UNIVERSITAT DE BARCELONA

FUNDAMENTAL PRINCIPLES OF DATA SCIENCE MASTER'S THESIS

Human explainability through an auxiliary Neural Network

Author:
Albert GARCIA

Supervisor:
Dr. Santi Seguí

A thesis submitted in partial fulfillment of the requirements
for the degree of MSc in Fundamental Principles of Data Science

in the

Facultat de Matemàtiques i Informàtica

June 30, 2020

UNIVERSITAT DE BARCELONA

Abstract

Facultat de Matemàtiques i Informàtica

MSc

Human explainability through an auxiliary Neural Network

by Albert GARCIA

Explainability in Deep Learning has become a hot topic in recent years due to the necessity of insights and justifications for predictions. Although this field has an extensive range of different approaches, this thesis explores the feasibility of a new methodology that seeks to provide human-interpretable explanations for each sample being processed by a Neural Network. The term black box is often used in the Explainability field, meaning that there is a lack in transparency within the model when processing data. The explored approach tries to deal with the black box by using the outputs of the hidden layers of a Neural Network as inputs for the model responsible for the explanations. This model is another Neural Network that can be seen as an auxiliary Neural Network to the main task. The predicted explanations are formed by a subset of a list of human-designed justifications for the possible outcomes of the main task. Using the predictions from both networks a cross comparison process is also performed in order to build confidence on the main predictions. Results successfully show how a significant proportion of incorrect outputs are questioned thanks to the predicted explanations.

Acknowledgements

First of all, this thesis would not have been possible without the guidance, support and help of my supervisor, Santi Seguí, to whom I owe a lot for giving me the opportunity to work as a Computer Vision Research Intern under his tutelage while letting me pursue the Master's thesis in a topic that piqued my interest. I also remain grateful and thankful for all the support provided to me by Pablo Laiz, a fellow PhD student and coworker.

I would also like to thank Michael DePass, a Master's colleague, for reviewing and spell checking my thesis.

Finally, huge thank you to my mother for her full support during all of my studies.

Chapter 1

Explainability in Deep Learning

1.1 Introduction

Deep Learning, a sub-field of Machine Learning, has revolutionised the world through its breakthrough predictions and forecasts. Deep Learning has outperformed Machine Learning methodologies in many ways by providing Artificial Intelligence (AI) models with the ability to learn deep and complex knowledge from large datasets. Not only has Deep Learning outperformed Machine Learning in terms of accuracy but it has also provided AI programmers with the ability to adapt a model to a specific task while at the same time adapting its complexity.

Although Deep Learning has changed the world, there are still several topics about it that need to be addressed if we want to keep using this technology for the better. Interpretability, explainability, uncertainty, fairness, ethics, confidence and safety are some of these topics. In the context of Machine Learning and AI, interpretability and explainability are sometimes used interchangeably. Nevertheless, there is a slight yet significant difference between these two terms: interpretability is about the cause-effect behaviour, or correlations, between the inputs and the outputs of a model while explainability is the extent to which the internal actions of a model can be explained in human terms. Notice how a model can be interpretable and not explainable at the same time, f.e., through experimental trials one can identify the correlations of the inputs with respect to the outputs (how inputs are affecting the outputs) while not knowing how the model is processing the input at a human level. Here is a simple and down to earth example for when there is interpretability and not explainability: at a school students are performing a chemistry experiment involving several components, students know what combinations of the components produce new products but at the same time they are completely ignorant as to how the new products are being formed. Explainability in this example would correspond to knowing all the chemical reactions and chains being produced to finally form the new product.

Now, explainability in the context of Deep Learning can be seen as knowing how the data is being processed through the Neural Network (NN) (also named Deep Neural Network, DNN, in Deep Learning) to produce the final output in human terms. Knowing the trained parameters and how these affect the input data is not enough in any way. Focusing on the explainability topic, an important trade-off appears between accuracy and explainability of models in AI. Turns out that the more complex an AI model is, and therefore the more potential it has to improve its accuracy, the less explainable it is. This has been found when comparing several Machine Learning methodologies with Deep Learning approaches. Figure 1.1 shows a qualitative plot of this trade-off to help build an intuition.

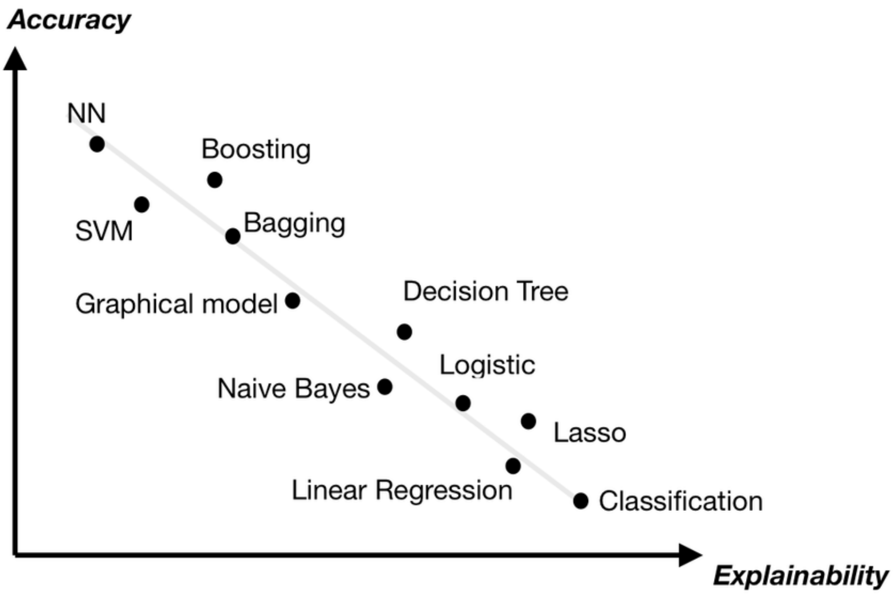


FIGURE 1.1: Qualitative plot of the accuracy and explainability trade-off. NN corresponds to the Deep Learning family while the rest of the points correspond to other Machine Learning families.

More generally speaking, when dealing with explainability in AI, the academic community usually refers to it as Explainable Artificial Intelligence, abbreviated as XAI, like the Wikipedia entry (Wikipedia contributors, 2020) does or like the explainability in AI survey (Tjoa and Guan, 2019) does. Along these lines, the popular term Black box also appears referring to a model lacking transparency with respect to the internal mechanisms at a high interpretation level.

1.2 Motivation

As previously mentioned Deep Learning and the use of Neural Networks has revolutionised the world by its breakthrough predictions and forecasts. It goes without saying that Deep Learning has also been applied to other tasks apart from prediction and forecasting. The range of applications is extremely extensive including Computer Vision (CV), Natural Language Processing (NLP), Time Series Analysis (TSA) and many more. Regardless of the precision and performance of these models there is a huge void when trying to look for human explanations of these models as seen in Figure 1.1. The reasons to ask for human explanations are really diverse. In almost all cases the motivation behind looking for these explanations is extremely significant to both users and designers of these NN models. Some possible reasons can be defined in order to build an intuition:

- Detection of undesired biases
- Legal requirements
- Build trust
- Provide transparency to the NN's black box
- Discover new interpretations and interactions

An example in a real context can be stated for each one of these reasons. Amazon has had several cases of unwanted bias when developing AI systems for automating tasks. One such case that was extremely popular was the case of a recruiting AI that showed a strong bias against women as reported by (Reuters and Jeffrey Dastin, 2018). Legal requirements can be enforced in sensitive and/or critical applications, f.e., when an AI system is involved in providing decisions in a field such as health. To build trust in an AI system is no easy task and can truly make an impact on the user experience. An example of when building trust strongly impacts the user experience would be in the field of autonomous vehicles. The fact that Neural Networks are black boxes is obvious but to be able to provide transparency to it would be a huge benefit in fields such as banking. Banks have restrictions when using Machine Learning and Deep Learning models when performing sensitive decisions related to client's money. Last but not least, to be able to understand what is happening inside of a NN in human terms can easily provide new discoveries in research fields, f.e., by discovering a hidden feature used by the NN to compute the outcome.

Many more reasons for justifying the search of human explanations in Deep Learning can be found. A big part of them revolve around the topic of Human-Computer interaction. Others are centred in fairness, ethics and safety. But in the end we can see how all of these have a huge impact and are, without a doubt, extremely significant for the development and improvement of AI systems. This is why XAI is a hot topic and will still be for years to come.

1.3 Background

Before getting into implementations and experiments we need to build an intuition on how explainability in Deep Learning is approached. There is a large number of researchers working on this hot topic and thus several lines of work have appeared. Each line of work has its own characteristics and properties regarding how to approach explainability. In order to explore the different lines of work, grouped by the characteristics of the methodologies being used, we will be referencing the XAI survey (Tjoa and Guan, 2019) as well as the Explainable Deep Learning guide (Xie et al., 2020). Since we are focusing on explainability in Deep Learning rather than including strict Machine Learning models we will be using the Explainable Deep Learning guide to build a map of the approaches. Even though the XAI survey provides a more general view of explainability in AI it is still a valuable resource to this thesis and should not be omitted.

The Explainability in Deep Learning guide proposes the following schematic for the different explainability methods in DNNs:

1. Visualisation
    - (a) Back-Propagation
    - (b) Perturbation
2. Distillation
    - (a) Local Approximation
    - (b) Model Translation
3. Intrinsic
    - (a) Attention Mechanisms
    - (b) Joint Training

Before briefly describing each one of these categories and subcategories it must be mentioned that these represent a reduction of all the lines of work being conducted, meaning that to create this category scheme the state-of-the-art approaches have been abstracted to one of these categories by defining the foundations it is based on. In other words, this categorisation has been constructed based on the foundations being used in each explainability work. Thus, this categorisation can be seen as a foundational categorisation of explainability in DNNs. Once this has been clarified, we now give a brief description for each category and subcategory in order to get an idea of the fundamentals being used in the explainability field.

1. Visualisation methods: the goal of these methods is to highlight characteristics or subsets of the input that heavily influence the output of the DNN.
    - (a) Back-Propagation: they identify the importance of input features by evaluating gradients being passed from output to input (back-propagation) during the training phase. The gradient computations performed provide a measure of how *sensitive* the output is to a specific input feature.
    - (b) Perturbation: they compute how a specific input feature affects the DNN's output by removing or altering the feature. These are able to compute the marginal relevance of each feature in a DNN with respect to the output.
2. Model distillation: they develop a separate *white box* (as opposed to black box) Machine Learning model that is trained to approximate the behaviour of the already trained DNN. The goal of this approach is to extract decision rules or features, from the mimicking Machine Learning model, that impact the output.
    - (a) Local Approximation: the Machine Learning model learns to mimic the DNN behaviour in a small subset of the data. This is motivated by the hypothesis that a DNN discriminates in a local region within the manifold formed by the data.
    - (b) Model Translation: same as the Local Approximation models with the exception that these learn over the whole data. It is worth noting that modelling causality, a relatively new hot topic in explainability, often falls within this sub-category.

