Explainability Analysis of Isolated Receptive Field Sub-Networks in Neural Networks

PERNA Luca, STROUK Timothée, LEDUC Galdwin

March 2025

Abstract

Understanding how Convolutional Neural Networks (CNNs) make classification is crucial to improve transparency and trust in AI models, particularly in sensitive domains like healthcare or autonomous systems. This work aims to investigate the role of receptive fields in CNNs and their potential for explainability.

Our methodology involves training a simple CNN on an image classification dataset, extracting neurons associated with receptive fields, and applying explainability techniques such as Grad-CAM and saliency Maps to assess their interpretability.

Through these experiments, we intend to analyze whether these neurons encode meaningful class-specific information and can therefore be regarded as independent sub-networks. Ultimately, our findings contribute to the broader goal of improving neural network interpretability by highlighting the utility of receptive field–based sub-networks for explainability and model analysis.

Github repository : Github Link

1 Introduction

Convolutional Neural Networks (CNNs) have established themselves as the cornerstone of various computer vision tasks, with a series of CNN-based architectures ([He+15; SZ15; KSH12]) being proposed in recent years. Despite their impressive performance, CNNs remain mostly opaque, often described as black-box models due to their limited interpretability. This lack of transparency has motivated the development of numerous explainability methods aimed at uncovering the internal mechanisms of CNNs and shedding light on how they make specific decisions.

Explainability techniques such as saliency maps, feature attribution, and concept-based methods ([SVZ14; San+24]) have significantly advanced our understanding of how CNNs detect and represent semantic concepts. These tools have revealed that deep models often rely on spatially localized features, and certain neurons appear to specialize in detecting high-level patterns corresponding to interpretable concepts. Building on these findings, we hypothesize that CNNs internally associate specific concepts to specific classes and that only a subset of these concepts would be sufficient to make accurate classification.

In this work, we intend to test this hypothesis by focusing on effective receptive fields [Luo+17]—the localized input regions that individual neurons or groups of neurons are responsive to. Rather than using receptive field visualizations solely for interpretation, we analyze their functional role in classification. We propose to decompose a trained CNN into subsets of neurons corresponding to distinct receptive fields. Then, we will evaluate whether these isolated subcomponents retain class-discriminative power with saliency maps and GradCam [Sel+19a; SVZ14]. This approach allows us to explore whether receptive fields can function independently and how their performance measures up to the full model.

By evaluating the classification capabilities of receptive field-based subnetworks, we seek to deepen the understanding of the internal structure of CNNs and provide further insight into the distribution and redundancy of information across the network. Ultimately, this study contributes to the broader goal of developing more interpretable and modular deep learning systems.

2 Problem definition

2.1 Definitions and Notations

Let $f : X \to Y$ be a Convolutional Neural Network (CNN) model, where X represents the input image space and Y denotes the set of possible class labels. Each convolutional layer in the network consists of a set of feature maps $\{M_1, M_2, \ldots, M_k\}$, where M_i represents the output of a convolutional filter applied to the input.

The **receptive field** of a neuron at spatial location (i, j) in layer l is defined as the subset of the input space that influences its activation, denoted as:

$$R_{i,j}^{(l)} \subseteq X. \tag{1}$$

Our objective is to extract the neurons associated with a given receptive field and construct a sub-network S, formally defined as:

$$S = \{N_k \mid N_k \in R_{i,j}^{(l)}\}, \tag{2}$$

where N_k are the neurons contributing to the receptive field $R_{i,j}^{(l)}$.
The key hypothesis is that S retains class-relevant information and can be analyzed independently using explainability techniques such as Grad-CAM and saliency maps.

2.2 Objective

Our primary goal is to **analyze the interpretability** of CNNs by isolating receptive field sub-networks while ensuring they retain classification capability. We will use several interpretability metrics to quantify how well the extracted sub-network S retains class information compared to the original CNN f:

- **Concept alignment**: measuring how well the receptive field sub-network captures human-interpretable features.
- **Feature importance**: assessing the contribution of the extracted neurons to classification performance.
- **Classification accuracy**: evaluating the ability of S to predict Y independently.

