UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

LUCAS MARQUES DORNELES

An approach based on class activation maps for investigating the effects of data augmentation on neural networks for image classification

Work presented in partial fulfillment of the requirements for the degree of Bachelor in Computer Science

Advisor: Prof. Dr. Joel Luís Carbonera
Co-advisor: Dr. Luan Fonseca Garcia

Porto Alegre
February 2024

ABSTRACT

Neural networks have grown more and more popular in the last few years as an effective tool for the task of image classification due to the impressive performance they have achieved on this task. In image classification tasks, it is common to use data augmentation strategies to increase the robustness of the trained networks to changes in the input images and to avoid overfitting. Although data augmentation is a widely adopted technique, the literature lacks a body of research analyzing the effects data augmentation methods have on the patterns learned by neural network models working on complex datasets. The primary objective of this work is to propose a methodology and set of metrics that may allow a quantitative approach to analyzing the effects of data augmentation in convolutional networks applied to image classification. An important tool used in the proposed approach lies in the concept of class activation maps for said models, which allow us to identify and measure the importance these models assign to each individual pixel in an image when executing the classification task. From these maps, we may then extract metrics over the similarities and differences between maps generated by these models trained on a given dataset with different data augmentation strategies. Experiments made using this methodology suggest that the effects of these data augmentation techniques not only can be analyzed in this way but also allow us to identify different impact profiles over the trained models.

Keywords: Artificial Intelligence. Machine Learning. Convolutional Neural Networks. Data Augmentation. Explainability. Image classification.

Uma abordagem baseada em mapas de ativação de classe para investigar os efeitos de aumento de dados em redes neurais para classificação de imagens

RESUMO

Redes neurais têm, nos últimos anos, crescido cada vez mais como ferramentas efetivas para a tarefa de classificação de imagens devido à alta performance que têm obtido nesta tarefa. Nessas tarefas de classificação, é comum o uso de estratégias de aumento de dados. Elas aumentam a robustez dos modelos contra variações nas imagens de entrada e ajudam a evitar overfitting. Embora aumento de dados seja uma técnica amplamente adotada, a pesquisa na área ainda falta com trabalhos que analisem os efeitos que métodos de aumento de dados têm nos padrões aprendidos por redes neurais trabalhando sobre datasets complexos. O principal objetivo deste trabalho é propor uma metodologia e conjunto de métricas que permitam uma abordagem quantitativa para a análise dos impactos que aumento de dados têm sobre redes neurais convolucionais na tarefa de classificação de imagem. Uma ferramenta importante utilizada nesta proposta de abordagem é o mapa de ativação de classes para os modelos de redes neurais, que nos permitem identificar e mensurar a importância que esses modelos dão para cada píxel individual numa imagem quando executam a tarefa de classificação. A partir destes mapas, podemos então extrair métricas sobre as similaridades e diferenças entre mapas gerados por modelos treinados em datasets com diferentes estratégias de aumento de dados. Experimentos feitos com esta metodologia sugerem que os efeitos destas diferentes técnicas de aumento de dados não apenas podem ser analisados desta maneira, mas também nos permitem identificar perfis de impacto diferentes sobre os modelos.

Palavras-chave: Inteligência Artificial. Aprendizado de Máquina. Redes neurais convolucionais. Aumento de dados. Explicabilidade. Classificação de imagens.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
ML	Machine Learning
CNN	Convolutional Neural Network
CAM	Class Activation Map
Grad-CAM	Gradient-Weighted Class Activation Map
MAD	Mean Average Difference
MSD	Mean Squared Difference
KLD	Kullback-Leibler Divergence
Class-KLD	Class Prediction Kullback-Leibler Divergence
TPs	True Positives
TNs	True Negatives
FPs	False Positives
FNs	False Negatives

CONTENTS

1 INTRODUCTION

Neural networks have been demonstrating surprising performance in various data-processing tasks over the last decade, especially in the field of image classification (TODESCATO et al., 2023; TODESCATO et al., 2024; PULS; TODESCATO; CARBONERA, 2024). Their high performance and capacity to learn complex patterns from training data are impressive, and both their customizability and ease of use help make them a highly active field of research.

When engaging with image classification problems, it is common to use data augmentation techniques to increase the robustness of the model to changes in the input images and also to avoid overfitting (WON; BAE; KIM, 2023; SHORTEN; KHOSHGOFTAAR, 2019). This increased robustness often means an increase in performance, making a model effective against a larger pool of input data. However, even with data augmentation being a widely adopted technique, there still lacks a body of research investigating the effects of data augmentation methods on the patterns learned by neural network models, especially those models working on more complex datasets. As such, understanding how data augmentation may alter the classification process of a neural network model may help us to select suitable data augmentation strategies for a given use case, or to provide insights for creating more robust and effective data augmentation techniques.

Analyzing the effects of data augmentation techniques, however, is not an easy nor straightforward task. Due to the nature of neural networks themselves, the process by which models learn to identify patterns in images, and how they use these patterns to classify the image, works in a manner that is hard to interpret. It is possible to analyze the weights inside the model (which may number in the millions) but they do not directly provide methods to comprehend the inner workings of neural networks and generate human-interpretable explanations for their choices. This makes analyzing neural networks a non-trivial task and a significant challenge in the field (ANGELOV; SOARES, 2020), lending it a reputation for working in a manner akin to a black box.

This black box property of neural networks has, in recent years, steadily given rise to the field of neural network interpretability, or explainability (ZHANG et al., 2021; LIU; WANG; MATWIN, 2018). One of the goals in this field is to understand why neural network models provide a given output in a given context and strive to make a model's choices explainable to humans. These challenges also relate directly to the trustworthiness of neural network models in society; as their adoption in various fields increases, so do the burdens in guaranteeing accountability for the choices they make (LIU; WANG; MATWIN, 2018; HAAR; ELVIRA; OCHOA, 2023; ZHANG et al., 2021). Work in the field is ongoing, however, and there are still no out-of-the-box, easy-to-use methods to interpret neural networks and their choice patterns (FAN et al., 2021; MUHAMMAD; YEASIN, 2020; OLAH et al., 2018).

Similarly, understanding how data augmentation itself may affect neural networks is still a complex issue (TANG; SHARMA; ZHANG, 2020; WON; BAE; KIM, 2023). Most works in this field are interested in discerning the impact of data augmentation in the performance of the models, comparing performance metrics such as accuracy and recall to measure their effectiveness concerning classification tasks. However, due mainly to the difficulties in interpreting neural networks, it is not trivial to determine how different techniques of data augmentation may affect the way neural networks work (SANTOS et al., 2022), and especially to do so at scale.

When analyzing data augmentation techniques, it is common to evaluate their impacts in one of two ways: either by considering performance metrics of models trained on differently augmented datasets in a given classification task (O'GARA; MCGUINNESS, 2019; NANNI et al., 2021; PEREZ; WANG, 2017; LI et al., 2018; CHEN et al., 2020), or by qualitative image analysis by applying semi-supervised analysis techniques over artifacts generated by explainability techniques (CAO et al., 2022; WON; BAE; KIM, 2023; UDDIN et al., 2020).

The first type of analysis compares the performance metrics (accuracy, F1-score, etc.) of models trained on differently augmented datasets. It investigates the impact that data augmentation techniques have quantitatively and at scale, but can only measure impact as understood by classification performance. This fact means it cannot offer insights into how the application of data augmentation impacted how the model processed each image during classification.

The second type of analysis works through the visual identification of important image regions. Through explainability techniques, such as *class activation maps* (CAM) (SELVARAJU et al., 2017), we may generate artifacts that can help us understand how an image is being used in a classification task by a neural network. This method by itself is usually adopted in a qualitative approach where a human must visually compare the artifacts for models with and without augmentation to determine the most important regions for both models, given an input image. An alternative technique involves manually annotating input images to provide some ground truth for region importance. A significant drawback to this type of technique, however, is its difficulty in scaling up. Analyzing hundreds or thousands of images would take an unreasonable amount of human effort in visually analyzing CAM pairs or annotating input data.