3. Intrinsic methods: these methods focus on designing a DNN that is able to provide explanations for the output at the same time. A property of such DNNs is that these can be jointly trained on the main task while optimising the corresponding explanations.
    (a) Attention Mechanisms: attention mechanisms are used in a DNN in order to provide explainable outputs by expressing the operations performed. An example of an attention mechanism is one that learns conditional distributions over the inputs yielding a downstream weight vector.
    (b) Joint Training: this method introduces an additional task to the main task being performed by the model while optimising both tasks at the same time. The aim of the additional task is to provide explanations to the outputs simultaneously, f.e., by providing explanations in natural language format or by associating deep features with human-level entities.

This foundational categorisation of the explainability methods used in Deep Learning helps build the background needed for this thesis. Additionally, there are many more approaches and methodologies being researched at the moment that are really promising such as modelling causality. An example of this is (Parafita and Vitrià, 2019), a novel explainability methodology in which they identify how features affect the model's output through the use of counterfactuals over a causal model of the data. Due to the extensive literature on explainability in Deep Learning, we let the reader refer to the papers cited in (Xie et al., 2020) to see representative contributions for each one of these categories and subcategories. Irrespective of this we also show in Figure 1.2 the map from the explainability guide in order to see some papers for each line of work.

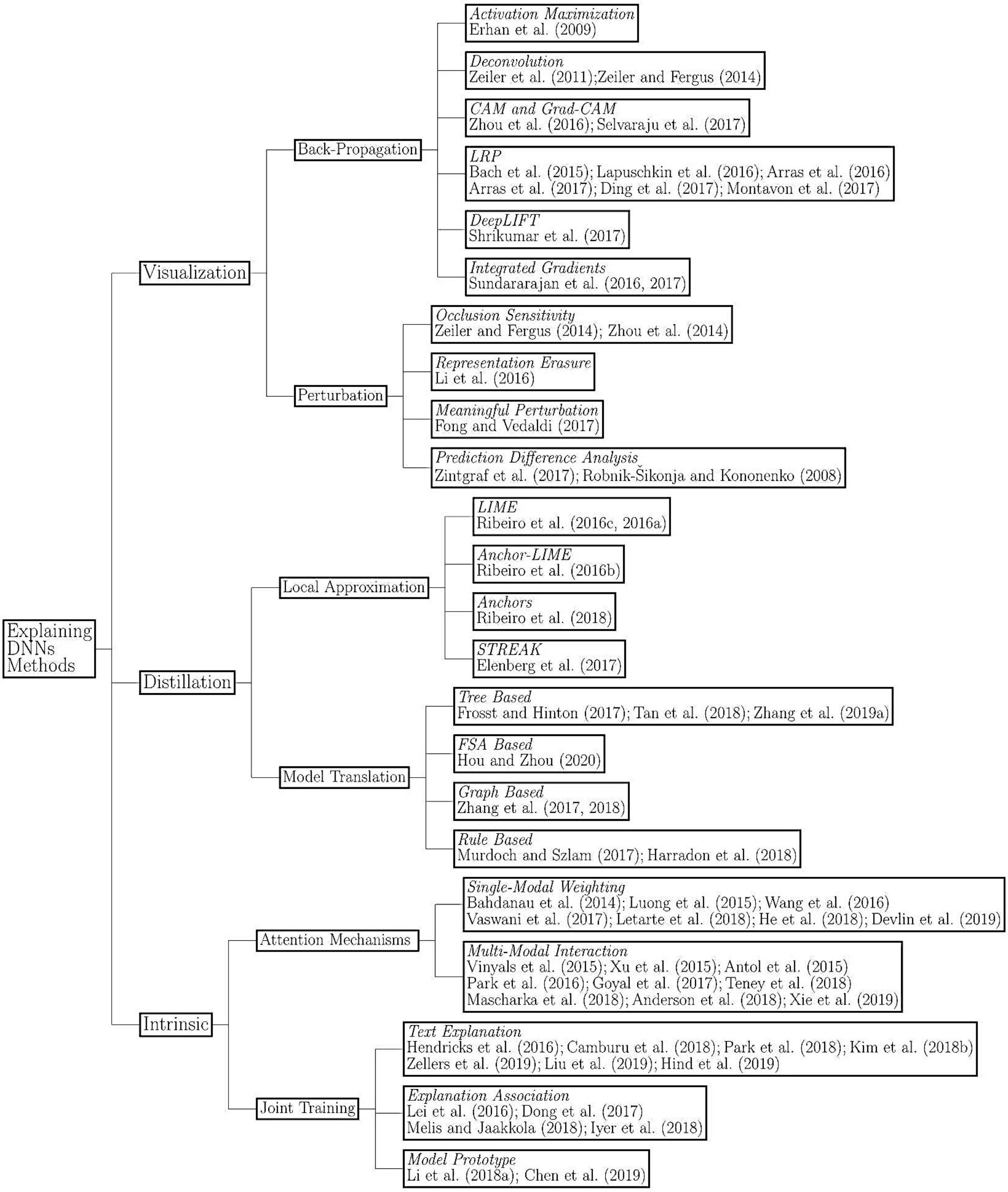

Explaining DNNs Methods
Visualization
Back-Propagation
Activation Maximization
Erhan et al. (2009)
Deconvolution
Zeiler et al. (2011);Zeiler and Fergus (2014)
CAM and Grad-CAM
Zhou et al. (2016); Selvaraju et al. (2017)
LRP
Bach et al. (2015); Lapuschkin et al. (2016); Arras et al. (2016)
Arras et al. (2017); Ding et al. (2017); Montavon et al. (2017)
DeepLIFT
Shrikumar et al. (2017)
Integrated Gradients
Sundararajan et al. (2016, 2017)
Perturbation
Occlusion Sensitivity
Zeiler and Fergus (2014); Zhou et al. (2014)
Representation Erasure
Li et al. (2016)
Meaningful Perturbation
Fong and Vedaldi (2017)
Prediction Difference Analysis
Zintgraf et al. (2017); Robnik-Šikonja and Kononenko (2008)
Distillation
Local Approximation
LIME
Ribeiro et al. (2016c, 2016a)
Anchor-LIME
Ribeiro et al. (2016b)
Anchors
Ribeiro et al. (2018)
STREAK
Elenberg et al. (2017)
Model Translation
Tree Based
Frosst and Hinton (2017); Tan et al. (2018); Zhang et al. (2019a)
FSA Based
Hou and Zhou (2020)
Graph Based
Zhang et al. (2017, 2018)
Rule Based
Murdoch and Szlam (2017); Harradon et al. (2018)
Intrinsic
Attention Mechanisms
Single-Modal Weighting
Bahdanau et al. (2014); Luong et al. (2015); Wang et al. (2016)
Vaswani et al. (2017); Letarte et al. (2018); He et al. (2018); Devlin et al. (2019)
Multi-Modal Interaction
Vinyals et al. (2015); Xu et al. (2015); Antol et al. (2015)
Park et al. (2016); Goyal et al. (2017); Teney et al. (2018)
Mascharka et al. (2018); Anderson et al. (2018); Xie et al. (2019)
Joint Training
Text Explanation
Hendricks et al. (2016); Camburu et al. (2018); Park et al. (2018); Kim et al. (2018b)
Zellers et al. (2019); Liu et al. (2019); Hind et al. (2019)
Explanation Association
Lei et al. (2016); Dong et al. (2017)
Melis and Jaakkola (2018); Iyer et al. (2018)
Model Prototype
Li et al. (2018a); Chen et al. (2019)

FIGURE 1.2: Explainability in Deep Learning map. Image source (Xie et al., 2020).

Chapter 2

Goals and Planning

2.1 Goals

Now that an introduction to the topic in question has been made, as well as building a conceptual map of how explainability in Deep Learning is approached, we can formulate the goal of this Master's thesis. The main objective of this thesis is to **design, implement and test an idea I came up with to approach explainability in Deep Learning through the use of an auxiliary DNN.** During a Summer Research Internship in Computer Vision at the University of Santiago de Compostela I was introduced to the explainability field. During my stay there I came across the idea of using an additional NN with the goal of providing human-interpretable explanations of the main NN's mechanisms and features. The auxiliary DNN should perform a classification task over a set of possible explanations regarding the main task. In order to fulfil the proposed goal, the following steps must be taken:

1. Acquire or create a dataset with the main task's targets as well as the explanations for each target.
2. Design and optimise the pair of models as well as determine how they should be interconnected.
3. Evaluate both models while focusing on a qualitative analysis of the predicted explanations.

The first step introduces an impactful decision for this thesis which is whether to find an already labeled dataset or to synthetically create one. Each one has its pros and cons but for the sake of exploration and completeness we have decided to perform an experiment for each case. Additionally, another impactful decision made for the thesis is to focus on Computer Vision tasks due to the fact that these are heavily graphical and relatively easy to asses by humans. All the source code implemented as well as the datasets used in the thesis can be found in A.

2.2 Planning

In order to achieve the previous goals, while following the decisions made, a continuous and gradual development process has been designed. This process follows a series of ordered tasks which correspond to the following:

- Design and creation of the synthetic dataset including the explanations
- Definition, implementation and optimisation of the pair of models for the synthetic dataset

- Evaluation and analysis of the trained models over the synthetic dataset
- Search for a real-world dataset with explanations
- Definition, implementation and optimisation of the pair of models for the real-world dataset
- Evaluation and analysis of the trained models over the real-world dataset
- Extraction of conclusions and thesis writing

A Gantt diagram has been designed to make the planning of the master's thesis throughout the course. The diagram is shown in Figure 2.1. Note that the days allotted as well as the ranging dates for each task are approximations, and thus are always subject to minor variations. From now on the thesis is split into project 1 and project 2. These projects correspond to the decision of using a synthetically created dataset and using a real-world dataset respectively.

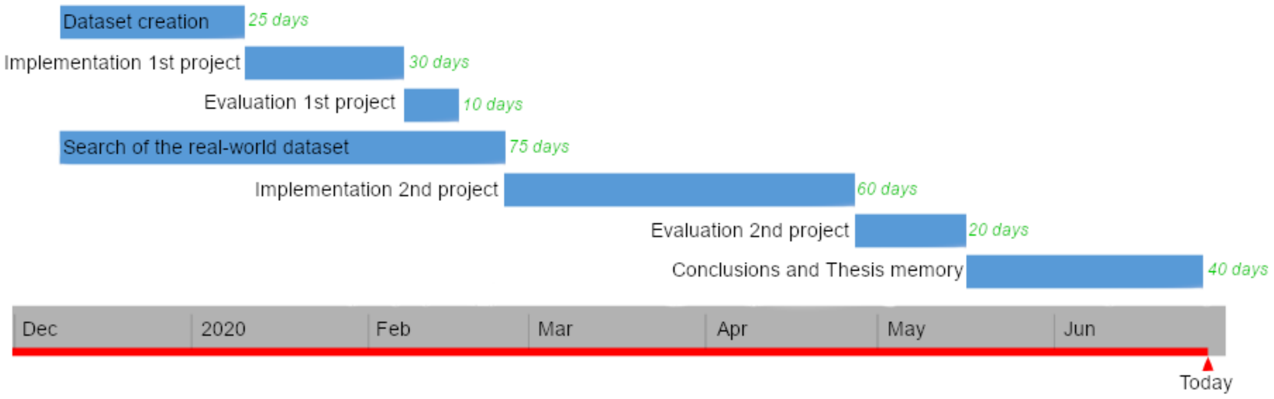


FIGURE 2.1: Gantt diagram showing the allotted days and the ranging dates for each plannified task.