A secondary objective is to evaluate whether these extracted sub-networks can be transferred across different CNN architectures to support model simplification and enable effective knowledge distillation.

2.3 Computational Complexity and Hardness

Let's now outline the key computational challenges:

- **Combinatorial Explosion:** Extracting the optimal sub-network requires searching through all neuron subsets, leading to an exponential search space.
- **Non-Linearity:** CNN activations are non-linear functions of input pixels, which makes analytically modeling neuron importance challenging.
- **Dependency on Network Depth:** The receptive field spans multiple layers, increasing the complexity of isolating relevant neurons while preserving semantic meaning.

Given the intractability of finding an exact solution, we instead turn to heuristic and approximation methods to address these challenges::

- **Gradient-Based Attribution:** Techniques such as Grad-CAM and saliency maps provide efficient neuron selection without exhaustive search.
- **Layer-Wise Pruning:** Progressive neuron elimination at each layer reduces the search space while maintaining network performance.
- **Knowledge Distillation:** Transferring knowledge from the extracted sub-networks to simpler architectures enables efficient reuse in other models.

Thus, by leveraging these techniques, we aim to make the sub-network extraction problem computationally feasible while preserving interpretability and classification performance.

3 Related Work

3.1 Receptive Field

The notion of the receptive field has long been central to the design and understanding of convolutional neural networks (CNNs). Initially defined as the region in the input space that influences a particular neuron's activation, the classical receptive field describes the idealized area covered by the convolutional kernel over multiple layers. However, recent studies have refined this concept by introducing the effective receptive field, which quantitatively demonstrates that not all regions within the classical receptive field contribute equally to the final response. Early work in this area [Luo+17] established the foundation by empirically showing the Gaussian-like distribution of influence across the receptive field. More recent numerical investigations [Jia+24] have expanded on this by detailing the relationship between architectural parameters and the effective receptive field.

A work that introduced the idea of using multiple Receptive field (RF) is the MoRF (Mixture of Receptive Field) [LYW24], which employs a dynamic neural network that uses attention mechanisms to select the most suitable receptive field from a predefined set. Extending the idea of using receptive field separately, the concept of decomposition and re-composition has been put forward to explicitly dissect a network into functional modules and understand how the receptive field properties can be recombined to achieve complex behaviors. The work by [KKS21] exemplifies this approach, offering both theoretical insights and empirical validations that help bridge the gap between network structure and receptive field behavior.

3.2 Visualization techniques

To better interpret the internal behavior of convolutional networks, a range of visualization techniques have been proposed. One of the most influential approaches is GradCAM [Sel+19a], which utilizes gradient information flowing into the final convolutional layer to produce visual explanations that highlight important regions in the input image. This method has become instrumental in both debugging and interpreting deep models by providing coarse localization maps that reveal class-discriminative regions.

Another widely studied visualization method involves the use of saliency maps, as explored in the pioneering work by ([SVZ14]). These maps indicate the sensitivity of the network's output with respect to small perturbations in the input, thus shedding light on the pixels or regions of the input that exert the strongest influence on the network's outputs. Alongside these techniques, recent research has started to explore higher-level conceptual visualization, sometimes referred to as concepts visualization. Although this area is still emerging, initial studies have shown promise in linking abstract network features with inter-

pretable semantic concepts, thereby extending the utility of visualization beyond pixel-level interpretations. The following work [San+24] also contributes to this growing body of research, further reinforcing the role of visualization techniques in clarifying the decision-making process of deep networks.

4 Methodology

This study begins its analysis by examining the explainability of CNNs through the pruning of inactive neurons, investigating how this affects both classification accuracy and interpretability. To assess these effects, it evaluates changes in performance and employs visualization techniques, here saliency maps and Grad-CAM, to illustrate model behavior.