Considering the limitations present in both common types of approaches, we identify a lack of scalable methods to obtain a more general and quantitative perspective of how different strategies of data augmentation impact the patterns a neural network learns while processing images, especially one that may apply to large image datasets without the need for human supervision.

In this work, we propose a methodology that combines the advantages of both of these common analysis types. The proposed approach uses explainability techniques, such as CAMs, to compare how the application of (or lack thereof) augmentation techniques affects the attribution of importance by a model to different image regions, doing so at scale and without the need for extensive human intervention. We hypothesize that by analyzing CAMs not in isolation but instead comparing maps generated by a baseline model against those generated by differently augmented models, we may be able to develop and use similarity metrics that help measure these differences in a scalable way. This approach should allow for a quantitative, scalable way to analyze the impacts of data augmentation on model behavior by comparing CAMs.

The results we obtained using the methodology proposed in the following chapters are promising. Although there remains difficulty in obtaining clear-cut answers using the metrics we chose, we can garner some insights about the models trained with augmented data. Additionally, investigating the correlations between the metrics of different augmentations also suggests the existence of separate profiles of impact across the data augmentation techniques we chose. In essence, this work attempts to tackle a single question: how can one use CAMs to automatically and quantitatively analyze the effects of data augmentation on a neural network model?

This work furthers the research into that question by proposing a basic, generically structured methodology to quantitatively evaluate the impacts of data augmentation in convolutional neural networks dealing with image classification tasks. Additionally, this work contributes by proposing an initial set of metrics that may be used within that methodology, as well as demonstrating the application of the methodology for a specific selection of metrics and augmentations, using the Grad-CAM (SELVARAJU et al., 2017) method of generating CAMs, applied to the CIFAR-10 dataset (KRIZHEVSKY; HINTON et al., 2009) and adopting the EfficientNet B0 (TAN; LE, 2019) neural network architecture. Another quality of this methodology is its extensibility; future works may also expand on this methodology, including but not limited to using more complex metrics and proposing new ways to analyze said metrics, or modify aspects of this methodology to fit other image-classification-related needs.

This work is structured as follows. In Chapter 2, we will expand on fundamental concepts to the understanding of the proposed methodology. In Chapter 3, we discuss works related to our goals. In Chapter 4, we present the proposed methodology, covering its' steps and their explanations. In Chapter 5, we discuss how the proposed methodology was applied in a specific context, as well as showcasing and analyzing our obtained results.

2 BACKGROUND

The main concepts needed to understand the methodology proposed in this work are covered in the following sections. First, we will introduce the concepts of machine learning and neural networks (Section 2.1). Then, we will cover the topics of data augmentation (Section 2.2) and model performance metrics (Section 2.3). Finally, we will explain the concept of CAMs (Section 2.4), a central concept in our work.

2.1 Machine Learning

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that focuses on developing algorithms capable of improving their performance through experience and learning from data (MITCHELL, 1997). Traditionally, ML is divided into three broad categories: Supervised learning, unsupervised learning, and reinforcement learning (MURPHY, 2012). This work focuses on supervised learning. In supervised learning, there are two main types of tasks: regression and classification. Regression tasks involve training a model to learn a mapping function between input data and continuous output values based on labeled input-output pairs (PATTERSON; GIBSON, 2017). Classification tasks, which are the focus of this work, have the objective of finding a mapping function between input data and discrete target labels (PATTERSON; GIBSON, 2017).

For our work, the input data we want to classify are images, and the labels are the different types of entities (objects, animals, etc) represented in the images. To do this, we make use of artificial neural networks.

Artificial Neural Networks, or simply Neural Networks, constitute a specific approach to ML. They consist of algorithms inspired by the structure of human brains, following a hypothesis that the layered, neuron-based structure of the human brain may be an effective structure for mimicking intelligent behavior (MURPHY, 2012). Neural networks are part of a broad family of techniques for machine learning called Deep Learning, in which hypotheses take the form of complex algebraic circuits with tunable connection strengths (RUSSELL; NORVIG, 2020). In this sense, artificial neural networks are structured as interconnected layers of simple processing units (which represent neurons) that have weights associated with the connections between units of adjacent layers. Each node in a layer is capable of processing inputs and producing outputs which may then be sent to nodes further in the chain of layers, similar to how, in the human brain, a signal may travel through a chain of neurons. In practice, however, the actual resemblance between neural networks and neural cells and structures is only superficial (RUSSELL; NORVIG, 2020).

Training artificial neural network models commonly involves applying three main steps: forward propagation, back-propagation, and gradient descent. Forward propagation consists of the process of passing the information through the layers of nodes, processing the initial input (images in our case) until it arrives at the final layer (GOODFELLOW; BENGIO; COURVILLE, 2016). Each layer of the neural network applies linear and non-linear transformations to the data it receives, propagating the processed signal forward until the final output is produced. From this final output, we determine the error produced between the expected value and the value obtained by the model. Determining the error is done by a cost function (also known as loss function) that measures this difference in some way, and many cost functions may be used for this purpose.

Back-propagation refers to the algorithm that is executed after the final layer output is generated. It propagates backward the cost function gradients in relation to the model parameters (the weights between nodes and biases associated with each node). These gradients may be understood as the contribution of each parameter to the final error produced by the model, or the sensibility of the network to that parameter; the higher, the more it is considered to have contributed to the error, and the more its value needs to be adjusted (GOODFELLOW; BENGIO; COURVILLE, 2016).

Gradient descent is then applied after the parameter gradients have been set. It consists of an optimization algorithm used to adjust the parameters of the model in the opposite direction to the gradients calculated during back-propagation (the gradients may also be understood as first-order derivatives), intending to minimize the cost function (GOODFELLOW; BENGIO; COURVILLE, 2016).

Specifically, in this work, we use a kind of artificial neural network called Convolutional Neural Networks (CNN). A CNN is a network that contains spatially local connections, at least in the early layers, and has patterns of weights that are replicated across the units in each layer (RUSSELL; NORVIG, 2020). A pattern of weights that is replicated across multiple local regions is called a kernel and the process of applying the kernel to the pixels of the image (or to spatially organized units in a subsequent layer) is called convolution (RUSSELL; NORVIG, 2020).

Figure 2.1 – Example of affine transformation augmentation.

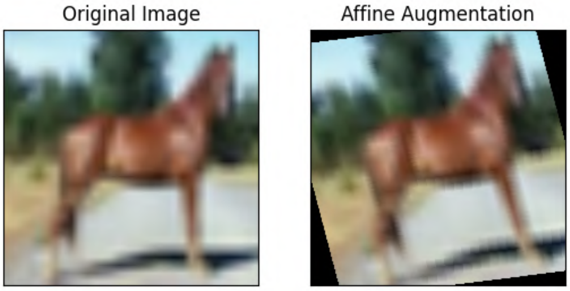
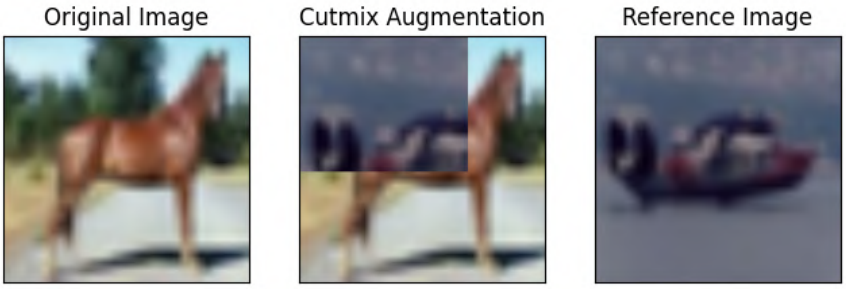


Figure 2.2 – Example of cutmix augmentation.



2.2 Data Augmentation

Data augmentation is one of the fundamental techniques this work is based on. They refer to a subset of regularization techniques, which work by introducing additional information to the ML model to better capture some properties of the problem being modeled (KHALIFA; LOEY; MIRJALILI, 2022). Image data augmentation, in particular, involves introducing variability in the training dataset itself, generating new data based on the original training images. This is done to increase the robustness of the model to variations in the input data.

