UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

FEATURE EXPLAINABILITY IN FINE GRAINED IMAGE CLASSIFICATION

Diploma Thesis

Athanasios Tsafonis

Supervisor: Emmanouil Vavalis

October 2023

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΠΕΞΗΓΗΣΙΜΟΤΗΤΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΣΤΗΝ ΛΕΠΤΟΜΕΡΗ ΤΑΞΙΝΟΜΗΣΗ ΕΙΚΟΝΩΝ

Διπλωματική Εργασία

Αθανάσιος Τσαφώνης

Επιβλέπων/πουσα: Εμμανουήλ Βάβαλης

Οκτώβριος 2023

Approved by the Examination Committee:

Supervisor **Emmanouil Vavalis**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Gerasimos Potamianos**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Dimitrios Samaras**
Professor, Department of Computer Science, Stony Brook University

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Athanasios Tsafonis

Diploma Thesis

FEATURE EXPLAINABILITY IN FINE GRAINED IMAGE CLASSIFICATION

Athanasios Tsafonis

Abstract

Explainability plays a pivotal role in establishing confidence in machine learning models, guaranteeing that their actions are comprehensible and can be reasonably explained to human stakeholders. In this research, a novel approach is introduced to elucidate the decision-making process of a fine-grained image classification model termed MMAL-Net. This model is particularly specialized in recognizing subtle differences within bird species and is evaluated using the CUB-200-2011 dataset. The central objective is to provide explanations regarding the pivotal attributes that contribute significantly to MMAL-Net's classification decisions. The process begins by utilizing the pre-trained MMAL-NET model to extract feature maps from each image. Subsequently, these feature maps serve as inputs for an attribute classification model named ATT-Net. The output of ATT-Net is a vector indicating the presence of various bird attributes within each feature map. The proposed approach involves training and evaluating the ATT-Net model. To assess attribute importance, the Grad-CAM algorithm is employed to compute weighted values associated with each attribute's relevance to its corresponding feature map. The same weighted values are also extracted from the MMAL-NET network following the application of Grad-CAM. The key step involves a quantitative analysis that measures the proximity between the attribute values obtained from ATT-Net and the MMAL-Net values for each image. For the evaluation part, a ground truth attribute list is created for every bird class, which is provided by experts in ornithology. Then, the MAP score metric is applied between the method's results and the ground truth attributes with a total percentage of 9%. The conclusion provided by this work is that the information a feature map gives might not be optimal for human-readable attribute prediction.

Github Implementation Code

Keywords:

Explainability, Grad-CAM, MMAL-Net, ATT-Net, MAP score

Διπλωματική Εργασία

ΕΠΕΞΗΓΗΣΙΜΟΤΗΤΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΣΤΗΝ ΛΕΠΤΟΜΕΡΗ ΤΑΞΙΝΟΜΗΣΗ ΕΙΚΟΝΩΝ

Αθανάσιος Τσαφώνης

Περίληψη

Η ερμηνευσιμότητα παίζει κομβικό ρόλο στην εδραίωση της εμπιστοσύνης στα μοντέλα μηχανικής μάθησης, εξασφαλίζοντας ότι οι ενέργειές τους είναι κατανοητές και μπορούν να εξηγηθούν λογικά σε ανθρώπους. Σε αυτήν την έρευνα, παρουσιάζεται μια νέα προσέγγιση για τον αποσαφηνισμό της διαδικασίας λήψης αποφάσεων ενός προηγμένου μοντέλου ταξινόμησης εικόνων με την ονομασία MMAL-Net. Αυτό το μοντέλο εξειδικεύεται στην αναγνώριση λεπτομερών διαφορών μεταξύ διαφορετικών ειδών πτηνών και αξιολογείται χρησιμοποιώντας το σύνολο δεδομένων CUB-200-2011. Η διαδικασία ξεκινά με τη χρήση του μοντέλου MMAL-Net για την εξαγωγή χαρτών χαρακτηριστικών από κάθε εικόνα. Στη συνέχεια, αυτοί οι χάρτες χρησιμεύουν ως είσοδοι σε ένα μοντέλο ταξινόμησης χαρακτηριστικών με την ονομασία ATT-Net. Το αποτέλεσμα του ATT-Net είναι ένα διάνυσμα που υποδεικνύει την παρουσία διαφόρων χαρακτηριστικών πτηνών εντός κάθε χάρτη χαρακτηριστικών. Για να αξιολογηθεί η σημασία των χαρακτηριστικών, χρησιμοποιείται ο αλγόριθμος Grad-CAM για τον υπολογισμό τιμών που σχετίζονται με τη σχετικότητα κάθε χαρακτηριστικού προς τον αντίστοιχο χάρτη χαρακτηριστικών. Οι ίδιες τιμές εξάγονται επίσης από το δίκτυο MMAL-Net μετά την εφαρμογή του Grad-CAM. Το κύριο βήμα της μεθόδου απαιτεί μια ποσοτική ανάλυση που μετρά την κοντινότερη απόσταση μεταξύ των τιμών των χαρακτηριστικών που έχουν ληφθεί από το ATT-Net και των τιμών του MMAL-Net για κάθε εικόνα, με σκοπό την ανάδειξη του σημαντικότερου χαρακτηριστικού. Για την αξιολόγηση των αποτελεσμάτων δημιουργείται μια λίστα χαρακτηριστικών για κάθε κατηγορία πτηνού, η οποία παρέχεται από ειδικούς στην ορνιθολογία. Στη συνέχεια, χρησιμοποείται το MAP score μεταξύ των αποτελεσμάτων της μεθόδου και της κάθε λίστας χαρακτηριστικών με συνολικό ποσοστό 9%. Το συμπέρασμα που προκύπτει είναι ότι οι πληροφορίες που παρέχοτνται από τους χάρτες χαρακτηριστικών ενδέχεται να μην είναι ιδανικές για την ανθρωποκεντρική πρόβλεψη των χαρακτηριστικών.

Table of contents

Chapter 1

Introduction

The act of explaining to a human why and how a machine learning model reached a decision is known as explainability. With complex models, you can't sufficiently comprehend how and why the inner mechanics affect the prediction. An artificial intelligence (AI) system like that, which hides its inputs and operations from the user or developer is referred to as a "Black box" [1, 2]. The parameters that one model may consider and the bias the model may contain must be made clear to the corresponding professionals who will use the model results. Therefore, once the model is implemented in the actual world the developers should be ready to explain it. The trust in a machine learning model is increased by the ability to comprehend a model. This becomes much more crucial in circumstances involving pivotal life challenges. Furthermore, explainability becomes crucial during the development part when a model is being debugged. It is also helpful in determining whether the models are appropriate for use in real-life situations [3]. Especially, in fine-grained categorization tasks, the ability to explain the model decisions for such detailed occasions is becoming more and more significant. As a branch of object recognition, fine-grained classification seeks to discern lower-level categories inside top-level categories. Differentiating between difficult-to-distinguish object classes is the main goal of every classification task. Examples include identifying different bird species, as well as different flower species, car brands, cancer categories, and many more. In contrast to general object recognition, fine-grained classification sometimes necessitates efforts from multiple aspects [4, 5, 6]. So, in every fine-grained classification model, the developer needs to know the reasons behind each image or text categorization decision. Since 2016, many computer vision research teams have made explainable AI a top priority for their research. However, after years of study and implementation, the Explain-

able AI sector has usually had trouble putting the principles of controlled, dependable, and intelligible AI into practice.

In this work, I propose a method for explaining the classifier's decision behind a state-of-the-art (SOTA) fine-grained image classification model, which is called MMAL-NET [7]. This network uses the CUB-200-2011 dataset, which includes over 11,000 bird images, 200 bird classes, and 312 bird attributes [8]. My goal is to explain which of all the possible attributes of a bird species were the most important for the MMAL-Net model to do the classification. Firstly, I use this already pre-trained and tested model to get the feature maps of every image out. I use these feature maps as inputs to my attribute classification model, which is called ATT-Net. The output of ATT-Net is a vector that contains which of the bird attributes are present on every feature map. After training and testing my model, I use the Grad-CAM algorithm [9, 10, 11] to calculate the weighted values of the corresponding attribute to each feature map. I also take the same weighted values from the MMAL-NET network after the Grad-CAM application, which correspond to the bird classification. I calculate which of the values I got from ATT-Net for each attribute of each image are mathematically closer to the MMAL-Net values for each image to get my results. These results show me which are the most important attributes that have been chosen to classify the bird species from the MMAL-Net network.

I am interested in using this method in every fine-grained classification problem to get out the explanation behind every decision made. These explanations can be used to raise the accuracy of the model and also check if there are annotation errors. My higher goal is for these results to be used in real-life problems that need categorization. So, I want to know how real are these results based on a human expert categorization ability. Do the attributes chosen by my method as the most important ones for every bird classification correspond to the attributes that a human, expert on identifying bird species, would choose to make the identification? This is why I created a custom ground truth provided by experts in ornithology to compare with my method's results.

In brief, the contributions of my work are:

1. An innovative explainability method for understanding which are the most important features behind every image classification of the MMAL-Net model.
2. A creation of a unique custom ground truth attribute list for every bird class provided by ornithology experts.
3. Extensive quantitative and qualitative evaluation of my method.

Chapter 2

Related Works

2.1 Fine-grained Classification

Fine-grained classification, also known as fine-grained categorization or fine-grained recognition, refers to the task of classifying objects or entities into highly specific and detailed categories within a broader domain. It involves distinguishing between closely related classes that share similarities but also have subtle differences. In fine-grained classification, the goal is to identify and categorize objects at a more granular level compared to traditional classification tasks [12]. For example, while a basic classification task might involve distinguishing between birds and airplanes, fine-grained classification aims to distinguish between different species of birds or between different brands of airplanes. Fine-grained classification typically requires more specialized models and techniques than general object recognition tasks [13]. This is because the differences between fine-grained categories are often subtle, such as variations in shape, color, texture, or other visual features. These subtle differences can be challenging to capture and distinguish for traditional classification algorithms. Applications in fine-grained classification include tasks such as species identification, product categorization, sentiment analysis, and medical diagnosis, where precise classification at a detailed level is required. Overall, fine-grained classification goes beyond basic categorization tasks, allowing for more specific identification and differentiation of closely related entities within a given domain [12].

2.1.1 Approaches in Fine-grained Classification

Several approaches have been developed to tackle the challenges in fine-grained classification.

1. **Transfer Learning:** Transfer learning involves taking a pre-trained model, typically trained on a large and diverse dataset (e.g., ImageNet), and adapting it for a specific task, such as fine-grained classification. The key idea is that the knowledge gained by the pre-trained model on the source task can be useful for the target task. There are two primary approaches to transfer learning. The first is feature extraction. In this approach, the pre-trained model's layers are used as a fixed feature extractor. The output from one of the intermediate layers (usually before the classification layer) is considered as the feature representation for the input data. These features are then fed into a new classifier (often a simple neural network or SVM) trained on the target dataset. The pre-trained weights are frozen during training, and only the classifier is trained on the target data. The next approach is fine-tuning, which takes the feature extraction approach a step further by allowing some of the layers in the pre-trained model to be updated during training on the target dataset. Typically, the earlier layers, which capture low-level features like edges and textures, are frozen or updated with a very low learning rate, while the later layers are fine-tuned to adapt to the specific features and patterns in the fine-grained dataset. This allows the model to retain some of the useful knowledge from the source task while adapting to the nuances of the target task [14].

2. **Attention Mechanisms:** Attention-based models have gained popularity in fine-grained classification. Attention mechanisms allow models to focus on specific parts or features of an input, which is particularly beneficial in fine-grained classification. These models can selectively attend to subtle details, textures, or patterns that are crucial for distinguishing between closely related classes. This granularity of feature selection is challenging to achieve with conventional models. Attention-based models can adapt their attention weights during inference based on the input data, unlike traditional models with fixed, handcrafted features. This adaptability makes them versatile and capable of handling diverse fine-grained classification tasks without extensive feature engineering. Fine-grained classification often requires precise localization of discriminative features within an image or sequence. These models inherently possess the ability to

pinpoint the relevant regions or elements, making them well-suited for tasks like object part localization or fine-grained image recognition [15]. Attention mechanisms have proven effective in transfer learning scenarios. Pre-trained attention-based models can be fine-tuned on specific fine-grained tasks, leveraging knowledge from a wide range of source domains. This transferability of knowledge allows for efficient model training and improved performance, even with limited labeled data. Attention maps generated by these models can provide valuable insights into the decision-making process. Users can visualize which parts of the input are most influential in the classification, aiding model interpretability and trustworthiness, which is often a challenge in deep learning. These mechanisms are not restricted to a single modality. They can be applied to various data types, including images, text, and speech, making them adaptable to fine-grained classification tasks in different domains. This versatility is a significant advantage when dealing with multi-modal data sources. Attention-based models have been designed to be computationally efficient, with mechanisms like self-attention and sparse attention. This efficiency is essential for real-time or resource-constrained fine-grained classification applications [15]. Some of these attention mechanisms include:

- **Spatial Transformer Networks (STN):** STNs use attention mechanisms to learn spatial transformations that can be applied to input data. They are commonly used in fine-grained image classification tasks to align and extract relevant image regions. However, a distinctive and often undervalued facet of STNs lies in their nexus with human cognition and perceptual processes. STNs draw inspiration from the way humans perceive and identify objects from diverse angles and positions. In a manner akin to how humans mentally manipulate or resize an object to discern it from varying vantage points, STNs can learn to perform analogous adjustments on neural network inputs. This connection to human cognitive processes underscores the potential of deep learning not only to replicate but also to potentially enrich our comprehension of visual information processing within the human brain. Furthermore, STNs have found utility across a spectrum of computer vision tasks, ranging from optical character recognition to image segmentation. Their capacity to adapt and transform data according to the task at hand is a testament to the flexibility and power of deep learning systems in emulating certain aspects of human intelligence [16].