4.1 Data Collection

We used the **Ship classification dataset** curated by Oleksander Schevchenko, available on Kaggle [Sch24], based itself on [Sha21]. It consists of images representing 10 distinct types of maritime vessels : Aircraft Carrier, Bulkers, Cargo Ship, Container Ship, Cruise Ship, General Cargo Ship, Sailboat, Submarine, Tanker and Trawler.

The dataset presents a real challenge due to the variability in image resolution, lighting conditions, viewing angles and backgrounds. It contains approximately 8000 training images and 800 test images. For our experiments, we standardized the images by resizing them to a uniform resolution of 188x188 pixels.

4.2 Original Model Architecture

The baseline CNN model, denoted as f, was constructed using PyTorch. Its architecture draws inspiration from standard CNN designs for image classification, comprising of 3 convolutional blocks, each consisting of a convolutional layer, a ReLU activation layer, and a pooling layer. This is followed by two fully connected layers, with a ReLU layer in-between. This original model reached an accuracy of 92% on the test dataset.

4.3 CNN Pruning

The core of our approach involves extracting relevant sub-networks from the original trained CNN. We achieved this by pruning neurons based on their activation patterns when processing a specific input stimulus, representative of a targeted class. This process aims to identify and retain the network neurons that are actively engaged by that stimulus, approximating a functionally relevant sub-network for that class. The implementation details are encapsulated in the PrunedCNN class and its helper function, *get_active_neurons()*.

4.3.1 Pruning methodology

There are two ways to prune a model based on a class C.
The first method identifies active neurons based on the network's response to a single input image, selected as a representative for the class of interest.

1. **Neuron Activation Identification** : We determine which neurons are active by checking whether their absolute activation sum across the spatial dimensions is greater than 0.

2. **Layer Pruning** : Using the identified active neurons, a new CNN called PrunedCNN is constructed, preserving only the active neurons in each convolutional layer. The weights and biases of active neurons are copied from the original model to maintain learned features.

3. **Fully connected Layer Adjustment** : A dummy forward pass is performed to determine the input size of the first fully connected layer after pruning. The modified architecture retains only the necessary dimensions. This does mean that the fully connected layers are randomized, and it will be one of the reasons for the model's reduced performances.

The second method uses all images from a certain class C to guide the pruning process. The following steps differ from the previous implementation:

1. **Neuron Activation Identification** : For each image, we keep in memory the output of each activation layers' neurons. We do not immediately delete the associated convolutional neurons.

2. **Layer Pruning** : We take the mean of each stored value, and we define a threshold (we used 0.05 for all 3 conv layers). If the mean is under the threshold, the corresponding convolutional layer is deleted.

Since both methods yield similar results, we focus on the second implementation, as it provides a more rigorous basis by using all class C images.

4.3.2 Important differences between Pruned and original model

It is important to notice that the newly created CNN does not have the same output form as the original model. While the original CNN gave to each class a score, and the prediction corresponds to the class with the highest score, the newly created model only evaluates the probability that the input image is of the same class that he was trained upon. Using a sigmoid function, this pruned CNN outputs either 1 if he predicts that the image is of the said class, 0 otherwise.
This difference requires careful consideration when comparing the performance of the original and pruned models. While the original model's accuracy reflects its ability to distinguish among all 10 classes, the pruned model's accuracy evaluates its performance on a binary task, determining whether an input belongs to the target class it was pruned for or to any other class.

4.4 Explainability methods

To visualize and compare the internal reasoning of the original model f and the pruned sub-networks S, we employ two standard gradient-based explainability techniques:

1. **Saliency Maps** : Based on the work of Simonyan et al. [SZ15] saliency maps highlight input pixels that most significantly influence the model's output score for a particular class. We compute the gradient of the target output score (the score for the predicted class in the original model f, or the single output score for the pruned model S) with respect to the input image pixels. The absolute value of these gradients is visualized as a heatmap, indicating pixel importance.