In the realm of data augmentation, several different techniques may be used to generate augmented images from training images. As it would be prohibitively time-consuming to test all of the most popular data augmentation techniques for this work, we decided to focus on seven techniques that represent a wide variety of different data augmentation approaches. These are:

- Affine transformation, which involves applying various linear and non-linear transformations (rotation, translation, scaling, shearing) in conjunction to one image. An example can be seen in Figure 2.1.
- Cutmix, which involves cutting a section of an image B with class Y and mixing it with an image A, which has a class X. The resulting image is then considered as having class X. An example is shown in Figure 2.2.

Figure 2.3 – Example of color jitter augmentation.

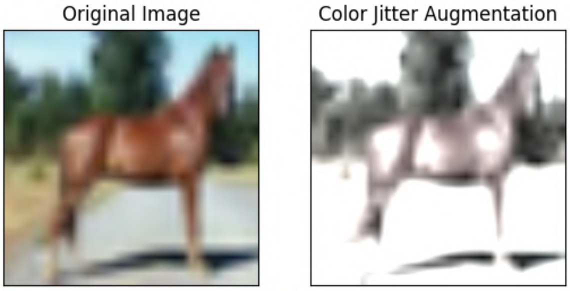
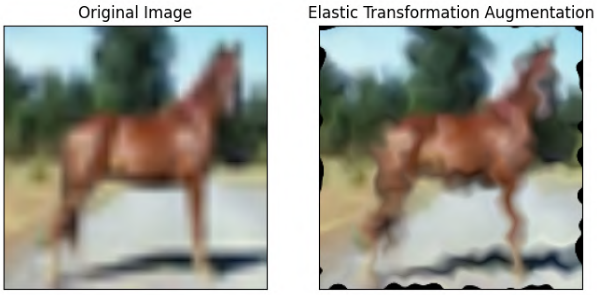


Figure 2.4 – Example of elastic transformation augmentation.



- Color Jitter, which randomly changes the brightness, contrast, saturation and hue of an image. An example is shown in Figure 2.3
- Elastic transforms, a transform that works based on displacement vectors applied on all pixels. An example is shown in Figure 2.4.
- Equalization, in which the histograms of each color channel of an image are equalized. Example in Figure 2.5
- Gaussian Blur, which applies Gaussian blurring to an image. An example is shown in Figure 2.6
- Random Cropping, which crops the input image at a random location and resizes the cropped area to be the same size as the input image's. An example is shown in Figure 2.7

Figure 2.5 – Example of equalization augmentation.

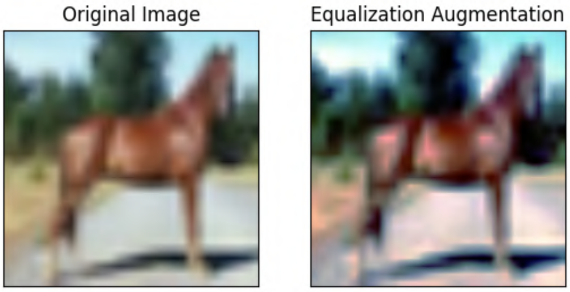


Figure 2.6 – Example of Gaussian blur augmentation.

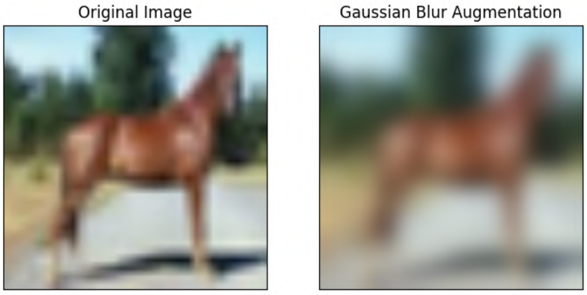
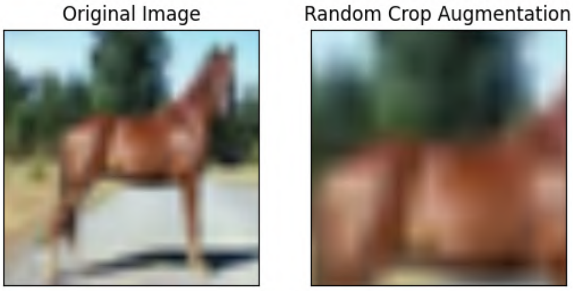


Figure 2.7 – Example of random cropping augmentation.



2.3 Performance Metrics

To evaluate the performance of CNNs, it is common to track metrics related to the correct and incorrect predictions made by the model. To track the performance of the models used in this work, we use accuracy, precision, F1-score and recall. As we are dealing with a multi-class problem, we use the macro averages of these metrics.

Accuracy, precision, F1-score, and recall are calculated based on the confusion matrix resulting from model inferences. Confusion matrices refer to the relationship between ground truth labels and predicted labels in classification problems. To trace these relationships, we rely on the concepts of true positives (TPs), true negatives (TNs), false positives (FPs) and false negatives (FNs). In classification tasks with only two target classes, positive represents the presence of a characteristic, and negative represents the lack of said characteristic. TPs refer to instances where the model predicts positively and the correct label is also positive; TNs refer to cases where the prediction was negative and the correct label was also negative; FPs refer to the model predicting positively but the correct label being negative; and FNs referring to cases where the model predicted negatively but the correct label is positive. By counting the number of instances of each of these cases, we may create a confusion matrix (FACELI et al., 2021). In multi-class problems with N classes, such as the one we are tackling, this matrix has an N by N shape, where the N lines and columns represent each of the N classes. The macro-average for a given metric is taken by averaging the sum of that metric over all N classes.

- Accuracy measures how often a classification ML model makes correct predictions overall.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

- Precision measures the ratio of correct positive predictions made against all positive predictions made by the model.

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

- Recall measures the ratio of correct positive predictions made against all instances that possess the characteristic.

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

- F1-Score combines precision and recall into a single metric, providing a way to assess the impact of both false positives and false negatives simultaneously.

$$F1 - Score = \frac{2\times(Recall\times Precision)}{Recall+Precision}$$

2.4 Grad-CAM and Class Activation Maps

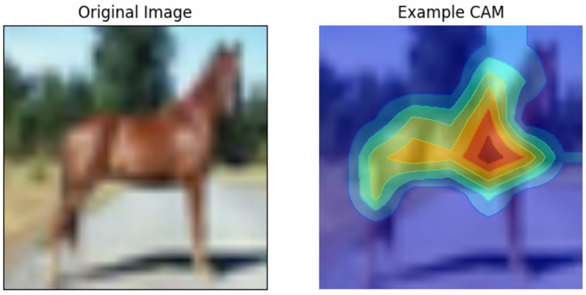
Explainability is still a significant challenge for neural networks. When models are trained, the patterns they learn are represented by the internal weights of the model, which may number in the millions or even billions. This makes analyzing image classification outputs of a model prohibitively complex if one were to inspect only a model's weights.

As tools to alleviate this problem, many techniques have been developed over the years. Deconvolutional neural networks, for example, can provide dense activation maps of an image by applying deconvolution operations and unpooling layers to the features extracted by CNNs (HAAR; ELVIRA; OCHOA, 2023); layer-wise relevance propagation (LWRP) can provide activation maps that not only indicate which parts of an image were important for classification but also which pixels support a different classification (HAAR; ELVIRA; OCHOA, 2023); and inversions allow recreation of the most important parts of an image by inverting what each layer of a model filters out to see what it considered important at that layer (HAAR; ELVIRA; OCHOA, 2023).

Amongst these, one of the most commonly used techniques to explain neural network classifications is that of CAMs. CAMs are a type of explainability method, commonly considered a specific type of saliency map, that indicates the discriminative image regions used by the neural network to identify that category (ZHOU et al., 2016). This explainability technique attributes the importance or contribution of each pixel of an input image to the final classification generated by the neural network. CAMs help establish how neural network models attribute importance to the different regions of an image (ZHOU et al., 2016), allowing visual analysis of what models consider important image regions.