- **Self-Attention Mechanisms (Transformers):** Transformers, equipped with their self-attention capabilities, have found versatile utility in achieving precise classification tasks across diverse domains. They can seamlessly adapt and excel in a multitude of domains, languages, and data types. Transformers can process text data in different languages with remarkable ease. Their self-attention mechanisms allow them to capture cross-linguistic similarities and variations. Just as a multilingual individual can grasp the subtle nuances of expressions and idioms in multiple languages, Transformers can discern the intricacies of semantics and context across diverse linguistic landscapes. Furthermore, they can adapt their attention patterns to suit the specific characteristics of the domain they are applied to. Transformers can learn the inherent patterns and structures of data from various domains, making them exceptionally adaptable to tasks as diverse as image recognition, sentiment analysis, and protein folding prediction. They are not just masters of space (as in understanding the relationships between elements in a sequence) but also time. Their self-attention mechanisms enable them to capture temporal dependencies within sequences. Also, they can generate coherent and contextually relevant text or content, making them indispensable in creative domains. Transformers can be thought of as knowledge synthesizers. They have the capacity to sift through vast amounts of information, extract valuable insights, and distill them into concise and coherent summaries. They mix and match information from diverse sources to produce holistic and informative outputs [17].

- **Non-local Neural Networks:** These networks utilize non-local operations to capture long-range dependencies in data, making them useful for fine-grained image classification tasks where global context is important. Non-local neural networks are a class of neural network architectures that incorporate non-local operations into their structure to capture long-range dependencies in data. This is in contrast to traditional convolutional neural networks, which primarily rely on local operations such as convolutions to process and analyze data. The concept of non-local operations in neural networks was introduced to address the limitations of local operations in capturing global context and dependencies in various types of data, with a particular focus on image processing tasks. Non-local neural networks are designed to capture relationships between distant pixels or elements in

an image or any data with a grid-like structure. In image processing, this is essential for tasks where understanding the global context is crucial. For example, in fine-grained image classification, recognizing subtle details or patterns often requires considering long-range dependencies between different parts of the image. The core innovation in non-local neural networks is the incorporation of non-local operations. These operations allow each element in the input data to interact with all other elements, regardless of their spatial proximity. This is in contrast to convolutional operations, which are local and limited to a small receptive field. Non-local operations are typically implemented using a self-attention mechanism. Self-attention allows the network to weigh the importance of different elements in the input when computing the output. Elements that are more relevant to a particular context receive higher attention scores, which allows the network to focus on capturing long-range dependencies. In fine-grained classification tasks, global context is crucial for accurate classification. Non-local neural networks excel in such tasks because they can capture the relationships between fine-grained features that may be distributed throughout the image. However, they can be computationally expensive, as they involve interactions between all elements in the input data. This can make training and inference with non-local neural networks more resource-intensive compared to standard CNNs [18].

- **Dense Attention Networks:** Dense Attention Networks (DANs) are a specialized class of neural networks that are particularly effective in capturing fine-grained details and patterns in various types of data, such as images. These networks operate by computing attention maps densely across an input, which allows them to focus on fine details at a much more granular level than traditional attention mechanisms. This capability is especially valuable in fine-grained image classification tasks, where distinguishing between subtle differences is crucial. The core of DANs is the attention mechanism, which is inspired by the human visual system. Attention mechanisms enable the network to focus on specific regions or features of the input, assigning different importance scores to different elements. DANs create dense attention maps by assigning attention scores to a grid or a set of local patches within the input. This fine-grained granularity allows the network to focus on even the smallest visual cues, such as texture, color, or shape details

that are indicative of specific classes or categories. DANs often employ multiple layers or scales of attention, enabling them to capture details at different levels of abstraction. This multi-scale analysis is vital for recognizing fine details while still considering the context and structure of the image [19].

- **Bilinear Attention Networks:** These networks employ bilinear pooling with attention mechanisms to model fine-grained interactions between different regions or elements in the data, often used in visual question answering and image-text tasks. Bilinear pooling is a mathematical operation that combines information from two sets of vectors. In the context of these networks, it is used to capture the interactions or relationships between different elements or regions. Furthermore, attention mechanisms are a fundamental part of Bilinear Attention Networks. They allow the network to focus on specific parts of the data, emphasizing relevant information and suppressing irrelevant details. In the context of visual question answering, an attention mechanism may be used to determine which regions in an image and which words in a question are most relevant to answering the question. This process involves assigning attention weights to different regions or elements based on their importance [20].

- **Dual Attention Networks:** Dual attention networks incorporate both spatial and channel-wise attention mechanisms to capture fine-grained features and interdependencies in data. Spatial attention focuses on the relationships between different spatial locations within input data, such as an image or a sequence of data. It aims to identify which parts of the data are more important or relevant for a given task. For instance, in image analysis, spatial attention can help the network focus on specific regions or objects within an image that are critical for understanding the content or making predictions. Channel-wise attention, on the other hand, focuses on the relationships between different channels or feature maps within the data. In deep neural networks, data is often processed through multiple convolutional layers, and each layer generates a set of feature maps (channels) that represent different aspects of the input. Channel-wise attention helps the network to determine which channels contain relevant information and should be emphasized while suppressing less informative channels. By incorporating both spatial and channel-wise attention, dual attention networks can capture complex rela-

tionships and dependencies in the data. They are particularly useful for tasks that involve understanding both spatial structure and the channel-wise distribution of information, such as object recognition in images [21].

- **Hierarchical Attention Networks:** These models utilize hierarchical attention mechanisms to process data at multiple levels of granularity, making them suitable for fine-grained tasks with complex structures, such as document classification. The hierarchical attention mechanism is applied at multiple levels within the hierarchy of the data. It allows the model to attend to and extract information from different levels of granularity simultaneously. Specifically, HANs typically have two levels of attention. Word-level attention and sentence-level attention. Word-level attention focuses on words within a sentence or sentences within a paragraph. It assigns weights to words or sentences based on their importance for the task. Sentence-level attention, on the other hand, focuses on sentences within a paragraph or paragraphs within a document. It assigns weights to sentences or paragraphs based on their relevance or significance. The hierarchical attention mechanism enables the model to process information from both the micro-level (words, sentences) and the macro-level (paragraphs, documents) simultaneously. This is beneficial for tasks that involve complex structures. HANs are particularly suitable for fine-grained tasks where understanding the relationships between different levels of data is crucial. They offer the advantage of capturing both local and global dependencies within the data, making them well-suited for tasks that require a deep understanding of hierarchical structures. They often outperform traditional models that only consider the data at a single level of granularity [22].

- **Region-based Attention Models:** These models segment an input into regions of interest and apply attention mechanisms to each region individually, which is useful in fine-grained object recognition tasks. The first step in these models involves dividing the input into smaller regions or patches. Each patch represents a portion of the input that will be processed individually. After segmentation, the model applies attention mechanisms to each region independently. This way, the model can focus its processing on the regions that are most relevant to the task. Within each region, the model extracts features and representations that are specific to that region. These features could be learned through convolutional

layers or other neural network components tailored for the particular task. Once each region has had its features extracted, the model can aggregate this information across all regions to create a comprehensive representation of the entire input. This can be achieved through pooling mechanisms or by applying additional attention mechanisms across the regions. The aggregated representation is then used for task-specific processing. Applications of these models can be seen in tasks such as fine-grained object recognition, image segmentation, and visual question answering [23].

- **Multi-modal Attention Models:** For fine-grained classification tasks that involve multiple data modalities (e.g., text and images), multi-modal attention models have been developed to effectively fuse and process information from different sources. A multi-modal attention model combines the information from different modalities in a way that allows it to focus on relevant parts of each modality and effectively fuse this information for decision-making. The text and image data are embedded into numerical vectors that can be processed by the model. Attention mechanisms enable the model to weigh the importance of different parts of the input data. In the context of multi-modal models, attention can be applied independently to both text and image modalities. These mechanisms combine the attended information from different modalities. Common fusion methods include concatenation, element-wise multiplication, or addition. The fused information is then used to make a final prediction or decision, such as category classification. Multi-modal attention models are trained on datasets where each data point contains both text and image information, along with corresponding labels. During training, the model learns to adjust its attention and fusion mechanisms to make accurate predictions [24].

3. **Metric learning:** Metric learning strategies strive to acquire a similarity measurement that optimizes the gap between various attribute groupings while concurrently reducing the fluctuations within each group. This not only contributes to heightened discrimination amidst intricately detailed categories but also empowers the model to grasp subtleties between closely related classes. Metric learning's primary objective is to elevate the likeness among instances from the same category while reducing the resemblance between instances from distinct categories. This approach effectively ad-

dresses the difficulties arising from substantial variations within the same class and minimal disparities between different classes, rendering it particularly suitable for intricate fine-grained tasks. Metric learning employs a range of strategies, including triplet networks, which train by creating trios comprising an anchor, a positive, and a negative example. These networks promote the anchor-positive distance being less than the anchor-negative distance by a predetermined threshold [25]. In addition, contrastive loss functions work to bring similar examples closer together and push dissimilar ones further apart within the embedding space. Meanwhile, Center loss prioritizes the mastery of tightly-knit clusters of examples for each class, thereby assisting the model in grasping class-specific differences [26]. Additionally, there exist Proxy-centric techniques that substitute the initial classes with proxy entities within the embedding realm, ultimately amplifying the distinction among fine-detailed classifications [27]. Meanwhile, Siamese networks present an alternative strategy, intending to align analogous instances in immediate proximity and position dissimilar ones at a considerable distance within the embedding dimension [28].

4. **Ensemble methods:** Combining multiple classifiers or models using ensemble techniques can improve the overall classification performance in fine-grained attribute classification tasks. Ensemble methods aggregate predictions from multiple models to make more accurate and robust predictions. Ensemble methods encompass a variety of techniques, including bagging, boosting, and stacking. Bagging techniques such as Random Forest create multiple models by training on different subsets of the data. Boosting methods like AdaBoost iteratively prioritize misclassified instances, while stacking combines predictions from multiple models with a meta-learner. They are particularly well-suited for fine-grained classification due to their ability to capture diverse feature representations and decision boundaries. They mitigate overfitting by aggregating predictions from multiple models, reducing biases and errors associated with individual classifiers. This is especially advantageous when dealing with limited labeled data [29].

2.1.2 Evaluation Metrics

The evaluation of the performance of fine-grained classification models relies on quantitative evaluation metrics to gauge the performance and efficacy of models. These metrics

are indispensable for evaluating a model's performance, aiding in model selection and refinement, and establishing whether a model's predictions align with the desired results. Various evaluation metrics exist, and the selection of a metric hinges on the particular problem and model objectives. Commonly used metrics include:

1. **Accuracy:** The proportion of correctly classified attributes [30].

2. **Precision, Recall, and F1-score:** These metrics provide a more detailed analysis of model performance by considering true positives, false positives, and false negatives [31, 30].

$$Precision = \frac{TP}{TP + FP} \tag{2.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.2}$$

3. **Mean Average Precision (MAP):** MAP measures the average precision across different attributes, providing an aggregated performance measure. Used in object detection and retrieval tasks, MAP considers precision-recall curves for each class and computes the average precision over all classes. It's well-suited for fine-grained classification where the class imbalance is common [31, 30].

4. **Specificity (True Negative Rate):** Specificity quantifies the ratio of correctly identified negatives to all real negative cases, and it is crucial when the priority is reducing false alarms [30].

5. **ROC Curve (Receiver Operating Characteristic Curve):** The ROC curve visually represents how a model performs at various threshold settings, aiding in the evaluation of the balance between correctly identified positives and false alarms [31, 30].

6. **AUC (Area Under the ROC Curve):** AUC is a measure that gauges how well a classification model performs overall. A greater AUC signifies a stronger ability to discriminate between different classes [31, 30].

7. **Mean Absolute Error (MAE):** MAE determines the mean of absolute discrepancies between predicted and observed values in regression assignments [30].

8. **Mean Squared Error (MSE):** MSE computes the mean of squared variances between predicted and real values in regression scenarios, magnifying the significance of substantial errors [30].

9. **R-squared (Coefficient of Determination):** R squared evaluates how much of the variance in the dependent variable can be explained by the independent variables, providing an assessment of how effectively the model aligns with the data [31, 30].

2.2 Interpretability in Machine Learning Models

In recent years, machine learning (ML) has made significant advancements and is now being utilized in various sectors. However, one of the main challenges associated with ML models is their lack of interpretability. Explainable machine learning (XML) is a swiftly evolving field that seeks to address this challenge by providing clarity into how ML models make predictions or decisions. As ML models become more complex, their decision-making processes become increasingly opaque, leading to a lack of trust and potential risks. In critical domains like healthcare, where ML models are used to diagnose diseases or determine treatment plans, it is crucial to understand why a model made a specific decision. Explainable machine learning helps to bridge the gap between the predictive power of ML models and the need for interpretability, providing human-understandable explanations for their predictions. Interpretability in machine learning models refers to the ability to understand and explain how a model makes predictions or decisions. Interpretability helps build trust in machine learning models and ensures that their behavior is understandable and justifiable to humans. Local and global interpretability are two different approaches to understanding the behavior of machine learning models. Local interpretability focuses on explaining the predictions of a machine-learning model on an individual instance or data point. The objective is to comprehend why a specific prediction was formed for that particular input. This approach is particularly useful for gaining insights into how the model behaves in specific cases. Global interpretability, on the other hand, aims to provide an overview of how a machine-learning model behaves across the entire input space. The goal is to understand the model's overall patterns, relationships, and general decision-making process. This approach is more concerned with understanding the model as a whole rather than explaining individual predictions [32].