2. **GradCAM** : As proposed by Selvaraju et al. [Sel+19b], Grad-CAM produces localization maps highlighting important image regions for a specific class prediction. It utilizes the gradients flowing into the final convolutional layer (conv3 in our architecture) to weigh the corresponding activation maps. The weighted sum, followed by a ReLU operation, forms a heatmap indicating the discriminative regions used by the network for its decision. This method is going to give particularly interesting results because it does not use the fully connected layers for the heatmap creation. Therefore the fact that they are randomly initialized is not going to matter.

These techniques are applied to both the original model f and each generated PrunedCNN S. For f, we target the output neuron of the predicted class. For S, we target its single output neuron. The resulting heatmaps are overlaid onto the input images for qualitative comparison.

5 Evaluation

5.1 Number of Pruned Neurons

As an introduction, we can look at the number of pruned neurons in each layer to gain insight into the concepts retained by the pruned model.

	Original Model	Pruned Model
Conv1 size	32 hidden neurons	32 hidden neurons
Conv2 size	64 hidden neurons	57 hidden neurons
Conv3 size	128 hidden neurons	72 hidden neurons

Number of Neurons in each layer

As observed above, no neuron were pruned from the first layer, which was expected, as this layer captures general information about shapes and lines. However, the last layer,which corresponds to more specific representations, experienced significant pruning (almost 44% removed). Based on this, it appears the model has likely lost specific information that could be important for distinguishing between classes other than the one it was pruned for.

5.2 Class-relative Accuracy

To assess the functional viability of the extracted sub-networks, we compare their classification capabilities to the original model.

- **Original Model (f) Accuracy:** We measure the standard multi-class classification accuracy of the fully trained model f on the test set. This represents the baseline performance, indicating the model's ability to distinguish between all 10 ship classes. Per-class accuracy is also reported to understand performance on individual categories.
- **Pruned model S_C Binary Accuracy:** For each pruned model S_C (pruned for class C), we evaluate its performance on the binary task of distinguishing class C from all other classes (Not C). We use the entire test set for this evaluation. The primary metrics reported are:
  - **Binary Accuracy:** The overall percentage of correctly classified test images.
  - **Precision:** The proportion of images predicted as class C that actually belong to class C, $Precision = \frac{TP}{TP+FP}$.
  - **Recall:** The proportion of actual class C images that were correctly identified, $Recall = \frac{TP}{TP+FN}$.

– **F1-Score:** The harmonic mean of precision and recall, $F_1 = \frac{2 \cdot (\text{Precision} \cdot \text{Recall})}{\text{Precision} + \text{Recall}}$.

Here are the results we obtained for a model pruned on class 5.

Class	Original model	Pruned model
0	0.9634	0.7323
1	0.7604	0.6116
2	0.9955	0.4430
3	0.9087	0.4796
4	0.9280	0.7468
5	0.9340	0.8065
6	0.8864	0.5500
7	0.9403	0.7327
8	0.9763	0.7105
9	0.9539	0.7123
Global average	0.9249	0.6525

Table 1: Accuracies Comparison, Pruned on Class 5

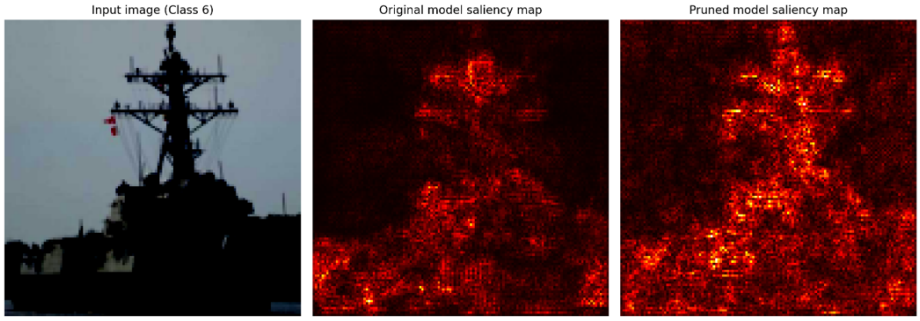
As shown above, the performance of our model is significantly reduced, which was expected. However, there are some notable observations. First, the class with the highest pruned accuracy is the one on which the model was pruned for (class 5). While still lower than the original model's performance, it remains a strong detector for class 5. Additionally, several other classes, such as 0, 4, and 7, continue to show decent performance. These higher scores likely stem from the similarities in the shapes of boats across these classes, suggesting that the pruned model retains useful data for multiple classes. Finally, it is important to remember that the fully connected layers are initialized randomly, making these results even more promising.