There are many ways of generating CAMs, such as Score-CAM (WANG et al., 2020), LayerCAM (JIANG et al., 2021), and Eigen-CAM (MUHAMMAD; YEASIN, 2020), but the method we will be using in this work is the Gradient-Weighted Class Activation Map (Grad-CAM) (SELVARAJU et al., 2017). Grad-CAM uses the gradients of any target label (for example ‘dog’ in a classification network) flowing into the last convolutional layer of a model to produce a CAM highlighting the important regions in the image for predicting the label. Given an image, a target layer, usually (but not necessarily) the final convolutional layer, and a target class, it forward-propagates the input through the CNN part of the model and obtains the raw score for the target category. The gradients of all outputs are then set to zero except for the desired class, which is set to 1. The result obtained by forward-propagation is then back-propagated up to the target layer that has the convolutional feature maps of interest, which are then combined to compute the coarse Grad-CAM localization. This localization represents which areas the model most strongly influences it to make a particular decision, meaning that a disturbance in that area would more strongly affect the model outcome for that class. Another interpretation of CAMs is that they represent the areas of the image the model is the most sensitive to for a given class. Lastly, as the visualization generated is coarse, it is upscaled to the image size by pointwise multiplication with guided back-propagation to generate the final class activation map. The resulting mappings will have pixels normalized in the $[0, 1]$ range. An example of this type of CAM can be seen in Figure 2.8. These maps will be one of the main tools utilized in this work.

Figure 2.8 – Example of CAM generated by Grad-CAM using the baseline model we trained as the selected model, the last convolutional layer as the target layer, and the predicted class as the target class.



3 RELATED WORKS

Although the fields of neural network explainability and data augmentation have become increasingly popular over time, the literature still lacks a body of work analyzing how data augmentation may affect explainability. Among those works that exist, here are a few that explore a field of research similar to ours.

Firstly, there is (TANG; SHARMA; ZHANG, 2020), which inspired our work and whose investigations on data augmentation and explainability we follow up on. Their work explores the impact of five different data augmentation methods on the CAMs of the models trained with augmented data in an image classification task. Their process is similar to ours, conducting an initial foray in constructing quantitative methods for evaluating data augmentation impacts, but doing so in a more specific way. The work made by (TANG; SHARMA; ZHANG, 2020) has its limitations, however. The authors selected MNIST, a simplistic grayscale dataset as the basis of their analysis. They also only trained one model per augmentation, subjecting the results to higher statistical randomness related to the impacts of starting states for the augmentation methods and neural networks. Our work improves on their initial exploration by using a more varied set of similarity metrics between CAMs; adopting CIFAR10 (KRIZHEVSKY; HINTON et al., 2009), a significantly more complex colored dataset, as the basis for the experiments; and training multiple models per augmentation to ensure higher statistic robustness of the results. Our work also expands on theirs by using more augmentation methods and measuring other similarity metrics between CAMs, as well as analyzing correlations between the metrics taken as they covary over the test dataset. Our work also differs from theirs by providing a structured methodology that may be applied to a large variety of image datasets, methods of generating CAMs, sets of augmentations, sets of similarity metrics between CAMs, and CNN architectures, the choice of architectures being limited by compatibility with CAMs.

In (WON; BAE; KIM, 2023), the authors also investigate the impacts of data augmentation on the interpretability of the models generated, but using a different methodology. It uses Grad-CAM and Information Bottlenecks through Attribution (SCHULZ et al., 2020) to generate attribution maps for images, which are then used to compare the CAMs generated by the different augmentation qualitatively to one another. Though it is thematically very close to our work, their work is interested in measuring how data augmentation affects interpretability as understood by qualitative measures (Human-model alignment, Faithfulness, and Human-understandable concepts,) whereas our work is interested in the use of quantitative measures that may be applied automatically over large datasets in order to compare the impacts of different augmentations.

Other related works cover only tangentially the matter of analyzing image data augmentation impact and explainability. In (UDDIN et al., 2020), for example, the authors investigate differences between the CAMs of different augmentations, analyzing their performance while introducing their own augmentation method. However, it is difficult to find works where the main focus is specifically on explainability with data augmentation, as a majority of articles that analyze and compare data augmentation methods are more interested in comparing their performances, such as (CHEN et al., 2020; RADHAKRISHNAN et al., 2017; YANG et al., 2022).

4 PROPOSED METHODOLOGY

In this chapter, we explain our proposed methodology for generating quantitative analyses from CAMs in image classification tasks. For clarity and brevity purposes, from here on out we will be referring to the models trained on augmented datasets as 'augmented models', and the model trained on the original CIFAR10 dataset (without applying any kind of augmentation) as 'baseline model'. Our main objective with this methodology is to propose a series of steps to analyze the impact of data augmentation on models trained on augmented data. To clarify, we are not interested in analyzing these augmented models or their artifacts in isolation, but rather by evaluating these models relative to the baseline model. Our intended analysis consists of comparing how data augmentation affects models in comparison to a baseline. In this way, our focus is on evaluating how the augmented models diverge from and co-vary with the baseline model. This approach allows us to collect metrics related to how the augmented models' CAMs diverge from the ones generated by the baseline model, and thus provide a quantitative analysis of their divergences.

The methodology consists of the following steps:

1. Select a set of data augmentation techniques A to be analyzed, and a set of starting states X. A starting state may be understood as some selection of variables or parameters that results in a specific initial configuration for the data augmentation methods, such as the seed given to a random number generator in the case of a data augmentation method that uses randomness. Note that the actual parameters of the augmentation itself (such as size of the crop for Random Cropping, standard deviation and kernel size for Gaussian blur, or the alpha and sigma for the Elastic Transformation method) should be the same across the $|X|$ states.
2. Select a CNN architecture and base dataset.
3. Select a method for generating CAMs.
4. Select a set of metric functions ME to be applied to the baseline and augmented CAMs.
5. Select a set of methods to be applied to the metrics for analysis.
6. Separate the dataset into training, test, and validation sets. The training, test and validation sets must be stable across all models. For clarity in later definitions, define the test set as I.

7. Train baseline model B of the selected CNN architecture based on the selected training and validation sets. The validation sets are used to monitor the model training and extract the version of the model with the best performance against the validation set across all epochs, and they are used this way for every single model.
8. Train $|X|$ versions of the chosen architecture for each augmentation, each of the $|X|$ models for a given augmentation a with starting state $x \in X$, and training on their respective augmented dataset. This will result in $|X| \times |A|$ augmented models $M_{a,x}, a \in A,\ x \in X$. All models, including the baseline, should be instantiated with the same starting weights. This step is done to eliminate other sources of impact from the model results such that the differences between the models are borne uniquely from the application of different augmentations. The set of augmented models will be defined as $M_{Aug} = M_{a,x}, \forall a \in A,\ \forall x \in X$, and the set of all models as $M = M_{Aug} \cup B$.
9. Collect performance metrics (precision, recall, etc) for each of the models.
10. Generate CAMs according to the selected method, generating augmented model CAM sets $S_m, \forall m \in M$. Defining the CAM-generating function as f which takes a model m and test image $i \in I$, a single CAM will be defined as $s_{m,i} = f(m,i), m \in M,\ i \in I$. The target layer for all models will be the last convolutional layer of the chosen architecture, and the target class will be the class predicted by the model for the respective test image. The CAM set S_m for model m will be defined as $S_m = s_{m,i}, m \in M,\ \forall i \in I$, meaning every individual CAM will be based off a different image in the test set, covering all of the test set.
11. Generate similarity metrics $model\_metrics\_set_{me,m}, \forall me \in ME,\ \forall m \in M_{Aug}$ between the augmented CAM sets $S_m, m \in M_{Aug}$ and baseline CAM set S_B. To do this, define the metric function $me \in ME$ between two individual CAMs as $me(s_{m,i}, s_{B,i}), m \in M_{Aug},\ i \in I$. The set of metrics $model\_metrics\_set_{me,m}$ between the CAMs of a given augmented model m and those of baseline model B for a given metric type me will be defined as $model\_metrics\_set_{me,m} = me(s_{m,i}, s_{B,i}), me \in ME,\ m \in M_{Aug},\ \forall i \in I$.
12. Average the metrics for each augmentation $a \in A$. The previous step generates $|X|$ values for a metric me, for each augmentation a, for a specific image $i \in I$. In this step, we aggregate these values in a mean, such that the value result for an aggregate metric function $me_{aggr}(a,i)$ based on metric function me for a given augmentation method a and test image i is defined by $me_{aggr}(a,i) = \sum_{x \in X} me(s_{M_{a,x},i}, s_{B,i}) \div$

$|X|, a \in A,\ i \in I$. The set of aggregate metric values for an augmentation method a and metric type me is defined as $aug\_method\_metrics\_set_{me,a} = me_{aggr}(a, i), \forall i \in I$. These aggregate values are used to give a better perspective on the mean differences of importance in image regions between augmented and baseline model CAMs, as interpreted by the metric me.