Chapter 3

Project 1

3.1 Explainability proposal

Before designing and implementing any of the two projects, the idea I came up with for approaching explainability in Deep Learning has to be developed. The core of the proposal revolves around the idea that the auxiliary network, responsible for the explanations, should always work in parallel to the main network. This idea follows the hypothesis that when processing data through a NN, explanations also play a role along the processing part. To give relevance to the modelling of the explanations, the model responsible for them can be designed to work in parallel to the main network, always working together whenever an input is processed. This core idea can be visualised in Figure 3.1.

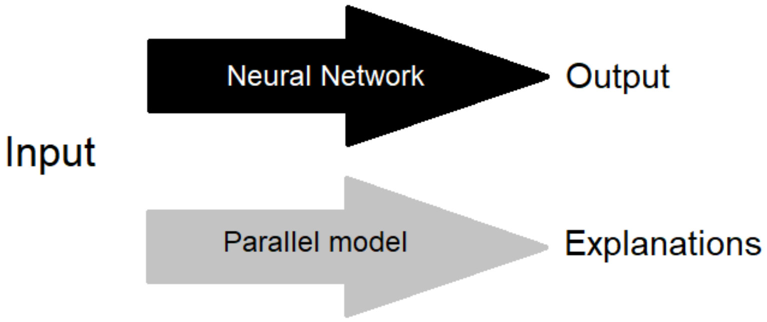


FIGURE 3.1: Core idea of the proposal made in the thesis. The model responsible for the explanations (big grey arrow) should always work in parallel with the main network (big black arrow).

Further developing the idea, in order to try to provide transparency to the main NN due to it being a complete black box, the parallel model can be designed such that its input comes from the hidden activations of the main NN. This is done with the expectation of clarifying what the main NN is internally doing by processing the hidden activations, and therefore the latent features, and outputting explanations of the main predictions. Following this inter-connectivity scheme, the explanations model can be seen as a *decoder* of the latent features into human-level explanations of the main task. This connectivity scheme is shown in Figure 3.2.

A question arises when focusing on the explanations model: should it be a Neural Network? The answers is not necessarily but at the same time it is a fairly good candidate for this *decoder* role. There are a couple of reasons for this. First of all the explanations model should be able to handle, manipulate and transform the hidden

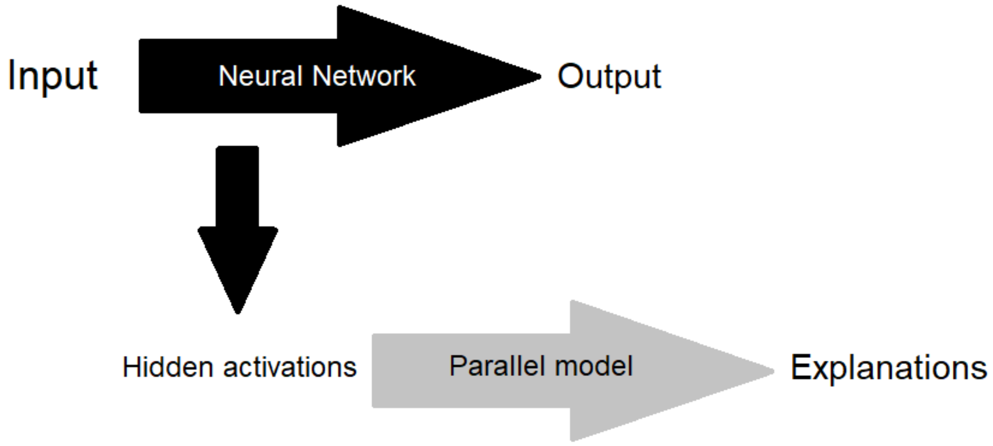


FIGURE 3.2: Connectivity scheme of the pair of models. The parallel model can be seen as a *decoder* of the latent features into human explanations.

activations from the main NN, and therefore the model should be able to correctly handle the shape of the inputted tensors as well as their distributions. The second reason is that, generally, hidden activations cannot be easily translated to human-interpretable explanations and thus the explanations model should have the learning capabilities for *decoding* these non-interpretable tensors. Additionally, when using a NN as the explanations model an advantage comes up. Since both models are composed of several layers the hidden activations can be sequentially inputted to the explanations model. It is hypothesised that this can potentially help the auxiliary NN when learning how to *decode* the main NN by providing deep features to deep layers. This final scheme, seen in Figure 3.3, corresponds to the architecture implemented in both projects.

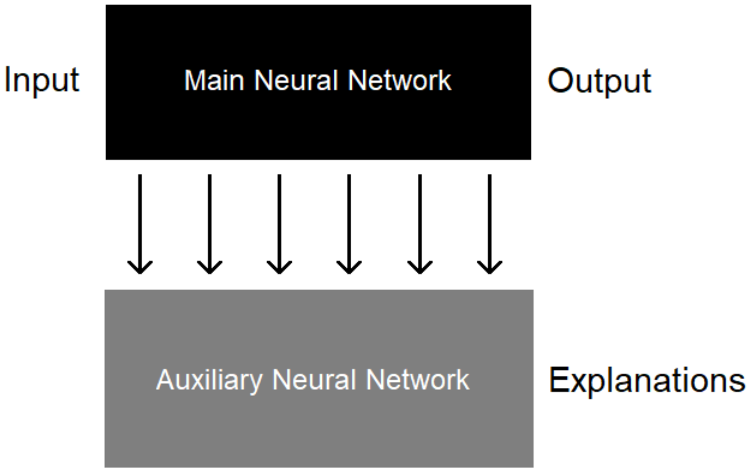


FIGURE 3.3: Connectivity scheme of the NN pair while taking advantage of the several layers of each NN. This corresponds to the final architecture followed in both projects.

As previously mentioned, the explanations should be provided by the dataset meaning that the auxiliary NN will perform a classification task over the whole set of possible explanations of the main output. Given a total of N different explanations provided by the dataset, the implementation of this classification task consists in performing N independent binary classifications at the output of the auxiliary NN. This proposal makes the explanations model yield a probability score for each explanation as seen in Figure 3.4. This means that post-processing will be usually needed such as adjusting the acceptance threshold.

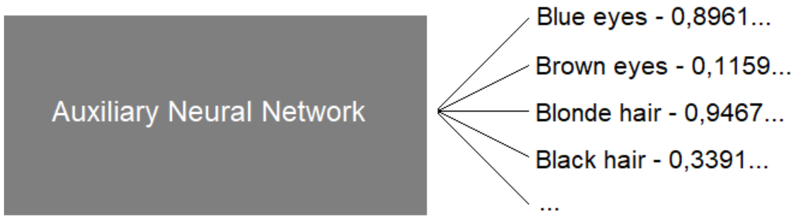


FIGURE 3.4: Output of the explanations NN. Given N different explanations, the model performs N binary classifications.

An important remark must made regarding the human explanations: these must be carefully designed and labeled. The explanations listed in the dataset must make total sense with respect to making an attempt at justifying the main predictions. Moreover, the explanations must not be duplicated nor redundant. The explanations must encode fundamental knowledge. Explanations that are too vague or too specific should be avoided. All of these considerations regarding the explanations must be taken into account to properly apply the proposed approach.

3.2 Projects setup

Both projects have been developed in the Google Colaboratory cloud environment. Google Colaboratory, shortened Google Colab or just Colab, is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud, specifically designed for Deep Learning development. This environment offers a free GPU accelerator with some usage limitations. The GPU corresponds to an NVIDIA Tesla T4 with 16 GB VRAM GDDR6. The environment also provides a total of 12 GB of RAM. This tool lets us efficiently implement and train the models designed in this thesis. The development of project 1 has been made with the Deep Learning framework called Keras (version 2.3.0) while using the TensorFlow (version 2.0) backend. On the other hand, due to implementation complications, project 2 has been developed with the Deep Learning framework PyTorch (version 1.5.0).

3.3 Synthetic dataset

The main task of the first project is a classification task of a total of 4 different image classes. This first project is designed to be a simple, easy-to-do, implementation of the proposal made in this thesis. The fact that a synthetic dataset is being used is coherent with this idea. Therefore, project 1 can be seen as a toy implementation of the explainability proposal. Nonetheless, this first implementation will let us detect possible weak and strong points. The synthetic dataset consist of a total of 4 different classes of images. For the sake of simplicity, each class is defined to be a simple 2D shape. These classes are: circle, triangle, square and rectangle. To clarify, the rectangle class is formed by strict rectangles, meaning that the two sides must have different lengths. The resolution of the generated images is 64 × 64. All the images are in grey scale (only 1 channel). To yield diversity within each class the scale, rotation and centre of each figure is randomly chosen within a predefined range. Figure 3.5 shows an example for each class.

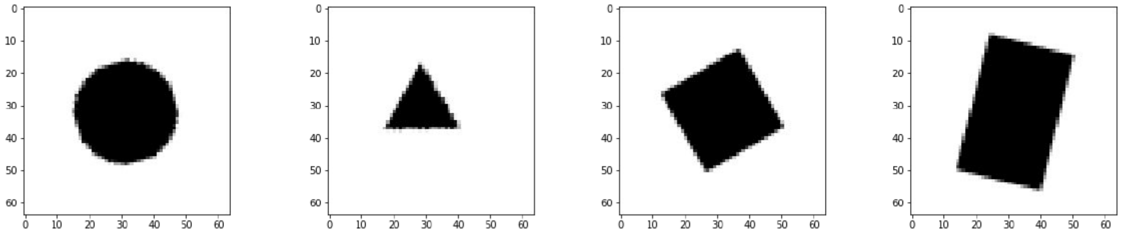
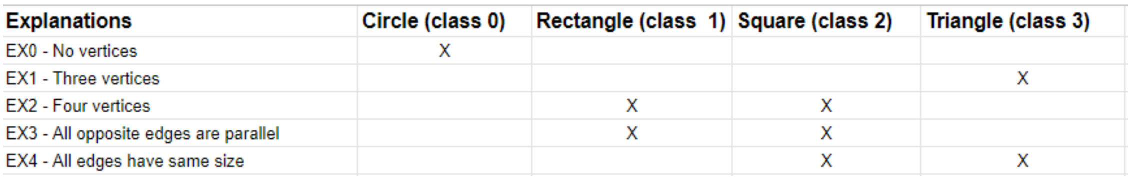


FIGURE 3.5: An example of each class generated in the synthetic dataset. From left to right circle, triangle, square and rectangle.