5.3 Explainability Map Analysis

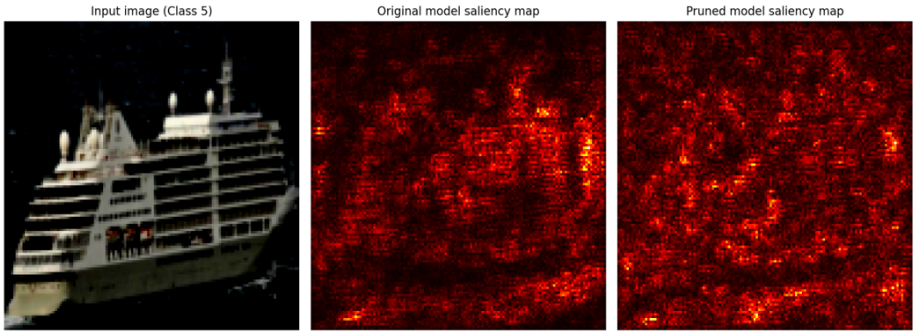
We used Saliency Maps and Grad-CAM to qualitatively assess whether the pruned models S_C retain meaningful, class-specific focus compared to the original model f.

5.3.1 Qualitative Comparison : Saliency Maps

We generate visualizations for both f and S_C on the same set of test images. We will present images from the class the Pruned model was trained on, and others from different classes.



Comparison of Saliency Maps on class 6 for model S_6



Comparison of Saliency Maps on class 5 for model S_6

As seen here, the pruning seems to accentuate the important parts of the image on the class it was pruned on class 6. There is a bit more noise, but that is nothing compared to the noise's importance on images not from class 6 (here class 5). The pruned model retains directional information and general shapes, but lacks representation of the whole boat shape. This tends to affirm our hypothesis in 5.2 about loss of class-specific information.

