 'PhysioNet' dataset contains a total of 13015 samples of heart sound recordings, lasting from 5 seconds to just over 120 seconds. Recordings were collected from different locations on the body, including aortic area, pulmonic area, tricuspid area and mitral area. The collected heart sound recordings were divided into two types: normal and abnormal heart sound recordings. The normal recordings were from healthy subjects and the abnormal ones were from patients with a confirmed cardiac diagnosis. Fig. (3.11) illustrates the normal and abnormal heart sound over time, while Fig. (3.12) shows the power spectrum over the normalized frequency for both normal and abnormal heart sound conditions (Liu et al., 2016; PhysioToolkit, n.d.). It is important to highlight that the 'PhysioNet' dataset is imbalanced as it contains 3158 samples of normal condition heart sounds and 9857 samples of abnormal sounds.

3.3.6.1 Pre-Processing of the heart sound dataset

The following types of features were extracted from the heart sound recordings:

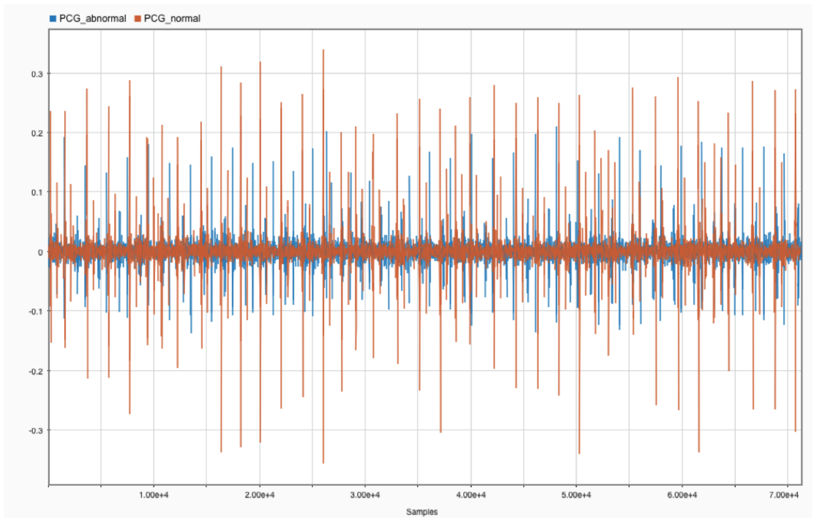


Figure 3.11: Normal and abnormal heart sound over time. Where the blue line represents the abnormal class, and the orange line refers to the normal class.

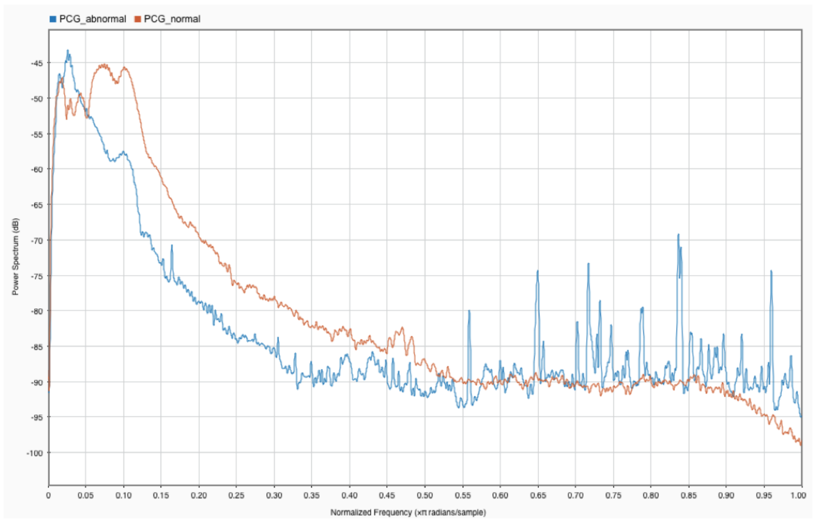


Figure 3.12: Power spectrum over the normalized frequency

- Statistical features: mean, median, and standard deviation.
- Signal processing features: dominant frequency, spectrum entropy, and Mel Frequency Cepstral Coefficients (MFCC).

Dominant frequency refers to the most relevant frequency in the sound spectrum

(Atienza et al., 2009). Spectrum entropy is defined as a measure of its spectral power distribution, and it is based on the Shannon entropy (Sharma and Parey, 2016). Spectrum entropy treats as a probability distribution the signal's normalized power distribution in the frequency domain. Then, it calculates the Shannon entropy of it, see (Pan et al., 2009) for detailed proof for spectrum entropy.

Mel Frequency Cepstral Coefficients is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency (Logan et al., 2000).

MFCCs are commonly derived as follows (ibid.):

- Divide the signals into frames
- Take the Fourier transform of each signal.
- Take the logs of the amplitude spectrum.
- Take the discrete cosine transform of the list of logs generated in the previous step.
- The MFCCs feautres are the amplitudes of the resulting spectrum.

Therefore, 27 features extracted from the audio recordings signals are described in Table 3.2.

Table 3.2: Features Summary

Features	*Quantity*	*Type*
Mean	1	Statistical
Median	1	Statistical
Standard Deviation	1	Statistical
Mean Absolute Deviation	1	Statistical
Quantile 25	1	Statistical
Quantile 75	1	Statistical
Signal IQR	1	Signal Processing
Sample Skewness	1	Statistical
Sample Kurtosis	1	Statistical
Signal Entropy	1	Signal Processing
Spectral Entropy	1	Signal Processing
Dominant Frequency Value	1	Signal Processing
Dominant Frequency Magnitude	1	Signal Processing
Dominant Frequency Ratio	1	Signal Processing
MFCC	13	Signal Processing

3.4 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. All the experiments were conducted with Python 3.6 using a personal computer with a 1.8 GHz Intel Core i5 processor, 8-GB RAM, and MacOS operating system. The default parameters were used for the algorithms used in the comparisons.

3.4.1 Performance Evaluation

The following metrics were used to assess the algorithms used in the experiments considered in this section:

Accuracy:

$$ACC(\%) = \frac{TP + TN}{TP + FP + TN + FN} \times 100, \tag{3.12}$$

Precision:

$$Precision(\%) = \frac{TP}{TP + FP} \times 100, \tag{3.13}$$

Recall:

$$Recall(\%) = \frac{TP}{TP + FN} \times 100, \tag{3.14}$$

Specificity:

$$Specificity(\%) = \frac{TN}{TN + FP} \times 100, \tag{3.15}$$

$F1$ Score:

$$F1\ Score(\%) = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100, \tag{3.16}$$

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.

3.4.2 Caltech-256 and Caltech-101 Dataset

Results for Caltech-256 are presented in Table 3.3. Both Caltech-256 and Caltech-101 datasets were divided into 80% for training and 20% for testing of the algorithms. The algorithms used for comparison are set to their default parameters.

Table 3.3: Performance Comparasion: Caltech-256 Dataset

Method	*Accuracy*	Parameters
xDNN	**75.41%**	259
MSVM (Cao et al., 2019)	70.18%	–
VGG–16 (He et al., 2016)	73.2%	138,000,000
VGG–19 (He et al., 2016)	70.62 %	138,000,000
ResNet–101 (Simonyan and Zisserman, 2014)	75.14 %	60,200,000
GoogleNet (Szegedy et al., 2015)	72.42 %	7,000,000
Softmax(7) (Zeiler and Fergus, 2014)	74.2%	–

Results presented in Table 3.3 demonstrate that the 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 3.4 showed that the proposed approach could surpass other state-of-the-art approaches in terms of accuracy.

Table 3.4: Performance Comparison: Caltech-101 Data set

Method	*Accuracy*	Parameters
SPP-net (He et al., 2015)	91.44%	35,000,000
xDNN (Angelov and Soares, 2020b)	90.62%	259
VGG–VD–16	90.32%	138,000,000
KNN	85.65%	4572
DT	54.42%	102

We compared the proposed xDNN approach with the best published single-label classifiers methods and achieved better result. There are couple of alternative methods that report higher results on Caltech problems, but they use additional information such as the context (Leng et al., 2019) or multiple labels 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.

3.4.3 iRoads Dataset

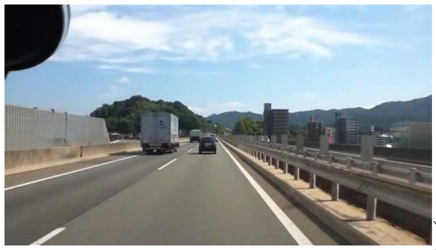
The iRoads dataset was divided into 80% for training of the algorithms and 20% for testing. Algorithms used for comparison use their default parameters and structure. Table 3.5 shows that the xDNN method provides the best result in terms of classification accuracy as well as time/complexity and simplicity 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.

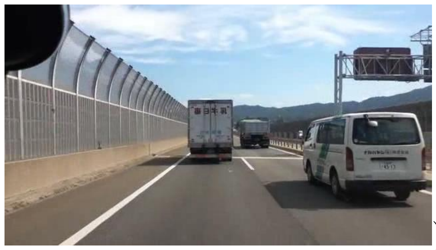
The explainable DNN surpasses in terms of accuracy the state-of-the-art VGG–16 algorithm which is a well-established convolutional deep neural network. Moreover, the proposed xDNN has at its top layer a set of a very small number of *MegaClouds* (27 or, on average, 4 *MegaClouds* per class) which makes it very easy to explain and visualize. For comparison, our earlier version of deep rule-based models, called DRB (Angelov and Gu, 2018a) 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., 2019a). 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.

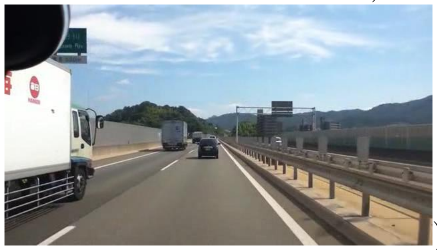
Table 3.5: Performance Comparasion: iRoads Dataset

Method	*Accuracy*	Time(s)	# Parameters
xDNN	**99.59%**	4.32	**27**
VGG–16 (He et al., 2016)	99.51 %	836.28	138,000,000
DRB (Angelov and Gu, 2019)	99.02%	**2.95**	521
SVM (Suykens et al., 1999)	94.17%	5.67	Not reported
KNN (Bishop, 2006)	93.49%	4.43	4656
Naive Bayes (Bishop, 2006)	88.35%	5.31	9313

MegaClouds generated by the proposed xDNN model can be visualized in terms of rules as illustrated by the Fig. (3.13).

IF (I $\sim$ ) OR

(I $\sim$ ) OR

OR (I $\sim$ )

THEN ‘Daylight scene’

Figure 3.13: xDNN rule generated for the ‘Daylight scene’.

3.4.4 COVID-19 identification

This subsection illustrates the results obtained by the xDNN for COVID-19 classification via CT-scan. The dataset was divided into 80% for training and 20% for testing purposes. The division has been made in terms of patients; therefore, we separated data of 96 patients for training and data for 24 patients for testing. Algorithms used in this comparison use their default parameters and structure. Results presented in Table 3.6 compare the performance of the xDNN algorithm with other state-of-the-art approaches, including traditional (*black-box*) deep neural network, Decision Tree, and AdaBoost.

Metric / Method	Accuracy	Precision	Recall	Specificity	F1 Score	AUC
xDNN	**97.38%**	**99.16%**	95.53%	**96.42%**	**97.31%**	**97.36%**
ResNet	94.96%	93.00%	**97.15%**	94.36%	95.03%	94.98%
GoogleNet	91.73%	90.20%	93.50%	90.17%	91.82%	91.79%
VGG-16	94.96%	94.02%	95.43%	94.51%	94.97%	94.96%
AlexNet	93.75%	94.98%	92.28%	92.32%	93.61%	93.68%
Decision Tree	79.44%	76.81%	83.13%	77.16%	79.84%	79.51%
AdaBoost	95.16%	93.63%	96.71%	94.98%	95.14%	95.19%

Table 3.6: Results considering different methods for the COVID-19 identification

The xDNN classifier provided better results in terms of all metrics than the other state-of-the-art approaches, including ResNet, GoogleNet, VGG-16, and Alexnet. Moreover, it also provided highly interpretable results that may be helpful for specialists (medical doctors).

Rules generated by the identified prototypes for COVID and Non-COVID patients are illustrated by Figs. (3.14) and (3.15), respectively. xDNN identified data of **18 patients with COVID-19 as prototypes and data of 11 patients non-infected as prototypes**. The training time for the xDNN algorithm was only 11.82 seconds for all images (an average of 5 milliseconds per image. On the other hand, the traditional deep learning approach may take hours for the same task and usually requires hardware accelerators such as GPUs and once trained is not flexible to new data. We have to stress that xDNN does not require full re-training if new data is presented- it keeps all prototypes identified so far and may add new if the data pattern requires that.

Balanced one-way ANalysis Of VAriance (ANOVA) (McHugh, 2011) was used to compare the results in terms of accuracy provided by the classification methods. The null hypothesis is that the accuracy results provided by the methods are the same. A cutoff value p less than 0.05 suggests that the accuracy of at least one of the algorithms is significantly different from the others. A $p = 4.38e-22$ was obtained and, therefore, the mean accuracy of the algorithms are not all the same; the null hypothesis was rejected.

The Tukey Honestly Significant Difference (HSD) test (McHugh, 2011) was performed to compare pairs of classifiers. Table 3.7 shows the results of the Tuckey HSD test for a 95% confidence interval for the true difference of the means.

Method 1	Method 2	meandiff	p-adj	lower	upper	Reject
xDNN	Resnet	-2.28	0.068	-4.6604	0.1004	False
xDNN	GoogleNet	-5.6583	0.001	-8.0387	-3.278	True
xDNN	Vgg16	-2.385	0.0493	-4.7654	-0.0046	True
xDNN	Alexnet	-3.7567	0.001	-6.137	-1.3763	True
xDNN	DT	-17.8783	0.001	-20.2587	-15.498	True
xDNN	Adaboost	-2.0583	0.1272	-4.4387	0.322	False
Resnet	GoogleNet	-3.3783	0.0015	-5.7587	-0.998	True
Resnet	Vgg16	-0.105	0.9	-2.4854	2.2754	False
Resnet	Alexnet	-1.4767	0.4709	-3.857	0.9037	False
Resnet	DT	-15.5983	0.001	-17.9787	-13.218	True
Resnet	Adaboost	0.2217	0.9	-2.1587	2.602	False
GoogleNet	Vgg16	3.2733	0.0023	0.893	5.6537	True
GoogleNet	Alexnet	1.9017	0.1912	-0.4787	4.282	False
GoogleNet	DT	-12.22	0.001	-14.6004	-9.8396	True
GoogleNet	Adaboost	3.6	0.001	1.2196	5.9804	True
Vgg16	Alexnet	-1.3717	0.5491	-3.752	1.0087	False
Vgg16	DT	-15.4933	0.001	-17.8737	-13.113	True
Vgg16	Adaboost	0.3267	0.9	-2.0537	2.707	False
Alexnet	DT	-14.1217	0.001	-16.502	-11.7413	True
Alexnet	Adaboost	1.6983	0.3061	-0.682	4.0787	False
DT	Adaboost	15.82	0.001	13.4396	18.2004	True

Table 3.7: Tukey Test Results

If the $p-adj < 0.05$ than the null hypothesis is rejected and the difference between the methods are statistically significant. As shown in Table 3.7 the proposed xDNN has results statistically different from 4 traditional approaches, including well known deep learning approaches as GoogleNet, VGG-16, and AlexNet.

Through the xDNN method we generated (extracted form the data) linguistic *IF...THEN* rules which involve actual images of both cases (COVID-19 and NO COVID-19) as illustrated in Figs. (3.14) and (3.15). Such transparent rules can be used in a clear 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.

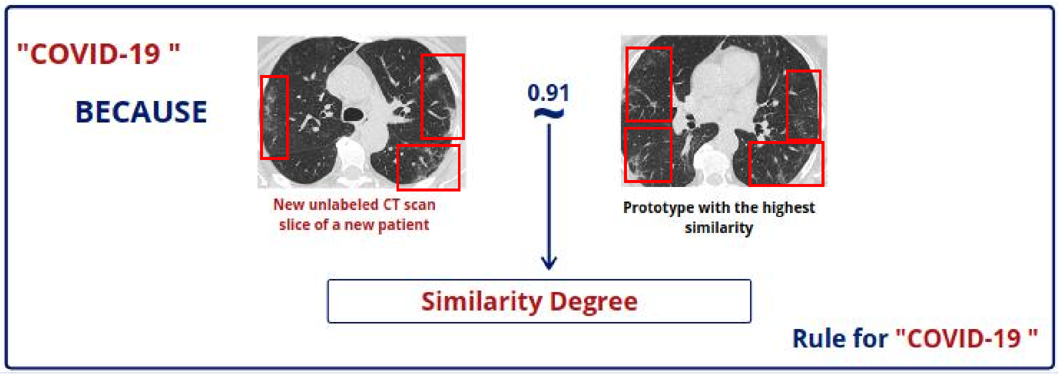

"COVID-19 "
BECAUSE
New unlabeled CT scan slice of a new patient
0.91
~
Prototype with the highest similarity
Similarity Degree
Rule for "COVID-19 "

Figure 3.14: Final rule given by xDNN classifier for the COVID-19 identification. Differently from typical deep neural networks, xDNN provides highly interpretable rules which can be visualised and used by human experts for the early evaluation of patients suspected of COVID-19 infection. The classification is done based on the similarity of the unlabeled CT scan slice to the identified prototypes.

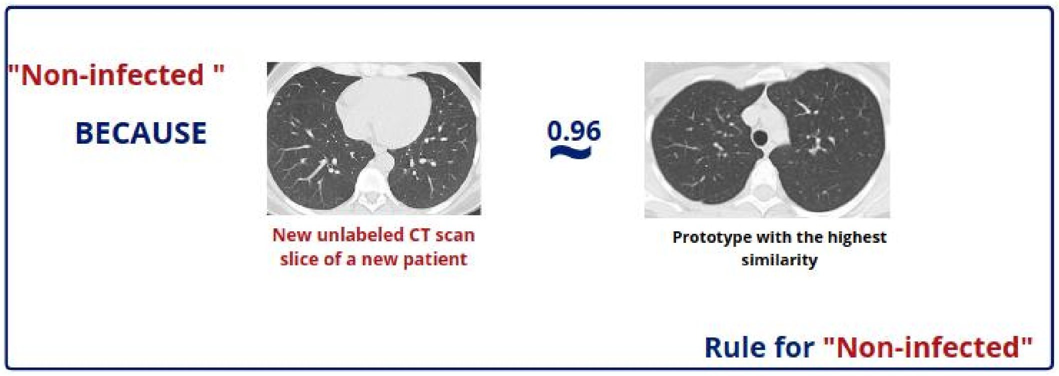

"Non-infected "
BECAUSE
New unlabeled CT scan slice of a new patient
0.96
~
Prototype with the highest similarity
Rule for "Non-infected"

Figure 3.15: Non-SARS-CoV-2 final rule given by the proposed eXplainable Deep Learning classifier.

The xDNN classifier demonstrated the best results in terms of performance than other state-of-the-art approaches, presenting an $F1$ score of 97.31% for the best case. Moreover, it also provides explanations in the form of *IF...THEN* rules using actual images of CT-scans with and without COVID-19. This is of great importance for medical specialists to understand and diagnose COVID-19 at early stages via computed tomography. In summary, the advantages of the proposed method include: i)high precision as compared with the top state-of-the-art algorithms; ii) high level of explainability (prototype-based, rule-based, visualisation using actual images); iii) no user- or problem- specific algorithmic meta parameters required.

3.4.5 Results for COMPAS dataset

The dataset used in this analysis contains a COMPAS recidivism risk decile scores, 2-year recidivism outcomes, and a number of demographic and crime-related variables on individuals who were scored in 2013 and 2014. The results obtained in this experiment are compared with the overall accuracy and groups accuracy with results provided by (Dressel and Farid, 2018). We also compare results on false positives (a defendant is predicted to recidivate but they do not) and false negatives (a defendant is predicted to not recidivate but they do). In this case, 80% of dataset was used for training and 20 % for testing of the algorithm.

In Table 3.8, LR_7 refers to logistic regression with 7 features, LR_2 is the logistic regression with 2 features, and NL-SVM refers to nonlinear SVM as defined by (ibid.).

As shown in Table 3.8, xDNN obtained a better performance in terms of overall accuracy than its competitors. Moreover, it is important to highlight that xDNN works per class, in parallel, therefore, it can obtain a more balanced result than other state-of-the-art approaches. As shown in Table 3.8, xDNN reduced the false positive rate for black people from 31.6% (best case with NL-SVM) to 30.2%, additionally, the false negative rate, when a defendant is predicted to not recidivate but they do, for white population has decreased from 46.1% in the best scenario with LR_2 to 29.6%.

Table 3.8: Experimental results

Results	LR$_7$	LR$_2$	NL-SVM	COMPAS	xDNN
Accuracy (overall)	66.6%	66.8%	65.2%	65.4%	**67.7%**
Accuracy (black)	66.7%	66.7%	64.3%	63.8%	**67.90%**
Accuracy (white)	66.0%	66.4%	65.3%	67.0%	**68.5%**
False positive (black)	42.9%	45.6%	31.6%	44.8%	**30.2%**
False positive (white)	25.3%	25.3%	20.5%	23.5%	36.8%
False negative (black)	24.2%	21.6%	39.6%	28.0%	34.1%
False negative (white)	47.3%	46.1%	56.6%	47.7%	**29.6%**

The proposed approach is prototype-based and it learns locally around the prototypes extracting the empirical data distribution called typicality (Angelov and Gu, 2019) as well as the data density.

The rule-based characteristic of the proposed method allows interpretability and explainability of the data. The prototypes identified for the 'Two year recidivism for black people' rule are demonstrated on Table 3.9.

Table 3.9: Identified Prototypes for the 'Two year recidivism for black people' rule

Features	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8	π_9
Number of Priors (NP)	14	13	8	8	10	7	2	0	21
Score Factor (SF)	1	1	0	1	0	1	0	0	1
Age above 45 (A45)	0	1	0	0	0	0	0	0	0
Age below 25 (A25)	0	0	0	0	0	0	0	1	0
African American (AA)	1	1	1	1	0	1	1	1	1
Female (F)	0	0	0	0	1	1	1	0	0
Misdemeanor (M)	0	0	0	1	0	0	1	1	0

Therefore, xDNN allows interpretability in terms of rule as follows:

$$\text{R:}\quad \text{IF}\quad \left(\begin{bmatrix} NP \\ SF \\ A45 \\ A25 \\ AA \\ F \\ M \end{bmatrix} \sim \begin{bmatrix} 14 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}\right) \text{ OR } \left(\begin{bmatrix} NP \\ SF \\ A45 \\ A25 \\ AA \\ F \\ M \end{bmatrix} \sim \begin{bmatrix} 13 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}\right) \text{ OR...OR } \left(\begin{bmatrix} NP \\ SF \\ A45 \\ A25 \\ AA \\ F \\ M \end{bmatrix} \sim \begin{bmatrix} 21 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}\right)$$
$$\text{THEN 'Two year recidivism'}$$

The parallel training and prototype nature of the xDNN approach favours fairer results than traditional approaches that use only an averaging of the history of the data, and may ignore relevant information about individuals. In contrast, the proposed approach works locally building multiple models that have higher chance to capture more diverse data distribution.

3.4.6 Results for Heart sound classification

To evaluate the performance of the considered methods the following indexes are considered: sensitivity (Se), specificity (Sp), and overall score (MAcc). These metrics are the same considered in the Physionet competition in which this dataset was obtained (Liu et al., 2016; PhysioToolkit, n.d.).

These indexes are calculated as:

$$Se = \frac{TP}{TP + FN}, \tag{3.17}$$

$$Sp = \frac{TN}{TN + FP}, \tag{3.18}$$

$$MAcc = \frac{Se + Sp}{2}. \tag{3.19}$$

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

Sensitivity is considered as an indicator of the classifier's ability to discover the true class. Specificity is considered as a index of the classifier's ability to define other classes. The overall score (MAcc) is given by the mean of sensitivity and specificity indexes.

The receiver operating characteristic (ROC) method is also considered in the analysis. As the ROC method is insensitive to both changes in class distribution and proportion of samples per class it provides a convenient way to evaluate the quality of evolving classifiers in nonstationary environment (Fawcett, 2006).

The 'PhisioNet' dataset was divided into 70% for training and 30% for testing purposes (as in the Physionet competition). Algorithms used during the experiments are with their default parameters and structure.

Table 3.10 summarizes the results obtained by xDNN and its competitors considering the 'Classification of Normal/Abnormal Heart Sound Recordings' dataset provided by Phisionet. Were considered 27 features inputs in the data space in order to determine if the patient heart sound is classified as normal or abnormal.

Table 3.10: Performance Comparasion: Heart sound classification

Method	Se	Sp	$MAcc$
xDNN	0.9082	**0.9526**	**0.9304**
ALMMo-0 (Soares et al., 2020b)	0.7930	0.9430	0.8680
AdaBoost & CNN (Potes et al., 2016)	**0.9424**	0.7781	0.8602
Ensemble of SVMs (Zabihi et al., 2016)	0.8691	0.8490	0.8590
Regularized Neural Network (Kay and Agarwal, 2016)	0.8743	0.8297	0.8520
MFCCs, Wavelets, Tensors & KNN (Bobillo, 2016)	0.8639	0.8269	0.8454
Random Forest + LogitBoost (Homsi et al., 2016)	0.8848	0.8048	0.8448
Ensemble of neural networks (Zabihi et al., 2016)	0.8982	0.9253	0.9117
Deep Structured Features (Rubin et al., 2016)	0.8450	0.8690	0.8380
Matrix norm sparse coding + 20 time-domain features (Whitaker et al., 2017)	0.8867	0.8816	0.8841

Table 3.10 shows that xDNN could obtain better results in terms of Sp and $MAcc$ than its competitors, including ALMMo-0. The AdaBoost & CNN could obtain a better performance in terms of sensitity, in other words, it had a better ability to discover the true class. However, xDNN showed a better performance in terms of specificity (classifier's ability to define other classes), due to its prototype-based nature. Moreover, it had the second best result in terms of sensitivity. Therefore, the proposed approach could obtain the best result in terms of overall score ($MAcc$). Fig. (3.16) illustrates the overall accuracy performance of the best considered approaches.