2.2.1 Inerpretable Model Types

Here I represent some interpretable models:

1. **Linear Models:** Linear models are a class of statistical and machine learning models that assume a linear relationship between input features and the target variable. They are widely used for both regression and classification tasks. The fundamental idea behind linear models is to fit a linear equation to the observed data, allowing predictions to be made based on the linear combination of input features. Within the realm of linear regression, there's an underlying assumption that the interplay between input features and the target variable can be elegantly captured through a linear equation. Going beyond the basics, linear regression models can be elevated with the incorporation of regularization approaches such as ridge regression, lasso regression, and elastic net regression. These innovative techniques inject penalty components into the model's objective function, effectively acting as safeguards against overfitting and elevating its prowess in generalization [32].

2. **Rule-Based Models:** Rule-based models are one of the most intuitive interpretable model types. They employ a set of predefined rules to make decisions. These rules can be simple if-then statements or more complex logical expressions. Rule-based models provide transparent decision-making processes and are highly interpretable. However, they may struggle with complex datasets and lack the ability to capture intricate patterns [33, 34, 35].

3. **Decision Trees:** Decision trees are hierarchical models that use a series of branching decisions to classify or predict outcomes. They provide a clear visual representation of decision paths, making them highly interpretable. Decision trees capture both linear and non-linear relationships in the data, and their simplicity allows for easy interpretation. Nonetheless, decision trees can suffer from overfitting and instability when dealing with noisy or high-dimensional data [36].

4. **Generalized Linear Models (GLMs):** Generalized linear models are a class of statistical models that extend linear regression to accommodate different types of data distributions. GLMs provide interpretable coefficients, allowing for a clear understanding of variable impacts on the outcome. They are well-suited for binary classification, re-

gression, and count data analysis. However, GLMs assume linearity between predictors and conclusions, which may limit their performance on highly complex datasets [32].

5. **Bayesian Networks:** Bayesian networks are a type of probabilistic graphical model used to depict connections between variables using directed acyclic graphs. They enable the explicit modeling of uncertainty and offer a clear representation of conditional dependencies. Bayesian networks are particularly adept at making informed inferences when faced with uncertainty and are efficient at managing missing data. Nonetheless, creating precise Bayesian networks demands expertise in the relevant domain and can be computationally demanding [37].

6. **Interactive Models:** Interactive models in the context of machine learning and AI involve combining human feedback with the algorithm's learning process. It is a subset of "human-in-the-loop" algorithms, where humans play an active role in shaping and improving the model's performance. The goal of interactive learning is to leverage the strengths of both machines and humans. Machines excel at processing large volumes of data and identifying patterns, while humans possess cognitive abilities such as reasoning, context understanding, and domain expertise. By combining these capabilities, interactive models aim to achieve more accurate, robust, and transparent results. One of the essential advantages of interactive learning is its ability to address the issue of limited or biased training data. Human feedback can help the model handle edge cases, fine-tune its behavior, and identify potential biases, improving its generalization and fairness [38].

7. **Naive Bayes:** Naive Bayes is a simple yet effective probabilistic classification algorithm based on Bayes theorem. It is widely used for tasks such as text categorization, spam filtering, sentiment analysis, and more. At its core, Naive Bayes calculates the probability of a sample belonging to a particular class based on the observed features of that sample. The "naive" in Naive Bayes comes from the assumption of feature independence, which simplifies the calculations. The algorithm presupposes that, when taking the class label into account, each feature's existence or non-existence is unrelated to that of any other feature. In simpler terms, the presence or absence of one feature has no bearing on whether another feature is present or absent. This assumption allows for a straightforward calculation of the probabilities [32].

8. **K-nearest Neighbor Models (KNN):** K-nearest neighbor (KNN) is a non-parametric classification and regression algorithm used for both supervised learning tasks. It is a simple yet effective algorithm that makes predictions based on the similarity or proximity of data points in the feature space. In KNN, the ”k” represents the number of nearest neighbors considered for making predictions. When presented with a new, unlabeled data point, KNN identifies the ”k” nearest neighbors to that point in the feature space. The class or label of the majority of these neighbors is assigned to the new data point as its predicted class or label. KNN doesn’t explicitly build a model during training. Instead, it stores the training dataset and calculates distances at prediction timem [32].

2.2.2 Interpretable learning by nature

Specific classic machine learning algorithms can inherently produce interpretable models without explicitly optimizing for interpretability during the training process. These interpretable models provide interpretability by themselves and, thus, can be considered “interpretable by nature”. These algorithms are often designed with simplicity and transparency in mind, which leads to models that can be directly explained and understood. By prioritizing high accuracy during the optimization process, these algorithms produce models that capture the underlying patterns in the data effectively. The resulting models tend to have a straightforward structure and feature importance that can be readily interpreted. Visualizations, such as decision trees or rule sets, are commonly used to gain insights into the model’s function and behavior [32]. Section 2.6.2 focuses on some ”interpretability by nature” methods.

1. **Decision Trees:** The Classification and Regression Trees (CART) algorithm was introduced by Leo Breiman in 1986. Breiman’s original paper is titled ”Classification and Regression Trees” and was published in the journal Machine Learning in 1984. CART is a popular and powerful algorithm used for both classification and regression tasks. It works by recursively partitioning the feature space into subsets, intending to create homogeneous groups that minimize the impurity (for classification) or the variance (for regression) within each group. The algorithm constructs a binary tree where each internal node represents a decision based on a particular feature, and each leaf node represents a predicted class or value [39]. Another decision tree algorithm is the ID2of3 [40]. The ID2of3 algorithm is designed for constructing decision trees in the context of machine learning and pattern recognition. It uses a hill-climbing search pro-

cess to learn M-of-N splits at each node. The idea behind M-of-N splits is that instead of making a binary decision at each node, it allows the node to have multiple branches, and a certain number of them need to be satisfied to reach a particular leaf node. Other algorithms include the C4.5 algorithm [41, 42], the C5.0T algorithm [43], which is an extension of C4.5, and the ID3 algorithm.

2. **Decision Rules:** The 1R algorithm, introduced by Robert Holte in 1993, is a simple and straightforward rule-based classification algorithm [44]. The 1R algorithm aims to find a single decision rule that can be used to classify instances in a dataset based on their attribute values. Despite its simplicity, it can often produce surprisingly good results, especially for datasets with categorical attributes. RIPPER algorithm [45] is primarily designed for classification tasks on structured data, where the target variable is discrete, and the input features can be categorical or continuous. It is an extension of the ”IREP” (Incremental Reduced Error Pruning) algorithm and uses a rule-based approach to build a set of IF-THEN rules for classification. Another algorithm is the CN2 algorithm [46], which is designed to learn a set of classification rules from labeled data, where each rule consists of an antecedent (condition) and a consequent (class label). The CN2 algorithm belongs to the category of inductive rule learning. Other algorithms in this category include the AntMiner+ algorithm [47], the Re-RX algorithm [48], and the C5.0R algorithm [43].

3. **Generalized Additive Models (GAMs):** Generalized Additive Models (GAMs) are a statistical modeling technique used for fitting non-linear relationships between predictors (independent variables) and a response (dependent variable). GAMs extend the concept of Generalized Linear Models (GLMs) by allowing for flexible and smooth functions of predictors, which makes them particularly useful for capturing complex, non-linear relationships in data. In a GAM, the response variable is presumed to conform to a distribution within the exponential family, which may include Gaussian, Poisson, or binomial. The fundamental concept behind GAMs revolves around representing the connection between the response variable and individual predictors as a summation of smooth functions about those predictors. The term ”additive” underscores the notion that each predictor’s impact operates independently of the others, and their influences are cumulatively combined, as discussed in reference [49].

$$(E(Y)) = \beta_0 + f_1(x_1) + f_2(x_2) + \cdots + f_k(x_k) \tag{2.3}$$

as given in [49] we have:

- $g(\cdot)$ is a link function that links the response variable to the linear predictor.
- $E(Y)$ is the expected value of the response variable Y given the predictors.
- β_0 is the intercept term.
- $f_1(x_1) + f_2(x_2) + \cdots + f_k(x_k)$ are smooth functions of the predictors $x_1 + x_2 + \cdots + x_k$ that capture the non-linear relationships.

2.2.3 Interpretable learning by design

This approach involves incorporating interpretability as a specific objective or constraint during the model development process. Unlike algorithms that produce interpretable models by nature, interpretable-by-design approaches provide more control over the degree of interpretability and allow for further improvements in human interpretability. By explicitly optimizing for interpretability, these approaches enable the development of white box models that are specifically tailored to be easily understandable by humans. They offer the advantage of flexibility, allowing the grade of interpretability to be managed and upgraded according to specific requirements. Interpretable by-design approaches encompass various techniques and methodologies. These models often possess clear decision rules, feature importance rankings, or explicit representations that align with human intuition and reasoning. By adhering to simpler architectures and prioritizing transparency, they enable users to comprehend the model's behavior and reasoning easily [32]. Section 2.6.3 focuses on some "interpretability by design" methods.

1. **Decision Trees:** Oblique Treed Sparse Additive Model (OT-Spam) is a specific type of region-specific predictive model that combines the flexibility of oblique trees with the interpretability and regularization of sparse additive models. It uses oblique trees to identify different regions or subgroups in the data and fits sparse additive models in each of these regions to capture the specific relationships between the variables [33, 50]. As mentioned earlier, oblique trees are decision trees that allow for splits that are not restricted to being axis-parallel. These types of trees can capture

more complex relationships between features and are well-suited for handling data with region-specific patterns. Furthermore, Sparse Additive Models (SpAM) are models that assume the overall response is a sum of smooth functions of individual features, with some form of regularization to encourage sparsity. The sparsity property makes the model more interpretable and robust by discouraging the inclusion of unnecessary features. Another approach is the Neural Decision Tree (NDT), which is a machine-learning model that combines the strengths of decision trees and neural networks. NDTs were introduced as a way to address the limitations of traditional decision trees, such as struggles with capturing complex patterns in data and handling high-dimensional feature spaces. This model attempts to bridge the gap between the interpretability of decision trees and the representational power of neural networks [51].

2. **Decision Rules:** The BOA model, introduced by Wang et al. in 2015, is a probabilistic rule-based machine learning model. It is designed to discover and represent rules in data by combining the ”Or” and ”And” logic operations within a Bayesian framework. The model allows for the incorporation of prior knowledge, represented as priors, which helps regulate the size and shape of the underlying rules. In the context of BOA, rules refer to logical expressions involving the input features and the target variable. These rules can be simple, single-condition statements or more complex combinations of conditions, including both ”And” and ”Or” operations. For example, a simple rule could be something like ”IF feature A is true, THEN predict class 1.” The model’s structure and parameters are learned from the data and are influenced by the prior knowledge or beliefs encoded in the priors. By incorporating priors, the BOA model can be guided toward generating rules that align with prior expectations or domain-specific knowledge. This feature makes BOA especially useful when there is some existing knowledge about the problem domain that can be utilized to shape the learned rules [33, 50]. Another technique is Falling Rule Lists (FRL) which is proposed by Cynthia Rudin and Yining Wang in their 2014 paper titled ”Falling Rule Lists” (published in the Journal of Machine Learning Research). It is designed for interpretable and transparent rule-based classification. The main goal of FRL is to create a predictive model that consists of a list of rules, where each rule is associated with a class label. These rules are ordered in such a way that they form a ”falling” sequence, meaning that the first rules are more important and cover more instances, while subsequent rules become more specific and

cover fewer instances. This approach allows the model to make predictions hierarchically, starting with the more general rules and then using the more specific rules for instances not covered by the previous rules [52].

3. **Decision Sets:** The Bayesian Rule Set (BRS) is a method designed to generate sets of rules that are easy to understand. It belongs to the category of machine learning models that seek to blend the advantages of rule-based systems with probabilistic models like Bayesian networks. The primary objective of BRS is to construct a model that is both interpretable and transparent, enabling humans to gain insights into the reasoning behind its predictions, thereby enhancing trust in its decisions. Even though it operates based on rules, BRS incorporates the concept of probabilistic reasoning by assigning probabilities to each rule, indicating the level of uncertainty associated with the rule's accuracy [53]. Another approach is the Interpretable Decision Sets (IDS) proposed by Lakkaraju et al. in their 2016 paper titled "Interpretable Decision Sets: A Joint Framework for Description and Prediction." IDS is designed to produce human-readable models that not only make accurate predictions but also provide meaningful explanations for those predictions. IDS generates a set of rules in the form of "if-then" statements, where each rule corresponds to a specific combination of conditions on the input features. These rules collectively form an interpretable decision set. The rules in the decision set are ordered based on their importance or relevance. This ordering allows for the prioritization of more critical rules in decision-making. The key advantage of IDS is that it provides transparency in the decision-making process, making it easier for users to understand and trust the model's predictions [34].

4. **Linear Models:** Supersparse Linear Integer Models (SLIM) is a machine learning technique proposed by Ustun and Rudin in 2014. SLIM is designed for building interpretable and efficient linear models, particularly in the context of binary classification problems. The method aims to find a linear decision boundary while using a minimal number of features. SLIM formulates the process of selecting features and coefficients for the linear model as an Integer Linear Programming (ILP) problem. ILP is a variant of linear programming where some or all variables must take integer values. SLIM includes a sparsity-promoting regularization term in the ILP objective function. This term encourages the selection of fewer features, leading to a highly interpretable and compact model. The SLIM algorithm allows for controlling the complexity of the final

model by adjusting the regularization strength or the number of allowed features [43].