A total of 3,000 images per class are generated. The number of samples to generate has to be within a reasonable range since a too small number will make training hard while a too big number will generate duplicated images and thus making the train-validation split useless. After computing the space of all possible images for each class the number 3,000 turns out to be a reasonable number of images to generate for each class while avoiding duplicated samples.

After generating all the images for each class, explanations must be given to each sample in the dataset. In this toy project, the explanations designed are identical within each class. The set of possible explanations for classifying these images are the following: no vertices, three vertices, four vertices, all opposite edges are parallel and finally all edges have the same size. One can notice how there are explanations common to different classes as well as explanations unique to some classes. Figure 3.6 shows how these explanations are attributed to the classes.



Explanations	Circle (class 0)	Rectangle (class 1)	Square (class 2)	Triangle (class 3)
EX0 - No vertices	X			
EX1 - Three vertices				X
EX2 - Four vertices		X	X	
EX3 - All opposite edges are parallel		X	X	
EX4 - All edges have same size			X	X

FIGURE 3.6: Table showing how the designed explanations are distributed across each class in the synthetic dataset.

Now that the synthetic dataset has been generated with all the labels needed, we can move on to the implementation, training and evaluation of the proposed model.

3.4 Implementation and training

Since we are developing a Deep Learning model for Computer Vision tasks, and specifically for classification in both tasks, Convolutional Neural Networks (CNN) are the models of choice. CNNs have proven to be a breakthrough improvement in Computer Vision tasks through Deep Learning methodologies. The layers used in this first project are: Convolutional layers, Batch normalization (BN) layers, Max Pooling layers, Concatenation layers, Flattening layers and Fully Connected (FC) layers (also called Dense layers). Convolution, BN, Max Pooling, Flattening and FC layers are used in both networks, the main network and the auxiliary network. On the other hand, the Concatenation layer is only used by the explanations network as the way for combining its hidden activations with the ones coming from the main network. Due to the low difficulty of classifying 2D grey shapes, the CNNs designed are fairly small.

Both CNNs are composed of a total of 4 processing blocks. The first 3 blocks are formed by Convolutional + BN + Max Pooling layers while the last block (responsible for making the classifications) is formed by Flattening + FC layers. The kernel sizes for both CNNs have been fixed to 4. The number of kernels per Convolutional layer for the main network is fixed to 5 while for the auxiliary network is fixed to 3. The Max Pooling layers use all the default hyperparameters (reduce the input's size by 2). The FC layers have as many units as classes to predict, meaning that the FC of the main network is composed of 4 units while the FC from the auxiliary network is composed of 5 units. The FC layer of the main network applies the Softmax activation, as done in single-label classification. On the other hand, the FC of the auxiliary network applies the Sigmoid activation to each output in order to perform independent classifications for each output. The last design decision to make is how to connect the hidden activations of the main CNN to the layers of the auxiliary CNN. This is done through the use of the Concatenation layers which concatenate a list of tensors through the specified axis. In our case we will concatenate through the channels axis to keep all the information split in different channels. All of these design decisions, as well as the final scheme implemented for this first project, are shown in Figure 3.7.

The main CNN is composed of a total of almost $1,500$ parameters while the auxiliary CNN is formed by just $1,000$ parameters. It must be noted that the architecture defined for the auxiliary CNN follows the same architecture as the main one. This technique, which we will refer to as Mirroring or Mirror CNN, is done for avoiding shape issues when connecting the hidden activations of the main CNN to the explanations CNN. Once the CNNs have been implemented and the dataset has been generated, the pair of models can be trained. The synthetic dataset is split 50/50 into train and validation sets always keeping the same proportion for each class (stratified splitting), meaning that each split will be composed of $1,500 \times 4 = 6,000$ samples. Both optimisation processes follow the same characteristics:

- The optimiser is Nadam with the default learning rate of $1e{-3}$.
- A total of 250 epochs are processed
- At the end of each training epoch the validation set is evaluated and the metrics stored (loss and accuracy)

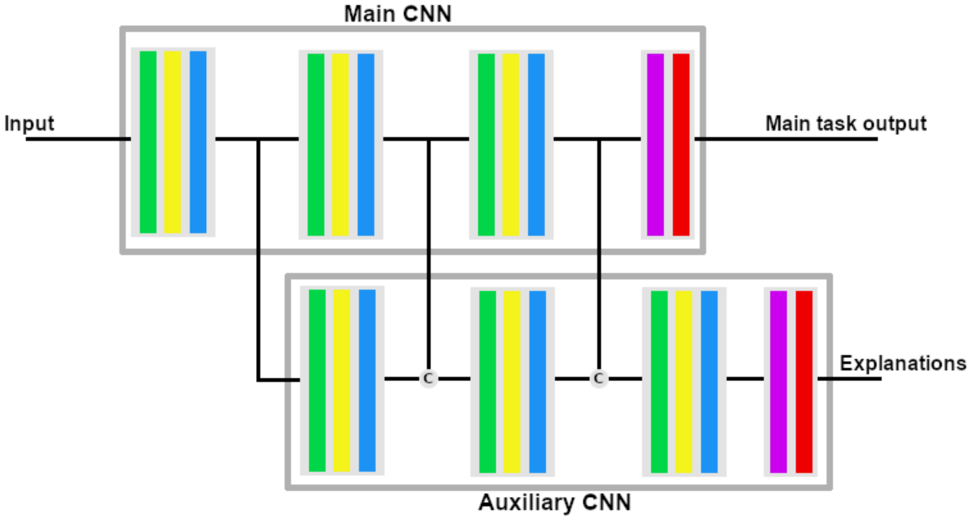


FIGURE 3.7: CNNs implemented for the first project. Each grey box represent a block of layers. Each coloured column represents a layer. The colours of each column encode the type of layer: green for Convolutional, yellow for BN, blue for Max Pooling, purple for Flattening and red for FC. The circle containing the capital letter "C" identifies the Concatenation operation.

- The loss of the main CNN is Categorical Cross Entropy while the loss for the auxiliary CNN is Binary Cross Entropy
- A constant learning rate decay is performed with a decay of $learningRate/epochs$
- Early stopping is used in the following way: the model with the best validation accuracy is always stored

Once the 250 epochs have been performed the best model found is loaded to perform evaluations on the predictions. As a final remark, the training process has been defined as a two-step process: during the first step the main CNN is trained alone; once the best parameters have been found then these are loaded and frozen (all parameters fixed) to finally train the auxiliary CNN, which corresponds to the second step.

3.5 Results and Evaluations

It goes without saying that the first thing we should look at is the accuracy achieved in both CNNs for each task. These values are presented in Table 3.1.

TABLE 3.1: All accuracies achieved by the best trained model proposed in project 1.

	Main CNN	**Explanations CNN**
Train accuracy	100%	99.02%
Validation accuracy	99.10%	98.43%

All accuracies greatly surpass the 98% value but this should not be surprising due to the extreme simplicity of the task being optimised. The main CNN is highly likely to easily learn how to detect the most important features to perform classification. At the same time the explanations CNN takes advantage of this and can also easily find correlations of the hidden features with respect to the explanations classes. Seeing these extremely high results we can subtract importance to the alternative of training both CNNs at the same time in hopes for achieving even higher performances. To have a better understanding of the 2D shape classification being done by the main CNN we can analyse its confusion matrix over the validation set. Table 3.2 corresponds to this confusion matrix.

TABLE 3.2: Confusion matrix of the main classification task over the validation set. Columns are ground truth classes and rows are predicted classes.

	Ground truth			
	Circle	Rectangle	Square	Triangle
Circle	1500	1	0	0
Rectangle	0	1466	1	19
Square	0	31	1499	0
Triangle	0	2	0	1481

The confusion matrix shows us that the most common mistake is confusing a rectangle for a square, which is not that surprising given that some rectangles are close to being squares. The following most common mistake is confusing triangles for rectangles which is indeed surprising and worth doing further analysis. The rest of the mistakes can be ignored due to their incredibly low numbers. We are in the perfect situation to take advantage of the predicted explanations for these mistakes, and more specifically to the triangles being confused for rectangles. We can come up with a formula to compute a *confidence* measure for each 2D shape class based on the explanations alone. Finally, with these measures we can make a cross-comparison between this *confidence* value and the probability predicted by the main CNN for each 2D shape class. The formula proposed to compute the *confidence* measure for a main class based only on predicted explanations is to first discretize each explanation probability using a threshold (default to 0.5), then multiply the discretized vector by the ground truth vector of explanations for the specific sample being analysed and finally normalise it. As an example, imagine the predicted probabilities vector of explanations is $(0.2, 0.1, 0.3, 0.5, 0.6)$, which discretized is the vector $(0, 0, 0, 1, 1)$, and we know the ground truth vector of explanations for each class. Now when computing the *confidence* measure for the class rectangle, which has explanations vector $(0, 0, 1, 1, 0)$, we do $(0, 0, 0, 1, 1) * (0, 0, 1, 1, 0) = (0, 0, 0, 1, 0)$, then we sum the vector giving a value of 1 and we finally normalise by the number of explanations this class has, meaning that the final measure is $1/2 = 0.5$. From an intuitive perspective, this measure computation is looking at how the corresponding explanations of a class match the predicted explanations independent from the rest of the predicted explanations. This measure ranges from 0 to 1 due to the fact that it is the proportion of how the thresholded explanations match the ground truth explanations for a specific class. In the end, we are able to compute a measure of *confidence* for each main class based only on the predicted explanations. Using this *confidences* vector along with the vector of probabilities predicted by the main CNN lets us do the desired cross-comparison. Obviously different methodologies, potentially better, for cross-comparing the predicted explanations with the predicted main classes can be formulated but due to the lack of time we will be using the one described here with the default threshold value.

When looking at all of these results, we can differentiate, at least, 3 different cases:

1. The largest predicted probability from the main classification task does not coincide, in the slightest, with the corresponding *confidence* value. Example presented in Figure 3.8.
2. The largest predicted probability from the main classification task is completely congruent with the corresponding computed *confidence* value but there is a tie with another class within the *confidences* vector. Example presented in Figure 3.9.
3. The largest predicted probability from the main classification task is, to some extent, congruent with the corresponding computed *confidence* value but due to the *confidences* vector another class is preferred. Example presented in Figure 3.10.