The area under the ROC curves confirms that xDNN is able to work efficiently in this classification problem, no matter if the distribution is changed to any other distribution or if the dataset is imbalanced. The area above the xDNN ROC curve refers in part to 5.81% of classification error with different assigned labels.

Thanks to its prototype-based nature, medical doctors can easily identify abnormal heart sounds by comparing a patient's sample with the identified prototypes from abnormal samples by xDNN.

Rules generated by the xDNN model provide a very intuitive representation for specialists. Moreover, each of the AnYa type fuzzy rules can be interpreted as a number of simpler fuzzy rules with single prototype connected by 'OR' operators. The transparent process provided by the xDNN model supports understandability of the system, differing from other machine learning approaches, which are called 'black box', since they hide (due to its nature) from users all the insights used to generate the final resulting structure.

Rule for the Normal class in xDNN top layer can be written as following:

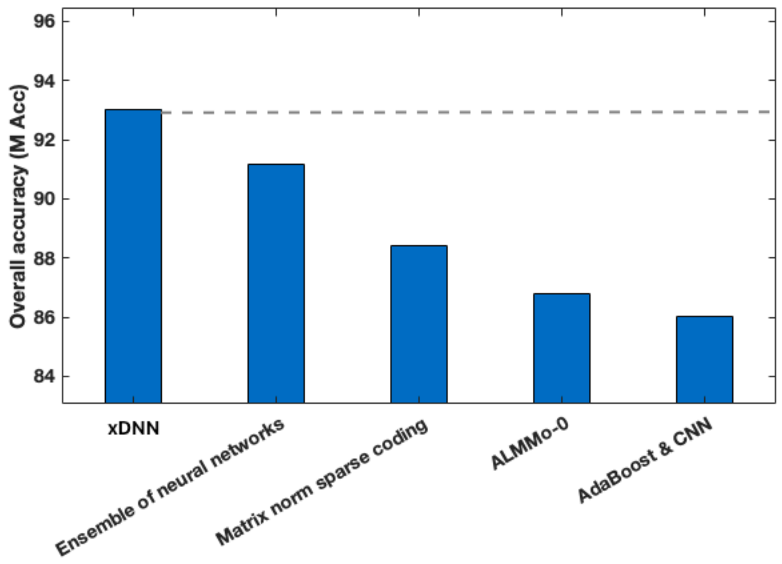


Figure 3.16: Overall accuracy performance of the best considered approaches

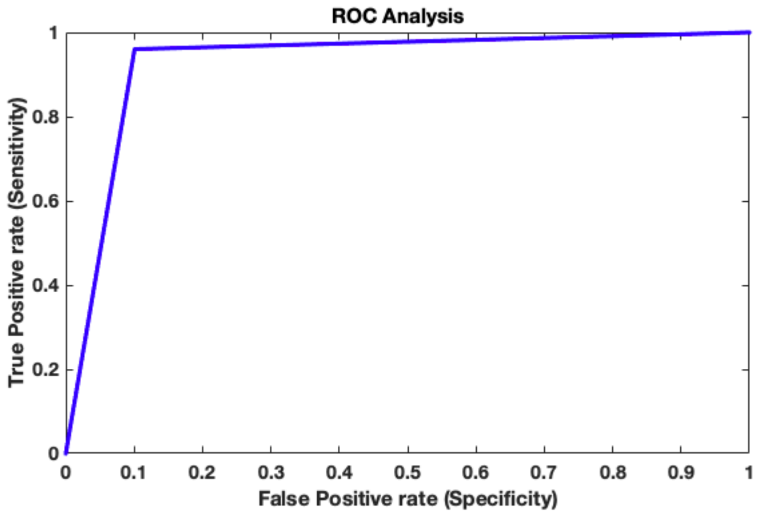


Figure 3.17: ROC analysis for heart sound classification using xDNN

The prototypes identified for the ‘Normal heart sound’ rule are demonstrated on Table 3.11.

In short, experiments have shown that xDNN is an efficient framework for heart sound classification tasks. Classification accuracies were higher than those produced by

$$\text{IF} \quad (\text{x} \sim \pi_1^1) \text{ OR } (x\sim \pi_1^2) \; OR \; (x\sim \pi_1^3) \text{ OR ... OR } (x\sim \pi_1^{20})$$
$$\text{THEN `Normal heart sound'}$$

Table 3.11: Identified Prototypes for the 'Normal heart sound' rule

Features	π_1^1	π_1^2	π_1^3	π_1^{20}
f_1	-2.7121e-05	1.5511e-04	-8.0804e-05	-5.4135e-05
f_2	1.5259e-04	0.0013	0	-1.2207e-04
f_3	0.0203	0.0795	0.0167	0.0096
f_4	0.0123	0.0441	0.0099	0.0057
f_5	-0.0084	-0.0220	-0.0067	-0.0034
f_6	0.0082	0.0237	0.0063	0.0030
f_7	0.0166	0.0457	0.0130	0.0064
f_8	1.4484	-0.4713	0.0276	0.2136
f_9	21.1471	15.7475	22.7916	15.4637
f_{10}	-2.7659	-1.5085	-2.9793	-3.5261
f_{11}	0.2868	0.3124	0.4749	0.3184
f_{12}	17.0982	41.0357	35.6619	21.0064
f_{13}	0.0669	0.0439	0.0278	0.0633
f_{14}	0.2244	0.2951	0.1133	0.2680
f_{15}	88.1961	100.0686	92.9163	87.5767
f_{16}	7.3405	2.4487	4.5780	5.2651
f_{17}	6.4674	7.0189	-2.6415	-4.2657
f_{18}	-0.0512	1.3058	-1.1482	6.2212
f_{19}	-2.5149	-2.9223	-3.8693	4.6041
f_{20}	-3.1430	-2.3074	-6.2024	-4.0199
f_{21}	-1.9638	0.8658	-6.2406	-7.7832
f_{22}	-0.1132	-4.5618	-3.1221	-3.3297
f_{23}	-0.2849	-5.7582	0.8459	-0.3391
f_{24}	1.6218	0.9306	-0.6360	-0.1036
f_{25}	-0.5334	-3.0779	-0.6840	-2.0954
f_{26}	-1.6926	-2.3390	1.9931	-3.0208
f_{27}	-2.0239	-0.8391	0.6190	-0.9700

state-of-the-art approaches considered for this problem not only identifying the major class but also the minor class (which is more important in this case). Differently from the state-of-the-art approaches which are 'black box', the proposed method produced transparent linguistic fuzzy rules, which are human interpretable, and helpful for specialists to make a full diagnosis about the patient situation.

3.5 Conclusion

In this chapter a new explainable-by-design method is presented. The explainable deep neural network (xDNN), directly addresses the bottlenecks of the traditional deep learning approaches and offers an explainable internal architecture. The xDNN approach requires very little computational resources (no need for GPUs) and short training times. xDNN is prototype-based approach where 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. Results have demonstrated that xDNN is able to produce great result on different challenging tasks, even surpassing state-of-the-art competitors as deep learning approaches, moreover, it also presents a transparent view of its decision structure mechanism.

Chapter 4

Deep Machine Reasoning

Traditional DNNs have as decision mechanism a flat *en bloc "winner takes all"* function which is also the last layer of the network. In xDNN, this popular decision concept is also applied.

Differently from traditional decision mechanisms, in this chapter, the Deep Machine reasoning (DMR) approach is introduced. DMR uses a multi-layer decision tree mechanism formed by pairwise comparison of top two classes in terms of minimum error in training. As consequence, the resulting Voronoi tessellation regions of the data clouds that are formed around each prototype (local zones of influence) are significantly different when binary decisions are made.

Furthermore, the DMR approach is also equipped with a mechanism for classes balancing through synthetic data (Gu et al., 2020a).The DMR synthesises data around prototypes which makes these synthetic data more likely to have the same class as the prototype. It starts by identifying a population of pairwise neighbouring data samples from minority classes around prototypes. Then, it imposes a Gaussian disturbance on these data samples, and, finally, it generates synthetic samples by creating linear interpolations between these extrapolations. The training dataset is then augmented through this synthetically generated data.

4.1 DMR Architecture

The architecture of the DMR approach is represented by 2 different steps (training and validation) as detailed in Figs. (4.1) and (4.2). The training phase is performed per class (except the last layer) and includes the following layers:

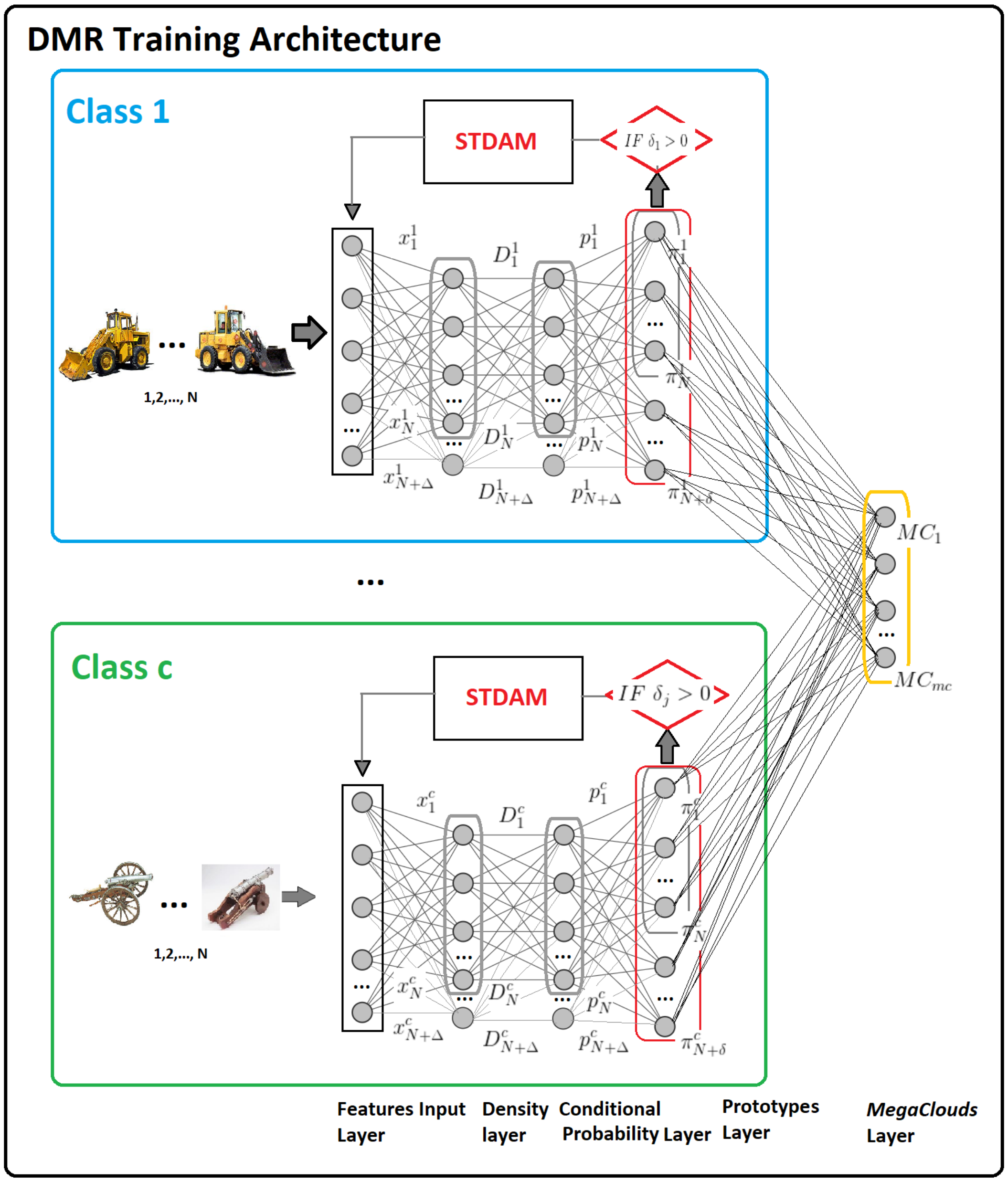


Figure 4.1: DMR Architecture during the training phase (STDAM stands for Synthetic Training Data Augmentation Mechanism).

4.1.1 Input (features) layer

This is the first layer which defines the data space. The number of inputs is determined by the nature of the problem that the data describe. In many problems these are clearly known physical or biomedical variables, e.g. velocities, pressure, temperature, etc. In image processing problems traditionally size, shape of objects or HoG (Mizuno et al., 2012) were used as well as more abstract methods like GIST (Solmaz et al., 2013). More recently, convolutional neural networks (CNN) like AlexNet (Krizhevsky et al., 2012), VGG–VD–16 (Simonyan and Zisserman, 2014), Inception (Szegedy et al., 2015), ResNet (He et al., 2016), Inception–Resnet (Szegedy et al., 2017) have proven to be very efficient to encode images and represent them as a highly abstract vector of the outputs from the Fully Connected Layer (FCL). The proposed DMR architecture is agnostic to the source of the features vector that the input layer represents. It can be any of the above.

4.1.2 Data 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 (Angelov and Gu, 2019) as shows the eq. 4.1:

$$D(x) = \frac{1}{1 + \frac{||x-\mu||^2}{||\sigma||^2}}, \tag{4.1}$$

where D is the density, μ is the global mean, and σ is the variance.

Data density, D represents the proximity of a data sample to the mean. If the data is normalized the range o the density will be in the range $0 < D \leq 1$ where $D = 1$ when $x = \mu$.

4.1.3 Conditional probability layer

The conditional probability or *typicality* τ, is estimated from the empirically observed data (eq.4.2) and its integral $\int_{-\infty}^{\infty} p(C|x)dx = 1$ is likely a multi-modal form of pdf (ibid.):

$$p(y|x) = \frac{\sum_{i=1}^{M} N_i D(x)}{\sum_{i=1}^{M} N_i \int_{-\infty}^{\infty} D(x)dx} \tag{4.2}$$

where N_i is the number of support of the $i - th$ *data cloud*, $\sum_{i=1}^{C}$; $N_i = N$.

4.1.4 Prototypes layer

The prototypes layer is the core of the DMR approach as it provides clear explanations models which can be represented in form of $IF...THEN$ rules. Prototypes, π, are the local peaks of the data density. The proposed DMR algorithm absorbs the new data samples by assigning them to the nearest prototype:

$$j^* = \underset{i=1,..,N;j=1,..,M}{\operatorname{argmin}} |x_i - \pi_j| \tag{4.3}$$

Therefore, each prototype is considered a *cloud* of data that it represents. Prototypes are independent from each other. Therefore, the resulting model's structure is flexible and can be changed (add or remove prototypes) without influencing the existing prototypes. As illustrated by Fig. (4.1), DMR network is trained *per class*, resulting in a set of prototypes *per class*. New prototypes are added when the following condition is satisfied (Angelov and Gu, 2019):

$$\begin{aligned} \mathit{IF}\ (D(x) \geq \max_{j=1,..,M} D(\pi_j)) \\ \mathit{OR}\quad (D(x) \leq \min_{j=1,..,M} D(\pi_j)) \\ \mathit{THEN}\ (\mathit{add\ a\ new\ data\ cloud}\ (j \leftarrow j+1)) \end{aligned} \tag{4.4}$$

If that is the case, then the vector of features of the current training data sample becomes a new prototype, π_{j+1} forms a new *data cloud* (Angelov and Soares, 2020b).

4.1.5 Synthetic data augmentation

This mechanism is not a separate layer, but a feedback process that gets information from the prototypes layer, augments the training data set (in the form of synthetically added features vectors close to the existing prototypes) and expands the size of the prototypes layer by balancing the amount of prototypes per class. This is an augmentation of the amounts of training data (by augmenting N to $N + \Delta$ made by feeding back the information from the prototypes layer. As a result, the size of the prototypes layer is expanded (by δ) so that the number of prototypes per class is being balanced. This is visualised in Fig. (4.1) where the red solid rectangle includes the black dotted one (original prototypes) but also adds prototypes which result from adding synthetic training data.

4.1.6 ***MegaClouds*** layer

In the *MegaClouds*layer the clouds (Voronoi tesselation regions) formed by the prototypes in the previous layer are merged if the neighbouring prototypes have the same class label.

4.2 Learning Procedure

The learning of DMR is summarised below by the following pseudo-code. The proposed architecture is feed-forward with the exception of the synthetic data augmentation mechanism which feeds back form the prototype layer back to the input layer. The proposed method can work both, in a batch mode as well as on a per sample basis, online.

DMR: Learning Procedure

```
1:  Read the first feature vector sample $x_i$ of class $c$;
2:  Standardise and normalise the data as detailed in (Angelov and Gu, 2019)
3:  Set $i \leftarrow 1; j \leftarrow 1; \pi_1 \leftarrow x_i; \mu \leftarrow x_1; N \leftarrow 1$
4:  FOR $i = 2, ...$
5:     Read $x_i$;
6:    Calculate $D(x_i)$ and $D(\pi_j)$ $(j = 1, 2, ..., M)$ according to eq. (4.1);
7:     IF eq. (4.4) holds
8:       Create new prototype: $j \leftarrow j + 1; \pi_j \leftarrow x_i; N \leftarrow N + 1$
9:     ELSE
10:      Search for the nearest prototype according to eq. (4.3);
11:      Update the nearest prototype as:
         $N \leftarrow N + 1;$
         $\pi_j \leftarrow \frac{N_j}{N_j+1}\pi_j + \frac{N_j}{N_j+1}x_i;$
12:        Balance the number of prototypes through synthetic data augmentation
    mechanism detailed below;
13:    END
14: END
```

Synthetic Data Generation

```
1:  FOR $j$ = 1,2,...,C DO
2:    Calculate the amount of synthetic data samples needed to balance the pair of
    classes $j$ and $j + 1$: $\delta = M_j - M_{j+1}$.
3:    UNTIL $\delta = 0$ DO
4:      $k = 1$
5:      Randomly select a pair of neighbouring data samples $(p_k, q_k)^*$ from the $0.3\sigma$
    zone around the prototype from the minority class;
6:        Apply Gaussian disturbance to $(p_k, q_k)^*$ by eq. (4.5) and obtain $(\hat{p_k}, \hat{q_k})^*$
```

```
        (Freudenberg et al., 2010);
            $(\hat{p}_k, \hat{q}_k)^* = (p_k + g_p, q_k + g_q)^*_k$    (4.5)
        where $g_p = [g_{p,1}, g_{p,2}, ..., g_{p,R}]^T$ and $G_q = [g_{q,1}, g_{q,2}, ..., g_{q,R}]^T$ are two $R$ dimensional
        randomly generated vectors sampled from the Gaussian distributions, $g_{p,l}, g_{q,l} \sim$
        $\mathbb{N}(0, \sigma)(l = 1, 2, ..., R)$ with $\sigma$ being the standard deviation.
 7:       Create random interpolation $\rho_k$ between $(\hat{p}_k, \hat{q}_k)^*$ as follows (Gu et al., 2020a):
            $\rho_k = \alpha_k^T \hat{p}_k + (1 - \alpha_k)^T \hat{q}_k$    (4.6)
        where $\alpha_k = [\alpha_{k,1}, \alpha_{k,2}, ..., \alpha_{k,R}]^T$ is a $R$ dimensional random vector, elements of
        which follows the uniform distribution within the range [0,1].
 8:       $k \leftarrow k + 1$
 9:    **END UNTIL**
10: **END FOR**
```

4.3 Multi-layer Decision Structure

The multi-layer decision architecture of DMR (see Fig. (4.2)) is formed by the following layers which are described in the next subsections.

4.3.1 Input (features) layer

The first layer is exact the same as in the training phase.

4.3.2 Ranked prototypes layer

In this layer, all the prototypes are rank ordered in terms of minimum error during the training. Then, the base is organized in terms of overlapping pairs: DMR starts with the top two prototypes (providing smaller error) and then the pair of the second best and the third; further on, the pair of the third and the forth, etc. In this way, all prototypes take part twice except the best one and the worst one, see Fig. (4.2). The output of this layer is the degree of similarity, S between the unlabeled data sample and the respective prototype. The activation functions of the neurons of this layer are defined as follows:

$$S_j = S(x_i, \pi_j) = \frac{1}{1 + \frac{||x_i - \pi_j||^2}{||\sigma_j||^2}}, \tag{4.7}$$

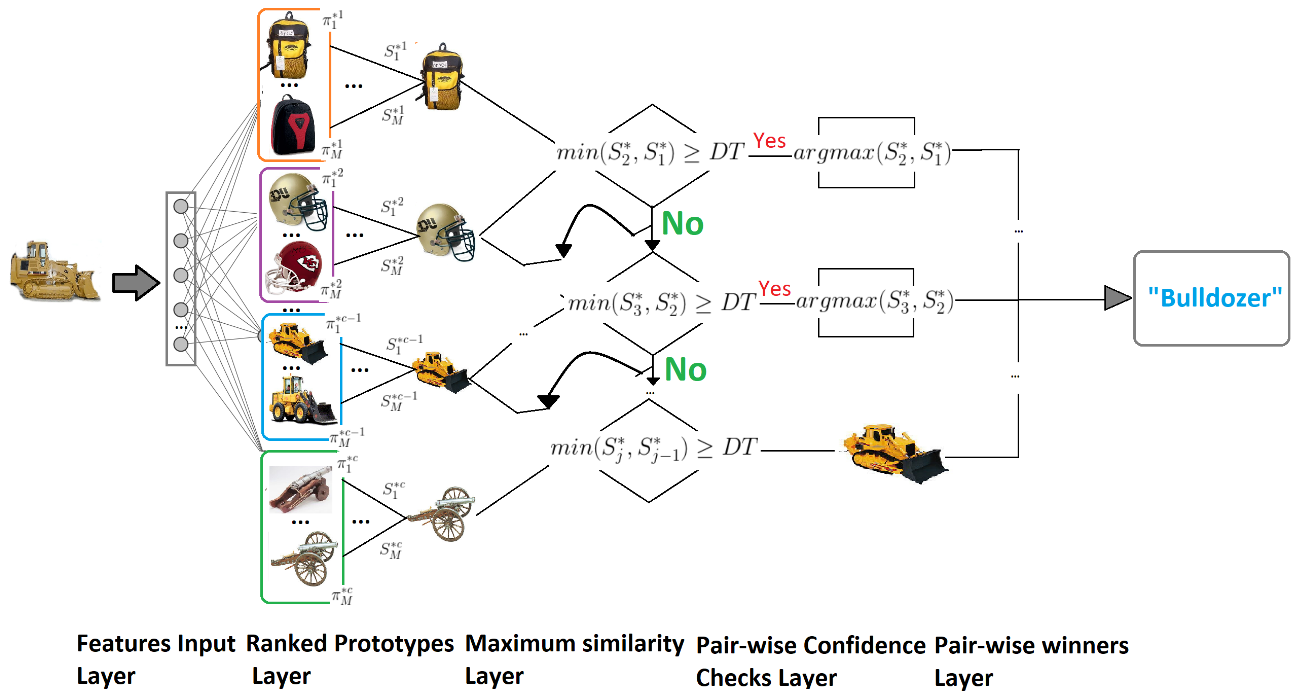


Figure 4.2: Multi-layer decision-making process of the DMR approach.

where $j = 1, 2, ..., M; i = 1, 2, ..., N$. It is easy to see that for similarity we use the same Cauchy function as the data density, eq. (4.1).

4.3.3 Maximum similarity layer

Each neuron of this layer is performing a simple max operation over the pair of similarity values that are coming form the previous layer, namely:

$$S_{j,j-1}^* = max(S_{j-1}, S_j) \tag{4.8}$$

The winner goes forward.

4.3.4 Pair-wise confidence checks layer

In this layer we check if the confidence in the best of the two potential outcomes is high enough. In this research thesis a threshold, Thr=0.9, which means 90% similarity of the new, unlabeled data sample to any prototype is used. The neurons of this layer are linked between each other forming a competitive layer. This link is activated if the confidence check fails (see Fig. 4.2). The flow of the information to the next layer is conditional on the outcome from the confidence check. First, the top two pairs of prototypes are checked. If the winner surpasses Thr it is the winner. Otherwise, the flow goes down to the next pair (in the same layer of the network, the key Fig. 4.2 is closed) and so on.

$$IF\ (min(S^*_{j,j-1}, S^*_{j-1,j-2}) \geq Thr)\ THEN\ (Pair-wise\ winners\ layer)$$
$$ELSE\ (Continue\ the\ tree)$$

4.3.5 Pair-wise winners layer

Pair-wise decisions are made to determine the winning prototype form the candidate pair (S^*_{j-1}, S^*_j), which passed the confidence check in the proceeding layer.

$$Label = argmax(S^*_{j,j-1}, S^*_{j-1,j-2}) \tag{4.9}$$

4.4 Numerical Experiments

The DMR approach is evaluated using several complex, well-known image classification benchmark data sets (Faces-1999, Caltech-101, and Caltech-256). Description of the datasets used for the experiments are given below:

4.4.1 Faces-1999

The Faces-1999 data set (Weber and Weber, 1999) contains 450 frontal real faces images from 27 different people. This data set is highly unbalanced. The resolution of the images are 896 x 592 pixels. Fig. (4.3) illustrates the Faces-1999 dataset.