2.2.4 Surrogate Models

Surrogate models serve as valuable tools in approximating inscrutable black-box models that elude human comprehension. These black-box models may encompass sophisticated machine learning algorithms or any model characterized by a multitude of parameters and intricate non-linear relationships. Given the inherent difficulty in grasping the inner workings of black-box models, surrogate models emerge as a means to craft more straightforward and intelligible models that capture the essence of the black-box model's behavior. Surrogate models can be either local or global by nature. A global surrogate model is built to approximate the overall behavior of the black-box model across the entire input space. It aims to capture the general trends and patterns of the black-box model, providing a more comprehensive understanding of its behavior. On the other hand, a local surrogate model is constructed to approximate the black-box model's behavior in a specific region of the input space. It aims to provide insights into how the black-box model behaves around a particular data point or within a small neighborhood. Local surrogate models are useful for understanding the black-box model's decision-making process in specific instances [32].

1. **Global Surrogate Models:** One method is the Decision Tree Extraction algorithm [54]. The key idea of the method is to take a pre-trained deep neural network (DNN) and convert it into an equivalent decision tree. This conversion process allows users to understand how the DNN makes decisions, particularly for individual examples. The resulting decision tree should ideally have similar behavior to the original deep neural network but in a more interpretable form. Another approach is the Binary Decision Tree method which is based on maximizing the difference in the average contribution of the split variable [55]. sp-Lime and k-Lime represent two distinct branches of the LIME algorithm family, both offering comprehensive insights on a global scale [56]. Some approaches use Random Forest (RF) as the underlying prediction model and then extract rules from the RF to gain interpretability and insight into the decision-making process of the model. Random Forest, as an ensemble learning technique, blends multiple decision trees to enhance prediction accuracy and boost generalization capabilities. Each decision tree in the Random Forest is trained on a bootstrapped sample of the data, and at each node, a random subset of features is considered for splitting. Some of these

methods include the RuleFit algorithm, the hill climbing with downhill moves (DHC) algorithm, and the Sparse Group Lasso (SGL) method [57].

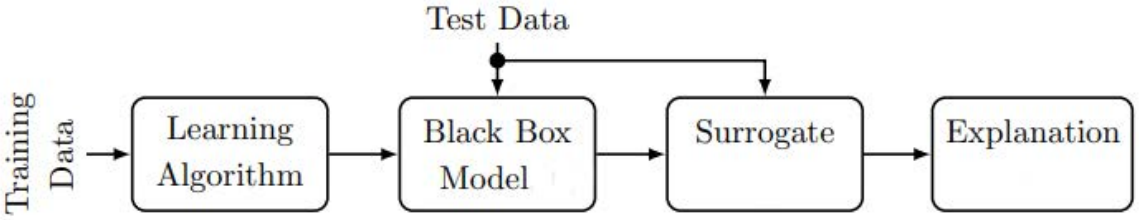

Figure 2.1: Fitting a global surrogate.

2. **Local Surrogate Models:** LIME (Local Interpretable Model-agnostic Explanations) is one of the most notable algorithms for explaining the predictions of black-box models. LIME starts by perturbing or generating synthetic data points around the instance for which you want an explanation. These synthetic data points are created by randomly modifying the features of the original instance while keeping other parts of the data constant. The black-box model makes predictions on the synthetic data points generated in the previous step. LIME assigns weights to these synthetic data points based on their similarity to the original instance. The more similar a synthetic data point is to the original instance, the higher the weight it receives. A simpler, interpretable model (e.g., linear regression) is then trained on the synthetic data points using the assigned sample weights. The local model tries to mimic the predictions of the black-box model in the neighborhood of the original instance. Finally, the coefficients and the behavior of the local model are used to explain why the black-box model made a particular prediction for the given instance [56, 58, 59]. Another popular and widely used method is SHAP (SHapley Additive exPlanation). The goal of SHAP is to provide a fair and consistent way of attributing the contribution of each feature to a specific prediction made by a model. It helps us understand how much each feature impacts the model's output for a given input instance. SHAP values ensure that the contributions of individual features add up to the difference between the model's output for a specific instance and the expected output across all instances [60]. Other approaches include MAPLE, which corresponds to Model Agnostic Supervised Local Explanations [61], and LORE, which corresponds to LOcal Rule-based Explanations [62].

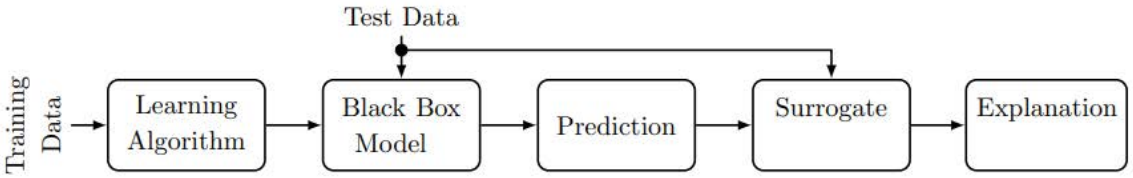

Figure 2.2: Fitting a local surrogate.

2.2.5 Explanation Generation

In this part, we discuss methods capable of directly producing an explanation, either at a local or global level. Unlike surrogate models, these approaches obtain the explanation instantly from the black box network itself [32].

1. **Global Explanation Generation:** The Leave-One-Out (LOO) approach is a method commonly used for feature importance evaluation in the context of machine learning and statistical modeling. It involves omitting one feature at a time from the prediction process to assess the impact of each feature on the model's performance. The idea is to measure how the model's performance changes when a specific feature is excluded. In LOO, for each data point in the dataset, the model is trained without using the feature of interest and then tested on that specific data point. This process is repeated for all data points and all features, resulting in a collection of models, each missing one of the features. By comparing the model's performance with and without each feature, one can infer its contribution to the overall prediction process [63]. The LOO approach can be computationally expensive, especially for high-dimensional datasets where the number of features is large. For each data point, it requires training a separate model without a specific feature, which can lead to a significant increase in computation time. As a result, the LOO approach may not be practical for high-dimensional data, and its application becomes more challenging as the feature space grows. Another method is the Global Sensitivity Analysis (GSA) technique and it is used to understand the behavior of black box models by examining how changes in input features affect the model's output. The method was introduced by Cortez and Embrechts (2011) and employs sensitivity measures such as range, gradient, and variance to quantify the impact of input variations on the model's predictions [64]. Golden Eye is another approach and its goal

is to provide insights into feature interactions and their impact on the classifier's output, even when the classifier's internal workings are not transparent or interpretable [65]. Some other methods include the Gradient Feature Auditing (GFA) algorithm [66], the Automatic Structure Identification (ASTRID) technique [67] and the iForest [68].

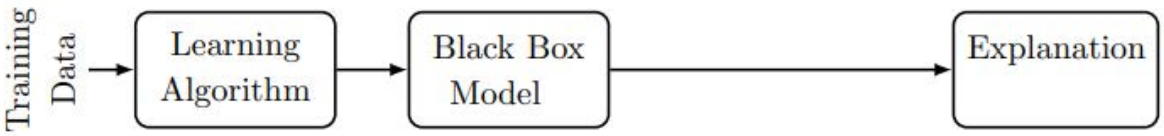

Figure 2.3: Generating global explanations.

2. **Local Explanation Generation:**

   - **Feature attribution:** Feature attribution approaches are very important when you have to deal with generating local explanations. The Local Gradients method aims to understand how the input features of a model influence its predictions for a specific instance [69]. The approach relies on the local gradients of the probability function with respect to the input features. The local explanation vector is essentially a set of gradients calculated for each input feature concerning the probability of a particular class label being predicted. These gradients indicate how sensitive the model's output is to changes in each input feature at a specific instance. Another approach is the Leave-One-Covariate-Out (LOCO) method and its main goal is to determine the importance of a specific input feature on the model's predictions. The basic idea behind LOCO is to compare the model's performance when all input features are used versus when the feature of interest is left out [70, 71]. Furthermore, there are ICE plots (Individual Conditional Expectation) that visualize the relationship between the target variable and an individual feature for each instance in the dataset [72]. Instead of showing the average behavior of the feature across all instances, ICE plots allow you to observe the individual behavior of the feature for each data point. By plotting ICE plots for multiple instances together, you can get insights into the heterogeneity in the relationship between the target variable and the feature across the dataset.

Another approach for feature attribution is the Grad-CAM algorithm, short for Gradient-weighted Class Activation Mapping. Grad-CAM's goal is to integrate gradient details into its framework [9, 10, 11]. I analyze Grad-CAM in a separate subsection because it's the main algorithm of this project.

- **Attention Based Models:** Attention-based models are a class of models that utilize attention mechanisms as a key component. These mechanisms allow the model to focus on relevant parts of the input when generating an output. One of the main advantages of attention mechanisms is their ability to highlight or assign higher weights to the most relevant parts of the input data. This way, the model can effectively emphasize the important features that contribute significantly to the output for a given task [15]. Context vectors are intermediate representations generated during the attention process. In the context of attention-based models, these context vectors are derived by attending to the input sequence and aggregating information from different parts of the sequence. They contain information about the relationship between the input tokens and capture the context needed for generating the output. The concept of context vectors is closely related to the attention mechanism's ability to consider surrounding information when processing each element in the sequence [73]. Some approaches include the Attention Based Summarization (ABS) method [74] and the Image Attention Based Models (IABMs) method [75].

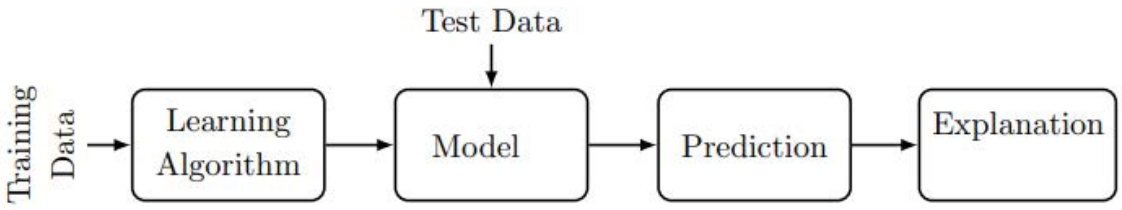

Figure 2.4: Generating global explanations.

2.2.6 Grad-CAM Algorithm

The Grad-CAM (Gradient-weighted Class Activation Mapping) algorithm is a technique used to understand the regions of an input image that contribute most to the predictions made by a convolutional neural network (CNN). Grad-CAM provides a way to generate

heatmaps highlighting the important regions in the image that the network focuses on during its decision-making process. The Grad-CAM algorithm builds upon the concept of Class Activation Maps (CAM) and leverages gradient information flowing into the final convolutional layer of the CNN [9, 10, 11]. The following steps outline the Grad-CAM process:

1. **Model Training:** To begin, a Convolutional Neural Network (CNN) model undergoes training on an extensive dataset that contains labeled examples, allowing it to acquire an understanding of the connections between input images and their respective labels. In the architecture of the model, it is essential to include a global average pooling (GAP) layer and a concluding fully connected (FC) layer prior to the application of the softmax activation function.

2. **Forward and Backward Pass:** To generate the Grad-CAM heatmap, an input image is fed through the trained CNN, and the forward and backward passes are performed. The forward pass calculates the activations of each layer, leading to the final prediction. The backward pass computes the gradients of the predicted class score with respect to the feature maps.

3. **Gradient Calculation:** The gradients flowing back from the final FC layer to the last convolutional layer are globally averaged. These gradients represent the importance of the feature maps regarding the predicted class. The resulting gradient values are pooled spatially, resulting in a single gradient value per feature map.

4. **Heatmap Generation:** The gradient values obtained in the previous step are combined with the feature maps to create a weighted sum, where each feature map is multiplied by its corresponding gradient value. This process results in a weighted activation map.

5. **Heatmap Visualization:** The weighted activation map is then processed by applying a ReLU activation function to discard negative values. Finally, the heatmap is resized to the original input image size and overlaid on the image using a color map to indicate the important regions responsible for the prediction.

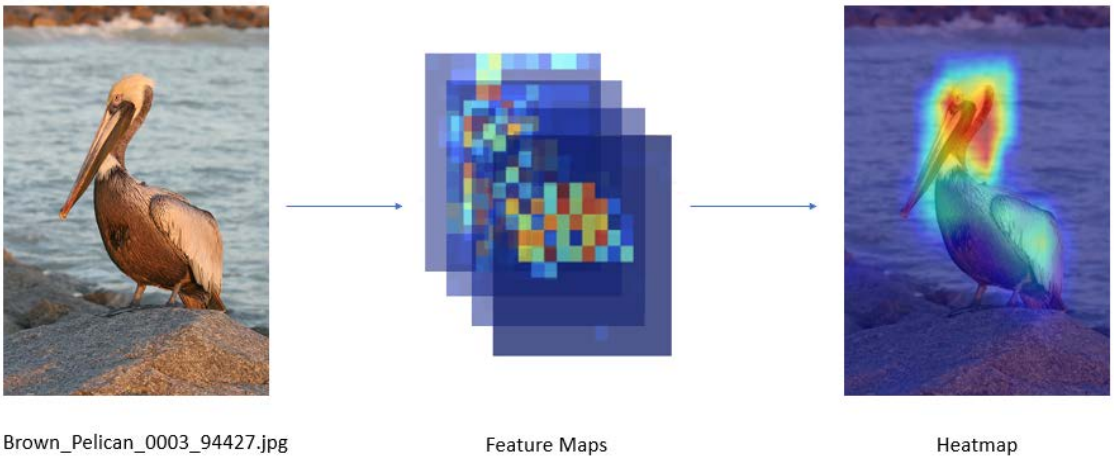

Figure 2.5: A Brown Pelican (Pelecanus occidentalis) image from the CUB-200-2011 dataset followed by the feature maps (target layers) from the MMAL-Net network and the final heatmap which is provided by the Grad-CAM algorithm.