13. The last step consists of using the selected methods to generate a quantitative analysis of the metrics to determine the impacts each augmentation has when compared to the baseline model, and then go over the generated analysis.

This methodology, as previously mentioned, is based on (TANG; SHARMA; ZHANG, 2020), but here we provide a structured set of steps that applies to different architectures, datasets, metrics, augmentation techniques, and CAM generation methods. For the exploration made in this work, we also provide a richer set of metrics between CAMs, each of them described in the following Section (Chapter 5).

5 EXPERIMENTS

In this chapter, we will discuss how we applied the methodology, delineating our choice of augmentations, metrics, and datasets in Section 5.1, as well as discussing our choices and showing the results we obtained using it in Section 5.2.

5.1 Experiment Design

In the first part of this section, we discuss our choices for dataset and base architecture. After that, we will expand upon important details concerning how datasets were pre-processed and how training was conducted. Lastly, we will discuss the metrics we used between CAMs, and briefly examine other important details of our application.

When considering our choice of dataset for this experiment, we found it important to find one that struck a balance between complexity, to allow for rich analysis, and training time. Considering this, we used CIFAR-10 (KRIZHEVSKY; HINTON et al., 2009), an image classification dataset with 60,000 images. It has properties that favor a richer analysis and more complex challenge than the MNIST dataset used by (TANG; SHARMA; ZHANG, 2020), and is also commonly used in the literature (INOUE, 2018; CUBUK et al., 2019; RATNER et al., 2017; LIU; DENG, 2015; HUSSAIN; BIRD; FARIA, 2019). CIFAR-10 has a relatively small set of classes at 10 labels, colored images representing relatively complex and varied patterns, a balanced distribution of samples in each class, and relatively low image resolution, making it an adequately challenging dataset for our image classification purposes.

To determine which base architecture to use, we selected three commonly used architectures in the literature (EfficientNet B0 (TAN; LE, 2019), ResNet-18 (HE et al., 2016), DenseNet-121 (HUANG et al., 2017)) and measured their performances in image classification with CIFAR-10. To compare them, we ran training routines for the three architectures for 30 epochs over the CIFAR-10 dataset and measured their test accuracy, precision, F1-score, and recall for each epoch. After obtaining the results for the model trainings (Figure 5.1 for accuracy, Figure 5.2 for precision, Figure 5.3 for recall, Figure 5.4 for F1-score, and Figure 5.5 for training time), we chose EfficientNet B0 for its balance between training time and performance.

As mentioned in Section 2.2, we chose seven data augmentation methods that we felt best represented a wide spread of different augmentation techniques, which would

Figure 5.1 – Evolution of the accuracy on the test set across 30 training epochs for the three candidate architectures.

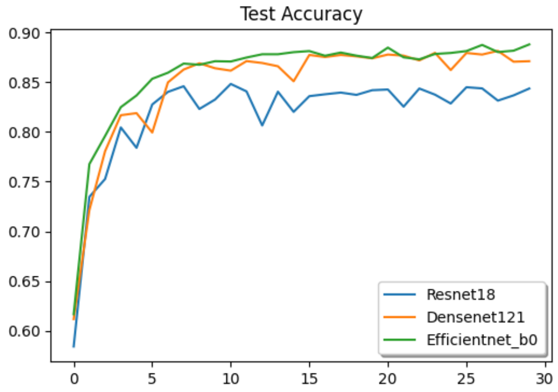


Figure 5.2 – Evolution of the precision on the test set across 30 training epochs for the three candidate architectures.

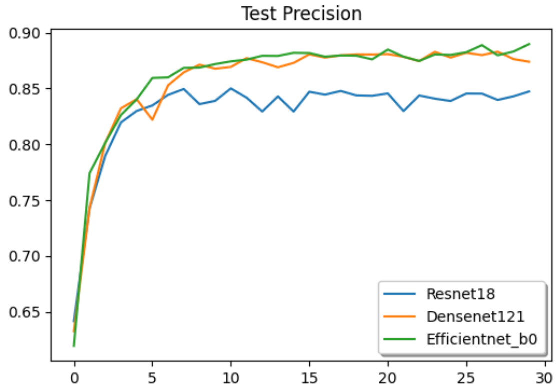


Figure 5.3 – Evolution of the recall on the test set across 30 training epochs for the three candidate architectures.

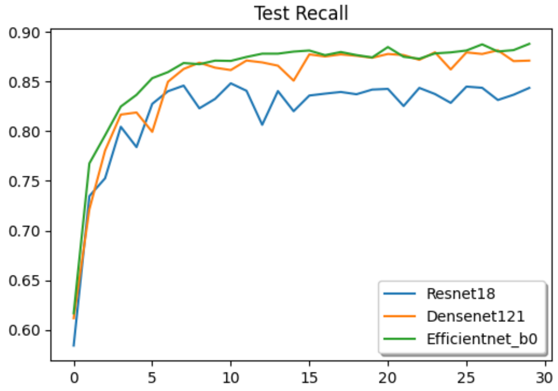


Figure 5.4 – Evolution of the F1-score on the test set across 30 training epochs for the three candidate architectures.

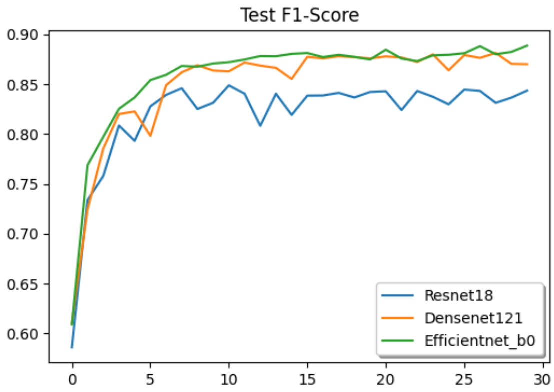
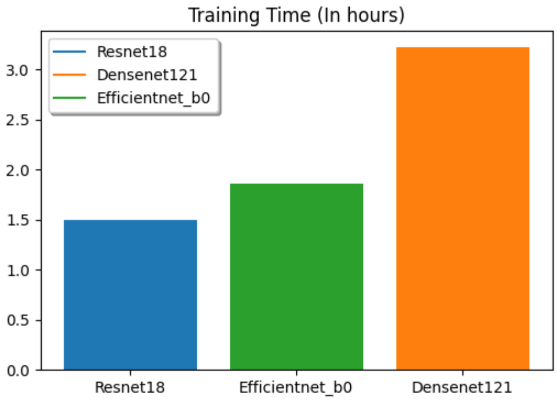


Figure 5.5 – Comparison of the amount of time necessary to train each of the candidate architectures over 30 epochs.



provide a richer ground for data augmentation analysis. The methods we chose were affine transformations, cutmix, color jitter, equalization, elastic transforms, random cropping, and gaussian blurring.

For the train-validation-test split, as CIFAR10 comes with a train-test split dataset, with 50,000 images for training and 10,000 images for testing, the only further modification we needed was a train-validation split. For that purpose, we decided on using random stratified sampling with 90% train to 10% validation split, with the same 45,000 base images for training and 5,000 for validation in all models. We do not use the validation set for hyperparameter optimization. We instead use the validation set to select the version of the model with the highest accuracy against it across all epochs. For the augmented datasets, we generated one augmented image for each training image, resulting in 90,000 total images for each augmented dataset. These augmentations are done online during batch fetching. For the training process, we adopted the cross-entropy loss as our loss function and adopted the Adam optimizer, with a learning rate at $1e^{-}3$ and weight decay at $1e - 5$. For training, validation, and testing routines, we used batch sizes of 32, and trained all models for 30 epochs each.