4.4.2 Caltech-101

The Caltech-101 data set (Griffin et al., 2007) contains 9144 images in divided into 102 categories(one background). The Caltech-101 dataset is highly unbalanced and is widely used as benchmark data set. The Caltech-101 dataset has about 40 to 800 images per category, where most classes have about 50 images. As described in the Chapter 3 of this thesis.

Figure 4.3: Samples of the Faces-1999 dataset (Weber and Weber, 1999).

4.4.3 Caltech-256

Caletch-256 has 30,607 images divided into 257 object categories (one of which is the background) (Griffin et al., 2007). It extends the Caltech-101 dataset adding more categories to it, which makes it more complex to classify. Similar to description for the Caltech-256 datatset presented in the Chapter 3 of this thesis.

4.5 Performance Evaluation

The performance of the classification methods is usually evaluated based on their accuracy metric which is defined as follows:

$$ACC(\%) = \frac{TP + TN}{TP + FP + TN + FN}, \tag{4.10}$$

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

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 training/testing (80% to 20%) sample sets. Default parameters were used for the competitors algorithms.

4.6 Results and Analysis

Computational simulations were performed to assess the accuracy of the proposed explainable tree-based deep learning method (DMR), against other state-of-the-art approaches.

4.6.1 Faces Data set

Table 4.1 shows that the proposed DMR method provides the best result in terms of classification accuracy than its state-of-the-art competitors. In this experiment, we considered a VGG–VD–16 pre-trained on ImageNet; a SVM model with RBF kernel and decision function of shape "one-vs-rest"; a KNN classifier with $k = 5$; and a Decision Tree (DT) with Gini impurity to measure the quality of a split and "log loss" and "entropy" both for the Shannon information gain, nodes are expanded until all leaves are pure or until all leaves contain less than 2 samples.

The number of model parameters for DMR (and xDNN) 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. However, the tree-based structure of the proposed DMR and the mechanism for balancing the classes allow the result to surpass all others. The propose deep reasoning through a layered pair-wise DT is exploiting and benefiting from the old principle of *divide et impera*.

Table 4.1: Performance Comparison: Faces-1999 Data set

Method	*Accuracy*	Parameters/Support
DMR	**96.71** %	39
VGG–VD–16	96.32%	138,000,000
xDNN	96.15%	42
SVM	95.51%	281
KNN	88.54%	225
DT	61.53%	27

4.6.2 Caltech-101 Data set

Table 4.2 shows the results considering the challenging Caltech-101 data set. It is possible to note through Table 4.2 that the proposed DMR method provides the best result in terms of classification accuracy. The proposed Caltech-101 is hugely unbalanced, and the inner data augmentation mechanism of the proposed DMR method favour the balance of the data, consequently, it improves the final classification result. Moreover, the intelligent tree-based structure of the proposed method allows interpretability and also favours the improvement in the classification accuracy of the given model.

The proposed explainable tree-based DNN surpasses in terms of accuracy the state-of-the-art VGG–VD–16 algorithm which is a well-established convolutional deep neural network. Moreover, it could also surpass other state-of-art approaches.

Table 4.2: Performance Comparison: Caltech-101 Data set

Method	*Accuracy*	Parameters/Support
DMR	<u>**94.31%**</u>	241
SPP-net (He et al., 2015)	91.44%	35,000,000
xDNN (Angelov and Soares, 2020b)	90.62%	259
VGG–VD–16	90.32%	138,000,000
KNN	85.65%	4572
DT	54.42%	102

4.6.3 Caltech-256 Data set

Results for Caltech-256 are presented in Table 4.3. Linear SVM and Softmax are trained on features from different layers (as indicated in brackets) from the convnet. Higher layers generally produce more discriminative features, for more details see (Zeiler and Fergus, 2014).

Table 4.3: Performance Comparison: Caltech-256 Data set

Method	*Accuracy*	Parameters
DMR	<u>**77.54**</u>%	562
xDNN (Angelov and Soares, 2020b)	75.41%	596
SVM(1) (Zeiler and Fergus, 2014)	24.6 %	–
SVM(2) (Zeiler and Fergus, 2014)	39.6%	–
SVM(3) (Zeiler and Fergus, 2014)	46.0%	–
SVM(4) (Zeiler and Fergus, 2014)	51.3%	–
SVM(5) (Zeiler and Fergus, 2014)	65.6%	–
SVM(7) (Zeiler and Fergus, 2014)	71.7%	–
Softmax(5)(Zeiler and Fergus, 2014)	65.7%	–
Softmax(7) (Zeiler and Fergus, 2014)	74.2%	–

These results demonstrate that the proposed DMR approach obtains the best classification accuracy ever reported for this complex problem, namely, 77.54%. The proposed approach not only surpasses all published competitors but also offers a clearly explainable model.

IF (Image ∼) OR (Image ∼)OR

... OR (Image ∼) THEN ”Mountain Bike”

DMR even surpasses the recently introduced by us xDNN approach (Angelov and Soares, 2020b), which reported the world best result on 5 December 2019 for this classification problem.

4.7 Conclusion

This chapter presents a prototype-based explainable DNN with DT inference and balanced amount of prototypes per class regardless of the possible imbalances of the training data. The proposed method offers two main novelties, namely: i) using a DT to determine the winning class label, and ii) balancing the classes by synthesising data around the prototypes determined from the available training data. It demonstrates excellent performance surpassing three well known benchmark problems (Caltech-101, Caltech-256 and Faces-1999). The proposed approach is explainable-by-design, computationally efficient (no need for GPUs, high degree of parallelization possible, no iterative search procedures and parameter optimisation). Furthermore, results demonstrated that DMR approach could surpass xDNN on the different classification tasks which demonstrates the efficiency of the novelties presented. It is a step towards bringing closer machine learning and automated reasoning into what we call *deep machine reasoning* aiming not only high levels of accuracy but also deeper understanding and insight about the decision process of the machine learning algorithm.

Chapter 5

Explaining Deep Learning Through Rules

This chapter introduces a new explainable approach to redesign a Deep Reinforcement Learning (DRL) model into a set of $IF...THEN$ rules. The DRL model provides the path planning policy for highway self-driving (Nageshrao et al., 2019). The DRL model maps the set of continuous state variables characterizing the position and velocities of the ego vehicle (EV) and the surrounding vehicles on a divided highway into a set of discrete actions in longitudinal and lateral direction.

State variables include meaningful affordance indicators of the road situation such as the longitudinal and lateral position and velocity of the host vehicle and relative longitudinal and lateral positions and velocities of the surrounding vehicles. The output of the model is a set of eight possible decisions/actions in longitudinal (maintain, accelerate, brake, and hard brake) and lateral (lane keep, change lane to right, and change lane to left) directions - Fig. (5.1).

The main idea is to provide an approximation of the DRL model with an alternative interpretable model with a similar performance. Therefore, the proposed approach is based on the following premises: i) the universal approximation ability of the rule-based models with fuzzy predicates; ii) the better interpretability of the prototype-based fuzzy rules (including visualization).

A density-based method for selecting the most important inputs and a two-stage hierarchical approach to group the adjacent prototypes in the data space is introduced. These two novel techniques allow us to reduce the number of prototypes needed and improve the explainability. This is achieved both linguistically as a set of hierarchical *IF…THEN* rules and through visualisation.

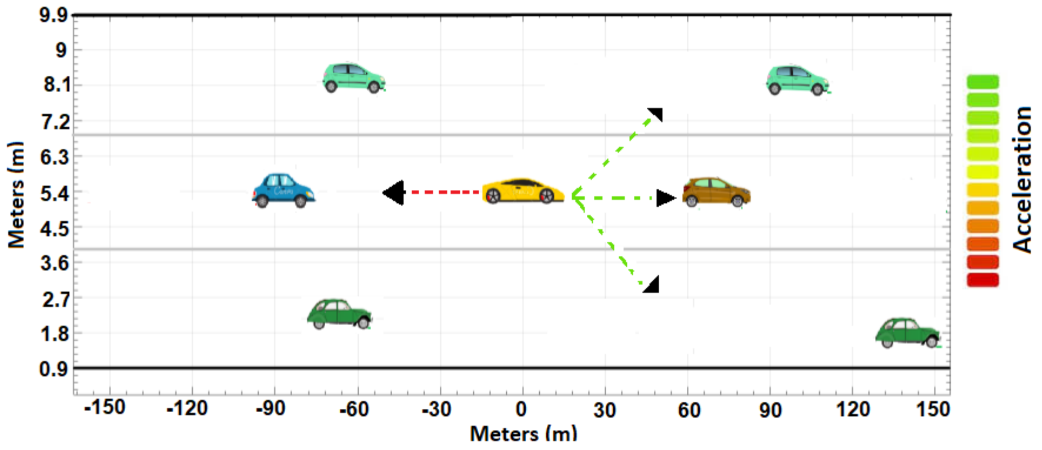


Figure 5.1: Example of host (*ego*) and surrounding vehicles on a highway, where the host vehicle is represented by the center vehicle (yellow car). The forwards arrows indicate the possible directions which the ego vehicle can move. The backwards arrow indicates the brake maneuver.

5.1 General Architecture

Let $T = \{(x_k, a_k)\}_{k=1}^{N}$ be training data set with $x_k \in \mathbb{R}^n$ denoting the state vector and $a_k \in \{1, ..., A\}$ denoting the action vector for each $k \in \{1, ..., N\}$. The layered architecture (Fig. (5.2)) can be seen as a mapping, $f : \mathbb{R}^n \to \mathbb{R}^A$; n is the number of inputs; A is the number of actions; i is the specific data sample/point k; N is the number of training data samples. Separate learning cycles are introduced for each action. Therefore, the data set is split into A sub-sets.

The learning process starts with analyzing the mutual proximity of the data (Angelov and Gu, 2019). As a result, a small number of prototypes are being selected which are actual data samples that are most representative locally (local peaks of data density). When prototypes are being formed only data samples that correspond to the same action are being considered. When prototypes that correspond to different actions are being put together in the data space a further level of analysis is being made, namely merging adjacent (in the data space) prototypes that correspond to the same action together forming so-called "*MegaClouds*". The *MegaClouds* can be visualized and also represented by *IF...THEN* rules. The general architecture of the proposed approach is given in Fig. (5.2).

As a result, we compose A parallel *IF...THEN* rules, each of which corresponds to one of the A actions and has the following form:

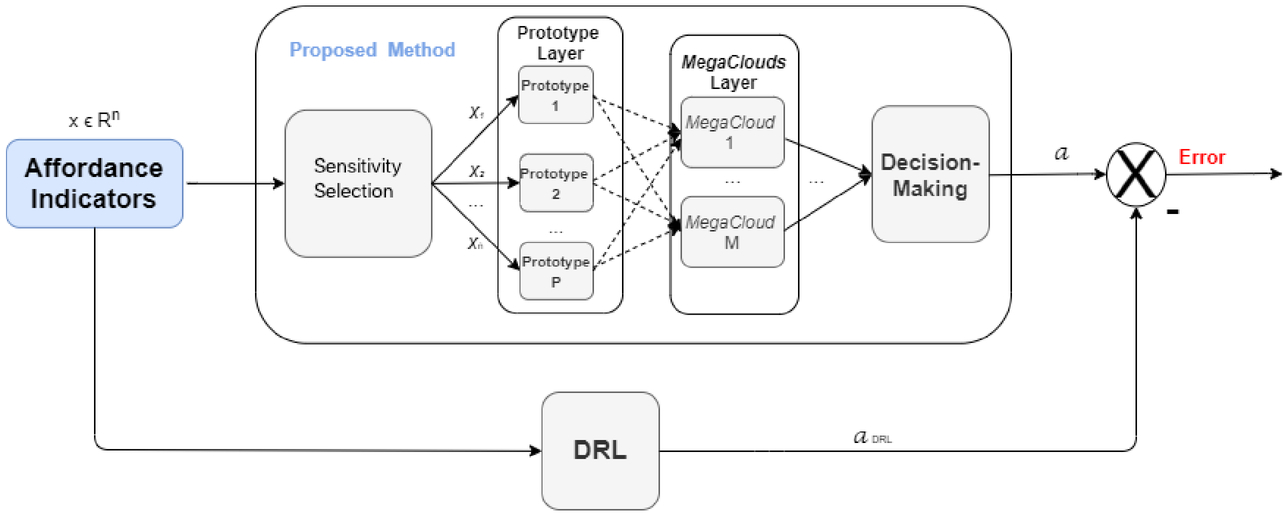


Figure 5.2: General structure of the proposed approach. a_{DRL} refers to the DRL output. The comparison between a and a_{DLR} is used to determine the accuracy of the proposed method.

$$\begin{aligned} \mathrm{R_l}:\ & IF\ (x \sim \pi_l^1)\ OR\ (x \sim \pi_l^2)\ OR\ ...\ OR\ (x \sim \pi_l^{P_l}) \\ & THEN\ (action\ \ l) \end{aligned} \tag{5.1}$$

where π_l^j $(j \in \{1, ..., \pi_l\})$ is the j^{th} prototype of the l^{th} action; P_l is the number of identified prototypes that represent the l^{th} action.

The identified prototypes are connected with logical "OR" (implemented as a $t-$conorm). Each of the conditions within the *IF…THEN* rules are fuzzy rules on their own but all of them have the same consequent pointing to the same action.

5.1.1 Learning rules from the data

The proposed method learns the prototypes associated with each action in separated steps. Therefore, the dataset is split during the training into sub-sets.

As each *IF…THEN* rule is identified separately for each action, unless specifically declared otherwise, all the mathematical notations in the algorithm consider the l^{th} action by default and the index l is omitted for clarity.

5.1.1.1 Pre-processing

Standardize the newly observed data sample, x. Standardization is performed on a per input basis:

$$\bar{x}_i = \frac{x_i - \mu(x_i)}{\sigma(x_i)} \tag{5.2}$$

where $x(i)$ denotes the standardized value of the i-th input of the data sample; $\mu(x_i)$ denotes the mean of the i-th input and $\sigma(x_i)$ denotes the standard deviation of the i-th input; $\mu \in \mathbb{R}^n$ denotes the vector of the mean values and $\sigma \in \mathbb{R}^n$ denotes the vector of the standard deviations.

Following the standardization the data is being normalized converting it to the range $[0, 1]$. Unity-based normalization of the i-th input is given by (Hastie et al., 2009):

$$\bar{x}_i = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)} \tag{5.3}$$

where $\bar{x}_i$ denotes the normalized values of the i-th input.

5.1.1.2 Parameters definition

The algorithm parameters are initialized as soon as the first data sample, $\bar{x}_1$, arrives to the system:

$$\begin{aligned} \mu_1 \leftarrow \bar{x}_1; \quad P \leftarrow 1; \quad \pi^1 \leftarrow \bar{x}_1 \\ \mathrm{C}^1 \leftarrow \{\bar{x}_1\}; \quad S^1 \leftarrow 1; \end{aligned} \tag{5.4}$$

Parameters include: i) mean, μ, initialized with the first normalized data point; ii) the number of prototypes being set to 1; iii) the prototype, π initialized with the first data sample; iv) initialize the first so-called *data cloud*, C^1 as a set of data points that are associated with the first prototype (Angelov, 2012). v) support, S, of the *data cloud* defined as the number of data points associated with a certain *data cloud* (Angelov and Gu, 2019).

Based on this initialization define the first *IF...THEN* rule for the given (l-th action) as follows:

$$\mathrm{R}_l : \quad IF\ (x \sim \pi_l^1)\ THEN\ (action\ l) \tag{5.5}$$

5.1.1.3 Data density calculation

Calculate the data density at the current data point, $\bar{x}$; as in Eq. (5.6).

$$D(\bar{x}) = \frac{1}{1 + \frac{||\bar{x}-\mu||^2}{||\sigma||^2}}; \tag{5.6}$$

where D is the data density, and σ_N denotes the standard deviation. $x = \{\bar{x}, ..., \bar{x}_N\} \in \mathbb{R}^n$ where N is the number of training data samples.

In this step we also identify the prototype π^{j^*} that is nearest to $\bar{x}$:

$$j^* = \underset{j\in\{1,...,P\}}{\operatorname{argmin}} \{||\bar{x} - \pi^j||^2\} \tag{5.7}$$

Then, using the density and the distance to the nearest prototype, π^j, we check the following condition (Angelov and Gu, 2019) based on which we determine if the current data point is going to be added to the set of prototypes or not.

$$\begin{aligned} &IF\ (D(\bar{x}) \geq \max_{j\in\{1,...,P\}}(D(\pi^j))) \\ &OR\ (D(\bar{x}_k) \leq \min_{j\in\{1,...,P\}}(D(\pi^j))) \\ &THEN\ (add\ a\ new\ data\ cloud) \end{aligned} \tag{5.8}$$

where $D(\pi^j)$ is density of the nearest prototype, π^j.

5.1.1.4 Prototype update

New data cloud is added if the condition 5.8 is met:

$$\begin{aligned} P \leftarrow P + 1; \quad \mathrm{C}^P \leftarrow \{\bar{x}\}; \\ \pi^P \leftarrow \bar{x}; \quad S^P \leftarrow 1; \end{aligned} \tag{5.9}$$

Otherwise, if the condition 5.8 is not satisfied, the nearest data cloud parameters are updated as:

$$\begin{aligned} C^{j^*} &\leftarrow C^{j^*} + \{\bar{x}\}; \\ \pi^{j^*} \leftarrow \frac{S^{j^*}}{S^{j^*}+1}\pi^{j^*} &+ \frac{S^{j^*}}{S^{j^*}+1}\bar{x}; \\ S^{j^*} &\leftarrow S^{j^*} + 1; \end{aligned} \tag{5.10}$$

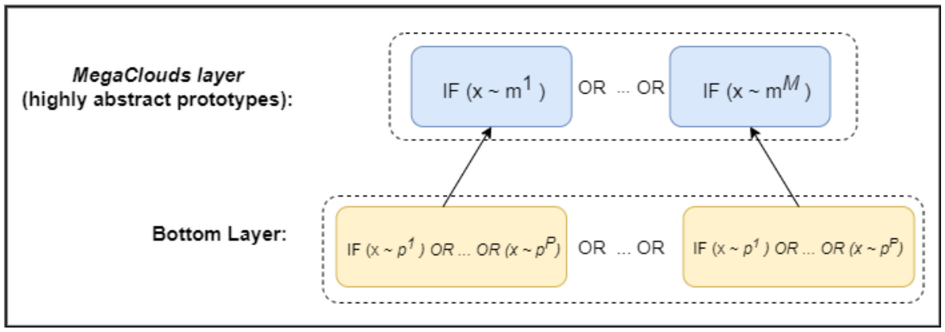

MegaClouds layer
(highly abstract prototypes):
IF (x ~ m^1)
OR ... OR
IF (x ~ m^M)
Bottom Layer:
IF (x ~ p^1) OR ... OR (x ~ p^P)
OR ... OR
IF (x ~ p^1) OR ... OR (x ~ p^P)

Figure 5.3: Hierarchical structure - *MegaClouds*, where m^M is the mean of the M-th *MegaCloud* associated with the l^{th} action

5.1.1.5 Rule update

The *IF…THEN* rule, $R_l$is updated with the identified prototypes:

$$\begin{aligned} R_l : \quad & \mathit{IF}\ (x \sim \pi_l^1)\ \ \mathit{OR}\ (x \sim \pi_l^2)\ \ \mathit{OR}\ ...\mathit{OR}\ (x \sim \pi_l^P) \\ & \qquad\qquad\qquad\qquad\qquad \mathit{THEN}\ (\mathit{action}\ \ l) \end{aligned} \tag{5.11}$$

5.1.2 Hierarchical organisation of the prototypes

Prototypes are organized in a hierarchical manner (Fig. (5.3)). At the bottom layer of the hierarchical architecture is the raw data set. In the layer above it is the set of prototypes (which are selected data points/samples). Each of the prototypes is a focal point in the data space forming *data clouds*, see Fig. (5.4) shaping the so-called Voronoi tessellation (Du et al., 1999). Since the prototypes are defined in isolation (per action) once they are put together in the data space quite often *data clouds* that correspond to the same action (describing *IF…THEN* rules with the same consequent) are adjacent. This allows us to merge these *data clouds* and, respectively fuzzy *IF…THEN* rules into so-called "*MegaClouds*" which have the same consequent (THEN part) and, respectively correspond to the same action. In this way, we minimise the amount of *IF…THEN* rules and improve the interpretablity of the model. We illustrate this in Fig. (5.3).

The membership function (MF) to a *MegaCloud* is formed as a Minkowski type kernel (Lee-Kwang et al., 1994). λ is the parameter of the kernel such that $\lambda \in (-\infty, \infty)$. We found experimentally that order of λ=6 gives best results in terms of accuracy. This can be explained with the fact that the higher the order of λ the more narrow the local Cauchy-type function becomes and less its generalization.

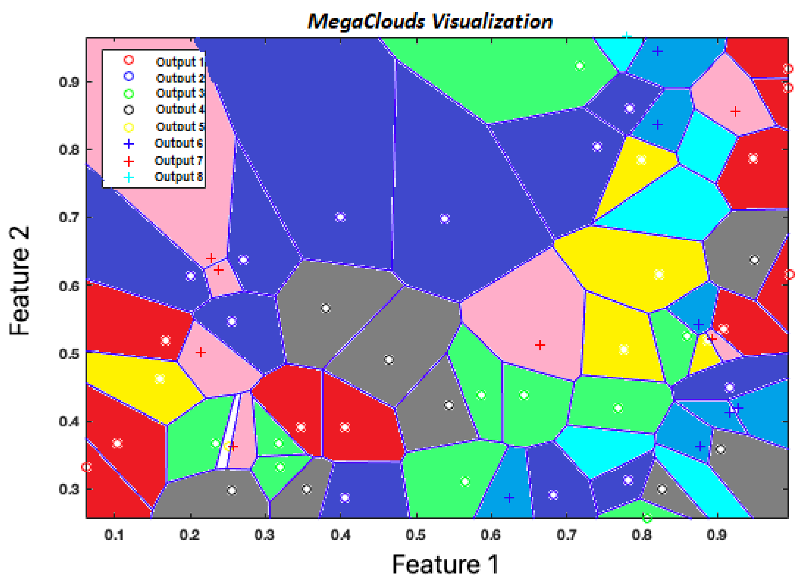


Figure 5.4: *MegaClouds* visualization in terms of Voronoi Tesselation

However, the lower the order of λ is the less accurate it is as it starts to blur with the neighbouring functions that are centred at other prototypes:

$$MF(\bar{x}) = \left(\frac{1}{P} \sum_{j=1}^{P} \left(\frac{1}{1 + \frac{||\bar{x} - \pi^j||^2}{||\sigma^2||}} \right)^{\lambda} \right)^{\frac{1}{\lambda}} \tag{5.12}$$

where MF is the degree of membership of the data point, $\bar{x}$ to the *MegaCloud*.

The membership function is multi-modal: each of its peaks is located at one of the prototypes, defined by a Cauchy-type function as described in Eq. 5.6 One can see that the prototypes, identified earlier, *pi* are the parameters of the MF. One can, however, find new parameters of the *MegaClouds* - the most obvious choice is the mean of each *MegaCloud* (since these Voronoi tessellation areas are adjacent by definition they form a larger area, see Fig. (5.4)), where m^M is the mean, $\sum_{j=1}^{P} \frac{\pi^j}{P}$, of the M-th *MegaCloud* associated with the l^{th} action; $m \in \mathbb{R}^n$.

The *IF...THEN* rules over the *MegaClouds* have the same form, but the key difference is that the number of conditions linked with T-conorm/disjunction (logical OR) are much less. They have the following format:

$$\begin{aligned} \mathrm{R}_l : \quad & IF\ (x \sim m_l^1)\ \ OR\ ...OR\ (x \sim\ m_l^M) \\ & \qquad\qquad\qquad\qquad THEN\ (action\ l) \end{aligned} \tag{5.13}$$

where m^M is the mean, $\sum_{j=1}^{P} \frac{\pi^j}{P}$, of the M-th *MegaCloud* associated with the l^{th} action; $m \in \mathbb{R}^n$.

5.1.3 Density-Based Input Selection

Inputs ranking and selection is a technique to reduce the dimensionality of a problem. A subset of relevant or more descriptive inputs facilitates model interpretation, and can produce better results due to the elimination of inputs that may confound the uncovering of patterns, trends, and relationships.

The contribution of each input using the density of data is estimated per input:

$$D(\bar{x}_i) = \frac{1}{1 + \frac{||x_i - \mu_i||^2}{(\sigma_i)^2}} \tag{5.14}$$

where $D(\bar{x}_i)$ denotes the density for i-th input of $\bar{x}_i$; $(i \in \{1, ..., n\})$.

The cumulative effect across all data samples for each input can be obtained according to the Eq. (5.15).

$$\Lambda_i = \Sigma_{j=1}^{N} D(\bar{x}_i). \tag{5.15}$$

The cumulative contribution for each input Λ_i can be ranked. The higher the value of Λ_i is for a particular input, the more descriptive and important is the i-th input (Soares et al., 2019b). The idea is that interesting inputs have higher density than other inputs - meaning that it conveys unique, different clean information, and as consequence it contributes more to the rule-based model result because the overlap between data clouds of different actions is less pronounced in these inputs. Less descriptive inputs are left one by one based on its Λ_i score until reduction in accuracy performance is noted. This sensitivity selection is helpful to reduce computational complexity which is an advantage especially in online implementation.