The Grad-CAM algorithm offers several benefits and finds applications in various areas. Grad-CAM helps in understanding the decisions made by CNN models by highlighting the regions in the input image that influence those decisions. This interpretability is crucial in domains where model transparency and explainability are required. By visualizing the heatmap, researchers and developers can identify cases where the model focuses on incorrect or irrelevant image regions. This insight can aid in model debugging and refinement, leading to improved performance. In medical imaging, Grad-CAM can be applied to highlight the specific regions in an image that contribute to a diagnosis. Doctors and healthcare professionals can utilize these heatmaps to gain insights into the model's decision-making process and validate its predictions [9, 10, 11]. In MMAL-NET we can observe from Grad-CAM heatmaps the important regions in every bird image. Grad-CAM can assist in object localization tasks by identifying the regions in an image where the model focuses its attention. This information is valuable for applications such as object detection and segmentation.

2.3 Concept-based Explanation

In this section, I present related work that provides a mechanism for explaining the model's decision with important image regions and corresponding concepts or attributes.

1. **Choose Your Neuron: Incorporating Domain Knowledge Through Neuron-Importance**

Selvarraju and colleagues suggest a technique that uses gradient calculations to determine the significance of each channel within a feature map. This is called Neuron Importance-based Weight Transfer (NIWT). The most important channels are identified by selecting the neurons with the highest importance scores. The authors assume that each channel represents a single, human-understandable attribute. They then propose to learn a mapping from the channel index to a specific attribute. To achieve this NIWT leverages a dataset of images to compute the importance of each neuron, and it simultaneously incorporates the corresponding domain knowledge representations for the training classes. Through this process, it acquires a learned mapping that associates class-related knowledge with the importance of neurons. This acquired mapping is subsequently applied to predict the importance of neurons using information about unseen classes. They then optimize the weights of the classifier in a manner that aligns with the predicted importance. In essence, by utilizing domain knowledge of previously unseen categories, it can forecast the low-level neurons that should contribute to the ultimate classification decision. After that, they train the network's weights so that the neurons anticipated to be significant indeed play a role in making the final decision. This approach allows them to bridge the gap between the description of a novel, unseen category, and the classifier's weights, enabling accurate prediction for these categories without requiring any actual images from them. This approach represents a novel breakthrough, as it is the first known zero-shot learning technique that establishes a direct link between domain knowledge and intermediate neurons within a deep network. Additionally, the method delivers an extra advantage by imbuing the neurons with interpretable semantics through the learned association from domain knowledge to neuron importance, effectively performing automatic neuron naming. They specifically concentrate on the demanding scenario of generalized zero-shot learning (GZSL). In contrast to the traditional zero-shot learning setup that assesses performance solely on unseen classes, GZSL extends the evaluation to encompass both seen and unseen classes, removing the unrealistic assumption inherent in standard zero-shot learning that test instances exclusively belong to unseen classes. This augmentation in complexity within GZSL underscores the difficulty of the task. To validate their proposal, they undertake comprehensive experimentation on two widely recognized datasets, namely Caltech-UCSD Birds (CUB) and Animals with Attributes 2 (AWA2). Their results ex-

hibit substantial improvements over existing methodologies. Furthermore, they meticulously scrutinize the caliber of their explanations, which ground the decision-making process of the classifier. They accomplish this through textual and visual instances that exemplify the effectiveness of their explanations in shedding light on the classifier's decisions [76]. This approach bears similarities to the method I am proposing. However, the assumption that a single channel is tied to just one attribute could be a problem because the features are implicitly learned. This could mean that multiple attributes may be associated with a single channel or vice versa.

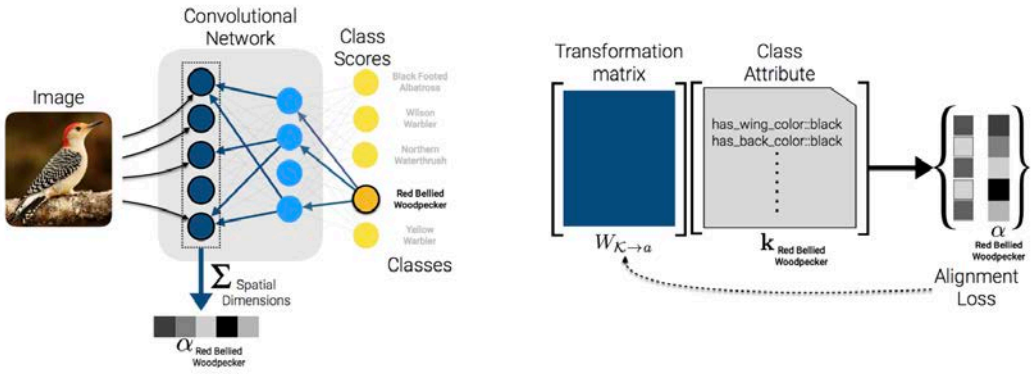

Figure 2.6: An overview of NIWT zero-shot learning approach. Image from Ramprasaath R. Selvaraju and P. Chattopadhyay et al., 2018.

2. **Generating Visual Explanations with Natural Language**

This approach is an interesting technique that combines computer vision and natural language processing to provide textual explanations for image classification. The core of this approach is the explanation mechanism. It operates on the features extracted by a classification algorithm, presumably for image classification tasks. These features are typically high-dimensional representations of images that the classification model uses for making predictions. The Explanation Sampler takes these fine-grained features and generates candidate textual descriptions. Each description corresponds to a specific attribute or aspect of the image. These descriptions likely help in breaking down the decision of the model into different components, making it more interpretable. The candidate textual descriptions are passed through a detection-based Grounding Model. This model's role is to associate each textual description with the corresponding bounding boxes in the image. It essentially aligns the text with the visual components, which

is crucial for creating meaningful explanations. The bounding boxes, along with the textual descriptions, are then evaluated by a Phrase Critic. This component assigns relevant scores to each attribute or description. These scores represent the importance or significance of each attribute in the context of the image. This step is vital for determining which aspects of the image contributed most to the model's decision. One of the limitations is that the importance scores are not directly tied to the decision-making process of the classification model. Instead, they are based on the model's features. This means that the explanation mechanism relies on the features extracted by the classification algorithm but does not directly explain the internal workings of the model or how it arrived at a specific decision. The explanation is more focused on providing insights into what features or attributes were relevant for the decision, rather than explaining the decision-making process itself [77].

3. **A Generalized Explanation Framework for Visualization of Deep Learning Model Predictions**

   This approach is designed to provide a comprehensive explanation of deep learning model predictions, particularly for image classification tasks. The authors introduce a system called GALORE, which consists of three key components: Attributive Explanation, Deliberative Explanation, and Counterfactual Explanation. Attributive explanation aims to provide pixel-wise importance information for the model's predictions, which is similar to Grad-CAM. By offering this pixel-wise information, attributive explanation helps users understand which areas of the image played a significant role in the model's prediction, making the decision more transparent and interpretable. Next, the deliberative explanation addresses the issue of ambiguity in model predictions. It provides a list of image regions that are identified as ambiguous between different classes. This is valuable for understanding why the model might have had difficulty making a clear decision. By highlighting ambiguous regions, it offers insights into the limitations of the model and potential areas for improvement. This component is particularly useful for quality control and model refinement. Counterfactual explanations are crucial for answering the ”why not” question. In other words, they help identify image regions that distinguish between closely similar classes. This can be particularly important in scenarios where similar classes are frequently misclassified. By explaining the differences between these classes, Counterfactual Explanation can help users under-

stand why the model made certain misclassifications. However, it's worth noting that while this component identifies regions that distinguish classes, it doesn't necessarily associate these regions with human-readable attributes. This might be a limitation if the goal is to provide more intuitive explanations. The key strength of the GALORE system is its comprehensive approach to explaining deep learning model predictions. It covers different aspects of explanation, including highlighting important regions, addressing ambiguity, and explaining misclassifications. These explanations can be valuable in various applications, including healthcare, autonomous vehicles, and security, where model interpretability and transparency are critical. However, the system's ability to provide truly human-readable attributes associated with image regions, especially in the case of counterfactual explanations, could be a potential area for further improvement if it's important for the user's understanding and decision-making process [78].

Several papers are working on explaining model decisions with important image regions and corresponding concepts. However, these methods require specially-designed recognition models which are not within this thesis's scope.

2.4 MMAL-Net

A part of my work has to do with extracting every image feature map and using them as inputs in my model (ATT-Net). I extract those activation maps from a model called MMAL-Net which represents Multi-branch and Multi-scale Attention Learning for Fine-Grained Visual Categorization. I chose this pre-trained model because it is one of the state-of-the-art (SOTA) models in fine-grained image classification using the CUB-2011 dataset, with 89.6% test accuracy. So how does MMAL-Net work? The acquired images of the object include not just the fundamental object structure but also capture additional elements, including part images at various scales and finer, intricate features, along with unprocessed images that incorporate the entire object. Their multi-branch network oversees the training of three different types of images, endowing their network with excellent classification capabilities and resilience when it comes to images of varying scales. Their method allows comprehensive training while providing minimal inference duration and learning the object's discriminative regions for recognition effectively. MMAL-Net uses an Attention Object Location Module (AOLM), which can forecast the position of the object. Furthermore, MMAL-Net uses an Attention Part

Proposal Module (APPM), which can suggest informative segment areas without the requirement of a bounding box or part annotations [7]. Three standard benchmark datasets are used to report the performances of the model but the only one I care about is the CUB-2011 dataset which I use in my project.

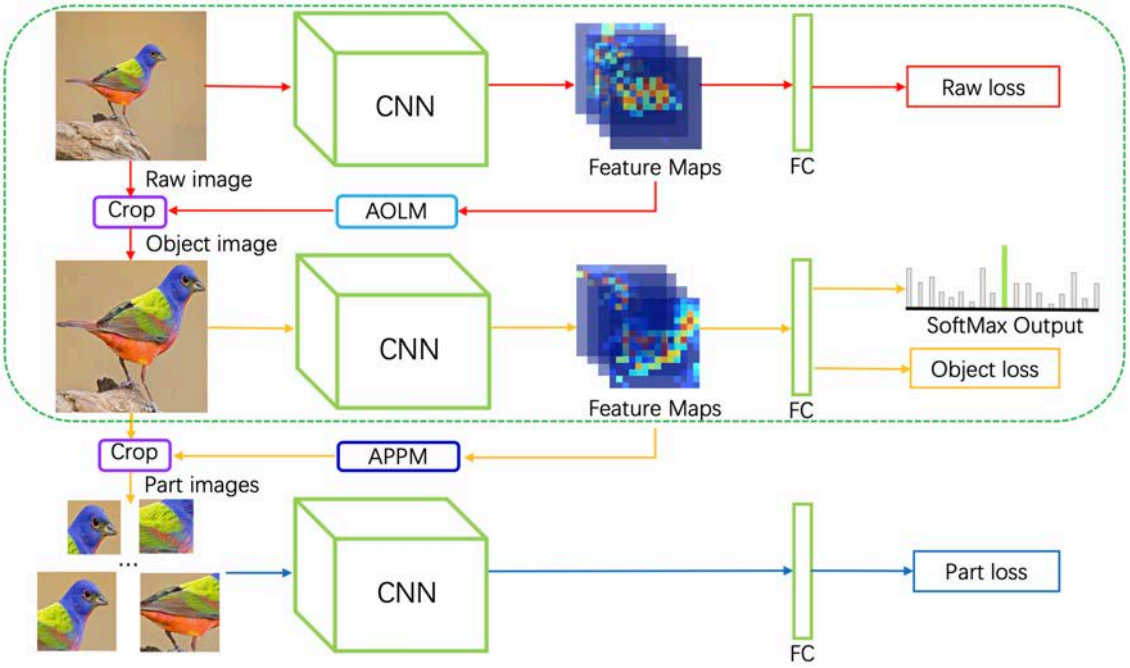

Figure 2.7: An overview of MMAL-Net in the training phase. Image from F. Zhang et al., 2020.

Chapter 3

Dataset

The CUB-200-2011 dataset is a widely recognized and extensively used benchmark for fine-grained image classification tasks in the field of computer vision. It is a collection of high-resolution images, specifically curated for fine-grained bird species classification. Each image is labeled with the corresponding bird species, making it an ideal resource for training and evaluating machine learning models. The dataset covers a diverse range of avian species with a focus on fine-grained visual distinctions among species.

```
1 has_bill_shape::curved_(up_or_down)
2 has_bill_shape::dagger
3 has_bill_shape::hooked
4 has_bill_shape::needle
5 has_bill_shape::hooked_seabird
6 has_bill_shape::spatulate
7 has_bill_shape::all-purpose
8 has_bill_shape::cone
9 has_bill_shape::specialized
10 has_wing_color::blue
11 has_wing_color::brown
12 has_wing_color::iridescent
13 has_wing_color::purple
14 has_wing_color::rufous
15 has_wing_color::grey
16 has_wing_color::yellow
17 has_wing_color::olive
18 has_wing_color::green
19 has_wing_color::pink
```

Figure 3.1: Illustration of the attribute list from CUB-200-2011 dataset.

This dataset comprises 11,788 images, featuring 200 distinct bird classes, with 5,994 im-

ages earmarked for training and 5,794 images for testing purposes. Each image is meticulously annotated, boasting a subcategory label, 15 part locations, 312 binary attributes, and a singular bounding box. The images themselves are of high resolution, exhibiting diverse dimensions that offer a treasure trove of visual data for classification tasks. The CUB-200-2011 dataset offers a comprehensive snapshot of birds in a multitude of poses, encompassing frontal, side, and various angles. This diversity presents a formidable challenge when it comes to differentiating between closely related species. Moreover, the dataset showcases an extensive spectrum of colors and textures displayed by different bird species, presenting both hurdles and opportunities for classification algorithms. It's worth noting that the avian subjects in the dataset may exhibit obstructions, such as branches, leaves, or other objects, adding complexity to the imagery. Additionally, the background complexity fluctuates across images, ranging from unadorned backgrounds to complex natural environments [8].