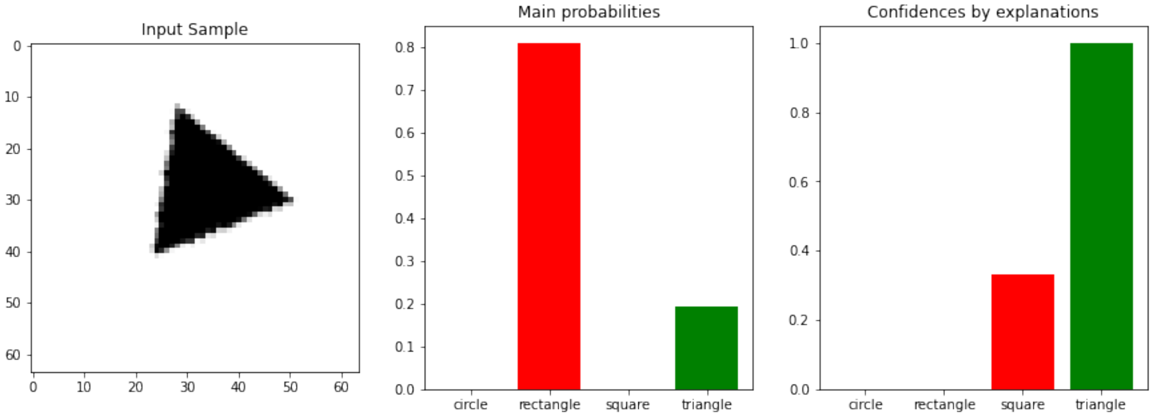


FIGURE 3.8: The predicted main class is not congruent at all with the corresponding *confidence* value based on the predicted explanations. The green bars identify the ground truth class.

We would expect to see an additional case corresponding to the situation in which the largest classification probability is congruent with the corresponding *confidence* value and at the same time the *confidences* vector shows that this mislabelled class is the preferred one. This case reflects the situation in which both the main CNN and the auxiliary CNN completely miss the correct target. To our surprise none of the mislabelled samples (61 in total) in the validation set show this situation. This points to the idea that the auxiliary CNN is useful for performing some sort of validation process over the predictions of the main CNN. We can consider that cases 1 and 3 present to the user and/or designer of the model a weak confidence with respect to the prediction of the main CNN, which is the important one in the end. This weak confidence can make the user/designer question whether that sample has been correctly or incorrectly classified, and probably calls for a manual supervision of the

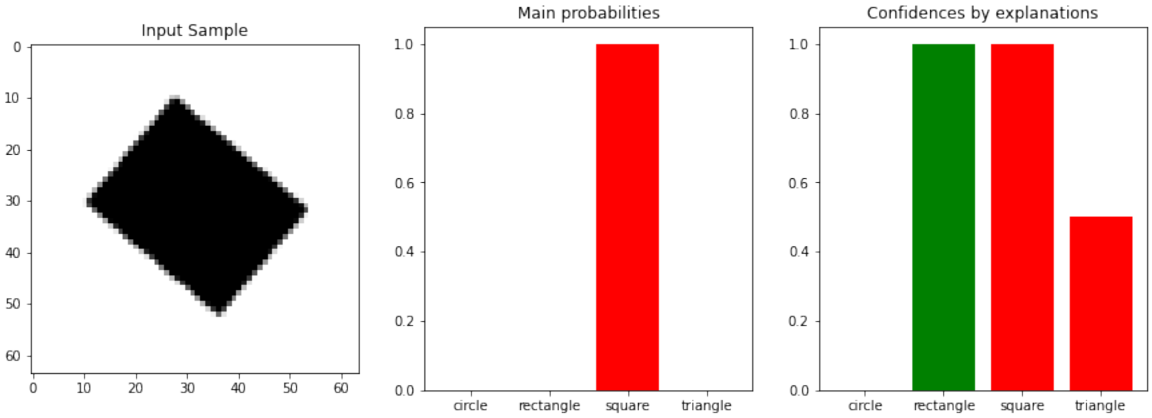


FIGURE 3.9: The predicted main class is completely congruent with the corresponding *confidence* value based on the predicted explanations but a tie appears within the *confidences* vector. The green bars identify the ground truth class.

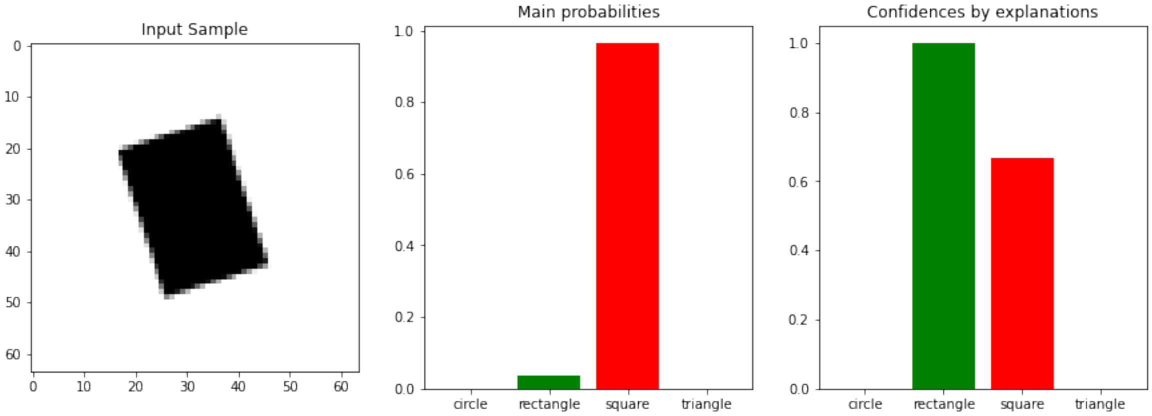


FIGURE 3.10: The predicted main class is somewhat congruent with the corresponding *confidence* value based on the predicted explanations but another class is preferred within the *confidences* vector. The green bars identify the ground truth class.

model. If we consider that cases 1 and 3 produce this effect over the user/designer then 57.38% of the misclassified samples from the validation dataset are questioned. This quantitative result is quite impressive considering the simple cross-comparison being made. Our hypothesis is that since the auxiliary CNN predicts explanations based only on the internal activations of the main CNN, and not the FC activations, it has a more direct and clear way to detect specific classes (explanations). Any mistake introduced by the main FC layer will not affect the explanations CNN. One final question arises when evaluating this cross-comparison technique: what about cross-comparing the correctly labelled samples from the validation set. It turns out that only 0.13% of all the correctly predicted classes on the main task are questioned, i.e., show case 1 or 3. Considering case 2 as a questionable prediction can be determined by the user/designer. Irrespective of this, we have decided to consider case 2 as a situation in which the main prediction is not questioned since the explanations fully coincide with the main predictions despite there being a tie. An example showing the cross-comparison over a correctly classified sample of the validation set can be seen in Figure 3.11.

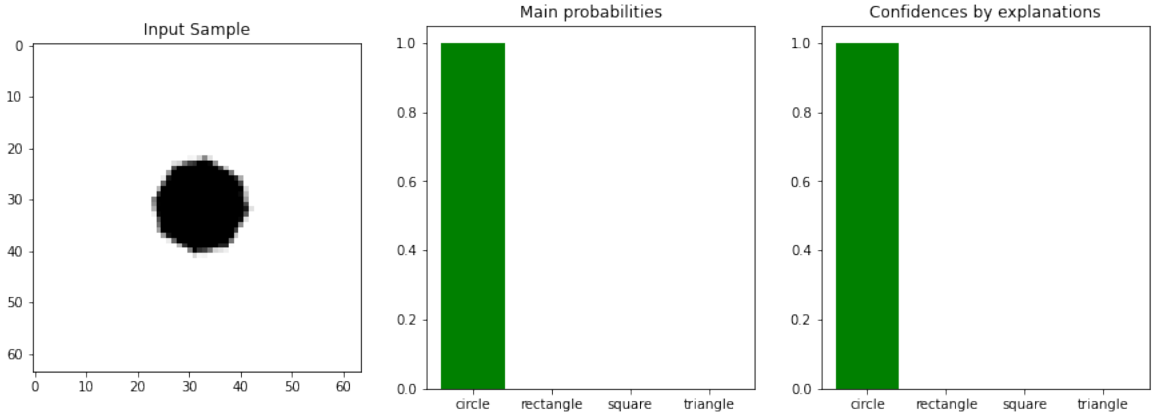


FIGURE 3.11: Cross-comparison of a correctly classified sample of the validation dataset. The green bars identify the ground truth class.

As an additional application of this cross-comparison method, out-of-distribution (OOD) samples are predicted as well as their explanations. The easiest way of generating OOD samples is by simply generating images with all of its values sampled randomly within the range of possible values (0 to 1). The cross-comparison validation has been performed over 100 OOD samples. Quantitative analysis shows that 20% of the OOD samples are completely congruent with the predicted explanations, meaning that 80% of them are questioned. To be able to question OOD samples with an accuracy around 80% seems like a, yet another, really good advantage of predicting the possible explanations following the proposed method in this thesis. Figure 3.12 shows an example of the cross-comparison over an OOD sample.

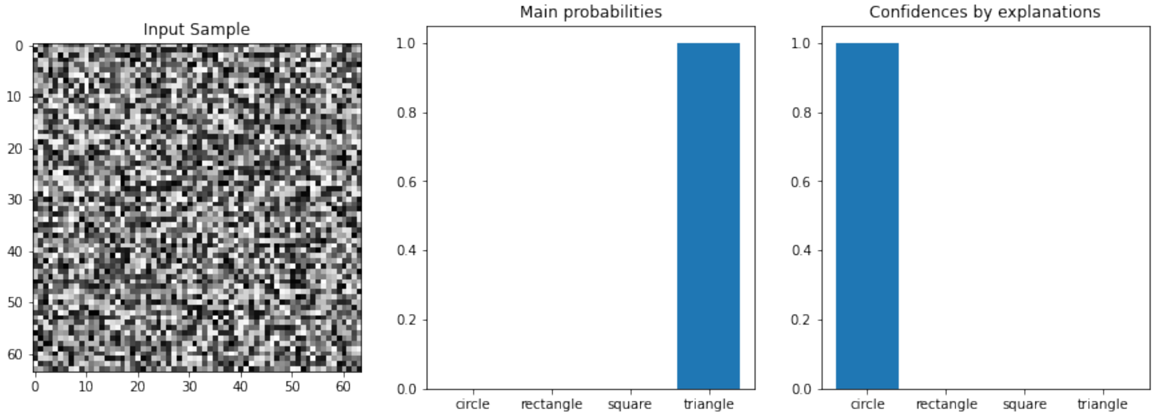


FIGURE 3.12: Cross-comparison of an out-of-distribution sample. The sample has been generated by uniformly sampling random values over the range of possible values.