5.2 Ford Dataset

The dataset used for learning the rule-based approximation was generated by simulating the DRL model described in (Nageshrao et al., 2019). It contains 256960 instances described by 20 different inputs as described by Table 5.1. The data set is divided into 8 different actions, each action represent a different state of the ego vehicle. The description of the actions are given below:

- Action 1 (Maintain): 217494 samples

- Action 2 (Accelerate by $+2m/s^2$): 12706 samples
- Action 3 (Brake by $-2m/s^2$): 6033 samples
- Action 4 (Hard brake by $-4m/s^2$): 4530 samples
- Action 5 (Lane change to left): 8078 samples
- Action 6 (Lane change left and also brake by $-2m/s^2$): 213 samples
- Action 7 (Lane change right): 7704 samples
- Action 8 (Lane change right and also brake by $-2m/s^2$): 102 samples

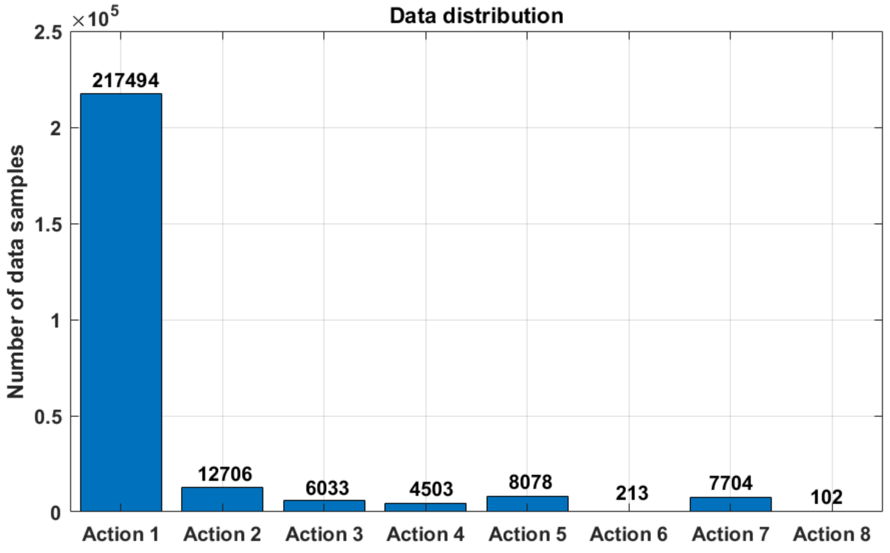


Figure 5.5: Data Distribution in terms of different maneuvers/actions by the ego vehicle, showing the clearly data imbalance nature of the the data set.

The dataset provided by Ford Motor Co. was obtained by a simulating DRL model representing driving policy of a self-driving vehicle in diverse traffic conditions. More details can be found in (Nageshrao et al., 2019).

The data set was divided into 80% for training and 20% for validation purposes as usual for such tasks (Dobbin and Simon, 2011). It is important to highlight that the analyzed dataset is imbalanced as illustrated in Fig. (5.5).

Table 5.1: Description of the inputs

Inputs	Description
1	Ego lateral position
2	Relative velocity between ego and center vehicles
3	Front left vehicle position longitudinal
4	Front left vehicle velocity
5	Front left vehicle lateral position
6	Front center vehicle position longitudinal
7	Front center vehicle velocity
8	Front center vehicle lateral position
9	Front right vehicle position longitudinal
10	Front right vehicle velocity
11	Front right vehicle lateral position
12	Rear left vehicle position longitudinal
13	Rear left vehicle velocity
14	Rear left vehicle lateral position
15	Rear center vehicle position longitudinal
16	Rear center vehicle velocity
17	Rear center vehicle lateral position
18	Rear right vehicle position longitudinal
19	Rear right vehicle velocity
20	Rear right vehicle lateral position

5.3 Performance Evaluation

In order to evaluate the performance of the proposed method the accuracy index is considered. Accuracy is defined as follows:

$$ACC(\%) = \frac{TP + TN}{TP + FP + TN + FN}, \tag{5.16}$$

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

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 experiments were executed using 80% of the data for training and 20% of the data for testing. The following methods were used for comparison: K-nearest Neighbors (KNN) (Cunningham and Delany, 2007), Naive Bayes (NB) (Rish et al., 2001), Support Vector Machine (SVM) (Suykens et al., 1999), Decision Tree (Safavian and Landgrebe, 1991), Random Forest (Breiman, 2001), XGBoost (Chen and Guestrin, 2016), and Catboost (Prokhorenkova et al., 2018). Parameters for XGBoost and CatBoost were optmized through Auto-sklearn hyper-parameter optimization (Feurer et al., 2015). For the other state-of-the-art approaches we consider the default parameters according to their references.

5.4 Results and Analysis

Computational simulations were performed to assess the accuracy of the explainable rule-based approach combined with the sensitivity selection method. Table 5.2 summarizes the results obtained by the proposed method considering different number of inputs.

Table 5.2: Performance Comparison for different actions and number of inputs

Acc / # Inputs [1]	Action 1	Action 2	Action 3	Action 4	Action 5	Action 6	Action 7	Action 8	**Overall**
20	98.46%	85.03%	75.52%	49.46%	80.16%	66.66%	92.96%	50.0%	94.54%
10	98.75%	88.02%	83.06%	91.34%	**91.72**%	72.72%	96.85%	38.09%	97.41%
7	**99.8**%	**93.3**%	**98.5**%	**94.3**%	90.02%	**92.7**%	**95.4**%	66.7%	**98.94**%
5	98.71%	83.97%	81.66%	78.06%	87.91%	80.00%	95.84%	**76.47**%	96.75%
3	98.36%	84.43%	77.92%	39.92%	78.98%	81.08%	93.06%	66.67%	94.89%

Table 5.3: Performance comparison for different actions with 7 inputs with the highest density

Accuracy / Method	Action 1	Action 2	Action 3	Action 4	Action 5	Action 6	Action 7	Action 8	**Overall**
proposed	**99.8**%	**93.3**%	**98.5**%	94.3%	90.02%	92.7%	95.4%	66.7%	**98.94**%
proposed(*MegaClouds*)	99.34%	89.82%	82.9%	81.45%	92.76%	72.0%	**97.31%**	72.72%	97.88%
SVM	88.65%	53.6%	0%	0%	**100.0**%	**100.0**%	74.7%	0%	87.08%
KNN	97.34%	83.1%	85.6%	84.7%	72.6%	**100.0**%	90.2%	70.00%	95.23%
Decision Tree	98.82%	88.16%	83.4%	82.26%	93.36%	82.92%	92.45%	70.00%	97.02%
Adaboost	94.3%	72.9%	88.0%	97.9%	87.9%	0%	86.0%	0%	92.95%
Discriminant analysis	91.0%	42.3%	33.9%	0%	36.6%	32.8%	38.6%	10.3%	85.43%
Random Forest	99.4%	90.2%	88.6%	**98.1%**	94.2%	75.2%	86.0%	**75.4%**	98.31%
XGBoost	99.2%	84.5%	86.9%	87.3%	89.9%	32.8%	89.4%	0%	97.12%
CatBoost	93.2%	78.7%	90.1%	92.2%	88.2%	52.3%	87.1%	0%	91.45%

Table 5.2 shows that, in general, the proposed autonomous method tends to be more accurate when the data space is reduced as we remove inputs with the lowest densities. Because of the parallel nature of the rule-based method, the sensitivity selection is able to work per action. Therefore, the proposed density-based input selection is able to create an individualized subset of inputs per action.

Table 5.4: Description of the 7 inputs with higher densities

Inputs	Description
Rv	Relative velocity between Ego and center vehicles
d^{FL}	Front left vehicle position longitudinal
V^{FL}	Front left vehicle velocity
d^{FC}	Front center vehicle position longitudinal
d^{FR}	Front right vehicle position longitudinal
V^{FR}	Front right vehicle velocity
d^{BC}	Rear center vehicle position longitudinal

Table 5.3, shows that, for the best scenario with 7 inputs (Table 5.4 and Fig. (5.6) details the 7 inputs with higher densities for action 1, similar trend happened with all the actions). The proposed autonomous method reached the best result in terms of overall accuracy compared to the other state-of-the-art methods. The rule-based method outperformed the other state-of-the-art approaches in terms of accuracy for 5 out of 8 actions. It is also possible to note through Table 5.3 that the SVM, Adaboost, Discriminant analysis, XGBoost, and CatBoost were not able to detect some of the actions, mainly actions 6 and 8 where the respective accuracy was 0%. Actions 6 and 8, refer to rare maneuvers during the driving simulation, are extremely difficult to recognise due the highly imbalanced data set, as illustrated in Fig. (5.5).

When all available information is considered as model's inputs it may cause overfitting, and then be prejudicial to the model's accuracy. Therefore, when the proposed density-based input selection method creates individualized subset of the most descriptive inputs per action it helps to overcome the dimensionality problem. For example, the overall accuracy of the proposed recommendation system increases from 94.54% to 98.94% using one third of the original dimensions of the input space, moreover, accuracy per action is also improved. Therefore, one can note that when a self-driving car needs to take an action such as "Lane change right and also brake by $-2m/s^2$" a different subset of inputs will have more impact than when a self-driving car needs to "Lane change left and also brake by $-2m/s^2$" as different maneuvers require different set of actions by the driver/agent.

Fig. (5.7) illustrates the confusion matrix for the best scenario. It is notable from the confusion matrix that even though the data set is imbalanced, the proposed

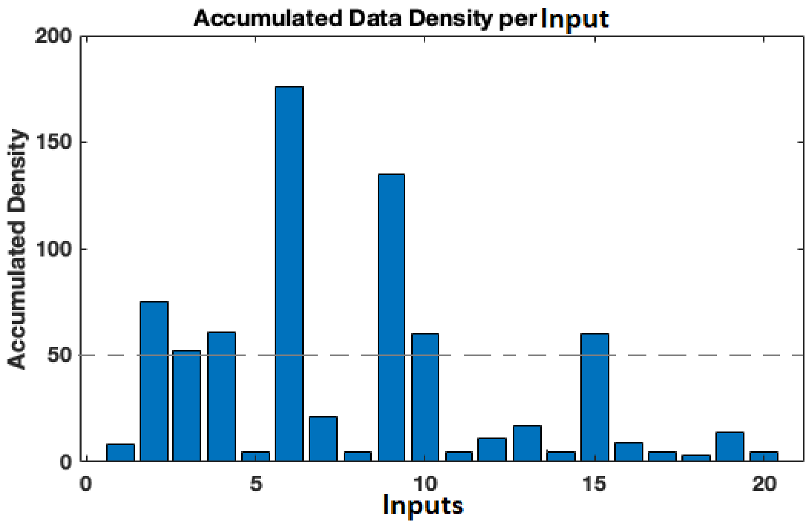
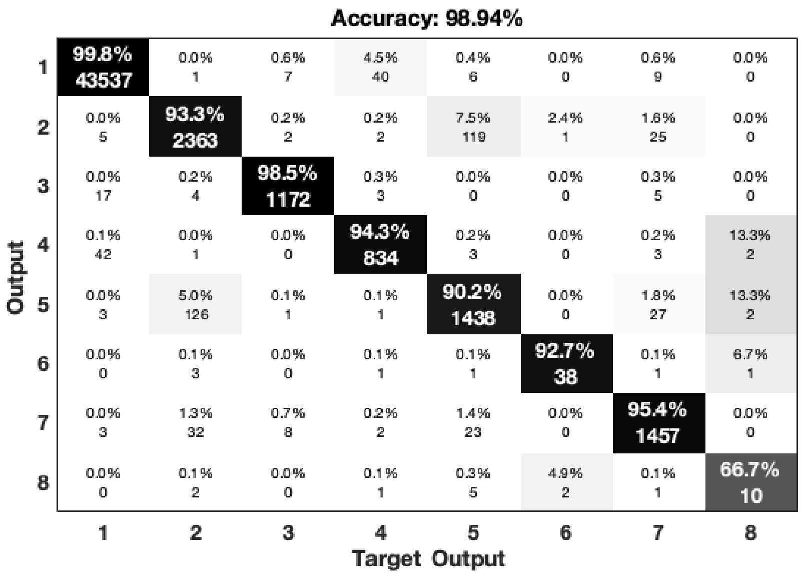


Figure 5.6: Accumulated density histogram, density bars above the dotted line denote the top seven density inputs.

method is able to correctly identify uniformly all the actions.


Accuracy: 98.94%

Output \ Target Output	1	2	3	4	5	6	7	8
1	99.8% 43537	0.0% 1	0.6% 7	4.5% 40	0.4% 6	0.0% 0	0.6% 9	0.0% 0
2	0.0% 5	93.3% 2363	0.2% 2	0.2% 2	7.5% 119	2.4% 1	1.6% 25	0.0% 0
3	0.0% 17	0.2% 4	98.5% 1172	0.3% 3	0.0% 0	0.0% 0	0.3% 5	0.0% 0
4	0.1% 42	0.0% 1	0.0% 0	94.3% 834	0.2% 3	0.0% 0	0.2% 3	13.3% 2
5	0.0% 3	5.0% 126	0.1% 1	0.1% 1	90.2% 1438	0.0% 0	1.8% 27	13.3% 2
6	0.0% 0	0.1% 3	0.0% 0	0.1% 1	0.1% 1	92.7% 38	0.1% 1	6.7% 1
7	0.0% 3	1.3% 32	0.7% 8	0.2% 2	1.4% 23	0.0% 0	95.4% 1457	0.0% 0
8	0.0% 0	0.1% 2	0.0% 0	0.1% 1	0.3% 5	4.9% 2	0.1% 1	66.7% 10

Figure 5.7: Confusion matrix for the best scenario (7 inputs)

Fig. (5.8) illustrates how the performance is affected as the data space is reduced with the removal of inputs with the lowest densities. One can see from Fig. (5.8) that the sensitivity selection helps to improve the computational complexity by reducing the number of inputs (less calculations are required due to the O (log n) nature of the proposed approach).

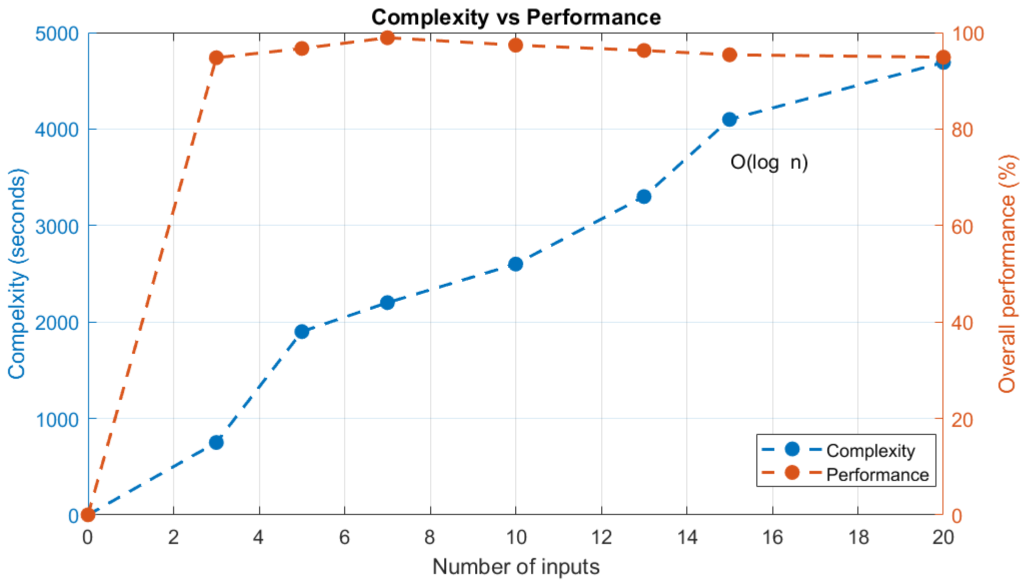


Figure 5.8: Computational complexity vs Overall performance

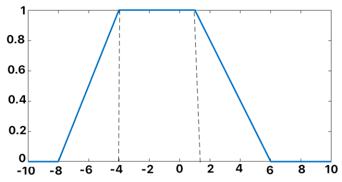
Besides the improvement in terms of accuracy, the hierarchical method contributed to improving interpretability of the proposed approach. It allowed to infer a meaningful approximation of the DRL model and enabled quick evaluation of its performance for specific use cases. Table 5.5 details the number of prototypes produced by the proposed recommendation system considering different layers and 7 inputs (best scenario).

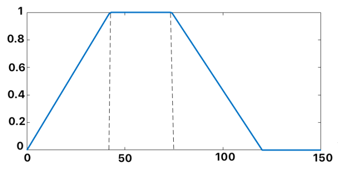
Table 5.5: Number of identified prototypes per Action for different hierarchical layers

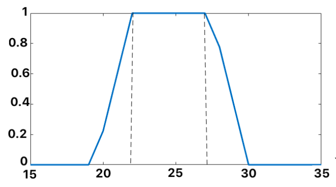
# Prototypes / Layer	Action 1	Action 2	Action 3	Action 4	Action 5	Action 6	Action 7	Action 8	**Total**
Bottom Layer	1315	1009	482	360	649	17	607	4	4443
MegaClouds	13	14	8	10	15	6	11	4	<u>**81**</u>

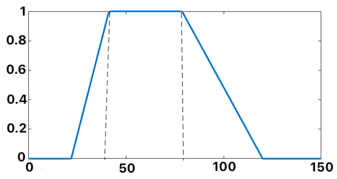
Generated trapezoidal fuzzy rules for the *MegaClouds* layer (highly abstract layer) can be illustrated in terms of inputs as illustrated by Fig. (5.9). It also can be visualized in terms of rules per prototype as given by Fig. (5.10).

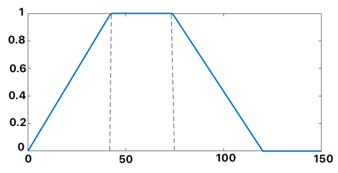
[1]Inputs with the highest density

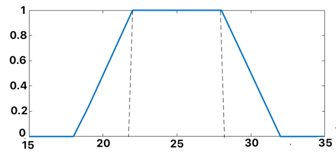
IF $(Rv \sim$  $)$

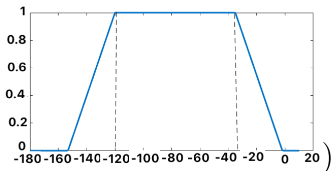
AND $(d^{FL} \sim$  $)$

AND $(V^{FL} \sim$  $)$

AND $(d^{FC} \sim$  $)$

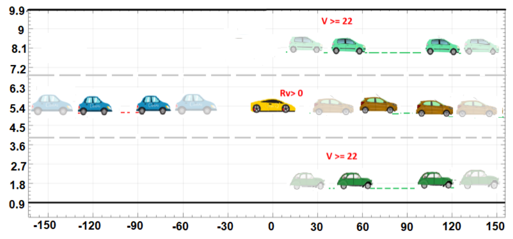
AND $(d^{FR} \sim$  $)$

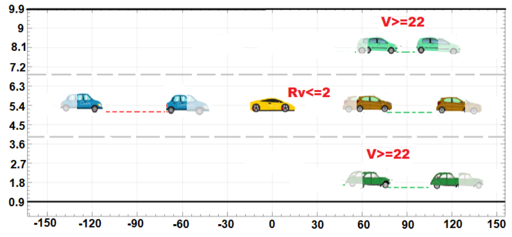
AND $(V^{FR} \sim$  $)$

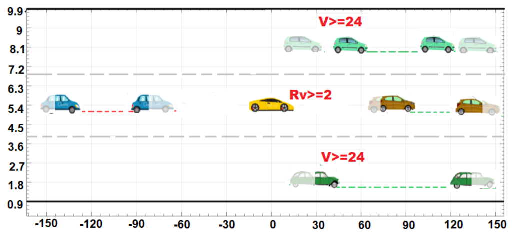
AND $(d^{BC} \sim$  $)$

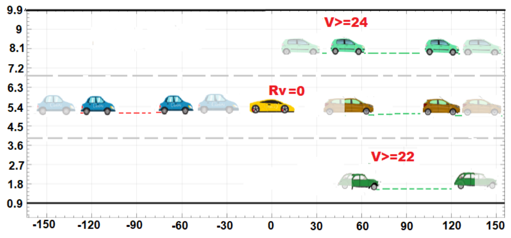
THEN 'Lane change left and also brake by $-2m/s^2$'

Figure 5.9: Trapezoidal rule per feature for 'Lane change left and also brake by $-2m/s^2$' (Action 6)

IF $(x \sim$  $)$ OR

$(x \sim$  $)$ OR

$(x \sim$  $)$ OR ... OR

$(x \sim$  $)$

THEN "Maintain"

Figure 5.10: Visual interpretation of trapezoidal rule for "Maintain" (Action 1), where the watermarked cars represent the soft trapezoidal fuzzy boundaries and the solid cars denotes the limits of the *MegaClouds*. R_v denotes the relative velocity between EV and center vehicle, and V denotes the velocities for the front left and front right vehicles, both R_v and V are in m/s

Fig. (5.11) illustrates the actions given by the proposed method along the time. This is helpful to analyse the driving behavior and sequence of events by specialists.

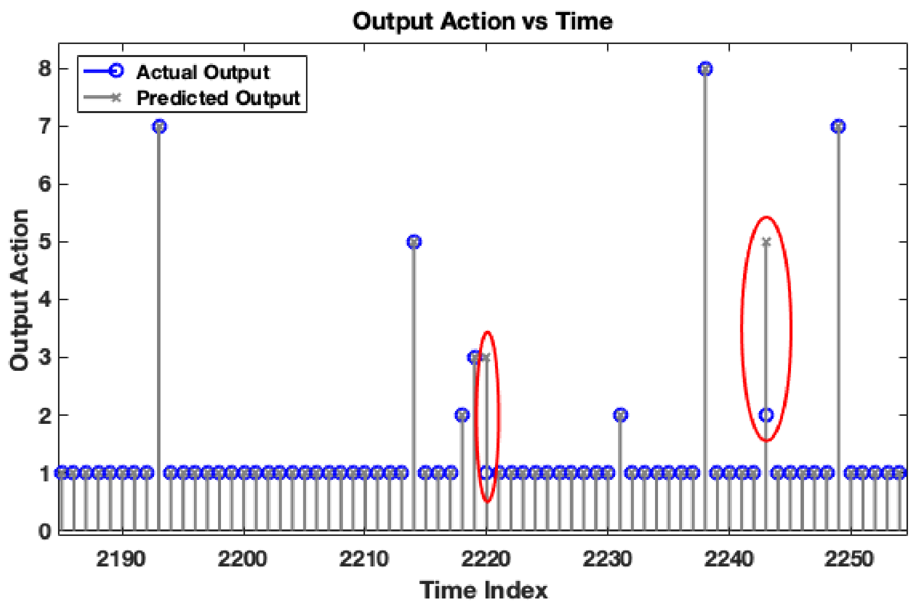


Figure 5.11: Action vs time. The red ellipsoid indicates the wrongly predicted action given by the proposed approach

In general, experiments have shown that the proposed explainable method is an efficient framework for this challenging task. Results showed advantages (98.94%) compared to similar methods for addressing the approximation task. Moreover, the proposed method in its top layer also produced transparent linguistic fuzzy rules, which are human interpretable. In addition, the hierarchical architecture allows to reduce the rule antecedents and to simplify the structure of the rule-based models.

5.5 Conclusion

In this chapter, a novel explainable rule-based machine learning model that can be used to approximate the decisions policy of a DRL agent is presented. The model is composed of a 0-order fuzzy rules. A new hierarchical mechanism to significantly reduce the number of generated fuzzy rules is also presented. In this case, adjacent (in the data space) prototypes which correspond to the same action are grouped and merged into so-called "*MegaClouds*". The method helped to improve the interpretability of the generated models. Moreover, it is also presented an input selection method based on ranking the density per input dimension in the data space contributed to improve the accuracy of the models as it creates individualized subsets of inputs per action, taking advantage of the parallel characteristic of the explainable self-organizing method.

Chapter 6

Detecting and Learning from Unknown

This chapter introduces a method and algorithm that departs from the traditional approach and offers a paradigm shift bringing the machine learning, in general, and pattern recognition and classification, in particular, extremely close to a fully unsupervised form. In a nutshell, it offers a self-learning locally generative models that work together and require extremely light supervision in the form of few data samples. It is able to automatically detect the unknown and to learn from it. This is in sharp contrast to the traditional approach where learning is, in essence, only an averaging of the history. The current approaches struggle to detect changes, dynamical evolution or appearance of new classes.

Methods like eClass (Angelov et al., 2008), FLEXFISClass (Lughofer, 2008) and other similar ones are called "evolving" classifiers. They are designed to take into account new coming data samples. However, when talking about new classes (rather than just new data samples) class label is required which means, that these methods are supervised learning methods. The proposed method is unsupervised in regards to the new data that represent a new class. There are also unsupervised evolving algorithms for clustering (Bezerra et al., 2016), but these methods do not deal with classification as the method proposed here. Another type of methods that claim to approach similar problems are the so-called zero-shot learning (ZSL) methods. They have an objective to transfer a learnt model to unknown classes without the acquisition of new features. However, the main problem with this type of technique is the dependence on additional information to relate unknown classes to previously trained models. Not always such information is available or possible to acquire (Lampert et al., 2009). In this respect, the ZSL approach is not unsupervised in terms of the new class and not a direct comparator.

The main challenges that the method, namely xClass, introduced in this chapter addresses are: i) to detect when a certain unlabeled (new) data sample is not from a class that was used in training, i.e. to have class label “Unknown” or “New”; ii) to learn from such new unlabeled data in an unsupervised manner. The proposed approach to address the first issue is based on the drop of the density that represent the confidence in a decision. The proposed approach to the second issue is by learning from the data for which the class is “New”. The proposed approach further selects prototypes out of the data samples of the “New” class according to their density in the same way as for the other/known classes. Because, the learning in the proposed approach is per class, all new data from a “New” class are analysed separately from the data from the known classes.