Figure 3.2: Bird images from CUB-200-2011 dataset.

Chapter 4

Proposed Method

4.1 Method Description

Applying the Grad-CAM algorithm, which is described in the 2.2.6 section, to the MMAL-Net I can get the heatmap from every image. Each heatmap provides the areas that include the most important attributes chosen to do the classification. So, we know these specific regions but these regions might include plenty of different attributes. The goal of my approach is to get to these specific attributes and not the general region of importance. However, I use these heatmaps for later evaluation. My proposed method consists of two steps. First, I use Grad-CAM to detect important regions. Then, I retrieve the attributes that are most important in those regions.

4.1.1 Using Grad-CAM to detect important regions

I take advantage of the Grad-CAM algorithm application on both the MMAL-Net model and ATT-Net. Just like CAM (Class Activation Mapping), Grad-CAM also utilizes the feature maps generated by the final convolutional layer of a Convolutional Neural Network (CNN). However, the distinction lies in how the feature maps, namely A_1, A_2, and A_3, are incorporated to create the ultimate heatmap. In CAM, these feature maps are weighted using coefficients obtained from the fully connected layer at the end of the network. On the other hand, in Grad-CAM, the weighting of these feature maps is determined by ”alpha values,” which are calculated based on gradient information. Consequently, Grad-CAM doesn't depend on a specific network architecture, as it can compute gradients across various types of neural network layers. Grad-CAM is used on a neural network that has completed its training

phase, meaning the neural network's parameters are unchanging. First Grad-CAM computes the gradient of y^c with respect to the feature map activations A^k of a convolutional layer, i.e. $\frac{\partial y^c}{\partial A^k}$, while y^c is the output for class c before the softmax activation function. The gradient's specific value computed in this stage is influenced by the chosen input image, as the input image dictates both the formation of feature maps A^k and the resulting class score y^c. In the context of a 2D input image, the gradient takes on a 3D form that mirrors the dimensions of the feature maps. These feature maps, amounting to k in number, each possess dimensions v in height and u in width. As a result, the composite shape of the feature maps becomes $[k, v, u]$ [9, 10, 11]. Correspondingly, the gradients computed during the initial step will also adopt the same shape of $[k, v, u]$.

After this phase, Grad-CAM computes the alpha values, which will be employed as weights for the upcoming step in the process. These alpha values correspond to the class c and feature map k, and their purpose is to modify the feature map A^k in the subsequent stage. The gradient calculations have a structure of $[k, v, u]$, as mentioned before. So, through applying pooling across the height v and width u dimensions, the outcome is a shape of $[k, 1, 1]$ or just $[k]$. These values represent the set of k alpha weights that I take from the Grad-CAM algorithm to apply to my method.

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{4.1}$$

So, let's say that α weights specifically represent the values after applying Grad-CAM on MMAL-Net. Every bird image has its own set of α_k values. k represents the number of feature maps, which is 2048 for every image. I also get the feature map values across channels at some specific spatial location (x, y). Let's call them S_k values.

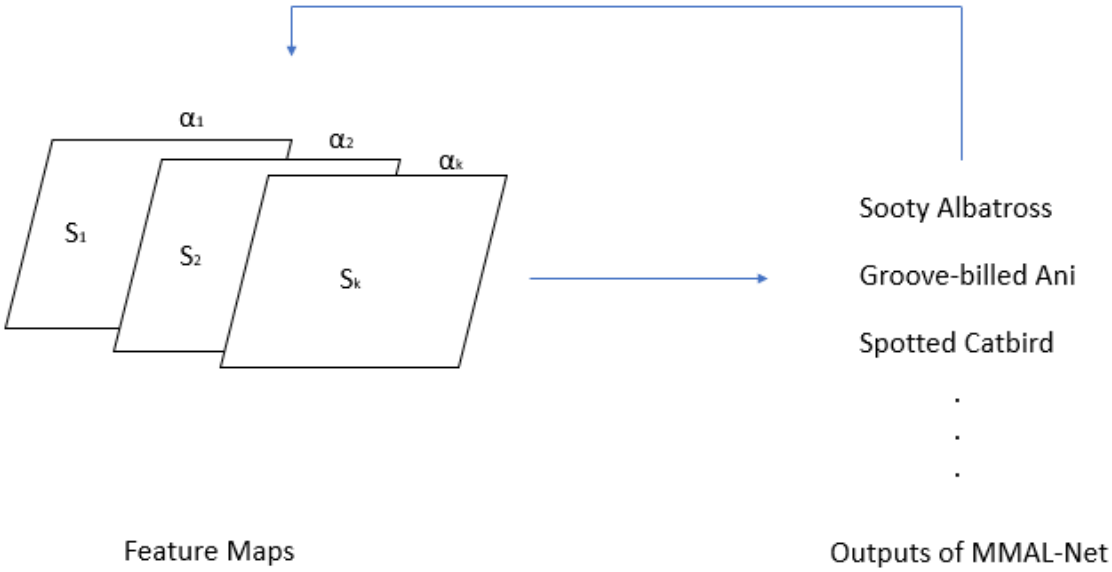

Figure 4.1: Overview of alpha and S values from MMAL-Net network.

4.1.2 Retrieve the important attributes of the important regions

The next step has to do with retrieving the important attributes of the important regions. To do this, I first train an attribute classification model (ATT-Net) which takes as input feature maps from the MMAL model and the output is a n-dim vector that contains the appearance scores of n predefined attributes. Next, I calculate the importance weights of each attribute with respect to the input feature map. I use the Grad-Cam algorithm for my attribute classification model, as I did in subsection 4.1.1 on MMAL-Net, and get the weights β_k, which represent the values that correspond to Grad-Cam weights for every attribute produced by the classification. k represents the number of 2048 channels. For a specific important region (x, y), I define the matching score of each attribute as the multiplication product of values β_k, with their corresponding S_k values, which were calculated at the first step of my method. Every image has 312 attributes, so I get 312 sets of $\beta_k * S_k$ values for every image.

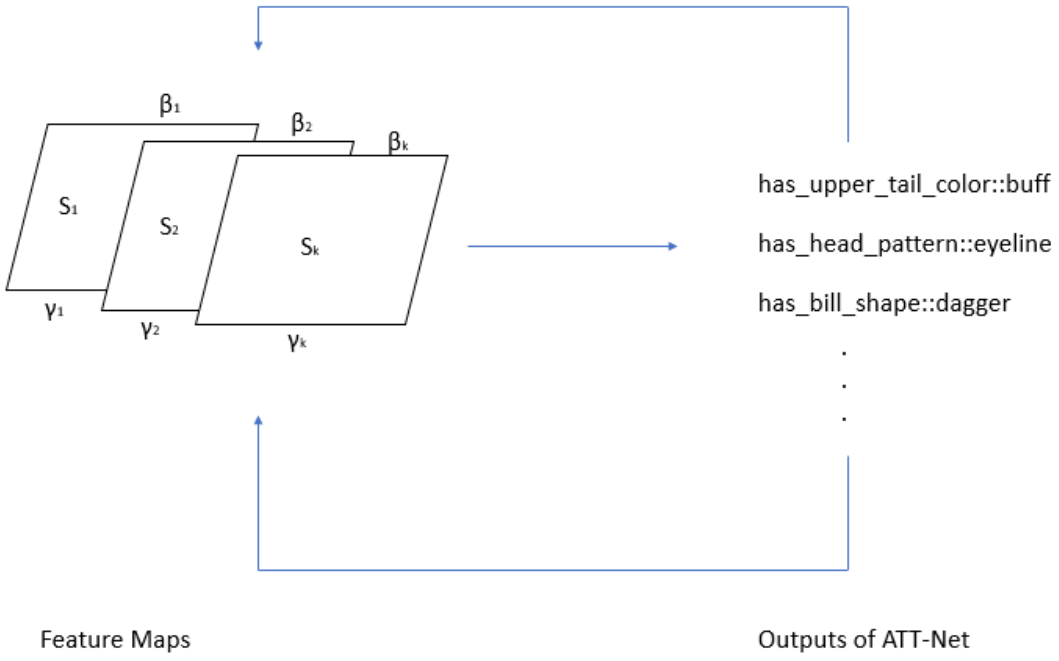

Figure 4.2: Overview of values from ATT-Net network.

To retrieve the most important attributes, I compare every product $\alpha_1 * S_1, \alpha_2 * S_2, \cdots, \alpha_k * S_k$ with all the products $\beta_1 * S_1, \beta_2 * S_2, \cdots, \beta_k * S_k, \gamma_1 * S_1, \gamma_2 * S_2, \cdots, \gamma_k * S_k$, etc. of the ATT-Net separately and rank the distances. The product $[\beta * S], [\gamma * S], [\delta * S]$, etc. which is closer mathematically to the product provided by the MMAL-Net $[\alpha * S]$ gives us the attribute which is the most important for the MMAL-Net model to classify the bird image with the corresponding α values. To calculate this distance I use the Euclidean distance formula.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2} \tag{4.2}$$

where x, y are two points in Euclidean n-space and x_i, y_i are Euclidean vectors. x vector represents the product $\alpha_k * S_k$ for one image and y vector represents the product $\beta_k * S_k$ for a specific attribute [79].

4.2 ATT-Net Model

I created a network to help me implement my method, which I named ATT-Net. It is an attribute classification model with inputs the target layer (chosen feature map) of the image and outputs a vector with the presence weights of the 312 attributes of the CUB-200-2011 dataset. Firstly, I use the MMAL-Net, which is described in section 2.3, to extract the feature maps I need as inputs to the ATT-Net model. These feature maps are layers with feature information in the MMAL-Net model. From every layer of the MMAL-Net, I can get different information about the features of every image. Early layers of the model attach great importance to features such as shape and size and later ones give more information about color, texture, etc. So the only way to be more effective and precise is to choose the target layers with the most information and test them. After trying different layers of the MMAL-Net model I choose to use the last convolutional layer of MMAL-Net, because it has the most information.

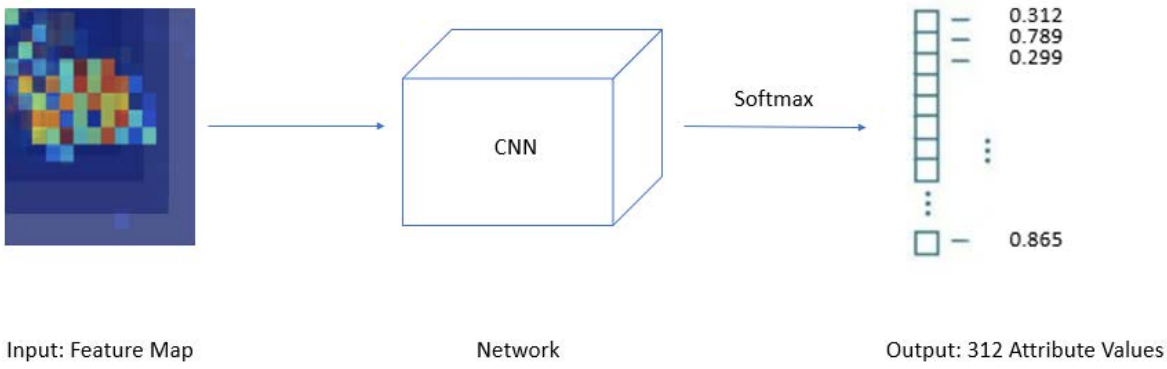

Figure 4.3: Overview of ATT-Net network.

CUB-200-2011 dataset has a list of 312 attributes, which include bird features such as red bill, yellow legs, head pattern with eyebrow, striped back pattern, etc. Also, it provides another list with the attributes that are present on each of the images of the dataset [8]. So I created a new dataset collecting the feature maps I have already extracted from MMAL-Net and a vector showing the presence or not of the 312 attributes for every image with 1 and 0 respectively. The feature maps are the inputs of ATT-Net and as labels I each time use the vector which gives me the presence or not of all the attributes in the image with the specific input feature map. So, the output of my model is a vector showing how much present an attribute is in the specific feature map compared to the other attributes. Following

this, the ATT-Net model undergoes training to grasp the connections between input feature maps and their associated attribute values. Within the model architecture, there's a global average pooling (GAP) layer and a concluding fully connected (FC) layer just before the application of the softmax activation function. For the optimization process, I employ the ADAM optimizer, and for the calculation of the loss I employ the mean squared error (MSE) loss formula as the model's loss function [80].

$$l(x, y) = L = \{l_1, \cdots, l_N\}^T, l_n = (x_n - y_n)^2 \tag{4.3}$$

Chapter 5

Results

5.1 Evaluation

One of the main goals of this project is for the results to be as close to the criterion of a human expert in bird identification. What exactly a human expert would see in a bird picture and what would help him/her with the final identification of the bird species? So, having help from some ornithology experts, for the evaluation of my method I created an attribute list for every class (bird species), which includes the most important attributes that a birdwatcher would choose to make the bird recognition. This list contains from 5 up to 10 attributes for each bird class. A combination of attributes is necessary for an expert to make a correct identification of a bird species. Some birds may need a bigger attribute combination than others, depending on how difficult the identification is with similar species. That's why the number of attributes for each ground truth class might be different.