To control the impact that the order of image batches could have during training, it was important to manage, across all augmented datasets, the order in which images are fed to the models during training. We discuss more of how this was achieved in Appendix B. The core idea was to guarantee, as much as possible, that the placement order for all augmented and unaugmented images in a dataset would be the same across all augmented datasets. For the unaugmented images, following this requirement meant placing them in the same order across augmented datasets. For the augmented images, we met this requirement through a different method. Instead of guaranteeing that the same augmented image appears in the same ordering position across datasets (which would be impossible as each combination of augmentation method and starting state applied over the same source image produces a different augmented image,) we instead use the source of the augmented images to order them. We guarantee that the source image for an augmented image at a specific position in the dataset will be the same across datasets and do the same for all augmented images.

Additionally, regardless of augmentation, all images were resized to 224×224 pixels (while preserving the aspect ratio) to fit the input format of EfficientNet B0 and normalized in order to improve results. To normalize the dataset, we calculate the mean and standard deviation of each color channel across all images in the dataset. From the mean and standard deviation, we alter the value of each RGB channel of a pixel by subtracting the respective color channel's mean from the pixel value of the channel and then dividing the result by that color channel's standard deviation. By doing this, every pixel's color channel value will have a mean of 0 and a standard deviation of 1 across the entire dataset. This significantly reduces the variability range of the dataset, accelerating the training convergence time.

Another important detail to cover is how we define the starting states X. Here, we choose three different states $X = [Seed1, Seed2, Seed3]$ for each augmentation, with each state representing a unique seed fed to all random generation libraries involved in the training of the respective augmented model. It is important to note that these states are stable across all augmentations, meaning that the seed used as the starting state for $M_{RandCrop,Seed1}$ is the same as that of $M_{Cutmix,Seed1}$, which is the same as $M_{Affine,Seed1}$, and so on.

For the CAM generation, as detailed previously, we chose the default Grad-CAM algorithm. In order to generate CAMs with this algorithm we must provide a target class alongside a model, the target layer of said model, and the input image. This is necessary as the CAM is a measure of which areas of an image the CNN model is the most sensible to with regard to a specific class. Another interpretation of CAMs is that regions with high values are regions where a change in image pixel values would significantly impact a model's output with regard to the given class. For the choice of target label, we used the model's predicted class for the respective test image. An alternative to this choice would've been to use the ground truth label instead of the predicted class, which would show us what the CAM would look like for a given label even if the model itself would not have made that classification choice. The problem with this lies with the usage of a CAM with a target class that the respective model would not have predicted, which would mean we are not analyzing the final decision-making process of the model representative of the patterns it has learned, but instead how the model would act if it had perfect performance. The upside to that approach, however, would be that the baseline and augmented model CAMs would always have the same class, and thus would theoretically make analysis easier and more straightforward. Both approaches have their own merits and it is not clear which of them is the ideal approach. Due to our interest in analyzing the behavioral patterns ultimately learned by the models, we decided to use the predicted label instead of the ground truth as the target class for CAM generation, even if the behavioral patterns they exhibit involve wrong classifications or classifications that differ between baseline and augmented models.

To evaluate the differences between the CAMs generated by the baseline and those generated by the different augmented models, we used the following metrics between CAMs: Mean Absolute Difference (MAD), Mean Squared Difference (MSD), Pearson Correlation, Spearman Correlation, and Overlap Rate. Additionally, we also use Kullback-Leibler Divergence (KLD) between the class predictions for each image, which here we will call Class Prediction Kullback-Leibler Divergence (Class-KLD). The discrete probability distributions used to calculate Class-KLD are explained ahead in Section 5.1. Each metric is explained as follows:

Mean Absolute Difference (MAD): It is the mean absolute error (MAE) metric applied between two CAMs. We reinterpret this error that MAE measures as being the mean pixel difference between the two CAMs. With this metric, we are purely interested in the magnitude of the difference between the maps, so this reinterpretation does not introduce any unintended problems in analyzing this metric. Although this metric has no specific unit, considering the range of values $[0, 1]$ a pixel can take, the maximum mean difference is 1, indicating that baseline and augment images have the maximum possible difference, and the minimum is 0, indicating two identical maps. This metric may be interpreted as a measure of the mean magnitude of the differences between a pair of CAMs, and can also be viewed as a degree of dissimilarity. Considering a pair of CAMs P and Q, and pixel indexes j for source image J with the total number of pixels $N = |J|$, where P_j represents the activation of pixel j in the CAM P, MAD is defined as

$$MAD(P,Q) = \frac{1}{N}\sum_{j}|P_j - Q_j|$$

Mean Squared Difference (MSD): It follows the same logic as MSD, but squaring the difference instead of taking the absolute of the difference. We use this metric in case it may exacerbate large differences between CAMs and aid in analyzing the magnitudes of differences from another perspective. MSD is defined as

$$MSD(P,Q) = \frac{1}{N}\sum_{j}(P_j - Q_j)^2$$

Pearson Correlation Coefficient: Pearson Correlation is a metric used to measure the linear relationship between two variables, reflecting the strength and direction of this relationship (BYLINSKII et al., 2018). It can be understood as a measuring of the degree to which two variables covary while maintaining a reasonably constant proportion. To use it, we flatten both CAM images into 1D vectors and then measure their covariance, using the baseline as P and augmented as Q. It is defined as

$$Pearson(P,Q) = \frac{cov(P,Q)}{\sigma(P)\times\sigma(Q)}$$

where $cov(P,Q)$ is the covariance between P and Q, and $\sigma(P)$ and $\sigma(Q)$ are the standard deviation of P and Q, respectively.

Spearman Correlation: Spearman follows in the same steps as Pearson, but as this metric weighs monotonicity more heavily than proportionality. That is, it measures how well the relationship between two variables can be described using a monotonic (but not necessarily linear) function. Alongside Pearson, it helps paint a clearer picture of the way baseline and augmentation covary. It is defined as

$$Spearman(P,Q) = \frac{cov(R(P),R(Q))}{\sigma(R(P))\times\sigma(R(Q))}$$

where $R(P)$ and $R(Q)$ refer to the rank-variable versions of P and Q, meaning P and Q transformed into a weak-ordering vector where any two items necessarily have a 'higher than', 'lower than' or 'equal to' relationship.

Overlap Rate: It is a specialization of the metric Intersection over Union (REZATOFIGHI et al., 2019) as applied to a specific selection of pixel regions. Intersection over Union works by analyzing two regions of an image and taking a ratio of the intersection between these regions over the union between these regions (REZATOFIGHI et al., 2019). Overlap Rate follows the same idea, but selects the region automatically by using the $Y\%$ pixels with the highest activations in the CAM, meaning the pixels with the highest value. This gives us an idea of how the important areas differ in spatial distribution, which will complement the information gained from MSD and MAD regarding the magnitude of their difference. It is important to note, also, that the percentage is not referring to a threshold of $Y\%$ of population, but rather to a threshold based on the value of the Y-th population percentile; this means that if $Y = 10$ and the 10th percentile highest value is 0.7, we will select all pixels with threshold value equal to or higher than 0.7. This may result in selecting more than 10% of the population if there should be multiple values equal to 0.7 in the population.

Class Prediction Kullback-Leibler Divergence: Class-KLD follows a different idea from the others. While the other metrics are applied directly between the CAMs, we apply Kullback-Leibler Divergence to the prediction matrix made by model $M_{a,x}$ as compared to the prediction made by the baseline model B. KLD is defined as an information-theoretic measure of the difference between two probability distributions (BYLINSKII et al., 2018). The idea behind the use of this metric is to measure how different the predictions made by the two models are for a test image. Between class prediction vectors P and Q, KLD is defined as

$$KLD(P, Q) = \sum_i Q_i log(\epsilon + \frac{Q_i}{P_i})$$

where, in our application, i refers to class in the vector of predictions, and ϵ is a regularization constant.