6.1 Concept and Basic Algorithm

The xClass pipeline includes the following steps:

1)**Pre-processing**, which includes different substeps like normalization and standardization, dealing with missing data, and feature selection (Kotsiantis et al., 2006). Specifically for image processing there are often other stages, such as rotation, augmentation, scaling, elastic deformation, etc (Ross, 2016). Even deep learning methods which claims to avoid handcrafting apply some of the cited steps.

2)**Learning phase**, which can be offline, when the full dataset is available; or it can be done online, when the data arrive in the form of a data stream (sample-by-sample). Evolving learning, ability of the algorithms to adapt their parameters and structure according to the non-stationary data streams, is a more sophisticated form of online learning (Angelov et al., 2010; Škrjanc et al., 2019).

3)**Generating outputs** for new unseen data, which is the **validation phase**. Different algorithms use different strategies in order to validate the model generated in the learning phase.

6.1.1 Pre-processing

The proposed method also starts with a pre-processing step which involves mostly the same steps depending on the specific problem. For example, for image processing we may also apply scaling, augmentation, rotation, etc. Practically for all problems normalization and standardization is required.

The proposed xClass method uses standardization and normalization as follows:

Firstly, it standardize the newly observed data sample, x.

$$\widehat{x}_i = \frac{x_i - \mu(x)_i}{\sigma(x)_i} \tag{6.1}$$

where $\widehat{x}$ denotes the standardized data sample. Outliers ($|\widehat{x}| \geq 3$) are ignored and not used for training. After that, the data is rescaled within the range $[0, 1]$ to consider them in the same proportion. It is important to highlight that in the proposed xClass method, the normalization is done upon the standardized data. Unity-based normalization of the i-th element of the j-th sample is given by:

$$\bar{x}_i = \frac{\widehat{x}_i - \min(\widehat{x}_i)}{\max(\widehat{x}_i) - \min(\widehat{x}_i)} \tag{6.2}$$

where $\bar{x}$ denotes the normalized data sample.

6.1.2 Parameters initialization

The prototype-based learning is the core of the proposed method which represents local focal points (the prototypes are focal points of locally valid generative models described by multimodal Cauchy distribution (Angelov and Gu, 2019)). The meta-parameters are initialized with the first observed data sample. The proposed algorithm works per class; therefore, all the calculations are done for each class separately.

$$P \leftarrow 1; \quad \mu \leftarrow \bar{x}_i; \tag{6.3}$$

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

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

$$\begin{aligned} \mathrm{C}_1 \leftarrow \{\bar{x}_1\}; \quad \pi_1 \leftarrow \bar{x}_1; \\ S_1 \leftarrow 1; \end{aligned} \tag{6.4}$$

where, π_1 is the prototype of C_1; S_1 is the corresponding support (number of members).

6.1.3 Density Estimation

$$D(\bar{x}_i) = \frac{1}{1 + \frac{||\bar{x} - \pi_j||^2}{||\sigma_i||^2}} \tag{6.5}$$

where π_j ($j = 1, 2, ..., P$) is the set of prototypes, and σ_i is the standard deviation. The data density also represents the degree of similarity between a given data sample, x_i and a prototype, π_j. Further, it can also be considered to represent the degree of confidence of associating a data sample, x_i with the prototype π_j (and labelling with the same label).

The reason it is Cauchy is not arbitrary (Angelov and Soares, 2020b). 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 (6.5). It can also be demonstrated that the so called *typicality*, τ, which is the weighted average of the data density, D, with weights representing the frequency of occurrence of a data sample (Angelov and Gu, 2019). Furthermore, the *typicality*, τ can be considered an empirically derived form of the pdf having the same properties, notably, it integrates to 1 an infinite range.

Density per feature f is obtained according to the equation (6.5), where D^f denotes the density for f-th feature of the $\bar{x}$ sample.

The cumulative effect across all data samples per feature can be obtained according to the equation (6.6).

$$\Lambda_i^f = \frac{\Sigma_{i=1}^{N} D_i^f(\bar{x}_i^f)}{N}. \tag{6.6}$$

The cumulative contribution for each feature Λ_i^f can be rank ordered, n represents the number of samples. The higher, the value of Λ_i^f is for a particular feature, the more important is the f-th feature. The rationale is that an interesting feature has higher density than other features - meaning that it conveys unique, different clear information, and, as a consequence, it contributes more to the classifier's result because the overlap between data of different classes is less pronounced for this feature.

Then the algorithm absorbs the new data samples one by one assigning them to the nearest (in the feature space) prototype:

$$n^* = \underset{j=1,2,\ldots,P}{\operatorname{argmin}}(||\bar{x}_i - \pi_j||^2) \tag{6.7}$$

Because of this form of assignment, the shape of the data partitioning is of the so-called Voronoi tesselation type (Okabe et al., 2009). 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 (Angelov and Gu, 2019).

6.1.4 Parameters Update

In case, the following condition (ibid.) is satisfied:

$$\begin{aligned} &\mathit{IF}\ (D_i(\bar{x}_i) \geq \max_{j=1,2,\ldots,P} D_i(\pi_j)) \quad \mathit{OR} \quad (D_i(\bar{x}_i) \leq \min_{j=1,2,\ldots,P} D_i(\pi_j)) \\ &\qquad\qquad\qquad\qquad \mathit{THEN}\ (\mathit{add\ a\ new\ data\ cloud}) \end{aligned} \tag{6.8}$$

It means that $\bar{x}_i$ is out of the influence area of π_j. Therefore, $\bar{x}_i$ becomes a new prototype of a new *data cloud* with meta-parameters initialized by equation (6.9).

$$P \leftarrow P + 1; \quad \mathrm{C}_P \leftarrow \{\bar{x}_i\}; \pi_P \leftarrow \bar{x}_i; \quad S_P \leftarrow 1; \tag{6.9}$$

Otherwise, data cloud parameters are updated by equation (6.10). 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).

$$\begin{aligned} \mathrm{C}_{n^*} \leftarrow \mathrm{C}_{n^*} + \{\bar{x}_i\}; \quad \pi_{n^*} \leftarrow \frac{S_{n^*}}{S_{n^*}+1}\pi_{n^*} + \frac{1}{S_{n^*}+1}\bar{x}_i; \\ S_{n^*} \leftarrow S_{n^*} + 1; \end{aligned} \tag{6.10}$$

One of the strongest aspects of the proposed approach is its high level of interpretability which comes from its prototype-based, local generative models as well as as its ability to be expressed as a set of linguistic $IF...THEN$ fuzzy rules of the following type:

$$\mathrm{R}: \quad \mathit{IF}\ (x \sim \pi_1) \quad \mathit{OR} \quad ... \quad \mathit{OR} \quad (x \sim \pi_P) \quad \mathit{THEN}\ (\mathit{Class} \quad c) \tag{6.11}$$

The fuzziness represents the degree of association/similarity to the prototypes. Indeed, the value of data density, D, can be interpreted as a membership function of the fuzzy set $(x \sim \pi)$ (Angelov and Gu, 2019). With a maximum 1 when $x = p$. The continuous typicality, τ given by the equation (6.12), is an empirically derived form of probability distribution. The value of τ even at the point $x = p_i$ is much less than 1 the integral of $\int_{-\infty}^{\infty} \tau dx = 1$.

$$\tau_i(x) = \frac{D_i(x)}{\int_x D_i(x)dx} \tag{6.12}$$

The typicality per class offers conditional probability that is the basis of a generative model, but within both, xDNN and xClass from the classifier design point of view, we are interested in the local peaks of the typicality which coincide with the peaks of the data density. Indeed, it can be demonstrated that since the mathematical expression of the typicality is a mixture of Cauchy expressions and of the data density is a Cauchy expression, the peaks of τ and D are at the same value of x^*. Data density, D is much easier to calculate and therefore, we use D rather than τ further.

6.2 Detect and Learn from Unknown

This is the most innovative part of the proposed algorithm in addition to the feature selection per class, which allows xClass to detect new data patterns autonomously and learn from them.

6.2.1 Drop of confidence (detect the novelty)

Unlabeled data samples become available as soon as the training process with labeled samples finishes. Then, the xClass classifier can continue to learn from these unknown data samples. The unlabeled training samples are defined as the set $\{u\}$, and the number of unlabeled samples is defined as U.

The first step in the weakly supervised learning process of xClass, is to extract the vector of confidence/degrees of closeness to the nearest prototypes for each unlabeled data sample defined as $\lambda(u_i)$, $i = 1, 2, ..., U$ follows:

$$\lambda(u_i) = \underset{j=1,2,...,P}{max} S(x_i, \pi_j), \tag{6.13}$$

where λ denotes the confidence degree.

The recursive mean $\overline{\mu}_i$ of the λ for the labeled data samples is used to detect sudden drop of the confidence generated by the xClass classifier when a new unknown class arrives and can be calculated as follows (Angelov, 2012):

$$\overline{\mu}_i = \frac{i-1}{i}\overline{\mu}_{i-1} + \frac{1}{i}\lambda_i, \overline{\mu}_1 = \lambda_1. \tag{6.14}$$

Then the m-σ rule is applied, for detailed explanation about the m-σ please refer to (Costa et al., 2015). New classes are actively added by the proposed xClass classifier when the inequality (6.15) is satisfied and rules are actively created. Otherwise, if the inequality is not satisfied the newly arrival unlabeled data samples are used for updating the structure and meta-parameters of the xClass classifier. Fig. (6.1) illustrates the drop of confidence of the proposed method when a new a unseen class arrives. The black line indicates the confidence of xClass. As the fall is detected, if the inequality (6.15) is satisfied this indicates that the label of this data sample is not any of the known to xClass labels. The options are that: a) This drop is a one off due to outlier, noise, randomness, or b) a number of such data samples above a drop of confidence is detected are close to each other in the data space (please note that they may not necessarily arrive one after the other as in Fig. (6.1)). Otherwise, if the condition given by the inequality (6.15) is not met the data sample is used to update the meta-parameters of the proposed method.

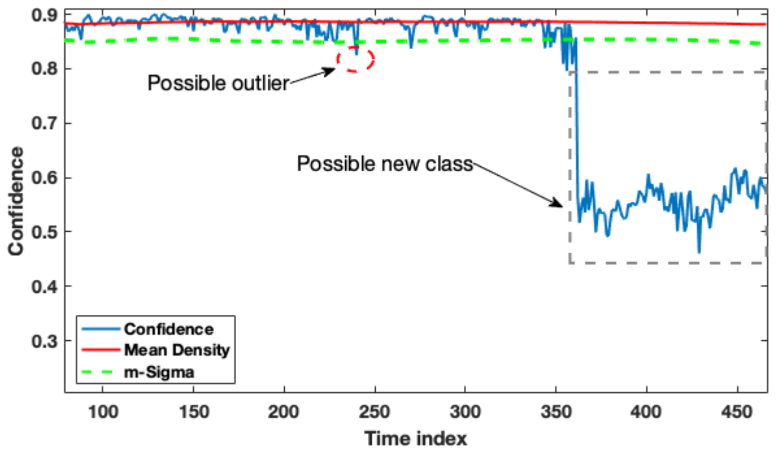


Figure 6.1: Drop of confidence of the proposed method when a new a unseen class arrives

$$\begin{aligned} IF\ \lambda(u_i) < (\bar{\mu}_i - m\sigma)\ THEN\ (u_i \in Possible\ new\ class\ detected) \\ ELSE\ (Update\ structure\ and\ meta-parameters) \end{aligned} \tag{6.15}$$

When the inequality (6.15) is satisfied, the arrival data sample is denoted as a potential outlier and temporally saved. When several of potential outliers are close to each other in the data space, have similar densities, they are denoted as 'new class 1', if more than one group is formed than new classes are formed as well and new labels as 'new class 2' are generated.

One or few labels for new detected classes are provided. The validation process is done through the 'winners-take-all' principle, which is given by,

$$Label = argmax(\lambda(u_{i+1})). \tag{6.16}$$

The general structure of the proposed xClass approach is illustrated by the block diagram presented in Fig. (6.2).

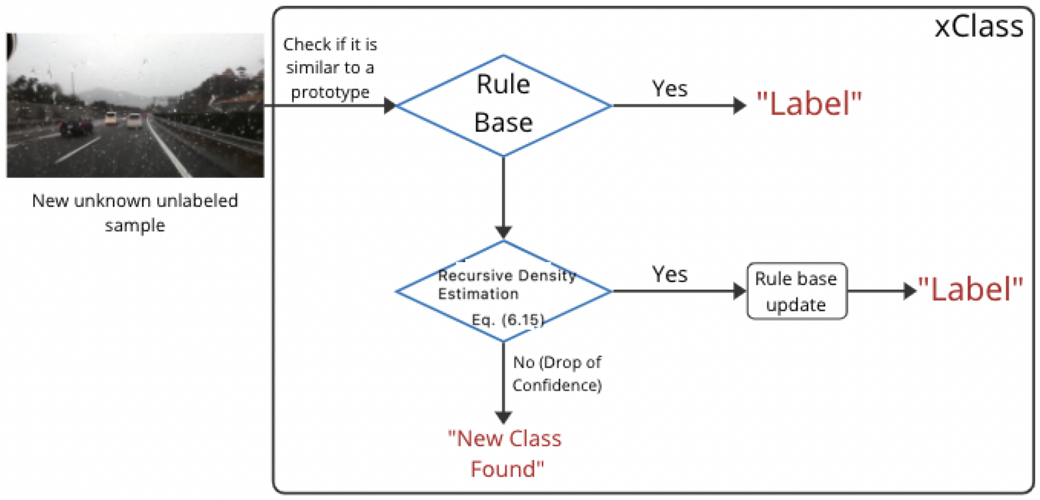

xClass
New unknown unlabeled sample
Check if it is similar to a prototype
Rule Base
Yes
"Label"
Recursive Density Estimation Eq. (6.15)
Yes
Rule base update
"Label"
No (Drop of Confidence)
"New Class Found"

Figure 6.2: General structure of xClass – block diagram

6.3 Results for Novelty Detection

In this section, the results obtained by xClass are demonstrated. Computational simulations were performed to assess the accuracy of the classification methods considering 4 different benchmark problems. All the experiments were conducted with Python 3.6 using a personal computer with a 1.8 GHz Intel Core i5 processor, 8-GB RAM, and MacOS operating system. The default parameters were used for the algorithms used in the comparisons. For this experiment, we considered a VGG–VD–16 pre-trained on ImageNet; a SVM model with RBF kernel and decision function of shape "one-vs-rest"; a KNN classifier with $k = 5$; a Decision Tree (DT) with Gini impurity to measure the quality of a split and "log loss" and "entropy" both for the Shannon information gain; LSTM has 32 layers, and uses "tanh" function for activation and "sigmoid" for recurrent activation. The results obtained during experimentation demonstrates that xClass offers:

– high precision as compared with the top state-of-the-art algorithms.

– ability to detect unseen/new data patterns autonomously and learn from them.

– ability to learn with extremely low supervision (few) labeled data samples for the newly detected classes.

– ability to autonomously select the most effective features per class.

– highly transparent interpretable model.

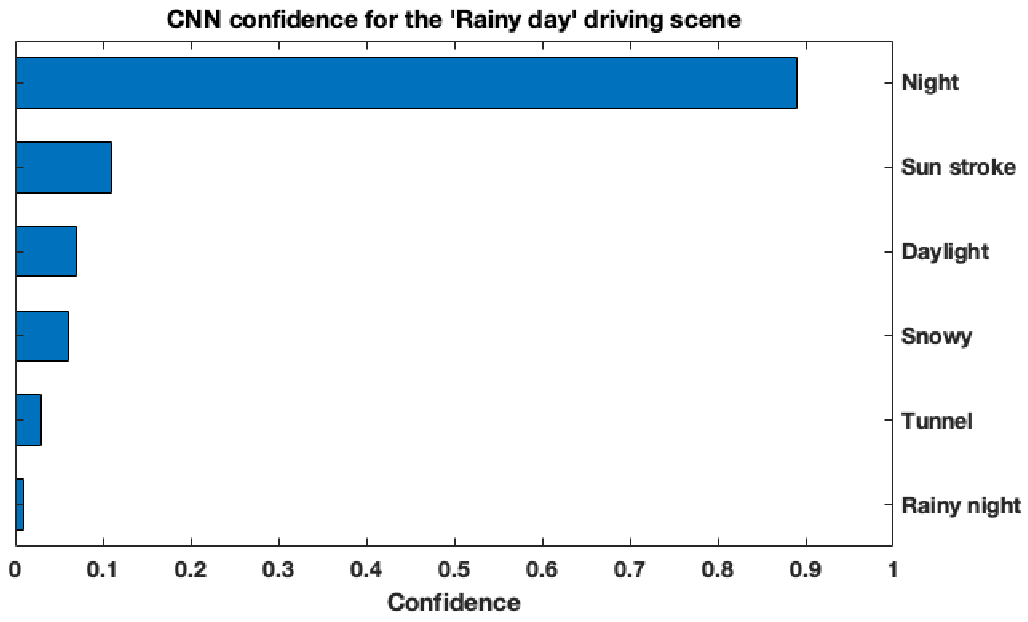


Figure 6.3: Wrong classification given by VGG–16 for a new unknown class (Rainy Day).

– no user- or problem- specific algorithmic parameter (except for feature selection which can be done by *ad hoc* decision).

– non-iterative algorithm able to learn continuously.

6.3.1 iRoads dataset

In the first experiment the iRoads dataset (Rezaei and Terauchi, 2013) was considered. The convolutional deep neural network VGG–16 was trained with 80% of the available iRoads dataset; however, images for the 'Rainy day' scenario were omitted of the training phase. After the training phase, 'Rainy day' trained images were presented to the neural network. As the VGG–16 approach was not trained for the presented situation, and it is not able to adapt its structure for the newly arrived class, it misclassified the 'Rainy Day' scenario with almost 90% confidence as a 'Night' scenario as illustrated by Fig. (6.3).

Therefore, the deep learning VGG–16 misclassifed with almost 90% of confidence the "Rainy day" driving scenario as a "Night" scenario as illustrated by Fig. (6.3). This is not surprising, because the VGG–16 (same as other mainstream deep neural networks) can only recognise what it was trained for and is not equipped with an exploratory mechanism to enable detection and learning from unknown data samples.

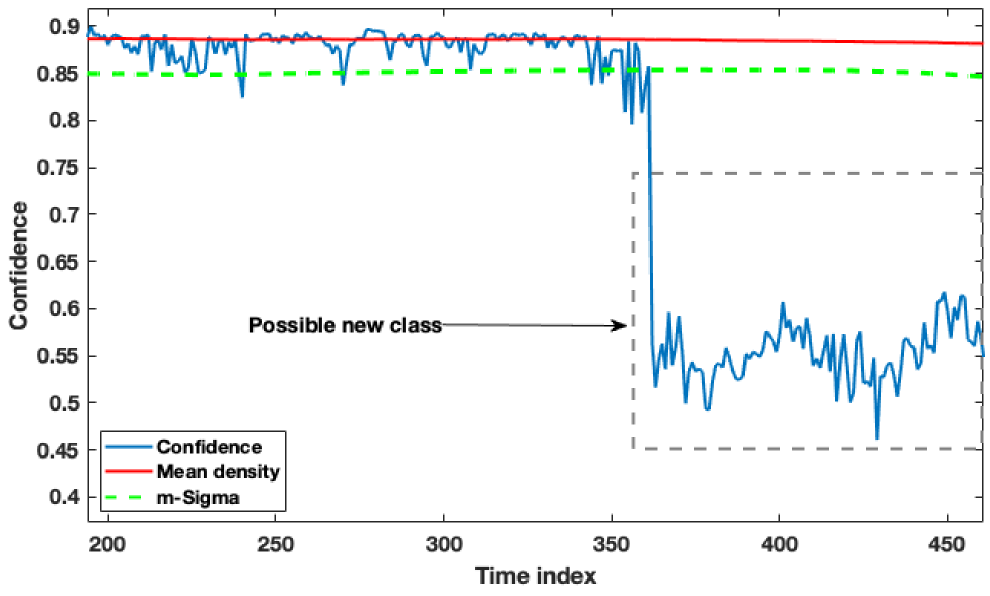


Figure 6.4: Sudden drop of confidence due the presentation of new unknown classes.

In such new situations, mainstream deep networks require a full retraining in order to correctly classify new classes. However,a full retraining of a deep neural network is usually time consuming, computational expensive, costly, and involves the human for labeling purposes.

The xClass exploratory mechanism is able to discover new classes as they arrive to the system due to its mechanism based on the recursive density estimation (Angelov, 2012) and Chebyshev inequality approach (Costa et al., 2015) as given by Fig. (6.4). The blue line indicates the confidence value (λ boundary) given by the xClass classifier, the red line indicates the the recursive density estimation value, the green line is the 3-σ. The sudden fall of the blue line indicates the moment when the unlabeled set of images belonging to an unknown class arrive to the system.

The proposed xClass classifier was trained with 80% of the available iRoads images of all classes except the "Rainy day" class. Then, the new unlabeled class was present to the proposed classifier, xClass was able to successfully detect the suddenly drastic fall in the confidence (Fig. (6.4)) and proactively create a new class as illustrated by Fig. (6.5). The prototype-based and non-iterative nature of the proposed method allowed to detect the fall in the confidence (λ) in real time, and differently, from traditional deep learning approaches, no retraining is required to learn the new class.

The proposed xClass classifier obtained 99.12% classification accuracy for unlabeled images using the 3-σ approach. The semantically meaningful label 'Rainy Day Scene'

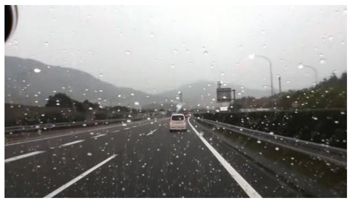
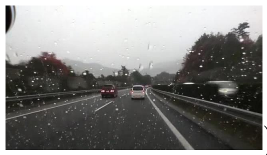
R_{new}: IF ($Image \sim$ ) THEN 'New class'

Figure 6.5: A new rule is proactively created when a sudden fall in the confidence is detected through the inequality (6.13). The proposed xClass classifier is highly interpretable due to its rule-based nature. This advantage favors human experts analysis as it provides a transparent structure, differently from the 'black box' approaches such as deep neural networks.

R_7: IF ($Image \sim$) OR ($Image \sim$ ) OR

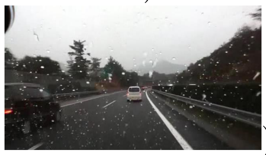
... OR ($Image \sim$ ) THEN 'Rainy day scene'

Figure 6.6: Final rule given by the xClass classifer for the new detected class. Label is attached during the validation phase. Differently from 'black box' approaches as deep neural networks, xClass provides highly interpretable rules which can be used by human experts for different analysis as necessary.

is optional and requires only one-off involvement by the human (by default it will stay as 'new class 1'). The final rule generated for this new class detected by the proposed xClass classifier is given by Fig. (6.6).

6.3.2 Faces-1999 dataset

As a second example, we consider the Faces-1999 dataset provided by Caltech (Faces, 1999). For the faces recognition problem, the xClass classifier is trained with just one type of face, differently from traditional approaches which are primed with all available classes (20 different types of faces). We used the final fully connected layer of VGG–16 for features extraction. For each image it produces 4096 values that can be considered as abstract features.

Traditional approaches are not equipped with exploratory mechanism, therefore,

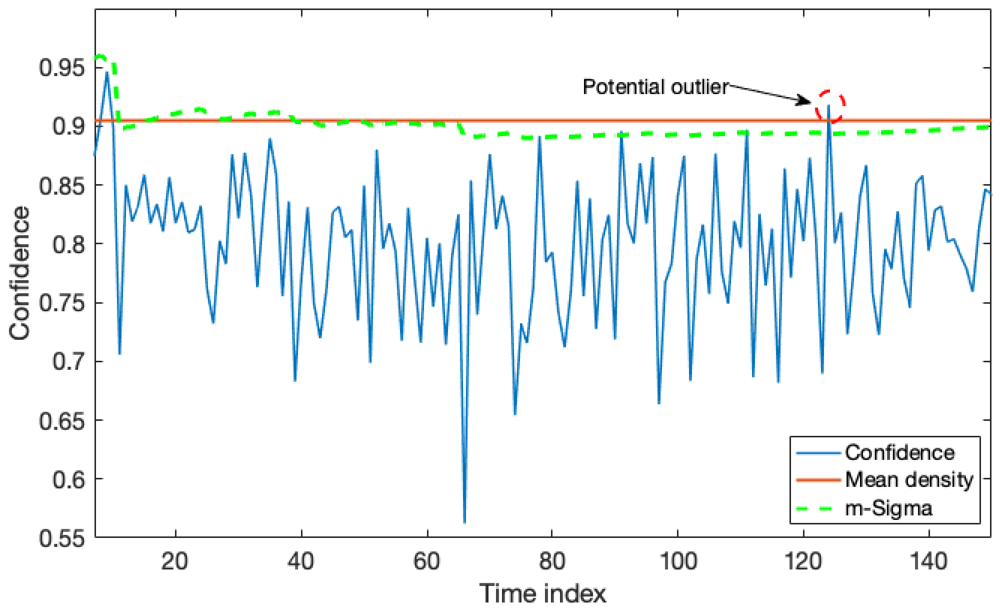


Figure 6.7: Sudden drop of confidence due the presentation of new unknown classes for the Faces-1999 dataset.

they are not able to discover discover new data patterns and adjust its parameters to correctly detect and classify the new arrival data. On the other hand, xClass was able to detect these new types of faces through the drop of confidence as illustrated by Fig. (6.7). So, as the new arriving data is not similar to any of the prototypes contained in the prototypes base of xClass, the algorithm is able to detect that it belongs to a new data distribution.