3.6 Comments on project 1

Overall, the results and analysis performed over the usage of the explanations CNN are really promising. Not only do the predicted explanations provide human-interpretable explanations but they can also be used as a way of building confidence on the main CNN's predictions. Apart from all of these, the predicted explanations could potentially show the user/designer undesired biases in the model's predictions. Irrespective of all of this, there are a couple of points to be made about this first project:

- The difficulty of the main task is extremely easy for a CNN, and even more considering we are not even working with RGB images. Again, this first approach performs a toy task.
- We do not have strong evidence that the auxiliary network has actually captured the meaning of each of the explanations at a human level. In other words, we have no confidence at all that the auxiliary CNN has learned that it has to detect 3 vertices, for example, and has to identify this detection with the explanation class defined by us.
- Explanations are identical across samples from the same shape class. This can potentially guide the auxiliary CNN into learning that if the input is a circle then its corresponding explanations should be predicted.
- The dataset is synthetic meaning that it has several weak points, f.e., being disconnected from real-world environments.

These points should not be taken lightly as they strongly affect the performance of the proposed model. This is why project 2 has been defined, to be able to overcome the issues of project 1. Project 2, therefore, is the next step in this research process of the proposed approach to explainability in Deep Learning.

Chapter 4

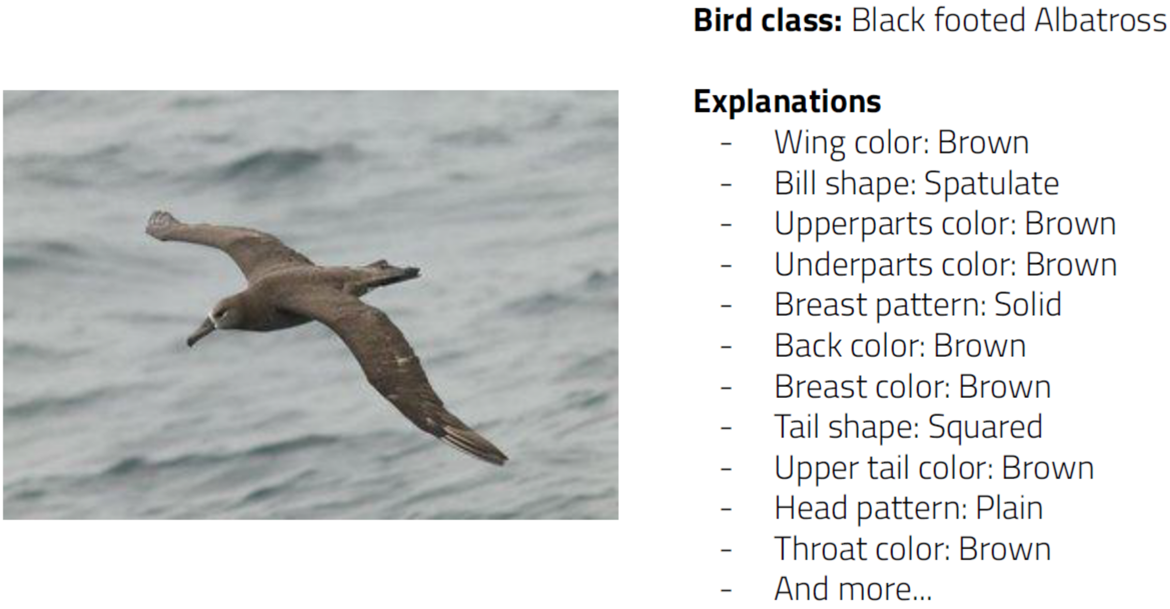
Project 2

4.1 Real-world dataset

One extremely important weak point from project 1 to overcome is the use of a synthetic dataset. In project 2 we aim to use a real-world dataset. After carefully looking for a fitting dataset (meaning that the explanations of each main task prediction must come along with the dataset) for a long time we have been able to find a good dataset with all the explanations needed. It turns out that finding a public dataset for Computer Vision with the needed explanations is hard to find. The dataset we found is composed of coloured images portraying birds classified by species. The provided explanations correspond to visual features of the bird visualised in the image. The dataset is composed of a total of 200 bird species and the list of visual features is formed by a total of 312 different features. Although there are a lot of explanations, all of these can be grouped, if needed, by the type of feature: the explanations are flattened values of specific traits of a bird, f.e., the trait *bill shape* comprises a total of 8 different values but these are flattened into 8 different explanations (curved bill shape, dagger bill shape, etc). All the labels, the main class as well as the visual features present in an image, have been manually labeled by humans. The authors of the dataset refer to these visual features (which we treat as explanations) as binary attributes. The public dataset has been obtained from (Wah et al., 2011) which corresponds to the original source. The dataset, named *Caltech-UCSD Birds-200-2011*, offers 60 images per bird class (for almost all of them). A 50/50 train-validation split over each bird class is performed. This means that we only have 30 samples per class for training the main CNN. This becomes a challenging task if we aim to achieve high accuracy over the main classification task. Due to the low number of samples available for training this could be seen as a few-shot learning task (or low-shot learning) which is well-known to be challenging since the goal is to capture the most generalisation potential from an extremely small dataset. To overcome this difficulty we rely on a heavy use of Computer Vision data augmentation. An example image from this dataset along with its visible visual features are shown in Figure 4.1.

4.2 Implementation and training

As previously stated, all the implementations within this second project have been developed using the Deep Learning framework PyTorch (version 1.5.0). Moreover, this project 2 implementation has been based on the work of (Saeed, 2018) in which they achieve a reasonably high accuracy on the selected dataset thanks to the application of heavy data augmentation as well as a state-of-the-art (SOTA) CNN architecture corresponding to ResNet18. Starting from their work, we will extend it with the addition of the auxiliary CNN. Even more, we will test whether ResNet50



Bird class: Black footed Albatross

Explanations

- Wing color: Brown
- Bill shape: Spatulate
- Upperparts color: Brown
- Underparts color: Brown
- Breast pattern: Solid
- Back color: Brown
- Breast color: Brown
- Tail shape: Squared
- Upper tail color: Brown
- Head pattern: Plain
- Throat color: Brown
- And more...

FIGURE 4.1: Birds dataset example with its corresponding species label as well as its explanations (visual features).

surpasses the results presented in (Saeed, 2018) in which they achieve a validation accuracy of 77.34% in the classification task. It goes without saying that we will be doing the same split so that a comparison of performances can be made. We will always be following the main decisions made by (Saeed, 2018) which correspond to the following:

- An already pretrained ResNet is loaded from the official PyTorch repository. The warm up performed corresponds to the well-known ImageNet classification task. This helps ResNet during the optimisation process by achieving slightly better accuracies, drastically reducing the training needed (only fine-tuning is needed) and usually avoiding overfitting.
- To fine-tune the pretrained ResNet a change has to be made to its architecture: the last FC layer has to be replaced with a new one (with the identity activation function) having as many units as the number of classes of the new classification task (200 in our case).
- The loss to use is the standard one for multi-class classification, Cross Entropy from PyTorch which already implements the Softmax function altogether with the Negative Log-Likelihood Loss in an efficient way. The optimiser chosen for fine-tuning the warmed up ResNet is Stochastic Gradient Descent (SGD) with a learning rate of $1e-4$ and a momentum of 0.9.
- An early stopping technique is used: the model saved always corresponds to the one that yielded the highest accuracy on the validation set. The validation set is evaluated at the end of each training epoch. A total of 200 epochs are processed.

An important issue arises when inspecting the images from the dataset: these have different sizes and some of them have an aspect ratio different than 1 (non-squared images), and thus the preprocessing pipeline must handle this. Moreover, this pipeline also includes the data augmentation techniques. The preprocessing applied to the loaded images, a key factor in achieving high accuracy for this extremely small dataset, is composed of the following steps:

1. First of all, the whole image is scaled by a factor such that the largest axis of the image has size 256.
2. A rotation of up to 45 degrees is randomly applied to the resized image with a 50% probability.
3. A crop of shape $224 \times 224 \times 3$ (expected input shape for the ResNet CNN) is randomly selected from the image.
4. A horizontal flip is randomly applied with a 50% probability. Vertical flips are not used since these make no sense in the context of bird images.
5. Finally, the values of the resulting image are normalised by subtracting the mean of the ImageNet dataset (RGB values $[0.458, 0.456, 0.406]$) and dividing by the standard deviation of ImageNet (values $[0.229, 0.224, 0.225]$).

This whole preprocessing pipeline ensures that the inputted image to the ResNet CNN has the expected shape as well as being normalised in the same way it has been pretrained. Additionally, the random rotation, the random crop and the random horizontal flip heavily augment the dataset (specially the random cropping technique). Some examples of this processing pipeline are shown in 4.2. This pipeline is modified to process the validation set: no data augmentation is applied and the random crop is replaced by a centre crop since it is the most likely crop to contain a centred bird. Validation examples are presented in Figure 4.3.

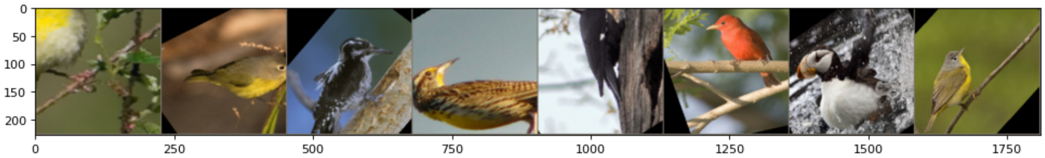


FIGURE 4.2: Training samples processed by the preprocessing pipeline. Modifications over the colour values have been reverted for visualisation purposes.

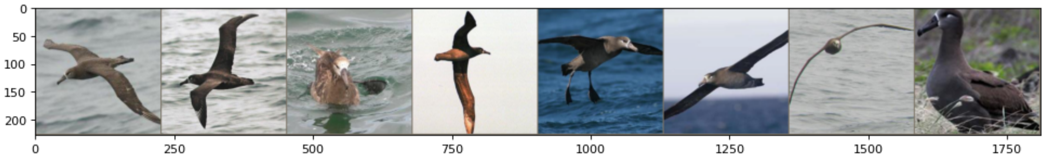


FIGURE 4.3: Validation samples processed by the preprocessing pipeline. Modifications over the colour values have been reverted for visualisation purposes.

Processing the whole training set (with size $= 30 \times 200 = 6,000$) corresponds to one training epoch. As previously stated, 200 epochs are processed and the saved model is always the one that achieves the maximum accuracy on the validation set.

TABLE 4.1: Accuracies achieved, on the birds classification task, by the best trained model for ResNet18 and ResNet50 architectures.

	ResNet18	**ResNet50**
Train accuracy	78.43%	90.01%
Validation accuracy	77.05%	81.03%

Table 4.1 shows the accuracies achieved, by the best model, for both ResNet18 and ResNet50 over the training and validation sets.

Unsurprisingly, ResNet50 (with $\sim$ 23 million parameters) outperforms ResNet18 (with $\sim$ 11 million parameters) by 5 percentage points on the validation set. From now on, ResNet50 is the CNN architecture to use. This means that for the final proposed model, the main CNN corresponds to a ResNet50 and the auxiliary CNN corresponds to a mirror of the ResNet50 architecture. The connections established from main CNN to auxiliary CNN have been chosen based on how ResNet architectures are formed. These are based on residual blocks as shown in the paper in which ResNet was first presented (He et al., 2015). The hypothesis is that at the end of each residual block deeper features, likely to be more relevant for the explanations, are computed. Therefore, the final implemented architecture for this second project is presented in Figure 4.4.