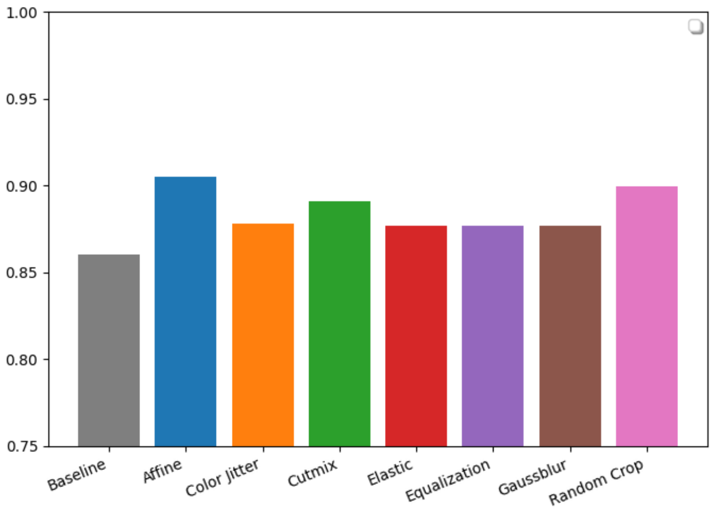
Having defined the base architecture, dataset, augmentations, and metrics, the application of the methodology is straightforward. First, we train the baseline and augmented models and then use Grad-CAM to generate CAM sets for all the models. From there, we collect the metrics between each augmented model's CAM set and the baseline model B's CAM set, considering all the images in the test set. After averaging the $|X|$ models' metrics for each augmentation, we then analyze these metrics.

To analyze the results we obtained, we adopted the following approaches:

- Analysis of performance metrics between the models. Although this is not the main focus of our work, determining their differences in effectiveness at the classification task is relevant information.
- Analysis of the similarity metrics value distribution over the test image set for every augmentation. Using boxplots, we inspect properties of the metric value distributions of the augmentations, which may be able to establish trends in the collected data. This analysis, however, is aggregative in nature; it does not permit an individual investigation into how each augmentation behaves for each test image.
- Correlation analysis for each metric across augmentations. By correlating the metric values across augmentations, we view another aspect of the behaviors between augmentations. For each metric $me \in ME$, we take the average metrics sets $aug\_method\_metrics\_set_{me,a}$ for all augmentations $a \in A$ and analyze how these sets of metric correlate with one another across augmentations. This analysis enables us to understand how the metrics covary between augmentations across the test image set, allowing us to make more nuanced observations about changes in behavior induced by the augmented dataset.

From these analyses, our objective is to build a comprehensive view of how the different models behave. The metric distribution plots, performance metric comparisons, and correlation maps are shown in Section 5.2, and the results as a whole are analyzed in Section 5.2.6.

Figure 5.6 – Bar plot of the accuracy performance metric for all models over the test image set.



5.2 Results

Below are the metric results taken between the baseline and augmented models. Firstly, we will analyze performance metrics related to the augmentations. Secondly, we will show the boxplot and correlation map results for every metric, analyzing them individually. Afterward, we will inspect the results more generally, making observations on trends we see and interpreting the results.

5.2.1 Performance Metrics

To measure model performance, we tracked, as previously mentioned, the accuracy, F1-measure, recall, and precision of all the models. Below are the results of these metrics over the test set for all augmentations, with the mean performance of the $|X|$ models being shown for each augmentation. Figure 5.6 shows the average test accuracy between the models, Figure 5.7 shows the average test precision, Figure 5.8 shows the recall comparison, and Figure 5.9 shows the F1-score comparison.

The results are consistent across metrics, implying that none of the models suffer particularly from the impact of false positives or false negatives. Additionally, although not an unexpected result, we note that utilizing any of the seven augmentation methods results in increased performance over the baseline model. All models display similar

Figure 5.7 – Bar plot of the precision performance metric for all models over the test image set.

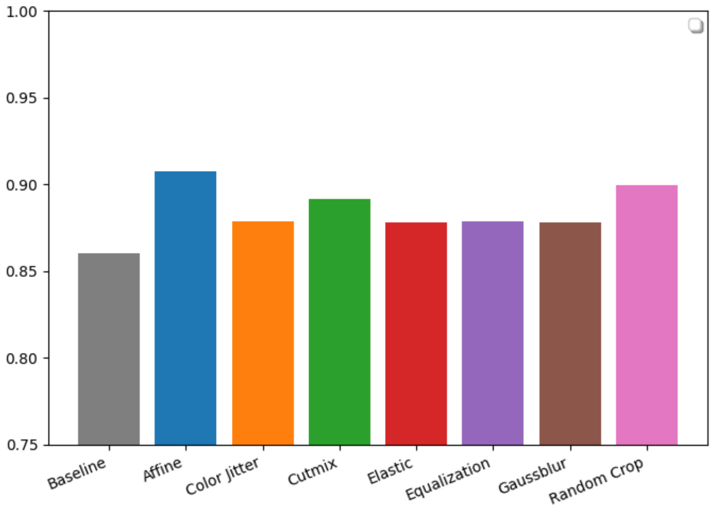


Figure 5.8 – Bar plot of the recall performance metric for all models over the test image set.

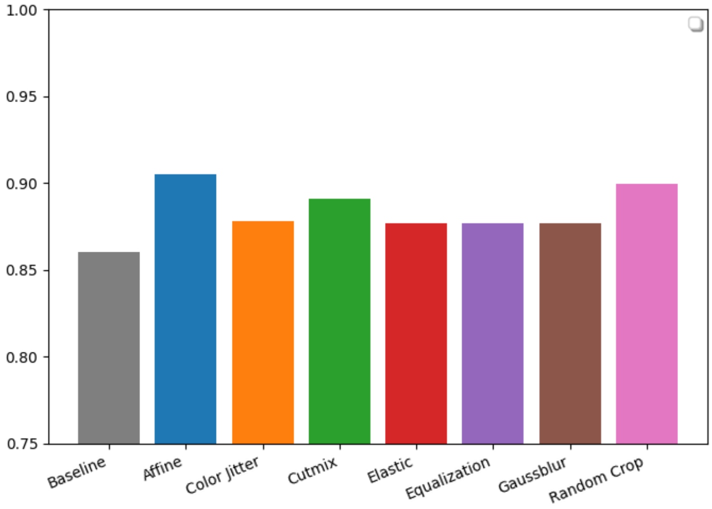
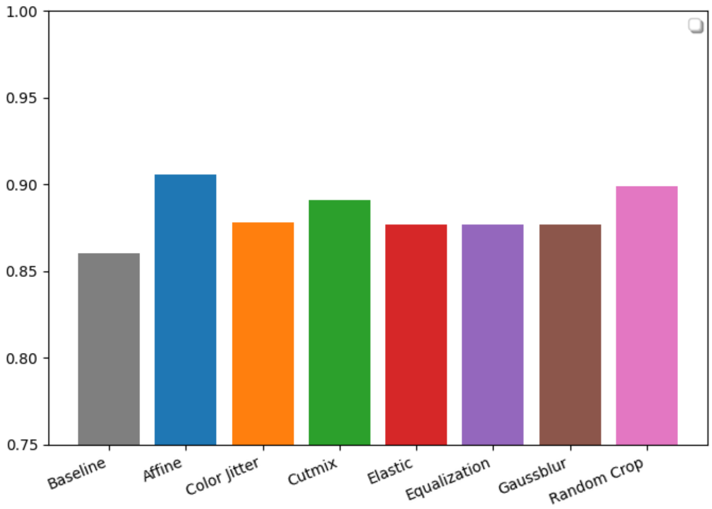


Figure 5.9 – Bar plot of the F1-score performance metric for all models over the test image set.



performance to one another, with affine transforms achieving the best performance across all performance metrics.

Also, as expected, it is hard to draw any conclusions about the impact of data augmentation by inspecting only performance metrics; to that end, we will have to analyze the CAM metrics. In the following sections, we will inspect these metrics individually.