I used some offline metrics such as Recall, Precision, and especially Mean Average Precision (MAP) to evaluate my method's results in comparison to the bird experts attribute list. I described the precision value in the 2.1.2 subsection and now I move on to the next step of calculating the Average Precision.

$$AP@K = \frac{\sum_{k=1}^{K}(Precision@k * rel_k)}{number\ of\ relevant\ results} \tag{5.1}$$

rel_k parameter is a relevance parameter that is equal to 1 when the k^{th} item is relevant or 0 when it is not. It is relevant when the predicted attribute k can be found inside its respective ground truth attribute list. Then, I calculate precision and rel_k iteratively using $k = [1, \cdots, K]$. In my occasion, K represents the total number of predicted attributes. To calculate the Mean Average Precision (MAP) for all occasions, I divide by the total number of images Q [31, 30].

$$MAP@K = \frac{1}{Q} \sum_{q=1}^{Q} AP@K_q \tag{5.2}$$

As mentioned in the 4.2 section, the output of the Grad-CAM algorithm is a heatmap. Having obtained the alpha values mentioned in the 4.1 section, Grad-CAM proceeds to employ each of these values as the weight assigned to its corresponding feature map. Subsequently, it computes a weighted summation of these feature maps, yielding the ultimate Grad-CAM heatmap. This heatmap is then subjected to a Rectified Linear Unit (ReLU) operation, serving to accentuate exclusively the positive values while converting all negative values to zero [9, 10, 11].

$$L^c_{Grad-CAM} = ReLU(\sum_k a^c_k A^k) \tag{5.3}$$

Having these heatmaps for every bird image, I know the specific region of the important attributes that my method is supposed to show me. So, I was able to do an early validation of my method checking if my method's results belong to the heated area of the Grad-CAM output.

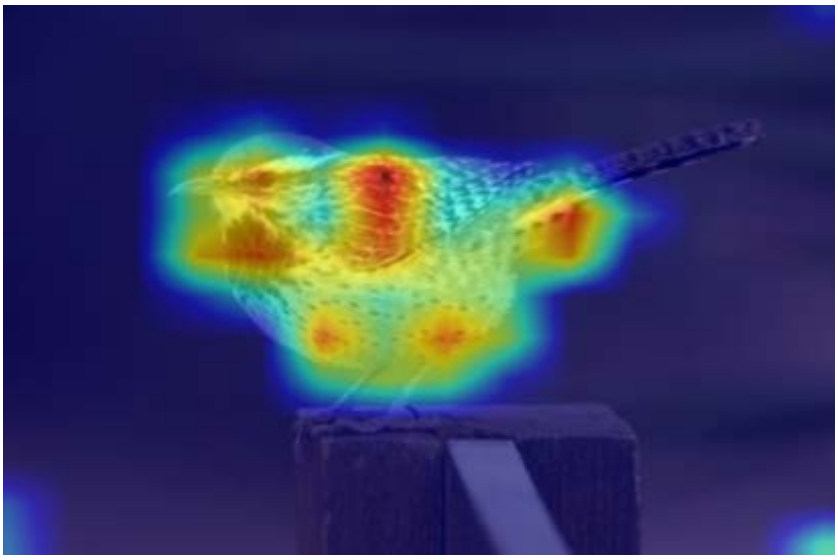

Figure 5.1: Visualization of Grad-CAM output (heatmap) for a Cactus Wren (Campylorhynchus brunneicapillus) image of the CUB-200-2011 dataset.

5.2 Experiments

The first thing I need to check is the results after training the ATT-Net network. To evaluate my training results I use some simple metrics such as accuracy, precision, recall, and F_1 score and save the results with the best F_1 score [31, 30].

$$F_1 = \frac{2 * precision * recall}{precision + recall} \tag{5.4}$$

To calculate which is the best F_1 score possible I tried different combinations of several parameters of my model. First I have to decide which layer of the MMAL-Net will give me the most complete information about features. MMAL-Net has 4 convolutional layers and layer 4 is the layer that I got the most efficient feature maps. Furthermore, I applied different learning rates (lr) and loss weights in my model to conclude that the best learning rate was 0.0001 and the most efficient loss weight was 0.25. For the optimization process, I employed the ADAM optimizer, a well-adopted algorithm for fine-tuning the parameters of machine learning models, particularly in neural networks. ADAM blends the advantages of AdaGrad and RMSProp by fine-tuning learning rates on a per-parameter basis using historical gradient data [81]. I also tried the Stochastic Gradient Descent (SGD) optimizer, which navigates the optimization landscape with a nimble and adaptable approach, seeking the elusive valley of minimal loss [82]. As you will see clearly in the following table, ADAM results are noticeably

better than SGD results. I also did some tests with more convolutional and linear layers in ATT-Net but the results were similar. In the following table, you can see the test f1 scores of my model, while combining all these parameters.

Combinations	Layer2	Layer3	Layer4
Adam/0.0001/0.2	46.879	51.745	51.945
Adam/0.00001/0.2	46.911	51.582	51.876
Adam/0.0001/0.25	48.757	51.789	**52.215**
Adam/0.0001/0.3	47.690	51.421	52.130
Adam/0.00001/0.3	47.509	51.457	52.123
Adam/0.0001/0.35	47.002	51.438	51.798
Adam/0.0001/0.4	45.786	51.230	51.589
SGD/0.0001/0.25	29.985	32.124	32.722
SGD/0.0001/0.35	29.567	31.834	32.001
SGD/0.00001/0.25	30.032	31.932	32.733
SGD/0.00001/0.35	28.975	31.765	31.921

Table 5.1: F1 results for test data after training.

After training and testing the feature maps with all these different parameters, I saved the model with the best test F1 score, which is 52.215%. I followed the same procedure but instead of using feature maps as inputs I used the images and the highest F1 score I could get was 55.351%. That way I could ensure what is the closest F1 score I could reach by trying different combinations of parameters. That F1 score suggests that my model doesn't learn efficiently. However, in models with so many predictions as outputs, the F1 score rarely overcomes 65%. Even though the test accuracy of my model is really high at 90.89%, this metric may not be suitable for my task because of the highly imbalanced custom datasets that I use.

The next step has to do with loading the feature maps from the best model path and applying them to my method. After calculating all the gradient values of importance from my method, I use the Euclidean distance formula, as mentioned in subsection 4.1, and the cosine similarity to calculate the distance between the two products. Cosine similarity is a measure used to determine how similar two non-zero vectors are, particularly in high-dimensional

spaces [79]. Instead of considering the vector's magnitudes, cosine similarity focuses on the angle between them. To test which of the two is more efficient I calculated the average position of the 5 most important attributes from the ground truth attribute list for every class, inside the 312 attribute list of my method's results for each class. As you can notice in the following table my method is more efficient with Euclidean distance.

Ground Truth Attribute	Euclidean Distance	Cosine Similarity
First Attribute	63.3167	89.3456
Second Attribute	76.4870	88.8237
Third Attribute	76.9521	89.2887
Fourth Attribute	80.8923	86.3976
Fifth Attribute	79.4589	86.2337

Table 5.2: Average position for the first five ground truth attributes.

After securing the best results I can get from my method, I use the MAP score, which is explained in detail in subsection 5.1, to compare my method's results with the ground truth attribute list of each class. The best MAP score that I get is 9% for the whole dataset.

My next objective has to do with thresholding the best results. I pass again the feature maps through ATT-Net and get the scores for the attributes. I use different threshold values for my predictions just to exclude the attribute values that are closer to zero. I use the best results file that gives me the information on the sorted attributes based on importance to filter it with the new predictions after thresholding. As a consequence, I get updated result lists for every different threshold value I tried. Then, I use again the MAP score to evaluate these updated results. As you can observe in the following table, every threshold value I try doesn't manage to get the 9% MAP score I already get without using any threshold value on my predictions.

Threshold Values	MAP Score
0	**9%**
0.1	1%
0.3	1%
0.5	0%

Table 5.3: MAP scores for different threshold values.

Thereafter, I represent a visualization of the results from five images of the CUB-200-2011. The left image of each set represents the human expert ground truth of this specific bird class. The central image shows the heated area which represents the region with the most important attributes calculated from the application of the Grad-CAM algorithm on the MMAL-Net classification model. The right image represents the top five predictions of my method on the specific image.

The first three examples represent three successful occasions of my method. In the first image, my method predicts two out of the five most important attributes that a human expert would choose to make the bird identification. In the second image, it predicts three out of five, and in the third image, it predicts two out of five again.

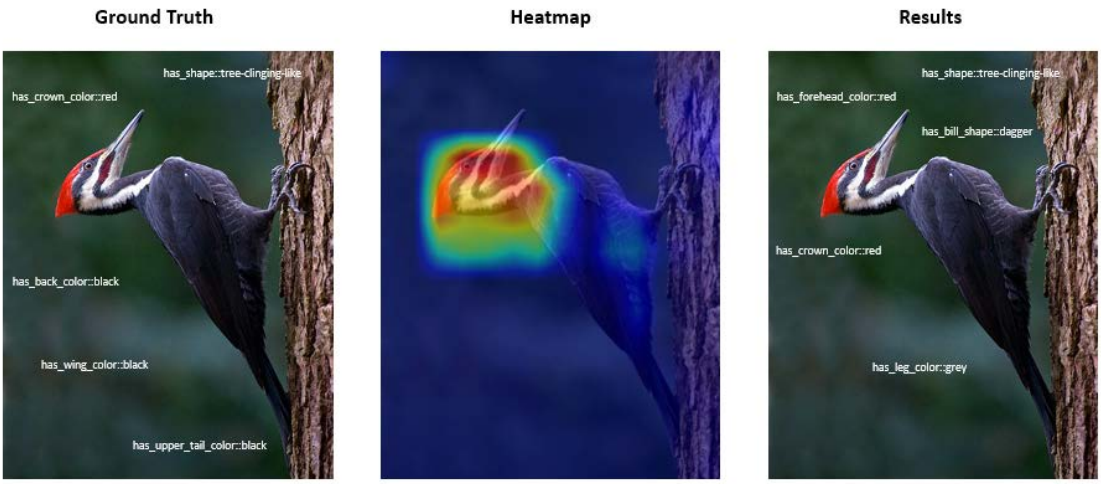

Figure 5.2: Overview of my results on an image of a Pileated Woodpecker (Dryocopus pileatus) from the CUB-200-2011.

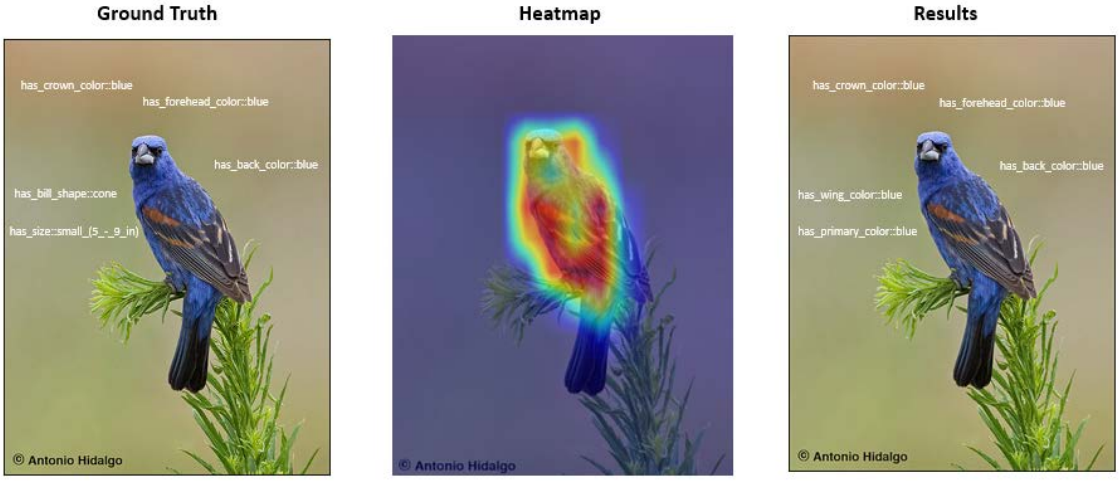

Figure 5.3: Overview of my results on an image of a Blue Grosbeak (Passerina caerulea) from the CUB-200-2011.

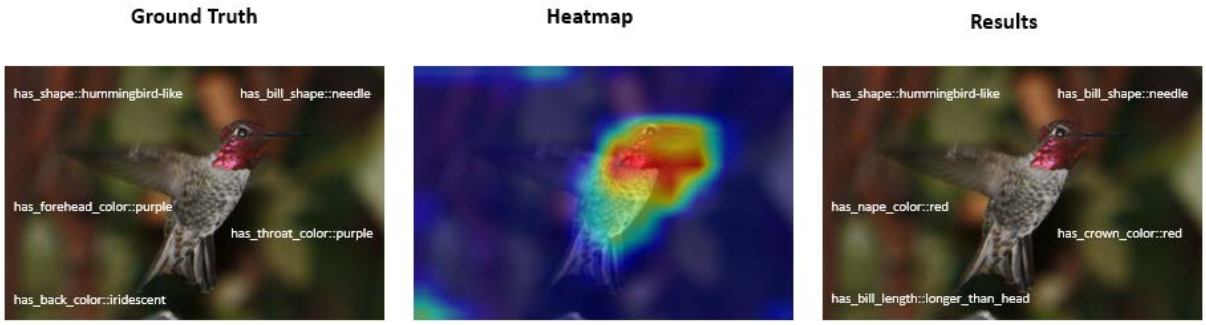

Figure 5.4: Overview of my results on an image of an Anna's Hummingbird (Calypte anna) from the CUB-200-2011.

The last two examples represent two unsuccessful occasions of my method. In these two images, not a single prediction matches the ground truth attributes.

Figure 5.5: Overview of my results on an image of a White-eyed Vireo (Vireo griseus) from the CUB-200-2011.