5.3.2 Qualitative Analysis : GradCAM

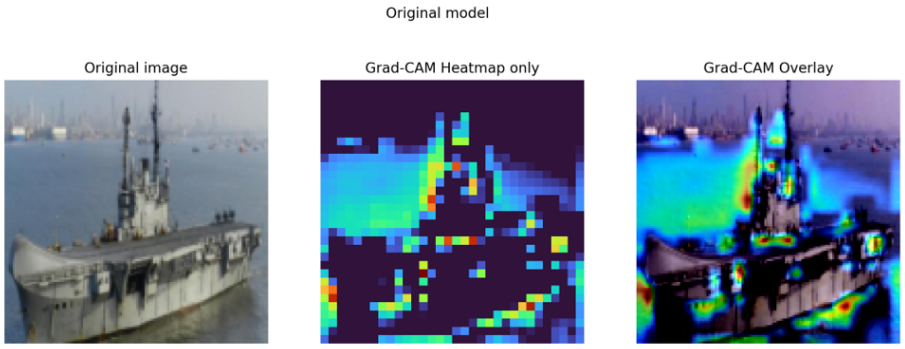


Figure 1: GradCAM Original Model on Class C

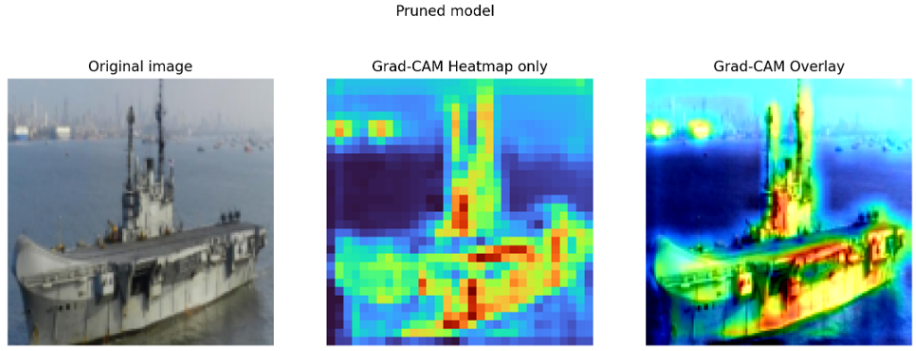


Figure 2: GradCAM Pruned Model on class C

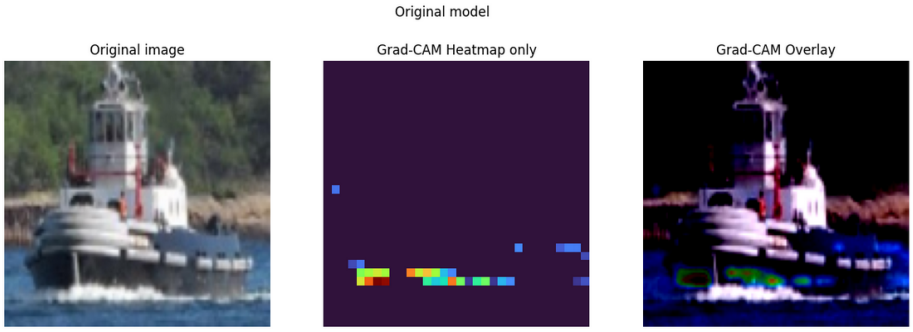


Figure 3: GradCAM Original Model on Class 9

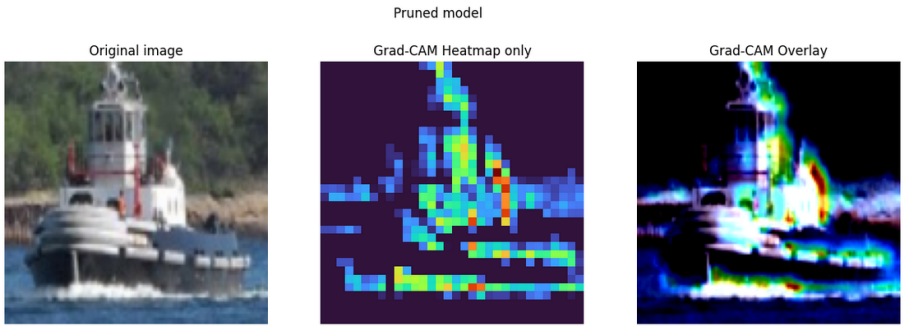


Figure 4: GradCAM Pruned Model on class 9

We observe two kind of behavior of the GradCam when pruning and extracting a sub network from the model.

The first behavior, illustrated in Figures 1 and 2, shows that the regions the model focuses on are complementary. This suggests that the receptive field is encoding class-related information. However, in some parts of the original network, the receptive field must be deactivated, highlighting an interesting aspect of how our effective receptive field correctly identifies key image regions.

The image is from class C.

The second behavior is demonstrated in Figures 3 and 4. In Figure 3, the original model focuses on a very specific and localized part of the image to make its classification. Intuitively, this small region should not provide enough information to determine the type of boat. Yet, in Figure 4, the receptive field becomes significantly larger and is concentrated on a clearly identifiable part of the boat, showcasing a more comprehensive focus. The image is not from class 9.

6 Conclusion

This project explored the use of activation-based pruning, guided by single-class exemplars, to extract functionally relevant sub-networks from a trained CNN, approximating the components tied to class-specific effective receptive fields. The goal was to assess whether these pruned sub-networks retain class-discriminative information and to examine their potential for advancing model interpretability.