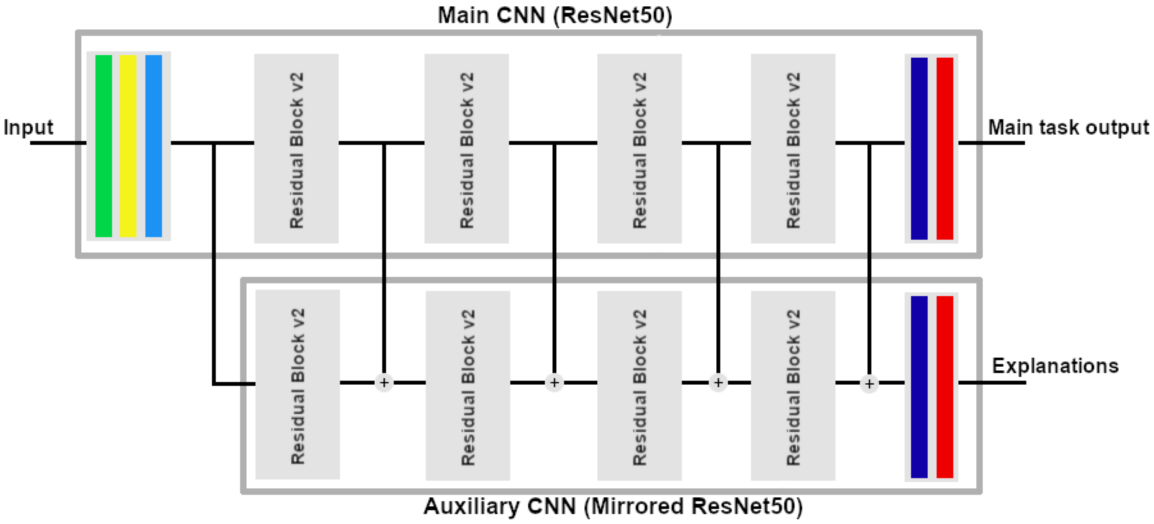


FIGURE 4.4: CNNs implemented for the second project. Each coloured column represents a layer. Colours assigned to each layer encode the same information as those of Figure 3.7. The additional colour present here, dark blue, corresponds to an Average pooling layer concatenated with a Flattening layer. The circle containing the character "+" identifies the Addition operation.

Note that ResNet version 2 has been selected which uses a bottleneck approach within each residual block. Note also how ResNet first applies an Average pooling layer before flattening the tensor. Average pooling is the operation of performing an average computation over each channel with a specific kernel size. ResNet decides to use an Average pooling layer such that each channel comes out with only 1 value (height and width equal to 1). This significantly reduces the dimensions of the tensor being outputted by the last residual block, meaning that the FC layer will need way fewer parameters. Also note how the established connections do not follow the pattern followed in project 1. This is due to the fact that the operation used to join the tensors, from the main CNN to the auxiliary CNN, has been changed to the addition operation. The addition operation between tensors can only be used when the tensors being added have exactly have the same shape. There are three reasons to use addition instead of concatenation:

1. By adding tensors the resulting shape is the same, and so the residual blocks can process the tensors in the same way. In other words, addition lets us be consistent when using ResNet.
2. When concatenating over the channels axis, the dimension of the resulting tensor increases. Addition keeps the same number of dimensions and thus the CNN does not need to make an increase in trainable parameters.
3. Finally, from an information point of view, the addition operation can be seen as joining together the information encoded from the main CNN into the information encoded on the auxiliary CNN at the same layer level. This can potentially help by reducing the number of parameters needed by the auxiliary CNN.

Again, once the main CNN has been trained and frozen, we can optimise the auxiliary CNN. One could argue that the mirrored ResNet50 (therefore the auxiliary CNN) could be warmed up by loading the parameters optimised during the training of the main CNN. This approach is also tested in order to see whether transferring the learned parameters of the main CNN for finetunning makes sense or not. All results achieved by the best model found, for each CNN (including the warm up approach), are presented in Table 4.2.

TABLE 4.2: All accuracies achieved, over the birds dataset, by the best model trained for each task, classification and providing explanations. The accuracies achieved by the warmed up auxiliary CNN are also shown.

	Main CNN	**Auxiliary CNN**	**Auxiliary CNN (warm start)**
Train accuracy	90.01%	85.53%	77.94%
Validation accuracy	81.03%	85.19%	78.38%

We can see how warming up the auxiliary CNN did not outperform random initialisation. This could point to the idea that the learned parameters on the main CNN are not fitted for detecting the desired explanations. This, at the same time, makes sense since the parameters of the main CNN have been optimised to yield high values on detecting specific features while the parameters of the auxiliary CNN should be optimised to translate the hidden activations into classes. There is a significant conceptual gap between translating hidden activations and detecting specific features of the input. Following the same line of thought, ImageNet initialization for the auxiliary CNN would not be useful either. The explanations provided by the non warmed up auxiliary CNN are the ones used for predictions and final evaluations.

4.3 Results and Evaluations

A similar evaluation process to the one performed in project 1 is conducted. In order to handle the large amount of classes, for both birds and explanations, only the top *N* classes (where *N* is a fixed value) will be usually shown for evaluating the predictions. First of all, let us look at the predictions computed by the trained model for the sample presented in Figure 4.1. The main predictions of this validation sample are presented in Figure 4.5.

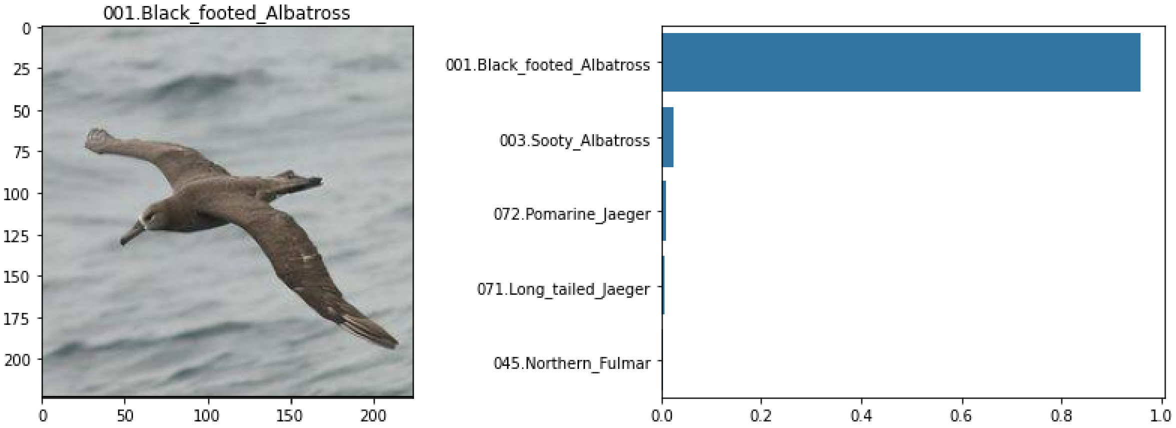


FIGURE 4.5: Main predictions of the validation sample shown in Figure 4.1. The ground truth class is located above the left image. The right plot corresponds to the top 5 classes by probability.

For this sample the main CNN has correctly identified the class of the bird with an extremely high probability. Interestingly enough, the second most likely class is another Albatross-like bird species just like the ground truth class. Now, we can take a look at the explanations predicted by the auxiliary CNN in Figure 4.6. To be able to predict these explanations is, by itself, an already impressive achievement. These raw explanation probabilities show an important issue which is repeated features with different values. In this specific case features *has shape* and *has nape color* appear two times within the top 18 explanations. Additionally, recall that there are a total of 312 explanations. Therefore, a post-processing step needs to be applied to these predictions. First of all, in order to handle the large amount of explanations being predicted a more strict threshold can be fixed, f.e., to a value of 0.8 (compared to the default value of 0.5). On the other hand, to avoid repeats in the top *N* explanations ranking, a cleaning process is performed to simply discard any predicted explanation concerning a feature that has already appeared with a higher probability. These two post-processing decisions are applied in order to try to fix the presented issues.

Once the post-processing has been defined, we can come up, again, with a cross-comparison methodology in order to try to verify the main CNN predictions based on the predicted explanations. The dataset has an advantage, which is that it provides a matrix with shape 200×312 (as many rows as bird classes and as many columns as visual features). This matrix stores, at each location, the probability that the image of a specific bird class (the row number) shows a visual feature (the column number). We can exploit this matrix by multiplying it by the post-processed vector of predicted explanations, which has a size of 312 (repeated features and probabilities lower than the fixed threshold are set to 0), in order to get a vector

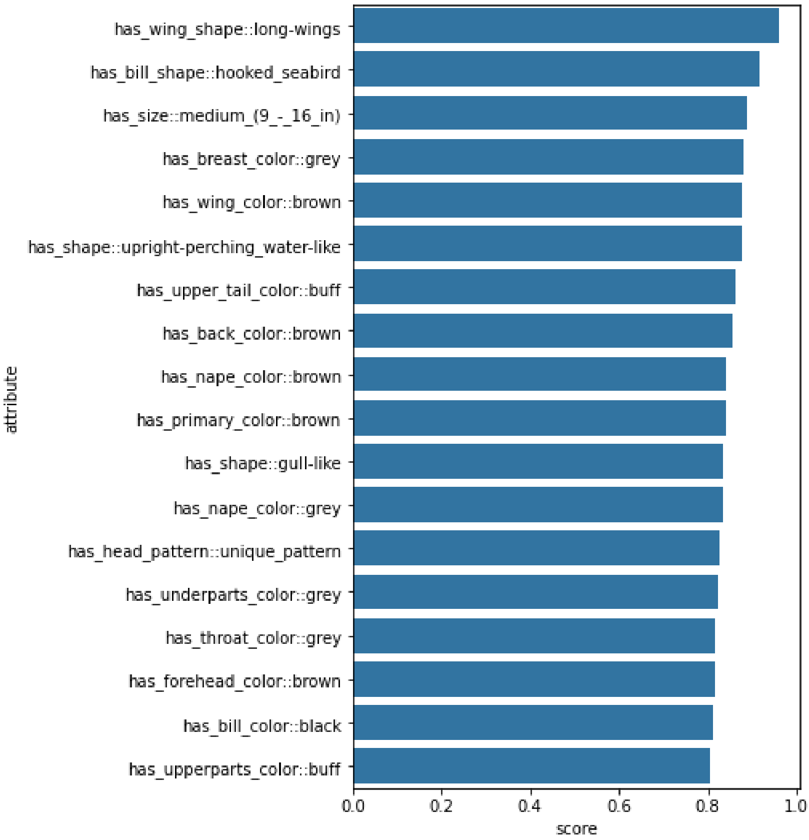


FIGURE 4.6: Auxiliary predictions of the validation sample shown in Figure 4.1. The top 18 explanations are shown in a horizontal bar plot.

of 200 values, each one of these corresponding to a *confidence* value measuring how the post-processed explanations fit each bird class. Finally, the top 5 (for example) bird classes can be taken in order to look for consistency with the main CNN prediction. This is applied to the predicted explanations for the same validation sample and the resulting classes can be seen in Figure 4.7.