Figure 5.6: Overview of my results on an image of a Black-billed Cuckoo (Coccyzus erythropthalmus) from the CUB-200-2011.

The 9% MAP score I managed to score highly suggests that my method's performance is limited. This also means that the explanation mechanism relies on how successfully the ATT-Net model can be trained. With 312 attributes as predictions for each output of every

image, the ATT-Net's training phase is a demanding task. Furthermore, it is clear that the attribute prediction performance depends on the feature maps of the bird prediction model. This suggests that the information provided by the feature map may not be as suitable as one might anticipate for predicting attributes in a way that is easily understandable to humans.

As a last step, I examined if some predictions were more explainable than others. To achieve that I wanted to see if the classes with the highest classification accuracy in MMAL-Net have bigger MAP scores after my attribute prediction method than the classes with the lowest classification accuracy. I calculated the MMAL-Net classifier's accuracy for each bird class. I selected the classes with the highest accuracy which was 100% and the classes with the lowest accuracy which was between 33-55%. I calculated the MAP scores for each occasion separately. However, the MAP scores were random, not giving any indication of how explainable the attributes of these classes can be.

MMAL-Net Classes	Accuracy	MAP score
Pileated Woodpecker	100%	30%
Sooty Albatross	100%	19%
Geococcyx	100%	6%
California Gull	33%	21%
Common Tern	50%	31%
Least Flycatcher	55%	4%

Table 5.4: MAP scores for the highest and lowest accuracies of classes from MMAL-Net model.

Chapter 6

Limitations and Future Work

After evaluating the results of my method, it is clear that there are some significant limitations that need to be addressed. What I noticed from my work is that the attribute prediction performance depends on the feature maps of the bird prediction model. The MMAL-Net in my occasion. So, different elaborations of feature maps from many state-of-the-art models for bird classification could give completely different results for my method. That's an indication that the information a feature map gives might not be as optimal as someone would expect for human-readable attribute prediction. Instead, one very optimistic approach would be to focus on explaining the classification decision by natural language, like short very detailed descriptions of a bird species, instead of a set of attributes. This could be a challenging approach but it is promising because it might provide the nearest results to a human-readable attribute prediction.

Chapter 7

Conclusion

Explainability in fine-grained classification tasks remains a serious challenge in machine learning. In this research, I introduce an approach aimed at clarifying the decision-making process of a fine-grained image classification model called MMAL-Net. My method provides insights into the most crucial characteristics utilized by the MMAL-Net network for classifying bird species. The procedure commences by employing the pre-trained MMAL-NET model to extract feature maps from individual images. These feature maps are subsequently used as inputs for an attribute classification model known as ATT-Net. To evaluate the significance of attributes, the Grad-CAM algorithm is utilized to calculate weighted values that indicate the relevance of each attribute to its corresponding feature map. These same weighted values are also extracted from the MMAL-NET network after applying Grad-CAM, which is aligned with bird classification. The pivotal step involves a quantitative analysis aimed at assessing the proximity between the attribute values acquired from ATT-Net and the MMAL-Net values for each image. My predictions are evaluated using as a ground truth an attribute list for every bird class, which includes the most important attributes that an expert in ornithology would choose to make the bird recognition. I used the MAP metric to validate the proximity of my results and managed to produce a MAP score of 9%. Upon reviewing the outcomes of my approach, it becomes evident that there exist notable constraints requiring attention. I observed that the effectiveness of attribute prediction relies heavily on the feature maps generated by the bird prediction model. Therefore, diverse modifications to feature maps sourced from various state-of-the-art bird classification models can yield vastly dissimilar outcomes when applied to my approach. This implies that the information within the feature maps may not align with the optimal expectations for human-readable attribute pre-

diction. My last contribution is the creation of the ground truth list that can be used for human criterion evaluations in similar projects.

Bibliography

[1] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138 – 52160, 2018.

[2] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv: Machine Learning*, 2017.

[3] M. A. Turk and A. P. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 1991.

[4] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *International Conference on Learning Representations (ICLR)*, 2014.

[6] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM Computing Surveys (CSUR)*, 51(5):1–42, 2018.

[7] F. Zhang, M. Li, G. Zhai, and Y. Liu. Multi-branch and multi-scale attention learning for fine-grained visual categorization (mmal net). *MultiMedia Modeling. MMM 2021. Lecture Notes in Computer Science*, 12572, 2021.

[8] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. *California Institute of Technology*, 2011.

[9] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[10] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[11] R. R. Selvaraju, V. K. Poornima, N. Sadeep, and M. Varun. Smooth-grad-cam++: An enhanced inference for visual explanations of deep convolutional networks. *Journal of Visual Communication and Image Representation*, 2021.

[12] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. *In Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

[13] T. L. Berg, C. Liu, S. Woo Lee, and M. L. Alexander. Birdsnap: Large-scale fine-grained visual categorization of birds. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[14] J. Plested and T. Gedeon. Deep transfer learning for image classification: a survey. *ArXiv, abs/2205.09904*, 2022.

[15] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *NeurIPS*, 2015.

[16] M. Jaderberg, K. Simonyan, and A. Zisserman. Spatial transformer networks. *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and Ł. Kaiser. Attention is all you need. *NeurIPS*, 2017.

[18] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. *CVPR*, 2018.

[19] J. Yin, C. Qi, W. Huang, Q. Chen, and J. Qu. Multibranch 3d-dense attention network for hyperspectral image classification. *IEEE*, 2022.

[20] J. Kim, J. Jun, and B. Zhang. Bilinear attention networks. *NeurIPS)*, 2018.

[21] Y. Chen, L. Chen, Y. Ma, Z. Liu, and M. Yang. Dual attention network for scene segmentation. *CVPR*, 2019.

[22] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. *Human Language Technologies (NAACL-HLT)*, 2016.

[23] S. Ren and et al. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 2015.

[24] A. Marino and et al. Learning cross-modal embeddings for cooking recipes and food images. *ICCV*, 2019.

[25] E. Hoffer and N. Ailon. Deep metric learning using triplet network. *International Workshop on Similarity-Based Pattern Recognition*, 2015.

[26] Y. Movshovitz-Attias and D. Mahajan. No fuss distance metric learning using proxies. *ICML*, 2017.

[27] S. Kim, D. Kim, M. Cho, and S. Kwak. Proxy anchor loss for deep metric learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3235–3244, 2020.

[28] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. 2015.

[29] T. G. Dietterich. Ensemble methods in machine learning. *International Workshop on Multiple Classifier Systems*, 2000.

[30] M. Sokolova and G. Lapalme. A systematic analysis of performance measures for classification tasks. *Inf. Process. Manag.*, 45:427–437, 2009.

[31] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. *Proceedings of the 23rd international conference on Machine learning*, 2006.

[32] N. Burkart and M. F. Huber. A survey on the explainability of supervised machine learning. *ArXiv, abs/2011.07876*, 2021.

[33] T. Wang, C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl, and P. MacNeille. Or's of and's for interpretable classification, with application to context-aware recommender systems. 2015b.

[34] H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable decision sets: A joint framework for description and prediction. *In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: ACM*, 2016.

[35] H. Lakkaraju, E. Kamar, R. Caruana, and J. Leskovec. Interpretable explorable approximations of black box models. *ArXiv, abs/1707.01154*, 2017.

[36] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan. Building interpretable classifiers with rules using bayesian analysis. *Department of Statistics Technical Report 609, University of Washington*, 2012.

[37] E. Charniak. Bayesian networks without tears. *AI Magazine*, 12:50–63, 1991.

[38] A. Holzinger, M. Plass, M. Kickmeier-Rust, K. Holzinger, G. C. Crisan, C.M. Pintea, and V. Palade. Interactive machine learning: experimental evidence for the human in the algorithmic loop. *Applied Intelligence*, 49(7):2401–2414, 2019c.

[39] L. Breiman, Jerome H. Friedman, Richard A. Olshen, and C. J. Stone. Classification and regression trees. *Biometrics*, 40:874, 1984.

[40] M. Craven and J. W. Shavlik. Extracting tree-structured representations of trained networks. *Advances in neural information processing systems*, 1996.

[41] J. R. Quinlan. Bagging, boosting, and c4.5. *AAAI/IAAI*, 1, 1996.

[42] J. R. Quinlan. C4.5: programs for machine learning. *Elsevier*, 2014.

[43] B. Ustun and C. Rudin. Supersparse linear integer models for optimized medical scoring systems. *Elsevier*, 2016.

[44] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11:63–90, 1993.

[45] W. Cohen. Fast effective rule induction. *Machine Learning Proceedings*, 1995.

[46] D. Martens, B. Baesens, and T. V. Gestel. Decompositional rule extraction from support vector machines by active learning. *IEEE Transactions on Knowledge and Data Engineering*, 2009.

[47] D. Martens, M. D. Backer, R. Haesen, J. Vanthienen, M. Snoeck, and B. Baesens. Classification with ant colony optimization. *IEEE Transactions on Evolutionary Computation*, 2007a.

[48] R. Setiono, B. Baesens, and C. Mues. Recursive neural network rule extraction for data with mixed attributes. *IEEE Transactions on Neural Networks*, 2008.

[49] X. Lin and D. Zhang. Inference in generalized additive mixed models by using smoothing splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(2):381–400, 1999.

[50] J. Wang, R. Fujimaki, and Y. Motohashi. Trading interpretability for accuracy: Oblique treed sparse additive models. *In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: ACM*, 2015a.

[51] R. Balestriero. Neural decision trees. *ArXiv, abs/1702.07360*, 2017.

[52] F. Wang and C. Rudin. Falling rule lists. *ArXiv, abs/1411.5899*, 2014.

[53] T. Wang, C. Rudin, F. Velez-Doshi, Y. Liu, E. Klampfl, and P. MacNeille. Bayesian rule sets for interpretable classification. *Data Mining (ICDM), 16th International Conference on: IEEE*, 2016.

[54] O. Bastani, C. Kim, and H. Bastani. Interpreting blackbox models via model extraction. *ArXiv, abs/1705.08504*, 2017.

[55] C. Yang, A. Rangarajan, and S. Ranka. Global model interpretation via recursive partitioning. *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 2018a.

[56] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should i trust you?: Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining: ACM*, 2016c.

[57] M. Mashayekhi and R. Gras. Rule extraction from decision trees ensembles: New algorithms based on heuristic search and sparse group lasso methods. *International Journal of Information Technology Decision Making*, 2017.

[58] M. T. Ribeiro, S. Singh, and C. Guestrin. Model-agnostic interpretability of machine learning. *ArXiv, abs/1606.05386*, 2016a.

[59] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should i trust you?: Explaining the predictions of any classifier. *In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: ACM*, 2016b.

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

[61] G. Plumb, D. Molitor, and A. S. Talwalkar. Model agnostic supervised local explanations. *Advances in Neural Information Processing Systems*, 2018.

[62] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti. Local rule-based explanations of black box decision systems. *Advances in Neural Information Processing Systems*, 2018a.

[63] E. Strumbelj, Z. Bosnic, I. Kononenko, B. Zakotnik, and C. Kuhar. Explanation and reliability of prediction models: the case of breast cancer recurrence. *Knowledge and information systems*, 2010.

[64] P. Cortez and M. J. Embrechts. Opening black box data mining models using sensitivity analysis. *In Computational Intelligence and Data Mining (CIDM)*, 2011.

[65] A. Henelius, K. Puolamaki, H. Bostrom, L. Asker, and P. Papapetrou. A peek into the black box: exploring classifiers by randomization. *Data mining and knowledge discovery*, 2014.

[66] P. Adler, C. Falk, S. Friedler, G. Rybeck, C. Scheidegger, and B. Smith andS. Venkatasubramanian. Auditing black-box models for indirect influence. *In Data Mining (ICDM), IEEE 16th International Conference on: IEEE*, 2016.

[67] A. Henelius, K. Puolamaki, and A. Ukkonen. Interpreting classifiers through attribute interactions in datasets. *ICML Workshop on Human Interpretability in Machine Learning (WHI)*, 2017.

[68] X. Zhao, Y. Wu, D. L. Lee, and W. Cui. iforest: Interpreting random forests via visual analytics. *IEEE transactions on visualization and computer graphics*, 2019.

[69] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and M. Zoeller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 2010.

[70] P. Hall, N. Gill, M. Kurka, and W. Phan. Machine learning interpretability with h2o driverless ai. 2017a.

[71] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 2018.

[72] A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 2015.

[73] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *CoRR, abs/1409.0473*, 2014.

[74] A. M. Rush, S. Chopra, and J. Weston. A neural attention model for abstractive sentence summarization. *Conference on Empirical Methods in Natural Language Processing*, 2015.

[75] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *In International conference on machine learning*, 2015a.

[76] M. Elhoseiny T. Sharma D. Batra D. Parikh S. Lee R. R. Selvaraju, P. Chattopadhyay. Choose your neuron: Incorporating domain knowledge through neuron-importance. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.

[77] L. A. Hendricks, A. Rohrbach, B. Schiele, T. Darrell, and Z. Akata. Generating visual explanations with natural language. *Applied AI Letters 2021*, 2021.

[78] P. Wang and N. Vasconcelos. A generalized explanation framework for visualization of deep learning model predictions. *IEEE Transactions on pattern Analysis and Machine Intelligence*, 2023.

[79] P. Danielsson. Computer graphics and image processing. 14:227–248, 1980.

[80] R. M. Neal. Pattern recognition and machine learning. *Pattern Recognition and Machine Learning*, 2006.

[81] D. P. Kingma and J. L. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2014.

[82] S. Sra, S. Nowozin, and S. J. Wright. Optimization for machine learning. *The MIT Press*, 2012.