By constructing pruned models (PrunedCNN or S_C) for various ship classes based on neuron activations from representative images, we compared their binary classification performance and explainability (using Saliency Maps and Grad-CAM) with that of the original multi-class model (f).
Our findings indicate that while performance drops significantly, the explainability maps reveal that pruned models often maintain focus on semantically relevant object parts, suggesting partial concept retention.

This work highlights both the potential and the challenges of decomposing CNNs into smaller and functionally relevant units that may offer greater interpretability. Further exploration of the connection between these empirically derived sub-networks and theoretical constructs, such as circuits or modules, could provide deeper insights into neural network functionality. Such understanding could pave the way for more interpretable and explainable AI systems.

References

[1] Otsu, N. Automatic threshold selection based on discriminant and least squares criteria. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[2] Beucher, S., & Lantuéjoul, C. Use of watersheds in contour detection. In *International Workshop on Image Processing: Real-Time Edge and Motion Detection/Estimation*, pages 17–21, 1992.

[3] Duda, R. O., Hart, P. E., & Stork, D. G. *Pattern Classification.* John Wiley & Sons, 2001.

[4] Sorour Mohajerani. 38-Cloud - Cloud Segmentation in Satellite Images. *Kaggle*, 2021. Available at: https://www.kaggle.com/sorour/38cloud-cloud-segmentation-in-satellite-images

[5] Danda, S., Challa, A., & Daya Sagar, B. S. (2016). A morphology-based approach for cloud detection. 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS),

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[SVZ14] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014. arXiv: 1312.6034 [cs.CV]. URL: https://arxiv.org/abs/1312.6034.

[He+15] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[SZ15] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV]. URL: https://arxiv.org/abs/1409.1556.

[Luo+17] Wenjie Luo et al. *Understanding the Effective Receptive Field in Deep Convolutional Neural Networks*. 2017. arXiv: 1701.04128 [cs.CV]. URL: https://arxiv.org/abs/1701.04128.

[Sel+19a] Ramprasaath R. Selvaraju et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization". In: *International Journal of Computer Vision* 128.2 (Oct. 2019), pp. 336–359. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. URL: http://dx.doi.org/10.1007/s11263-019-01228-7.

[Sel+19b] Ramprasaath R. Selvaraju et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization". In: *International Journal of Computer Vision* 128.2 (Oct. 2019), pp. 336–359. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. URL: http://dx.doi.org/10.1007/s11263-019-01228-7.

[KKS21] Hiroaki Kingetsu, Kenichi Kobayashi, and Taiji Suzuki. *Neural Network Module Decomposition and Recomposition*. 2021. arXiv: 2112.13208 [cs.LG]. URL: https://arxiv.org/abs/2112.13208.

[Sha21] Vinayak Shanawad. 2021. URL: https://www.kaggle.com/datasets/vinayakshanawad/ships-dataset.

[Jia+24] Longyu Jiang et al. "Numerical investigation of the effective receptive field and its relationship with convolutional kernels and layers in convolutional neural network". In: *Frontiers in Marine Science* 11 (Oct. 2024). DOI: 10.3389/fmars.2024.1492572.

[LYW24] Dongze Lian, Weihao Yu, and Xinchao Wang. "Receptive Fields As Experts in Convolutional Neural Architectures". In: *Proceedings of the 41st International Conference on Machine Learning*. Ed. by Ruslan Salakhutdinov et al. Vol. 235. Proceedings of Machine Learning Research. PMLR, 21–27 Jul 2024, pp. 29531–29544. URL: https://proceedings.mlr.press/v235/lian24b.html.

[San+24] Antonio De Santis et al. *Visual-TCAV: Concept-based Attribution and Saliency Maps for Post-hoc Explainability in Image Classification*. 2024. arXiv: 2411.05698 [cs.CV]. URL: https://arxiv.org/abs/2411.05698.

[Sch24] Oleksander Schevchenko. 2024. URL: https://www.kaggle.com/datasets/oleksandershevchenko/ship-classification-dataset.