After the detection of these new classes, an extremely weak supervision (1% training data labeled) and weak supervision (10% training data labeled) is provided in order to label these newly arrived. After, the labeling phase, the classification task was performed. As one can see from Fig. (6.8) and (6.9), the proposed xClass method can surpass its state-of-the-art competitors as they require more labeled data to provide good results. With just 1% of training data is clearly visible the advantage of xClass. On real scenarios the labeling process is very time consuming and is not always possible. The classification curve is given by Fig. (6.9).

Fig. (6.7) illustrates the sudden drop in the confidence when new unknown classes are presented to xClass classifier; the xClass uses the drop of confidence based on the density of the data to discover new classes. Traditional approaches are not equipped with exploratory mechanisms as the proposed xClass method; therefore, they are not able to detect new data patterns and adapt their structure to this situation. It

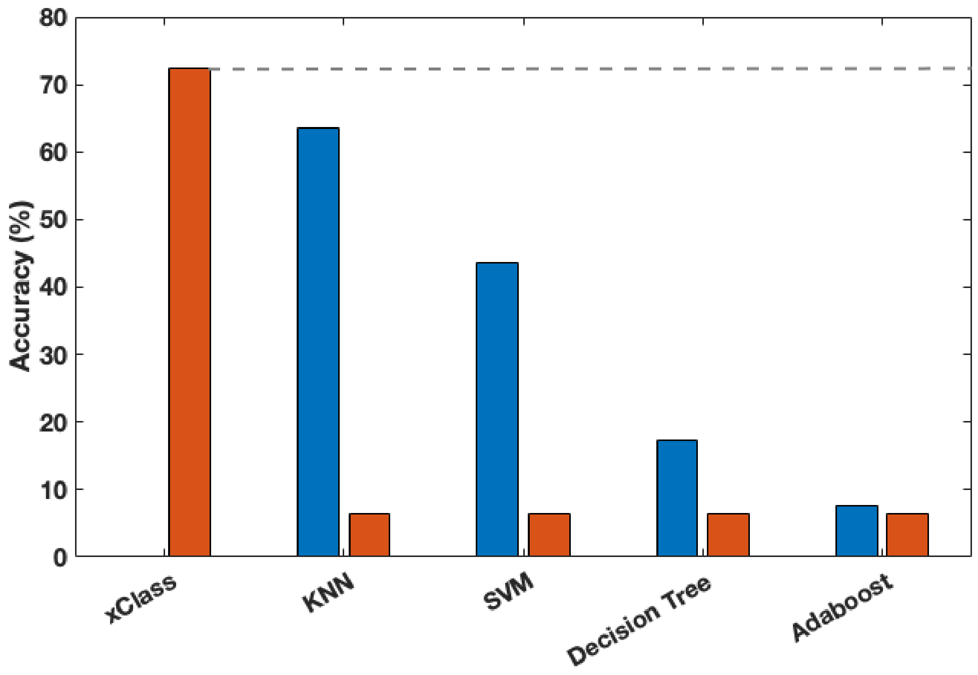


Figure 6.8: Accuracy for extremely weak supervision classification for the Faces-1999 dataset. Red bars illustrate the results obtained by state-of-the-art approaches when just one class is provided during the training phase. The blue bars indicate the results when all the classes are provided.

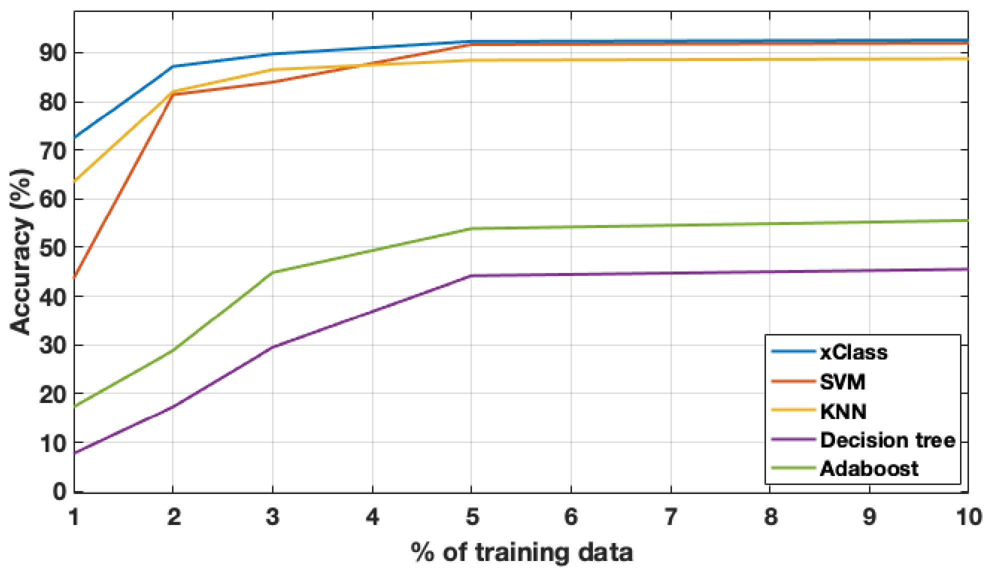


Figure 6.9: Classification curve for different number of training samples for the Faces-1999 dataset.

is notable that the proposed xClass classifier can obtain better results without the necessity for huge number of labeled data, differently from traditional approaches. The performance curve is given in Fig. 6.9, as illustrated, with xClass still producing better classification rates when more training data is provided.

6.3.3 Caltech-101 dataset

As a third case, we consider the Caltech-101 dataset (Fei-Fei et al., 2007). As in the other experiments the proposed xClass classifier was primed with 80% of data samples from the first class for training, and then, used its exploratory mechanism to discover the other classes autonomously and learn from them based on the data density according through the drop of confidence as detailed in Fig. (6.10); as illustrated in Fig. (6.11), traditional approaches are not able to detect new data patterns after the training phase (traditional approaches were trained with just 1 class), and then, tend to produce results with low accuracy. Unlike supervised methods which are data hungry, the proposed xClass approach could obtain high classification accuracy with extremely weak supervision (Fig. (6.11)), in order word, with less training data as possible. The acquisition of labeled data requires enormous human efforts and it is very time consuming. Fig. (6.12) gives the evolution of the performance of the proposed exploratory classifier as more training samples are provided. As it is illustrated by Fig. (6.12), the xClass classifier is able to produce better results in terms of accuracy,

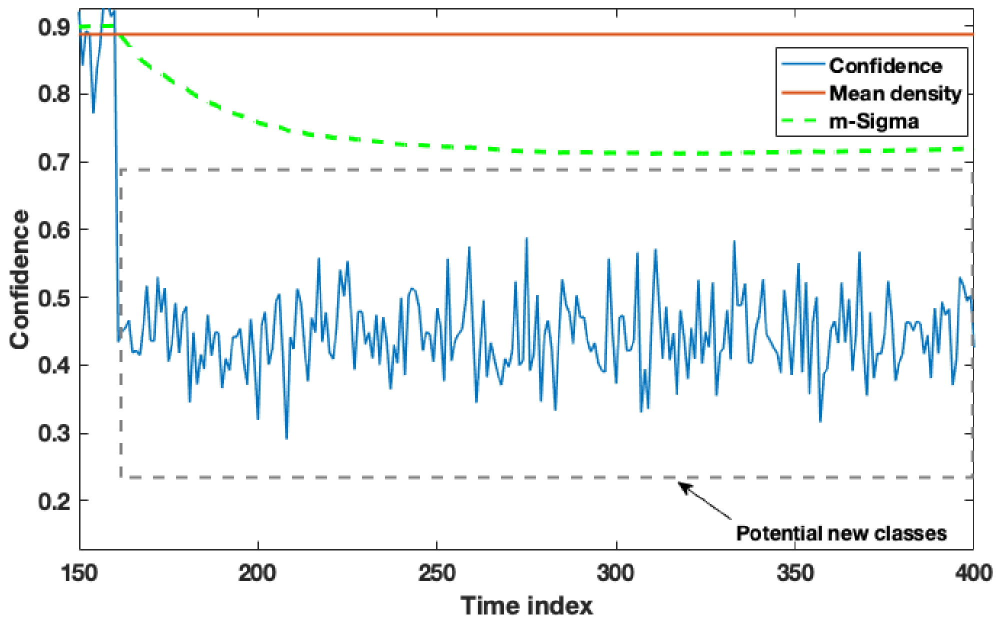


Figure 6.10: Sudden drop when new unknown are classes are presented to the xClass method – Caltech-101 dataset.

demonstrating its efficiency to detect and learn from unknown effectively.

The Caltech-101 dataset is constituted of 101 different classes. However, in the experiment only 10 classes were used. Supervised methods such as Decision tree, k-nearest neighbors (KNN), Adaboost, and SVM require information about all the available classes beforehand, in order to produce better results (the red bars in Fig. (6.11) illustrate the results obtained when just one class is used in the training phase). In comparison, the proposed extremely weakly supervised approach requires just the knowledge about one class beforehand as illustrated by Fig. (6.10) as the other classes are discovered through its exploratory mechanism.

The blue bar in Fig. (6.11) illustrates the classification results when just 1% of labeled training data is provided for all classes. The proposed exploratory xClass classifier could obtain almost 90% of classification accuracy. State-of-the-art approached have the necessity for labeled training data to produce acceptable results as illustrated in Fig. (6.12). Even when more labeled training data is provided, the proposed xClass classifier still produce better results in terms of accuracy than its competitors. Furthermore, the Zero-Shot learning method proposed by (Long and Shao, 2017) was reported to provide 57% accuracy for the same problem which is significantly poorer result than the one obtained by the proposed xClass method. In addition to the significantly higher accuracy than the Zero-Shot learning method, the

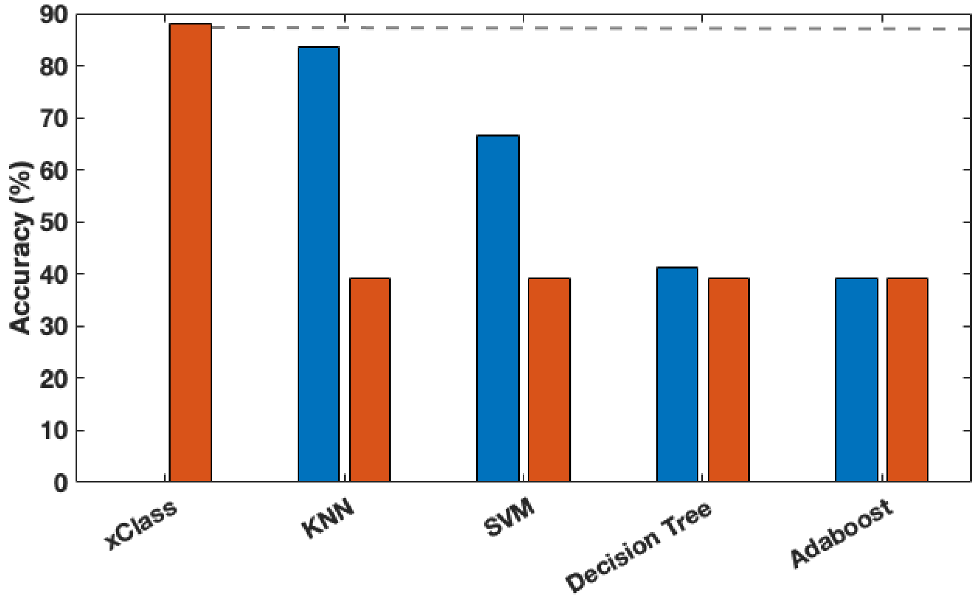


Figure 6.11: Accuracy for extremely weak supervision classification for the Caltech-101 dataset.

proposed xClass method also has the advantage of allowing human inspection of the decision-making process (this makes it explainable).

6.3.4 Vehicles dataset

In the fourth case, we consider the vehicles dataset (Soria et al., 2011), which is a non-image based dataset. xClass is, firstly, trained with just one sample of the first class, and then, it has to autonomously detect the other classes based on the empirically observed data and the sudden drop of confidence (Fig. (6.13)).

The inner parallel feature selector of the proposed approach selected 7 out of the 18 original features differently for each class. This is helpful to improve the interpretability of the proposed classifier. Results obtained by xClass and its competitors are given in Fig. (6.14).

It is important to highlight that SVM, KNN, Decision Tree, Adaboost, Long short-term memory (LSTM) are all supervised methods, and they were trained with all available classes beforehand (red bars in Fig. (6.15) illustrate the results obtained by the traditional supervised approaches if just one class is used in the training phase). However, the proposed xClass approach could obtain better results in terms of accuracy even though it uses an extremely weak supervision (Fig. (6.15)).

Fig. (6.13) illustrates the drop of confidence when new unseen classes are presented

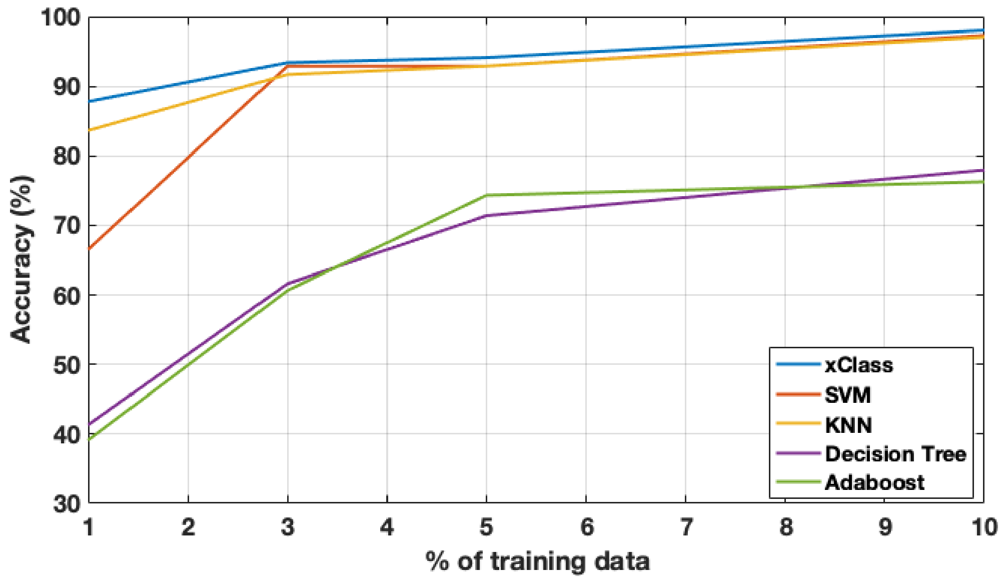


Figure 6.12: Classification curve for different number of training samples for the Caltech-101 dataset.

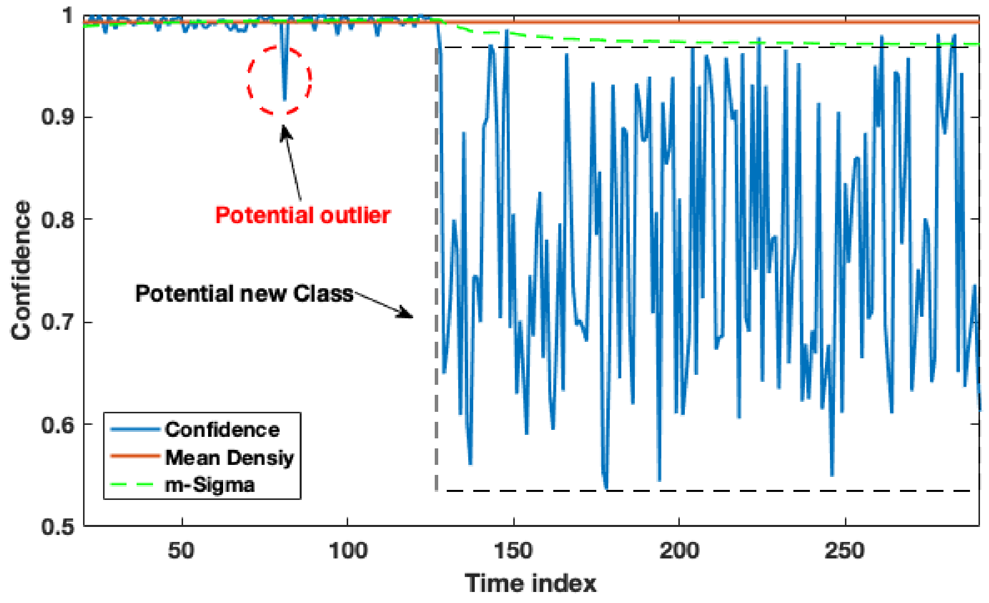


Figure 6.13: Sudden drop of confidence due the presentation of new unknown classes – Cars dataset.

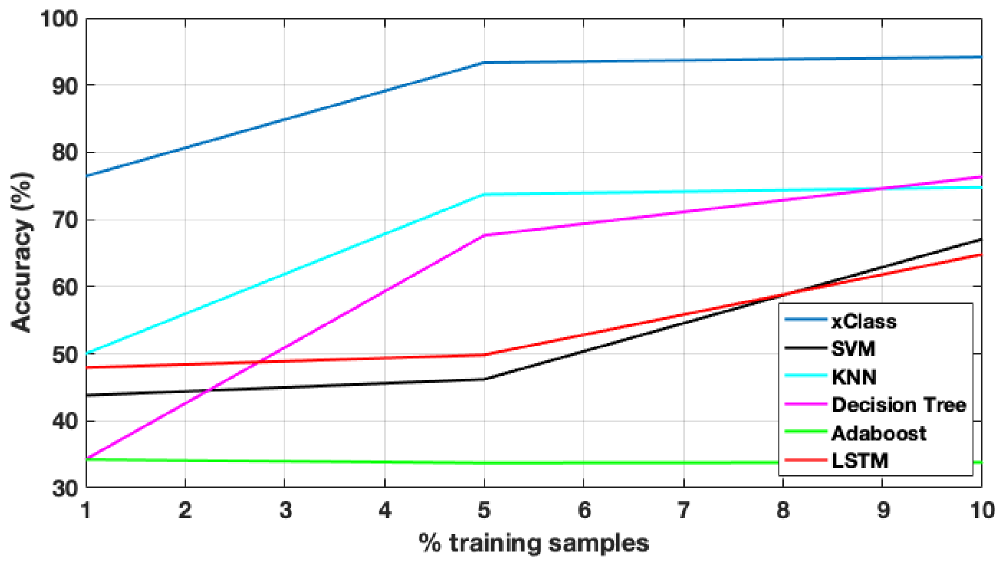


Figure 6.14: Classification curve for different number of training samples for the Cars dataset.

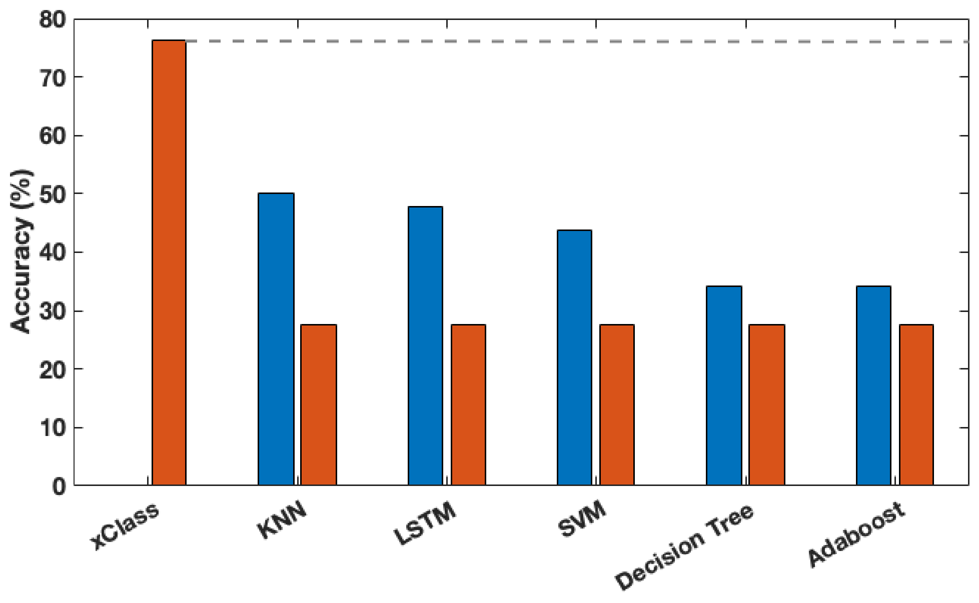


Figure 6.15: Accuracy for extremely weak supervision classification for the Cars dataset.

to the proposed classifier. Differently from traditional approaches which require the knowledge of all available classes beforehand, the proposed xClass uses its exploratory mechanism to autonomously discover this new class with basis just on the empirical data. Red bars on Fig. (6.15) shows the results obtained by state-of-the-art methods if just one class is presented during the training phase, as they are not able to detect new arrivals data patterns and adapt they structure to this scenario, they wrongly classify the new arrived data samples as the known class. Different types of supervision (extremely weak, weak, full) is provided during experiments, in all cases the proposed method could provide better results in terms of classification performance than its competitors as illustrated by Fig. (6.14). It is possible to note through Fig. (6.15) that the results obtained for extremely weak supervision with xClass surpass its competitors in more than 25% in terms of classification performance, which indicates the efficiency of the proposed method.

As given by Fig. (6.14), xClass is able to improve its results if more training data and all classes are provided. For validation purposes, 20% of the data samples were used in all cases and labels for newly detected classes by xClass are attached during this phase. The AnYa fuzzy rule (Angelov and Gu, 2019) for the newly identified class R_{new} can be written as follows:

$$R_{new}: \text{ IF } (x \sim \begin{bmatrix} 104 \\ 41 \\ 66 \\ 10 \\ 23 \\ 635 \\ 73 \end{bmatrix}) \; THEN \text{ `}NewClass1'$$

where x is the set of selected features given by the density-based feature selector. x can be written as follows:

$$x = \begin{bmatrix} COMPACTNESS \\ CIRCULARITY \\ PR.\ AXIS\ ASPECT\ RATIO \\ MAX.\ LENGTH\ ASPECT\ RATIO \\ PR.\ AXIS\ RECTANGULARITY \\ SCALED\ VARIANCE\ MINOR \\ SKEWNESS\ ABOUT\ MAJOR \end{bmatrix}$$

During the validation stage labels are attached to the newly identified rules. The final format for the first newly identified rule is given as follows:

$$R_2: \text{ IF } (x \sim \begin{bmatrix}104\\41\\66\\10\\23\\635\\73\end{bmatrix}) \; OR \; (x \sim \begin{bmatrix}90\\34\\55\\6\\17\\224\\65\end{bmatrix}) \; OR...OR \; (x \sim \begin{bmatrix}113\\53\\62\\11\\24\\688\\72\end{bmatrix}) \;\; THEN \text{ `}SAAB'$$

6.4 Conclusion

The xClass approach could surpass its competitors in terms of accuracy for all experiments using extremely weak supervision, as well as, full supervision. Moreover, the proposed algorithm produced highly transparent interpretable results, which are helpful for human experts analysis. No user- or problem- specific algorithmic parameter (except for feature selection which can be done by *ad hoc* decision) are required which is also an advantage provided by the proposed xClass classifier.

xClass have been tested on four challenging problems, including adversarial autonomous cars scenarios classification, imbalanced faces detection, and objects detection. Not only we achieved higher accuracy (in one of the problems outperforming by 25% the other methods), but, more significantly, we only used the knowledge of just a single class at the start of the process and extremely weakly labeled data and we generated interpretable models with smaller number of features used. Furthermore, the proposed xClass method demonstrated the ability to learn from unknown without retraining, which is one of the biggest problems of deep learning based on neural networks. As illustrated the convolutional deep learning misclassified an unknown class with high confidence, on the other hand, the proposed approach was able to detect a sudden drop in the confidence and learn from this unknown data, then it was able to proactively create a new class for this new scenario.

Chapter 7

Conclusion and Future Work

7.1 Summary of Research and Finding

Deep learning has attracted attention due to its proven ability of obtain highly accurate results in challenging problems (Goodfellow et al., 2016; LeCun et al., 2015). However, the main criticisms over deep learning approaches are over the fact that they are "black-box" approaches, because they usually have million of parameters that are extremely difficult to interpret and relate to the physical nature of the problem. Moreover, they also require large amounts of labeled data, computing resources, and long training in order to obtain acceptable results (Rudin, 2019).

Therefore, in this research thesis new explainable-by-design deep learning approaches are proposed as an alternative to traditional deep learning methods. The methods proposed in this thesis offers two main achievements of deep learning, higher accuracy on different extremely challenging problems, and also a transparent decision structure that allow users to inspect and understand the decisions taken by the algorithm. The characteristics of interpretability and explainability offered by the algorithms proposed by this thesis are essentially important for high-risk applications, such as self-driving cars, medical treatment, or court decisions(ibid.).

This new deep learning architecture and its variants combines inference and learning in synergy. The prototype nature of these methods offers a feed-forward architecture that it is non-iterative and non-parametric which favours the efficiency in terms of time and computing resources. From the user's point of view, the decisions taken by the approach is clearly understandable to human users. The next section presents the limitation of the work proposed in this thesis.

7.2 Limitations

In this section the limitations of the proposed approaches are highlighted.

1) Dependency on external features: although, the explainable deep learning approach and its variants proposed in this research thesis produces high accuracy results on different challenging problems, even surpassing traditional deep learning approaches, they must rely on external features to produce these results. In other words, the quality of the external features have high influence on the final results.

2) No optimization: the work presented on this thesis offers different explainable feed-forward architectures that allows minimum computational resources when compared to mainstream deep learning methods. However, parameters and optimization can be introduced to improve the quality of the prototypes selected during the training phase, and so, improve the results in terms classification metrics.

3) Prototypes base for large datasets: large datasets (millions of data points) may produce large prototype bases that can negatively affect the interpretability of the model. Therefore, strategies to reduce the prototype base may be necessary to favour the interpretability.

4) Rules: rules illustrated in thesis were produced to be understandable by humans. However, the methods proposed here compare the similarities between vectors of numbers which can be harder to interpret by users.