Notice the strong impact of the post-processing steps applied to the predicted explanations. When no post-processing is applied the ground truth class is at rank 5. Once repeated features are cleaned the ground truth class jumps to rank 3. Finally when applying both cleaning and a stricter threshold the ground truth class reaches the rank 2 position. Based on all of these findings, the post-processing is always applied with a threshold of 0.8. This can be applied in the same way as it has been applied in project 1, with the only difference that we now have to fix the number N for the top N classes based only on explanations. For example, Figure 4.8 shows a misclassified bird being questioned by the top 10 predicted classes based on the predicted explanations.

A quantitative analysis is performed in the same way as it has been done in project 1, i.e., computing the proportion of questioned validation samples for the correctly and incorrectly classified ones. In this case the N parameter is fixed to 10 for identifying when the predicted explanations are not consistent with the predicted bird class. From the correctly classified samples in the validations set a total of 85% of them present consistent explanations. Meanwhile, from the misclassified validation samples the proportion of not consistent explanations, and therefore

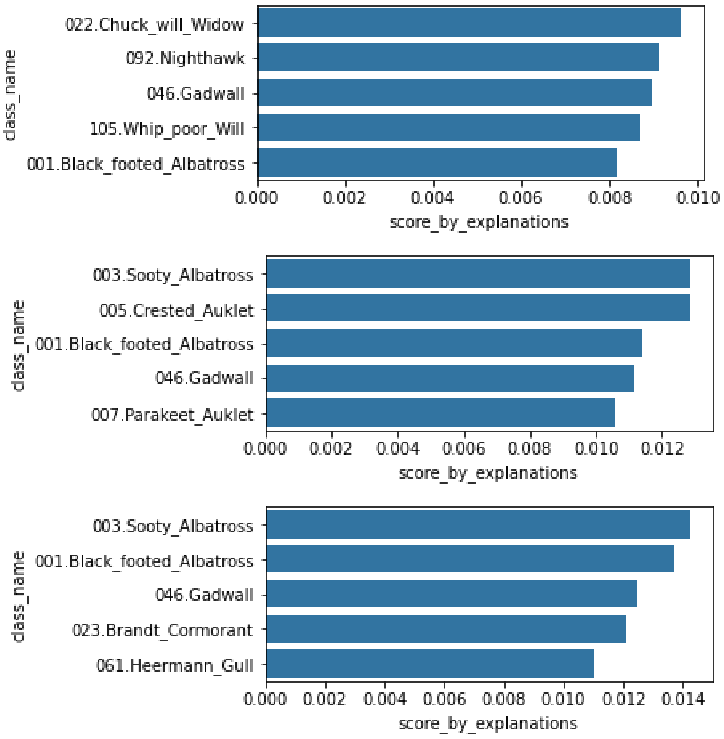


FIGURE 4.7: Top 5 classes by explanations for the validation sample shown in Figure 4.1. From top to bottom the top 5 classes when: no post-processing is applied, cleaning repeated features, cleaning repeated features and applying stricter threshold.

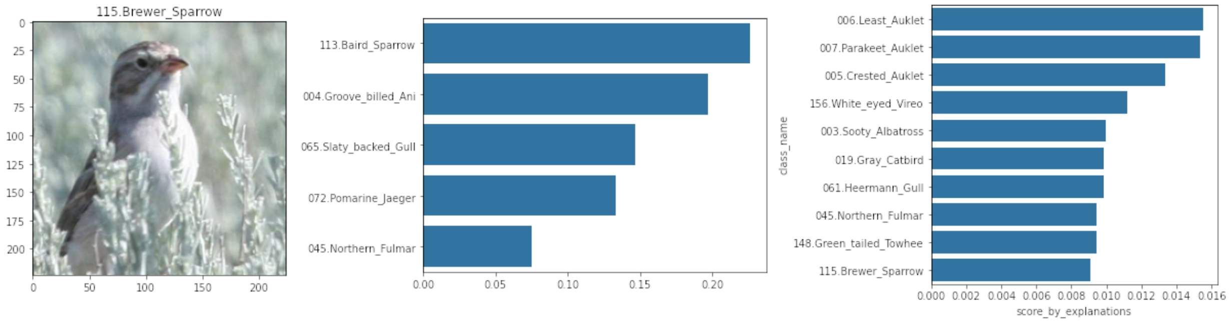


FIGURE 4.8: From left to right: input image depicting a Brewer Sparrow bird, the predicted probabilities for the bird class and finally the top 10 predicted bird classes based on explanations. Notice how the most likely predicted class is not present in the top 10 classes by explanations.

questioned misclassifications, is located around 35%. In absolute values almost 700 correctly classified samples are questioned whereas around 415 incorrectly classified samples are questioned. Whether the trade-off between questioning correctly and incorrectly classified samples is worth doing depends on the application and the resources available as well as the performance of both models. When cross-comparing $1,000$ random input images all of them have been questioned. This is not that surprising given the fact that it is highly unlikely that the predicted class appears in the top 10 classes out of the 200. Even though the high number of classes helps with this, the cross-comparison process perfectly questioned all OOD samples.

4.4 Comments on project 2

As mentioned before, project 2 is the next step towards a more strict research of the proposed explainability approach for Deep Learning. Several weak points from project 1 have been fixed. First of all, an extremely important point is that we have high variability of explanations between and within classes. This gives us confidence that the auxiliary CNN will actually capture the information encoded on each explanation through the optimisation process, all thanks to the high variability of explanations. By learning when an image has or does not have a specific explanation (visual feature), regardless of the bird class, is what makes the auxiliary CNN be forced to learn what we humans want it to learn. Additionally, the selected dataset is a real dataset and presents lots of visual similarities as well as a low number of samples, making the whole task a challenge. These two points, that have been fixed in project 2, were the most worrisome. To our surprise, some samples with wrong labels have been detected in the birds dataset but the proportion of these errors is low enough to still use it as it is.

Regarding the results and evaluations found, as well as considering the difficulty of the dataset and the extremely low amount of samples per class, they can be seen as fairly good. In this second project all out-of-distributions samples have been correctly questioned. On the other hand, the number of correctly classified samples being questioned versus the number of incorrectly classified samples being questioned does not seem that good, however the importance of these consistencies heavily depends on the application in question, the available resources and the model performance.

All in all, being able to predict explanations at a human level, as done with this real-world dataset of birds, turns out to be extremely useful. Not only does it provide confidence to the designer/user of a Deep Learning system but it can also be used for additional processes like the one being tested here: to cross-compare explanations with the main prediction in order to detect possible inconsistencies.

Chapter 5

Conclusions

5.1 Final comments

Both projects followed the same implementation and testing steps. These steps are the ones that have been defined when proposing the goals of the thesis. The completion of the proposed steps means that we have achieved what we aimed for in the thesis, i.e., to implement and test an approach to improve explainability in Deep Learning. Moreover, the results seen in both projects are quite impressive. Even though checking the class-explanations consistency on the second project has been a more difficult task and has not performed as well as it has on project 1, the predicted explanations for the birds dataset provide an invaluable source of information regarding how the main CNN processes the input image through the use of human-defined explanations. As previously mentioned, being able to predict human-interpretable explanations can be used for additional tasks like the one tested here (cross-comparing the main predictions). Out-of-distribution detection is another obvious application, also tested in the thesis. Detection of unwanted biases can also be implemented. Nevertheless, there are a couple of weak points regarding the proposed explainability approach in Deep Learning which are the following:

1. To be able to have a datasets with human explanations annotated for the target of each sample is a quite hard thing to achieve. These kinds of dataset are usually extremely expensive to build in terms of time and resources.

2. Using human-defined explanations for conveying possible explanations of the main predictions can be not ideal. The problem with the source of these explanations is the source itself. Humans can be mistaken when interpreting and justifying a task such as classification. Moreover, using human-defined explanations heavily restricts the possible explanations search space. In other words, using human explanations means that the predicted explanations are enclosed within the knowledge that is already available by humans and this in turn means that no new insights will emerge.

3. The proposed explainability approach could be categorised as an Intrinsic model approach, as explained in 1.3 Background. This kind of approach has its own strong and weak points as expected. Due to the lack of time, the pair of models have always been trained on a two step process, first the main CNN and then the auxiliary CNN, although training both at the same time can potentially help each others learning processes. Also due to the lack of time, hyperparameter searching has not been feasible.

In conclusion, the master's thesis has been a success in terms of exploration, implementation and testing of the proposed approach. Extremely interesting insights have been found as well as possible real-world applications.

Appendix A

Master's thesis source code

A.1 GitHub repository

Code implementation and datasets of the thesis published in:
`https://github.com/maxby12/TFM-UB`

Bibliography

He, Kaiming et al. (Dec. 2015). “Deep Residual Learning for Image Recognition”. In: *arXiv e-prints*, arXiv:1512.03385, arXiv:1512.03385. arXiv: `1512.03385 [cs.CV]`.

Parafita, Álvaro and Jordi Vitrià (Sept. 2019). “Explaining Visual Models by Causal Attribution”. In: *arXiv e-prints*, arXiv:1909.08891, arXiv:1909.08891. arXiv: `1909.08891 [stat.ML]`.

Reuters and Jeffrey Dastin (2018). “Amazon scraps secret AI recruiting tool that showed bias against women”. In: [Online; accessed 12-June-2020]. URL: `https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G`.

Saeed, Muhammad Mujtaba (2018). *Caltech-Birds-Classification*. `https://github.com/Muhammad-MujtabaSaeed/Caltech-Birds-Classification`.

Tjoa, Erico and Cuntai Guan (July 2019). “A Survey on Explainable Artificial Intelligence (XAI): Towards Medical XAI”. In: *arXiv e-prints*, arXiv:1907.07374, arXiv:1907.07374. arXiv: `1907.07374 [cs.LG]`.

Wah, C. et al. (2011). *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology.

Wikipedia contributors (2020). *Explainable artificial intelligence — Wikipedia, The Free Encyclopedia*. [Online; accessed 5-June-2020]. URL: `https://en.wikipedia.org/w/index.php?title=Explainable_artificial_intelligence&oldid=959167960`.

Xie, Ning et al. (Apr. 2020). “Explainable Deep Learning: A Field Guide for the Uninitiated”. In: *arXiv e-prints*, arXiv:2004.14545, arXiv:2004.14545. arXiv: `2004.14545 [cs.LG]`.