The next section refers to the contributions proposed on this research thesis.

7.3 Contributions

The work described in this thesis offers explainable-by-design architectures for different problems as an attempt to overcome the "black-box" nature of the traditional deep learning:

1) A new prototype-based deep learning architecture that combines inference and learning in synergy.The method uses data density as its core mechanism (Angelov and Gu, 2019). Furthermore, xDNN it is non-iterative and non-parametric, and do not requires extremely powerful hardware resources (GPUs) to obtain highly accurate results. Also, the approach is clearly understandable to human users.

2) An extension of xDNN with a decision tree decision-making process and balanced amount of prototypes per class. The method offers two main novelties: i) using a decision tree to determine the winning class label, and ii) balancing the classes by synthesising data around the prototypes determined from the available training data.

3) A new explainable approach to redesign a Deep Reinforcement Learning (DRL) model into a set of $IF...THEN$ rules. It provides an approximation of the DRL model with an alternative interpretable model with a similar performance. The approach relies on the following premises: i) the universal approximation ability of the rule-based

models with fuzzy predicates; ii) the better interpretability of the prototype-based fuzzy rules (including visualization).

4) A fully autonomous extremely weakly supervised approach (xClass) which is able to learn from just a single class and a small set of labeled data samples. Then, as new classes, unknown to users, appears the proposed xClass method is able to successfully discover this and learn from the data autonomously through a recursive data density mechanism.

The proposed approaches demonstrated very high performance on the benchmarking datasets and in real applications. Including, a own set of CT-scans for Covid-19 identification and a FORD dataset for affordance indicators prediction. Besisdes, the very high performance the proposed approaches also present interpretability and explainability of the generated models and their decisions as an alternative to the traditional “black-box” approaches.

7.4 Future Work

The following directions are to be considered in the future for improvement of the proposed systems:

1) Parameter/prototypes optimization needs to be investigated, in order to improve the effectiveness of the learning algorithm. The main goal here is to find most optimum and parsimonious set of prototypes that will produce the best results without affecting interpretability. (Duan et al., 2021) offers an interesting approach that to modularize deep learning via pairwise learning with kernels which can be useful for the prototype algorithms presented in this work.

2) In order to improve the interpretability of the decisions taken by the proposed networks attention mechanisms can be used as in (Kim et al., 2021). Attention mechanisms offers a visual perspective of the decisions taken by the networks that may produce insights that are more interesting than rules for real applications and research purposes. In this sense, xDNN and its variants can offer more than possibility of interpretation for the same problem.

3) Distance metrics/dissimilarity are the core of algorithms based on prototypes. Therefore, the investigation of methods that are agnostic in terms of distance metrics are of great interest as demonstrated by (Chen et al., 2015).

4) Another research perspective that is extremely important is the extension of the algorithms for object detection problems. Object detection is an important and extremely difficult class of problem that it is extremely important for different high stake applications as autonomous vehicles. The work presented by (Du et al., 2022) offers an interesting perspective in this direction as it presents an object detector algorithm that can detect out-of-distribution (OOD) objects in different scenarios.

Therefore, this section presents new research perspectives that can be useful to improve the work presented in this research thesis and also to bring a new path to explainable-by-nature or anthropomorphic algorithms.

Appendix A

List of Publications

Publications

The following publications have been generated while developing this thesis, and to an extent has guided the thesis into what it has become:

Journal publications

Plamen Angelov and Eduardo Soares (2021). "Detecting and learning from unknown by extremely weak supervision: exploratory classifier (xClass)". In: *Neural Computing and Applications*, pp. 1–13

Plamen P. Angelov et al. (2021). "Explainable artificial intelligence: an analytical review". In: *WIREs Data Mining and Knowledge Discovery*, e1424. DOI: `https://doi.org/10.1002/widm.1424`

Eduardo Almeida Soares et al. (2020c). "Explaining deep learning models through rule-based approximation and visualization". In: *IEEE Transactions on Fuzzy Systems*

Plamen Angelov and Eduardo Soares (2020b). "Towards explainable deep neural networks (xDNN)". in: *Neural Networks* 130, pp. 185–194

Eduardo Soares et al. (2020b). "Autonomous Learning Multiple-Model zero-order classifier for heart sound classification". In: *Applied Soft Computing* 94, p. 106449

Xiaowei Gu et al. (2020b). "A self-adaptive synthetic over-sampling technique for imbalanced classification". In: *International Journal of Intelligent Systems* 35.6, pp. 923–943

Conference publications

Plamen Angelov and Eduardo Soares (2020a). "Towards Deep Machine Reasoning: a Prototype-based Deep Neural Network with Decision Tree Inference". In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 2092–2099

Eduardo Soares et al. (2019a). "Actively semi-supervised deep rule-based classifier applied to adverse driving scenarios". In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8

Eduardo Soares et al. (2019b). "Explainable density-based approach for self-driving actions classification". In: *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, pp. 469–474

Mona Alghamdi et al. (2019). "Self-organising and self-learning model for soybean yield prediction". In: *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*. IEEE, pp. 441–446

Dataset publications

Eduardo Soares and Plamen Angelov (2020). *A large dataset of real patients CT scans for COVID-19 identification*. Version V1. Harvard. DOI: 10.7910/DVN/SZDUQX. URL: https://doi.org/10.7910/DVN/SZDUQX

Other research publications

Eduardo Soares et al. (2021). "An Explainable approach to Deep Learning from CT-scans for Covid Identification". In: *TechRxiv*, pp. 1–8

Eduardo Soares and Plamen Angelov (2021). "RADNN: Robust to Imperceptible Adversarial Attacks Deep Neural Network". In: *TechRxiv*, pp. 1–8

Farshad Firouzi et al. (2021). "Harnessing the Power of Smart and Connected Health to Tackle COVID-19: IoT, AI, Robotics, and Blockchain for a Better World". In: *IEEE Internet of Things Journal*, pp. 1–1. DOI: 10.1109/JIOT.2021.3073904

Eduardo Soares et al. (2020a). "SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification". In: *medRxiv*

Eduardo Soares and Plamen Angelov (2019). "Fair-by-design explainable models for prediction of recidivism". In: *arXiv preprint arXiv:1910.02043*

Paulo Vitor de Campos Souza et al. (2020). "Autonomous data density pruning fuzzy neural network for optical interconnection network". In: *Evolving Systems*

References

Adadi, Amina and Mohammed Berrada (2018). “Peeking inside the black-box: a survey on explainable artificial intelligence (XAI)”. In: *IEEE access* 6, pp. 52138–52160.

Ai, Tao, Zhenlu Yang, Hongyan Hou, Chenao Zhan, Chong Chen, Wenzhi Lv, Qian Tao, Ziyong Sun, and Liming Xia (2020). “Correlation of chest CT and RT-PCR testing in coronavirus disease 2019 (COVID-19) in China: a report of 1014 cases”. In: *Radiology*, p. 200642.

Alam, Mohammed S and Son T Vuong (2013). “Random forest classification for detecting android malware”. In: *2013 IEEE international conference on green computing and communications and IEEE Internet of Things and IEEE cyber, physical and social computing*. IEEE, pp. 663–669.

Alghamdi, Mona, Plamen Angelov, Raul Gimenez, Mariana Rufino, and Eduardo Soares (2019). “Self-organising and self-learning model for soybean yield prediction”. In: *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*. IEEE, pp. 441–446.

Alpaydin, Ethem (2016). *Machine learning: the new AI*. MIT press.

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

Angelov, Plamen, Dimitar Filev, and Nik Kasabov (2010). *Evolving intelligent systems: methodology and applications*. Vol. 12. John Wiley & Sons.

Angelov, Plamen and Xiaowei Gu (2018a). “Deep rule-based classifier with human-level performance and characteristics”. In: *Information Sciences* 463, pp. 196–213.

— (2019). *Empirical approach to machine learning*. Springer.

Angelov, Plamen, Edwin Lughofer, and Xiaowei Zhou (2008). “Evolving fuzzy classifiers using different model architectures”. In: *Fuzzy sets and systems* 159.23, pp. 3160–3182.

Angelov, Plamen and Eduardo Soares (2020a). “Towards Deep Machine Reasoning: a Prototype-based Deep Neural Network with Decision Tree Inference”. In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 2092–2099.

— (2020b). “Towards explainable deep neural networks (xDNN)”. In: *Neural Networks* 130, pp. 185–194.

Angelov, Plamen and Eduardo Soares (2021). “Detecting and learning from unknown by extremely weak supervision: exploratory classifier (xClass)”. In: *Neural Computing and Applications*, pp. 1–13.

Angelov, Plamen and Ronald Yager (2011). “Simplified fuzzy rule-based systems using non-parametric antecedents and relative data density”. In: pp. 62–69.

Angelov, Plamen P and Dimitar P Filev (2004). “An approach to online identification of Takagi-Sugeno fuzzy models”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 34.1, pp. 484–498.

Angelov, Plamen P and Xiaowei Gu (2018b). “Toward anthropomorphic machine learning”. In: *Computer* 51.9, pp. 18–27.

Angelov, Plamen P., Eduardo A. Soares, Richard Jiang, Nicholas I. Arnold, and Peter M. Atkinson (2021). “Explainable artificial intelligence: an analytical review”. In: *WIREs Data Mining and Knowledge Discovery*, e1424. DOI: `https://doi.org/10.1002/widm.1424`.

Ardila, Diego et al. (2019). “End-to-end lung cancer screening with three-dimensional deep learning on low-dose chest computed tomography”. In: *Nature medicine* 25.6, pp. 954–961.

Arrieta, Alejandro Barredo et al. (2020). “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”. In: *Information Fusion* 58, pp. 82–115.

Assael, Yannis M, Brendan Shillingford, Shimon Whiteson, and Nando De Freitas (2016). “Lipnet: End-to-end sentence-level lipreading”. In: *arXiv preprint arXiv:1611.01599*.

Atienza, Felipe et al. (2009). “Real-time dominant frequency mapping and ablation of dominant frequency sites in atrial fibrillation with left-to-right frequency gradients predicts long-term maintenance of sinus rhythm”. In: *Heart Rhythm* 6.1, pp. 33–40.

Azar, Ahmad Taher, Hanaa Ismail Elshazly, Aboul Ella Hassanien, and Abeer Mohamed Elkorany (2014). “A random forest classifier for lymph diseases”. In: *Computer methods and programs in biomedicine* 113.2, pp. 465–473.

Bach, Sebastian, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek (2015). “On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation”. In: *PloS one* 10.7, e0130140.

Bengio, Yoshua, Aaron Courville, and Pascal Vincent (2013). “Representation learning: A review and new perspectives”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.8, pp. 1798–1828.

Bezerra, Clauber Gomes, Bruno Sielly Jales Costa, Luiz Affonso Guedes, and Plamen Angelov (2016). “A new evolving clustering algorithm for online data streams”. In: *2016 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*. IEEE, pp. 162–168.

Bhattacharyya, Siddhartha, Sanjeev Jha, Kurian Tharakunnel, and J Christopher Westland (2011). "Data mining for credit card fraud: A comparative study". In: *Decision support systems* 50.3, pp. 602–613.

Bhavsar, Himani and Mahesh H Panchal (2012). "A review on support vector machine for data classification". In: *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 1.10, pp. 185–189.

Biehl, Michael, Barbara Hammer, and Thomas Villmann (2013). "Distance measures for prototype based classification". In: *International Workshop on Brain-Inspired Computing*. Springer, pp. 100–116.

— (2016). "Prototype-based models in machine learning". In: *Wiley Interdisciplinary Reviews: Cognitive Science* 7.2, pp. 92–111.

Bien, Jacob and Robert Tibshirani (2011). "Prototype selection for interpretable classification". In: *The Annals of Applied Statistics*, pp. 2403–2424.

Bishop, Christopher M (2006). *Pattern recognition and machine learning*. springer.

Blair, Alan and Abdallah Saffidine (2019). "AI surpasses humans at six-player poker". In: *Science* 365.6456, pp. 864–865.

Bobillo, Ignacio J Diaz (2016). "A tensor approach to heart sound classification". In: *2016 Computing in Cardiology Conference (CinC)*. IEEE, pp. 629–632.

Bramer, Max (2007). "Avoiding overfitting of decision trees". In: *Principles of data mining*, pp. 119–134.

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Campos Souza, Paulo Vitor de (2020). "Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature". In: *Applied soft computing* 92, p. 106275.

Cao, Jianfang, Min Wang, Yanfei Li, and Qi Zhang (2019). "Improved support vector machine classification algorithm based on adaptive feature weight updating in the Hadoop cluster environment". In: *PloS one* 14.4, e0215136.

Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko (2020). "End-to-end object detection with transformers". In: *European Conference on Computer Vision*. Springer, pp. 213–229.

Castelvecchi, Davide (2016). "Can we open the black box of AI?" In: *Nature News* 538.7623, p. 20.

Chattopadhay, Aditya, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian (2018). "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 839–847.

Chen, Runjin, Hao Chen, Jie Ren, Ge Huang, and Quanshi Zhang (2019). "Explaining neural networks semantically and quantitatively". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9187–9196.

Chen, Tianqi and Carlos Guestrin (2016). “Xgboost: A scalable tree boosting system”. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794.

Chen, Xue-Wen and Xiaotong Lin (2014). “Big data deep learning: challenges and perspectives”. In: *IEEE access* 2, pp. 514–525.

Chen, Zetao, Stephanie Lowry, Adam Jacobson, Zongyuan Ge, and Michael Milford (2015). “Distance metric learning for feature-agnostic place recognition”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 2556–2563.

Cortes, Corinna and Vladimir Vapnik (1995). “Support-vector networks”. In: *Machine learning* 20.3, pp. 273–297.

Costa, Bruno Sielly Jales, Clauber Gomes Bezerra, Luiz Affonso Guedes, and Plamen Parvanov Angelov (2015). “Online fault detection based on typicality and eccentricity data analytics”. In: *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–6.

Cover, Thomas and Peter Hart (1967). “Nearest neighbor pattern classification”. In: *IEEE transactions on information theory* 13.1, pp. 21–27.

Cunningham, Padraig and Sarah Jane Delany (2007). “k-Nearest neighbour classifiers”. In: *Multiple Classifier Systems* 34.8, pp. 1–17.

Dara, Suresh and Priyanka Tumma (2018). “Feature extraction by using deep learning: A survey”. In: *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*. IEEE, pp. 1795–1801.

Dean, Jeff, David Patterson, and Cliff Young (2018). “A new golden age in computer architecture: Empowering the machine-learning revolution”. In: *IEEE Micro* 38.2, pp. 21–29.

Demidova, Liliya, Evgeny Nikulchev, and Yulia Sokolova (2016). “Big data classification using the SVM classifiers with the modified particle swarm optimization and the SVM ensembles”. In: *International Journal of Advanced Computer Science and Applications* 7.5, pp. 294–312.

Deng, Li and Dong Yu (2014). “Deep learning: methods and applications”. In: *Foundations and trends in signal processing* 7.3–4, pp. 197–387.

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

Dieber, Jürgen and Sabrina Kirrane (2020). “Why model why? Assessing the strengths and limitations of LIME”. In: *arXiv preprint arXiv:2012.00093*.

Dino, Hivi Ismat and Maiwan Bahjat Abdulrazzaq (2019). “Facial expression classification based on SVM, KNN and MLP classifiers”. In: *2019 International Conference on Advanced Science and Engineering (ICOASE)*. IEEE, pp. 70–75.

Dobbin, Kevin K and Richard M Simon (2011). “Optimally splitting cases for training and testing high dimensional classifiers”. In: *BMC medical genomics* 4.1, p. 31.

Dorogush, Anna Veronika, Vasily Ershov, and Andrey Gulin (2018). “CatBoost: gradient boosting with categorical features support”. In: *arXiv preprint arXiv:1810.11363*.

Doshi-Velez, Finale and Been Kim (2017). “Towards a rigorous science of interpretable machine learning”. In: *arXiv preprint arXiv:1702.08608*.

Dressel, Julia and Hany Farid (2018). “The accuracy, fairness, and limits of predicting recidivism”. In: *Science advances* 4.1, eaao5580.

Du, Qiang, Vance Faber, and Max Gunzburger (1999). “Centroidal Voronoi tessellations: Applications and algorithms”. In: *SIAM review* 41.4, pp. 637–676.

Du, Xuefeng, Xin Wang, Gabriel Gozum, and Yixuan Li (2022). “Unknown-Aware Object Detection: Learning What You Don’t Know from Videos in the Wild”. In: *arXiv preprint arXiv:2203.03800*.

Duan, Kai-Bo and S Sathiya Keerthi (2005). “Which is the best multiclass SVM method? An empirical study”. In: *International workshop on multiple classifier systems*. Springer, pp. 278–285.

Duan, Shiyu, Shujian Yu, and José C Prıncipe (2021). “Modularizing deep learning via pairwise learning with kernels”. In: *IEEE Transactions on Neural Networks and Learning Systems*.

Faces, Caltech (1999). *Computational vision at caltech*.

Favarò, Francesca M, Nazanin Nader, Sky O Eurich, Michelle Tripp, and Naresh Varadaraju (2017). “Examining accident reports involving autonomous vehicles in California”. In: *PLoS one* 12.9, e0184952.

Fawcett, Tom (2006). “An introduction to ROC analysis”. In: *Pattern recognition letters* 27.8, pp. 861–874.

Fei-Fei, Li, Rob Fergus, and Pietro Perona (2004). “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*. IEEE, pp. 178–178.

— (2007). “Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories”. In: *Computer vision and Image understanding* 106.1, pp. 59–70.

Feurer, Matthias, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter (2015). “Efficient and robust automated machine learning”. In: *Advances in neural information processing systems*, pp. 2962–2970.

Figueiredo, Mauricio and Fernando Gomide (1999). “Design of fuzzy systems using neurofuzzy networks”. In: *IEEE transactions on Neural Networks* 10.4, pp. 815–827.

Firouzi, Farshad et al. (2021). “Harnessing the Power of Smart and Connected Health to Tackle COVID-19: IoT, AI, Robotics, and Blockchain for a Better World”. In: *IEEE Internet of Things Journal*, pp. 1–1. DOI: `10.1109/JIOT.2021.3073904`.

Freudenberg, James S, Richard H Middleton, and Victor Solo (2010). "Stabilization and disturbance attenuation over a Gaussian communication channel". In: *IEEE Transactions on Automatic Control* 55.3, pp. 795–799.

Friedman, Nir, Dan Geiger, and Moises Goldszmidt (1997). "Bayesian network classifiers". In: *Machine learning* 29.2, pp. 131–163.

Fukushima, Kunihiko and Sei Miyake (1982). "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition". In: *Competition and cooperation in neural nets*. Springer, pp. 267–285.

Garcia, Cristiano, Daniel Leite, and Igor Škrjanc (2019). "Incremental missing-data imputation for evolving fuzzy granular prediction". In: *IEEE Transactions on Fuzzy Systems* 28.10, pp. 2348–2362.

Gawehn, Erik, Jan A Hiss, and Gisbert Schneider (2016). "Deep learning in drug discovery". In: *Molecular informatics* 35.1, pp. 3–14.

Ghai, Bhavya, Q Vera Liao, Yunfeng Zhang, Rachel Bellamy, and Klaus Mueller (2020). "Explainable active learning (xal): An empirical study of how local explanations impact annotator experience". In: *arXiv preprint arXiv:2001.09219*.

Gilpin, Leilani H, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal (2018). "Explaining explanations: An overview of interpretability of machine learning". In: *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*. IEEE, pp. 80–89.

Goodfellow, Ian, Yoshua Bengio, Aaron Courville, and Yoshua Bengio (2016). *Deep learning*. Vol. 1. 2. MIT press Cambridge.

Goodman, Bryce and Seth Flaxman (2017). "European Union regulations on algorithmic decision-making and a "right to explanation"". In: *AI magazine* 38.3, pp. 50–57.

Graf, Arnulf BA, Olivier Bousquet, Gunnar Rätsch, and Bernhard Schölkopf (2009). "Prototype classification: Insights from machine learning". In: *Neural computation* 21.1, pp. 272–300.

Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton (2013). "Speech recognition with deep recurrent neural networks". In: *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, pp. 6645–6649.

Gregor, Karol, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra (2015). "Draw: A recurrent neural network for image generation". In: *International Conference on Machine Learning*. PMLR, pp. 1462–1471.

Griffin, Gregory, Alex Holub, and Pietro Perona (2007). "Caltech-256 object category dataset". In:

Grigorescu, Sorin, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu (2020). "A survey of deep learning techniques for autonomous driving". In: *Journal of Field Robotics* 37.3, pp. 362–386.

Grosan, Crina and Ajith Abraham (2011). “Rule-based expert systems”. In: *Intelligent systems*. Springer, pp. 149–185.

Gu, Xiaowei, Plamen Angelov, and Eduardo A. Soares (2020a). “A self-adaptive synthetic over-sampling technique for imbalanced classification”. In: *International Journal of Intelligent Systems* 35.6, pp. 923–943.

Gu, Xiaowei, Plamen P Angelov, and Eduardo A Soares (2020b). “A self-adaptive synthetic over-sampling technique for imbalanced classification”. In: *International Journal of Intelligent Systems* 35.6, pp. 923–943.

Gunning, David (2017). “Explainable artificial intelligence (xai)”. In: *Defense Advanced Research Projects Agency (DARPA), nd Web* 2.2.

Gunning, David and David Aha (2019). “DARPA’s explainable artificial intelligence (XAI) program”. In: *AI Magazine* 40.2, pp. 44–58.

Gurney, Jeffrey K (2013). “Sue my car not me: Products liability and accidents involving autonomous vehicles”. In: *U. Ill. JL Tech. & Pol’y*, p. 247.

Hagras, Hani (2018). “Toward Human-Understandable, Explainable AI”. In: *Computer* 51.9, pp. 28–36. DOI: 10.1109/MC.2018.3620965.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). “The elements of statistical learnin”. In: *Cited on*, p. 33.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). “Spatial pyramid pooling in deep convolutional networks for visual recognition”. In: *IEEE transactions on pattern analysis and machine intelligence* 37.9, pp. 1904–1916.

— (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hearst, Marti A., Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf (1998). “Support vector machines”. In: *IEEE Intelligent Systems and their applications* 13.4, pp. 18–28.

Hinton, Geoffrey E, Simon Osindero, and Yee-Whye Teh (2006). “A fast learning algorithm for deep belief nets”. In: *Neural computation* 18.7, pp. 1527–1554.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780.

Holzinger, Andreas, Chris Biemann, Constantinos S Pattichis, and Douglas B Kell (2017). “What do we need to build explainable AI systems for the medical domain?” In: *arXiv preprint arXiv:1712.09923*.

Holzinger, Andreas, Peter Kieseberg, Edgar Weippl, and A Min Tjoa (2018). “Current advances, trends and challenges of machine learning and knowledge extraction: from machine learning to explainable AI”. In: *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*. Springer, pp. 1–8.

Holzinger, Andreas, Georg Langs, Helmut Denk, Kurt Zatloukal, and Heimo Müller (2019). “Causability and explainability of artificial intelligence in medicine”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 9.4, e1312.

Homsi, Masun Nabhan, Natasha Medina, Miguel Hernandez, Natacha Quintero, Gilberto Perpiñan, Andrea Quintana, and Philip Warrick (2016). "Automatic heart sound recording classification using a nested set of ensemble algorithms". In: *2016 Computing in Cardiology Conference (CinC)*. IEEE, pp. 817–820.

Hosny, Ahmed, Chintan Parmar, John Quackenbush, Lawrence H Schwartz, and Hugo JWL Aerts (2018). "Artificial intelligence in radiology". In: *Nature Reviews Cancer* 18.8, pp. 500–510.

Hu, Junlin, Jiwen Lu, and Yap-Peng Tan (2015). "Deep transfer metric learning". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 325–333.

Ibrahim, Mark, Melissa Louie, Ceena Modarres, and John Paisley (2019). "Global explanations of neural networks: Mapping the landscape of predictions". In: *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 279–287.

Jang, J-SR (1993). "ANFIS: adaptive-network-based fuzzy inference system". In: *IEEE transactions on systems, man, and cybernetics* 23.3, pp. 665–685.

Jebara, Tony (2004). "Multi-task feature and kernel selection for SVMs". In: *Proceedings of the twenty-first international conference on Machine learning*, p. 55.

Johnson, Kristin, Frank Pasquale, and Jennifer Chapman (2019). "Artificial intelligence, machine learning, and bias in finance: toward responsible innovation". In: *Fordham L. Rev.* 88, p. 499.

Kanai, Sekitoshi, Yasuhiro Fujiwara, and Sotetsu Iwamura (2017). "Preventing gradient explosions in gated recurrent units". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 435–444.

Kangin, Dmitry and Plamen Angelov (2015). "Recursive svm based on teda". In: *International Symposium on Statistical Learning and Data Sciences*. Springer, pp. 156–168.

Kasabov, Nikola K and Qun Song (2002). "DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction". In: *IEEE transactions on Fuzzy Systems* 10.2, pp. 144–154.

Kay, Edmund and Anurag Agarwal (2016). "Dropconnected neural network trained with diverse features for classifying heart sounds". In: *2016 Computing in Cardiology Conference (CinC)*. IEEE, pp. 617–620.

Keerthi, S Sathiya and Chih-Jen Lin (2003). "Asymptotic behaviors of support vector machines with Gaussian kernel". In: *Neural computation* 15.7, pp. 1667–1689.

Kim, Hanjae, Sunghun Joung, Ig-Jae Kim, and Kwanghoon Sohn (2021). "Prototype-guided saliency feature learning for person search". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4865–4874.

Kong, Weifang and Prachi P Agarwal (2020). “Chest imaging appearance of COVID-19 infection”. In: *Radiology: Cardiothoracic Imaging* 2.1, e200028.

Kotsiantis, Sotiris B, Dimitris Kanellopoulos, and Panagiotis E Pintelas (2006). “Data preprocessing for supervised leaning”. In: *International journal of computer science* 1.2, pp. 111–117.

Kramer, Oliver (2013). “K-nearest neighbors”. In: *Dimensionality reduction with unsupervised nearest neighbors*. Springer, pp. 13–23.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25, pp. 1097–1105.

Kulkarni, Vrushali Y and Pradeep K Sinha (2012). “Pruning of random forest classifiers: A survey and future directions”. In: *2012 International Conference on Data Science & Engineering (ICDSE)*. IEEE, pp. 64–68.

Lampert, Christoph H, Hannes Nickisch, and Stefan Harmeling (2009). “Learning to detect unseen object classes by between-class attribute transfer”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 951–958.

Langer, Markus, Daniel Oster, Timo Speith, Holger Hermanns, Lena Kästner, Eva Schmidt, Andreas Sesing, and Kevin Baum (2021). “What do we want from Explainable Artificial Intelligence (XAI)?–A stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research”. In: *Artificial Intelligence* 296, p. 103473.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). “Deep learning”. In: *nature* 521.7553, pp. 436–444.

LeCun, Yann, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel (1989). “Backpropagation applied to handwritten zip code recognition”. In: *Neural computation* 1.4, pp. 541–551.

Lee, Sang Jun and Keng Siau (2001). “A review of data mining techniques”. In: *Industrial Management & Data Systems*.

Lee-Kwang, Hyung, Yoon-Seon Song, and Keon-Myung Lee (1994). “Similarity measure between fuzzy sets and between elements”. In: *Fuzzy Sets and Systems* 62.3, pp. 291–293.

Leite, Daniel, Pyramo Costa, and Fernando Gomide (2013). “Evolving granular neural networks from fuzzy data streams”. In: *Neural Networks* 38, pp. 1–16.

Leng, Jiaxu, Ying Liu, and Shang Chen (2019). “Context-aware attention network for image recognition”. In: *Neural Computing and Applications* 31.12, pp. 9295–9305.

Leslie, Christina, Eleazar Eskin, and William Stafford Noble (2001). “The spectrum kernel: A string kernel for SVM protein classification”. In: *Biocomputing 2002*. World Scientific, pp. 564–575.

Li, He, Kaoru Ota, and Mianxiong Dong (2018a). “Learning IoT in edge: Deep learning for the Internet of Things with edge computing”. In: *IEEE network* 32.1, pp. 96–101.

Li, Oscar, Hao Liu, Chaofan Chen, and Cynthia Rudin (2018b). "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions". In: *Thirty-Second AAAI Conference on Artificial Intelligence*.

Li, Zhuohan, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joey Gonzalez (2020). "Train Big, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers". In: *International Conference on Machine Learning*. PMLR, pp. 5958–5968.

Liaw, Andy, Matthew Wiener, et al. (2002). "Classification and regression by randomForest". In: *R news* 2.3, pp. 18–22.

Lipton, Zachary C, John Berkowitz, and Charles Elkan (2015). "A critical review of recurrent neural networks for sequence learning". In: *arXiv preprint arXiv:1506.00019*.

Liu, Chen et al. (2018). "Memory-efficient deep learning on a SpiNNaker 2 prototype". In: *Frontiers in neuroscience* 12, p. 840.

Liu, Chengyu et al. (2016). "An open access database for the evaluation of heart sound algorithms". In: *Physiological Measurement* 37.12, p. 2181.

Logan, Beth et al. (2000). "Mel Frequency Cepstral Coefficients for Music Modeling." In: *International Society for Music Information Retrieval (ISMIR)*. Vol. 270, pp. 1–11.

Long, Yang and Ling Shao (2017). "Learning to recognise unseen classes by a few similes". In: *Proceedings of the 25th ACM international conference on Multimedia*, pp. 636–644.

Lu, Hongfang and Xin Ma (2020). "Hybrid decision tree-based machine learning models for short-term water quality prediction". In: *Chemosphere* 249, p. 126169.

Lughofer, Edwin David (2008). "FLEXFIS: A robust incremental learning approach for evolving Takagi–Sugeno fuzzy models". In: *IEEE Transactions on fuzzy systems* 16.6, pp. 1393–1410.

Lundberg, Scott and Su-In Lee (2017). "A unified approach to interpreting model predictions". In: *arXiv preprint arXiv:1705.07874*.

MacCarthy, Mark (2019). "An Examination of the Algorithmic Accountability Act of 2019". In: *Available at SSRN 3615731*.

Mamdani, Ebrahim H and Sedrak Assilian (1975). "An experiment in linguistic synthesis with a fuzzy logic controller". In: *International journal of man-machine studies* 7.1, pp. 1–13.

Marcus, Gary (2018). "Deep learning: A critical appraisal". In: *arXiv preprint arXiv:1801.00631*.

McCulloch, Warren S and Walter Pitts (1943). "A logical calculus of the ideas immanent in nervous activity". In: *The bulletin of mathematical biophysics* 5.4, pp. 115–133.

McHugh, Mary L (2011). "Multiple comparison analysis testing in ANOVA". In: *Biochemia medica* 21.3, pp. 203–209.

Medsker, Larry R and LC Jain (2001). "Recurrent neural networks". In: *Design and Applications* 5.

Mizuno, Kosuke, Yosuke Terachi, Kenta Takagi, Shintaro Izumi, Hiroshi Kawaguchi, and Masahiko Yoshimoto (2012). “Architectural study of HOG feature extraction processor for real-time object detection”. In: *2012 IEEE Workshop on Signal Processing Systems*. IEEE, pp. 197–202.

Mohri, Mehryar, Afshin Rostamizadeh, and Ameet Talwalkar (2018). *Foundations of machine learning*. MIT press.

Mokhtari, Karim El, Ben Peachey Higdon, and Ayşe Başar (2019). “Interpreting financial time series with SHAP values”. In: *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering*, pp. 166–172.

Moraes, Rodrigo, João Francisco Valiati, and Wilson P GaviãO Neto (2013). “Document-level sentiment classification: An empirical comparison between SVM and ANN”. In: *Expert Systems with Applications* 40.2, pp. 621–633.

Müller, Berndt, Joachim Reinhardt, and Michael T Strickland (1995). *Neural networks: an introduction*. Springer Science & Business Media.

Nageshrao, Subramanya, H Eric Tseng, and Dimitar Filev (2019). “Autonomous highway driving using deep reinforcement learning”. In: *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE, pp. 2326–2331.

Nebel, David, Marika Kaden, Andrea Villmann, and Thomas Villmann (2017). “Types of (dis-) similarities and adaptive mixtures thereof for improved classification learning”. In: *Neurocomputing* 268, pp. 42–54.

Ng, Ming-Yen et al. (2020). “Imaging profile of the COVID-19 infection: radiologic findings and literature review”. In: *Radiology: Cardiothoracic Imaging* 2.1, e200034.

Nguyen, Anh, Jason Yosinski, and Jeff Clune (2015). “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436.

Nielsen, Michael A (2015). *Neural networks and deep learning*. Vol. 25. Determination press San Francisco, CA.

Noble, William S (2006). “What is a support vector machine?” In: *Nature biotechnology* 24.12, pp. 1565–1567.

O’Shea, Keiron and Ryan Nash (2015). “An introduction to convolutional neural networks”. In: *arXiv preprint arXiv:1511.08458*.

O’Mahony, Niall, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh (2019). “Deep learning vs. traditional computer vision”. In: *Science and Information Conference*. Springer, pp. 128–144.

Okabe, Atsuyuki, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu (2009). *Spatial tessellations: concepts and applications of Voronoi diagrams*. Vol. 501. John Wiley & Sons.

Oord, Aaron van den, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu (2016). "Wavenet: A generative model for raw audio". In: *arXiv preprint arXiv:1609.03499*.

Orr, Mark JL et al. (1996). *Introduction to radial basis function networks*.

Oshiro, Thais Mayumi, Pedro Santoro Perez, and José Augusto Baranauskas (2012). "How many trees in a random forest?" In: *International workshop on machine learning and data mining in pattern recognition*. Springer, pp. 154–168.

Oyedotun, Oyebade K and Adnan Khashman (2017). "Prototype-incorporated emotional neural network". In: *IEEE transactions on neural networks and learning systems* 29.8, pp. 3560–3572.

Padarian, José, Alex B McBratney, and Budiman Minasny (2020). "Game theory interpretation of digital soil mapping convolutional neural networks". In: *Soil* 6.2, pp. 389–397.

Pak, Myeongsuk and Sanghoon Kim (2017). "A review of deep learning in image recognition". In: *2017 4th international conference on computer applications and information processing technology (CAIPT)*. IEEE, pp. 1–3.

Pal, Mahesh (2005). "Random forest classifier for remote sensing classification". In: *International journal of remote sensing* 26.1, pp. 217–222.

Pan, YN, J Chen, and XL Li (2009). "Spectral entropy: a complementary index for rolling element bearing performance degradation assessment". In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 223.5, pp. 1223–1231.

Pasquale, Frank (2015). *The black box society*. Harvard University Press.

Pedreschi, Dino, Fosca Giannotti, Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, and Franco Turini (2019). "Meaningful explanations of Black Box AI decision systems". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 9780–9784.

Perner, Petra (2008). "Prototype-based classification". In: *Applied Intelligence* 28.3, pp. 238–246.

Phillips, PJ, CA Hahn, PC Fontana, DA Broniatowski, and MA Przybocki (2020). *Four Principles of Explainable Artificial Intelligence Online: http://dx. doi. org/10.6028/NIST*.

PhysioToolkit, PhysioBank (n.d.). "PhysioNet: components of a new research resource for complex physiologic signals". In: *Circulation. v101 i23. e215-e220* ().

Potes, Cristhian, Saman Parvaneh, Asif Rahman, and Bryan Conroy (2016). "Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds". In: *2016 computing in cardiology conference (CinC)*. IEEE, pp. 621–624.

Pradhan, Ashis (2012). "Support vector machine-a survey". In: *International Journal of Emerging Technology and Advanced Engineering* 2.8, pp. 82–85.

Prasatha, VS, Haneen Arafat Abu Alfeilate, AB Hassanate, Omar Lasassmehe, Ahmad S Tarawnehf, Mahmoud Bashir Alhasanatg, and Hamzeh S Eyal Salmane (2017). "Effects of distance measure choice on knn classifier performance-a review". In: *arXiv preprint arXiv:1708.04321*, p. 56.

Prokhorenkova, L, G Gusev, A Vorobev, AV Dorogush, and A Gulin (2018). "CatBoost: unbiased 467 boosting with categorical features". In: *Advances* 468.

Rai, Arun (2020). "Explainable AI: From black box to glass box". In: *Journal of the Academy of Marketing Science* 48.1, pp. 137–141.

Raub, McKenzie (2018). "Bots, bias and big data: artificial intelligence, algorithmic bias and disparate impact liability in hiring practices". In: *Ark. L. Rev.* 71, p. 529.

Rezaei, Mahdi and Mutsuhiro Terauchi (2013). "Vehicle detection based on multi-feature clues and Dempster-Shafer fusion theory". In: *Pacific-Rim Symposium on Image and Video Technology*. Springer, pp. 60–72.

Rish, Irina et al. (2001). "An empirical study of the naive Bayes classifier". In: *IJCAI 2001 workshop on empirical methods in artificial intelligence*. Vol. 3. 22, pp. 41–46.

Rokach, Lior and Oded Maimon (2005). "Decision trees". In: *Data mining and knowledge discovery handbook*. Springer, pp. 165–192.

Rosenblatt, Frank (1958). "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6, p. 386.

Ross, Lou (2016). "The Image Processing Handbook, John C. Russ and F. Brent Neal. CRC Press, Boca Raton, FL, 2015, 1053 pp. ISBN: 978-1498740265." In: *Microscopy and Microanalysis* 22.3, pp. 733–733.

Rubin, Jonathan, Rui Abreu, Anurag Ganguli, Saigopal Nelaturi, Ion Matei, and Kumar Sricharan (2016). "Classifying heart sound recordings using deep convolutional neural networks and mel-frequency cepstral coefficients". In: *2016 Computing in cardiology conference (CinC)*. IEEE, pp. 813–816.

Rudin, Cynthia (2019). "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead". In: *Nature Machine Intelligence* 1.5, pp. 206–215.

Rudin, Cynthia and Joanna Radin (2019). "Why are we using black box models in AI when we don't need to? A lesson from an explainable AI competition". In: *Harvard Data Science Review* 1.2.

Safavian, S Rasoul and David Landgrebe (1991). "A survey of decision tree classifier methodology". In: *IEEE transactions on systems, man, and cybernetics* 21.3, pp. 660–674.

Saralajew, Sascha, Lars Holdijk, Maike Rees, and Thomas Villmann (2018). "Prototype-based neural network layers: incorporating vector quantization". In: *arXiv preprint arXiv:1812.01214*.

Schmidhuber, Jürgen (2015). "Deep learning in neural networks: An overview". In: *Neural networks* 61, pp. 85–117.

Schuldt, Christian, Ivan Laptev, and Barbara Caputo (2004). “Recognizing human actions: a local SVM approach”. In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* Vol. 3. IEEE, pp. 32–36.

Schuster, Mike and Kuldip K Paliwal (1997). “Bidirectional recurrent neural networks”. In: *IEEE transactions on Signal Processing* 45.11, pp. 2673–2681.

Schwarting, Wilko, Javier Alonso-Mora, and Daniela Rus (2018). “Planning and decision-making for autonomous vehicles”. In: *Annual Review of Control, Robotics, and Autonomous Systems.*

Sejnowski, Terrence J (2018). *The deep learning revolution.* Mit Press.

Selvaraju, Ramprasaath R, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra (2017). “Grad-cam: Visual explanations from deep networks via gradient-based localization”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 618–626.

Shaha, Manali and Meenakshi Pawar (2018). “Transfer learning for image classification”. In: *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA).* IEEE, pp. 656–660.

Sharma, Vikas and Anand Parey (2016). “A review of gear fault diagnosis using various condition indicators”. In: *Procedia Engineering* 144, pp. 253–263.

Shen, Yuan, Shanduojiao Jiang, Yanlin Chen, Eileen Yang, Xilun Jin, Yuliang Fan, and Katie Driggs Campbell (2020). “To explain or not to explain: A study on the necessity of explanations for autonomous vehicles”. In: *arXiv preprint arXiv:2006.11684.*

Shi, Heshui, Xiaoyu Han, Nanchuan Jiang, Yukun Cao, Osamah Alwalid, Jin Gu, Yanqing Fan, and Chuansheng Zheng (2020). “Radiological findings from 81 patients with COVID-19 pneumonia in Wuhan, China: a descriptive study”. In: *The Lancet Infectious Diseases.*

Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman (2013). “Deep inside convolutional networks: Visualising image classification models and saliency maps”. In: *arXiv preprint arXiv:1312.6034.*

Simonyan, Karen and Andrew Zisserman (2014). “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556.*

Škrjanc, Igor, Jose Antonio Iglesias, Araceli Sanchis, Daniel Leite, Edwin Lughofer, and Fernando Gomide (2019). “Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: a survey”. In: *Information Sciences* 490, pp. 344–368.

Smagulova, Kamilya and Alex Pappachen James (2019). “A survey on LSTM memristive neural network architectures and applications”. In: *The European Physical Journal Special Topics* 228.10, pp. 2313–2324.

Soares, Eduardo and Plamen Angelov (2019). “Fair-by-design explainable models for prediction of recidivism”. In: *arXiv preprint arXiv:1910.02043.*

Soares, Eduardo and Plamen Angelov (2020). *A large dataset of real patients CT scans for COVID-19 identification*. Version V1. Harvard. DOI: `10.7910/DVN/SZDUQX`. URL: `https://doi.org/10.7910/DVN/SZDUQX`.

— (2021). “RADNN: Robust to Imperceptible Adversarial Attacks Deep Neural Network”. In: *TechRxiv*, pp. 1–8.

Soares, Eduardo, Plamen Angelov, Sarah Biaso, Michele Higa Froes, and Daniel Kanda Abe (2020a). “SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification”. In: *medRxiv*.

Soares, Eduardo, Plamen Angelov, Bruno Costa, and Marcos Castro (2019a). “Actively semi-supervised deep rule-based classifier applied to adverse driving scenarios”. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8.

Soares, Eduardo, Plamen Angelov, Dimitar Filev, Bruno Costa, Marcos Castro, and Subramanya Nageshrao (2019b). “Explainable density-based approach for self-driving actions classification”. In: *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, pp. 469–474.

Soares, Eduardo, Plamen Angelov, and Xiaowei Gu (2020b). “Autonomous Learning Multiple-Model zero-order classifier for heart sound classification”. In: *Applied Soft Computing* 94, p. 106449.

Soares, Eduardo, Plamen Angelov, and Ziyang Zhang (2021). “An Explainable approach to Deep Learning from CT-scans for Covid Identification”. In: *TechRxiv*, pp. 1–8.

Soares, Eduardo A, Heloisa A Camargo, Suzana J Camargo, and Daniel F Leite (2018). “Incremental gaussian granular fuzzy modeling applied to hurricane track forecasting”. In: *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, pp. 1–8.

Soares, Eduardo Almeida, Plamen P Angelov, Bruno Costa, Marcos Castro, Subramanya Nageshrao, and Dimitar Filev (2020c). “Explaining deep learning models through rule-based approximation and visualization”. In: *IEEE Transactions on Fuzzy Systems*.

Solmaz, Berkan, Shayan Modiri Assari, and Mubarak Shah (2013). “Classifying web videos using a global video descriptor”. In: *Machine vision and applications* 24.7, pp. 1473–1485.

Song, Yan-Yan and LU Ying (2015). “Decision tree methods: applications for classification and prediction”. In: *Shanghai archives of psychiatry* 27.2, p. 130.

Soria, Daniele, Jonathan M Garibaldi, Federico Ambrogi, Elia M Biganzoli, and Ian O Ellis (2011). “A ‘non-parametric’version of the naive Bayes classifier”. In: *Knowledge-Based Systems* 24.6, pp. 775–784.

Souza, Paulo Vitor de Campos, Eduardo A Soares, Augusto Junio Guimarães, Vanessa Souza Araujo, Vinicius Jonathan S Araujo, and Thiago Silva Rezende (2020). “Autonomous data density pruning fuzzy neural network for optical interconnection network”. In: *Evolving Systems*.

Stilgoe, Jack (2020). “Who Killed Elaine Herzberg?” In: *Who’s Driving Innovation?* Springer, pp. 1–6.

Stock, Pierre and Moustapha Cisse (2018). “Convnets and imagenet beyond accuracy: Understanding mistakes and uncovering biases”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 498–512.

Suthaharan, Shan (2016). “Support vector machine”. In: *Machine learning models and algorithms for big data classification*. Springer, pp. 207–235.

Suykens, JAK, Lukas Lukas, Paul Van Dooren, Bart De Moor, Joos Vandewalle, et al. (1999). “Least squares support vector machine classifiers: a large scale algorithm”. In: *European Conference on Circuit Theory and Design, ECCTD*. Vol. 99. Citeseer, pp. 839–842.

Svozil, Daniel, Vladimir Kvasnicka, and Jiri Pospichal (1997). “Introduction to multi-layer feed-forward neural networks”. In: *Chemometrics and intelligent laboratory systems* 39.1, pp. 43–62.

Swain, Philip H and Hans Hauska (1977). “The decision tree classifier: Design and potential”. In: *IEEE Transactions on Geoscience Electronics* 15.3, pp. 142–147.

Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi (2017). “Inception-v4, inception-resnet and the impact of residual connections on learning”. In: *Thirty-first AAAI conference on artificial intelligence*.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich (2015). “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

Takagi, Tomohiro and Michio Sugeno (1985). “Fuzzy identification of systems and its applications to modeling and control”. In: *IEEE transactions on systems, man, and cybernetics* 1, pp. 116–132.

Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu (2018). “A survey on deep transfer learning”. In: *International conference on artificial neural networks*. Springer, pp. 270–279.

Tan, Xiao-hui, Wei-hua Bi, Xiao-liang Hou, and Wei Wang (2011). “Reliability analysis using radial basis function networks and support vector machines”. In: *Computers and Geotechnics* 38.2, pp. 178–186.

Tang, Duyu, Bing Qin, and Ting Liu (2015). “Document modeling with gated recurrent neural network for sentiment classification”. In: *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1422–1432.

Tarwani, Kanchan M and Swathi Edem (2017). “Survey on recurrent neural network in natural language processing”. In: *Int. J. Eng. Trends Technol* 48, pp. 301–304.

Taunk, Kashvi, Sanjukta De, Srishti Verma, and Aleena Swetapadma (2019). “A brief review of nearest neighbor algorithm for learning and classification”. In: *2019*

International Conference on Intelligent Computing and Control Systems (ICCS). IEEE, pp. 1255–1260.

Tay, Yi, Mostafa Dehghani, Dara Bahri, and Donald Metzler (2020). "Efficient transformers: A survey". In: *arXiv preprint arXiv:2009.06732.*

Ting, Daniel Shu Wei, Lawrence Carin, Victor Dzau, and Tien Y Wong (2020). "Digital technology and COVID-19". In: *Nature medicine* 26.4, pp. 459–461.

Tjoa, Erico and Cuntai Guan (2020). "A survey on explainable artificial intelligence (xai): Toward medical xai". In: *IEEE Transactions on Neural Networks and Learning Systems.*

Tritscher, Julian, Markus Ring, Daniel Schlr, Lena Hettinger, and Andreas Hotho (2020). "Evaluation of Post-hoc XAI Approaches Through Synthetic Tabular Data". In: *International Symposium on Methodologies for Intelligent Systems.* Springer, pp. 422–430.

Varshney, Kush R and Homa Alemzadeh (2017). "On the safety of machine learning: Cyber-physical systems, decision sciences, and data products". In: *Big data* 5.3, pp. 246–255.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need". In: *arXiv preprint arXiv:1706.03762.*

Voulodimos, Athanasios, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis (2018). "Deep learning for computer vision: A brief review". In: *Computational intelligence and neuroscience* 2018.

Wang, Danding, Qian Yang, Ashraf Abdul, and Brian Y Lim (2019a). "Designing theory-driven user-centric explainable AI". In: *Proceedings of the 2019 CHI conference on human factors in computing systems*, pp. 1–15.

Wang, Qiang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao (2019b). "Learning deep transformer models for machine translation". In: *arXiv preprint arXiv:1906.01787.*

Weber, M and M Weber (1999). "Caltech frontal face database". In: *California Institute of Technology.*

Wexler, Rebecca (2017). "When a computer program keeps you in jail: How computers are harming criminal justice". In: *New York Times* 13.

Whitaker, Bradley M, Pradyumna B Suresha, Chengyu Liu, Gari D Clifford, and David V Anderson (2017). "Combining sparse coding and time-domain features for heart sound classification". In: *Physiological measurement* 38.8, p. 1701.

Xiao, Jianli (2019). "SVM and KNN ensemble learning for traffic incident detection". In: *Physica A: Statistical Mechanics and its Applications* 517, pp. 29–35.

Xing, Wenchao and Yilin Bei (2019). "Medical health big data classification based on KNN classification algorithm". In: *IEEE Access* 8, pp. 28808–28819.

Xiong, Wayne, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig (2016). "Achieving human parity in conversational speech recognition". In: *arXiv preprint arXiv:1610.05256*.

Yin, Wenpeng, Katharina Kann, Mo Yu, and Hinrich Schütze (2017). "Comparative study of CNN and RNN for natural language processing". In: *arXiv preprint arXiv:1702.01923*.

Yoo, Seung Hoon et al. (2020). "Deep learning-based decision-tree classifier for COVID-19 diagnosis from chest X-ray imaging". In: *Frontiers in medicine* 7, p. 427.

Zabihi, Morteza, Ali Bahrami Rad, Serkan Kiranyaz, Moncef Gabbouj, and Aggelos K Katsaggelos (2016). "Heart sound anomaly and quality detection using ensemble of neural networks without segmentation". In: *2016 computing in cardiology conference (CinC)*. IEEE, pp. 613–616.

Zadeh, Lofti A. (1965). "Information and control". In: *Fuzzy sets* 8.3, pp. 338–353.

— (1983). "The role of fuzzy logic in the management of uncertainty in expert systems". In: *Fuzzy sets and systems* 11.1-3, pp. 199–227.

Zakariah, Mohammed et al. (2014). "Classification of large datasets using Random Forest Algorithm in various applications: Survey". In: *International Journal of Engineering and Innovative Technology (IJJEIT)* 4.3.

Zeiler, Matthew D and Rob Fergus (2014). "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer, pp. 818–833.

Zhang, Mi, Faen Zhang, Nicholas D Lane, Yuanchao Shu, Xiao Zeng, Biyi Fang, Shen Yan, and Hui Xu (2020). "Deep Learning in the Era of Edge Computing: Challenges and Opportunities". In: *Fog Computing: Theory and Practice*, pp. 67–78.

Zhong, Guoqiang, Li-Na Wang, Xiao Ling, and Junyu Dong (2016). "An overview on data representation learning: From traditional feature learning to recent deep learning". In: *The Journal of Finance and Data Science* 2.4, pp. 265–278.

Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba (2016). "Learning deep features for discriminative localization". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929.

Zhou, Lina, Shimei Pan, Jianwu Wang, and Athanasios V Vasilakos (2017). "Machine learning on big data: Opportunities and challenges". In: *Neurocomputing* 237, pp. 350–361.

Zhuang, Fuzhen, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He (2015). "Supervised representation learning: Transfer learning with deep autoencoders". In: *Twenty-Fourth International Joint Conference on Artificial Intelligence*.