5.2.2 Overlap Rate (Top 20, Top 10, Top 5)

In this section we will show boxplot and correlation map results for the overlap rate metric. In Figure 5.10, we show the results for overlap rate with a threshold of $Y = 20$, and in Figure 5.11 we show the correlation map across augmentations for overlap rate with $Y = 20$. We do the same for $Y = 10$ (Figure 5.12 for the boxplot, Figure 5.13 for the correlation map) and for $Y = 5$ (Figure 5.14 showing the boxplot, Figure 5.15 showing the correlation map).

Analyzing the differences between overlap rate boxplots, we can see that as the importance threshold increases (and Y lowers from 20 (Figure 5.10) to 10 (Figure 5.12) to 5 (Figure 5.14)), the observed overlap rate lowers for all augmentations. This behavior indicates that, although the augmented models may consider similar image regions to be similarly important overall, they focus on different parts of the image as we filter out the less important regions of the CAMs. This result is not necessarily unexpected,

Figure 5.10 – Boxplot for overlap rate with threshold $Y = 20$, showing the distribution of the metric values over the test set for each augmentation.

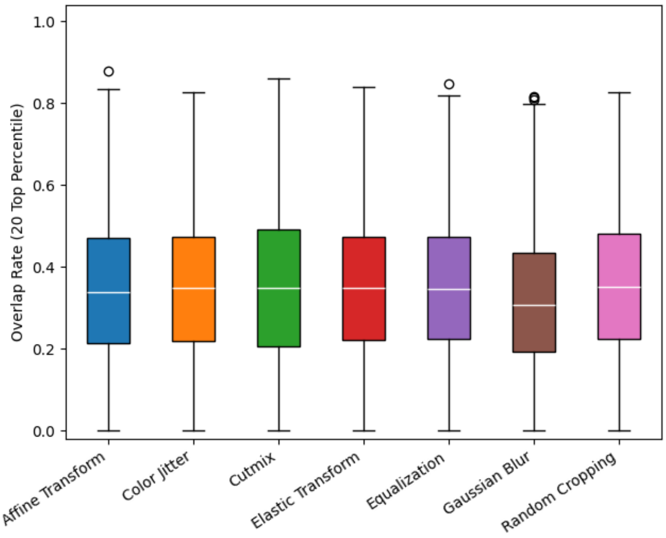
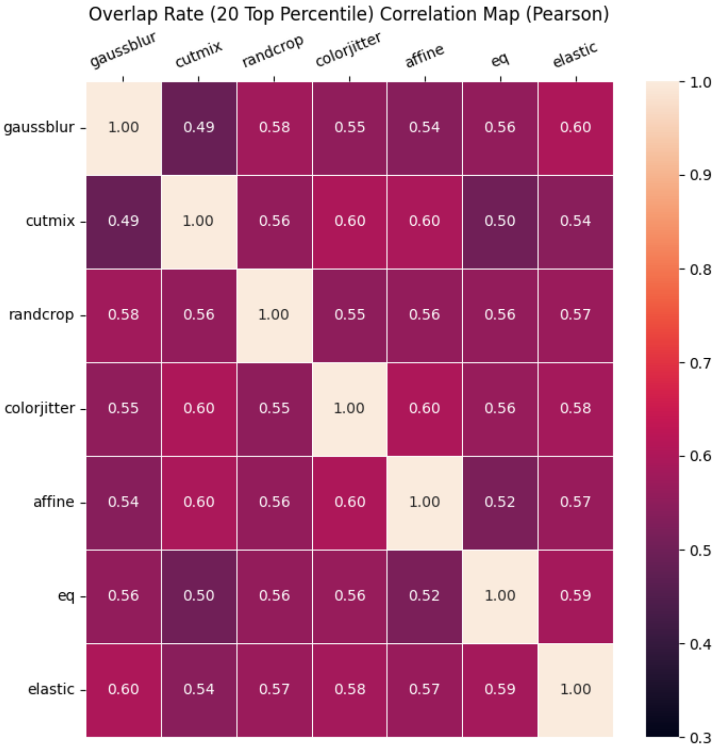


Figure 5.11 – Correlation map for overlap rate with threshold $Y = 20$, showing the correlation of the metric between augmentations over the test set.



	gaussblur	cutmix	randcrop	colorjitter	affine	eq	elastic
gaussblur	1.00	0.49	0.58	0.55	0.54	0.56	0.60
cutmix	0.49	1.00	0.56	0.60	0.60	0.50	0.54
randcrop	0.58	0.56	1.00	0.55	0.56	0.56	0.57
colorjitter	0.55	0.60	0.55	1.00	0.60	0.56	0.58
affine	0.54	0.60	0.56	0.60	1.00	0.52	0.57
eq	0.56	0.50	0.56	0.56	0.52	1.00	0.59
elastic	0.60	0.54	0.57	0.58	0.57	0.59	1.00

Figure 5.12 – Boxplot for overlap rate with threshold $Y = 10$, showing the distribution of the metric values over the test set for each augmentation.

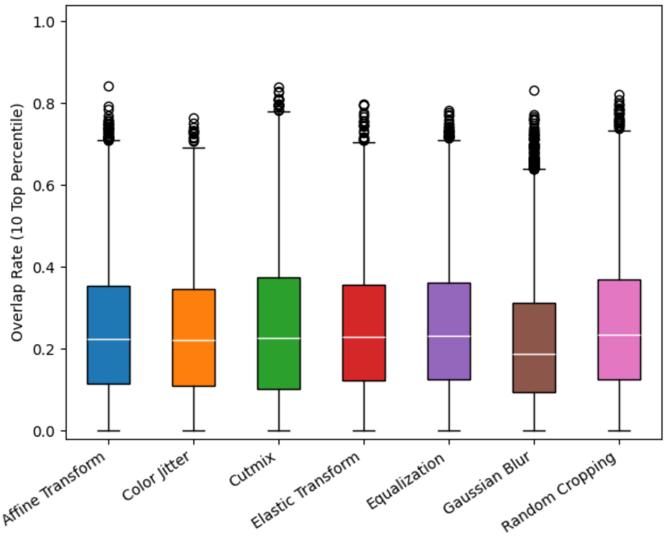
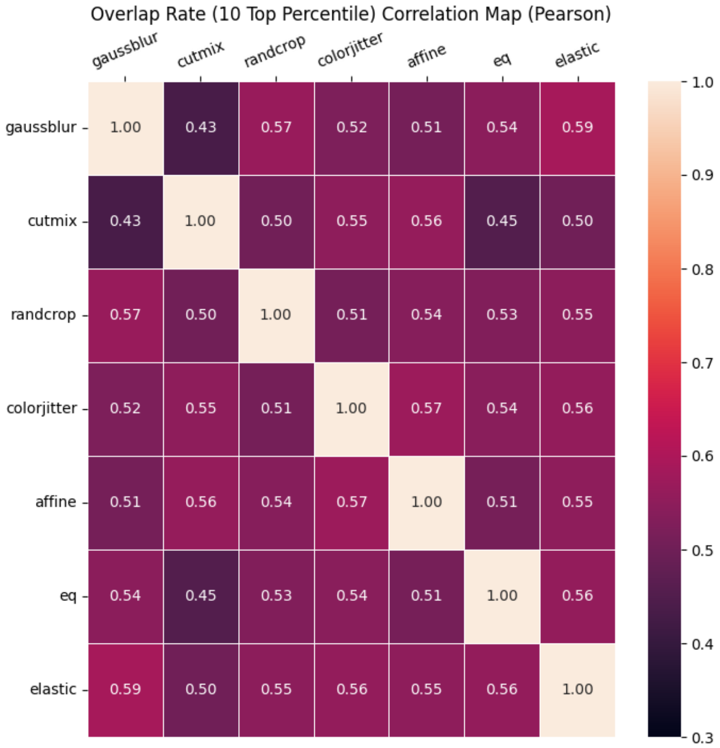


Figure 5.13 – Correlation map for overlap rate with threshold $Y = 10$, showing the correlation of the metric between augmentations over the test set.


Overlap Rate (10 Top Percentile) Correlation Map (Pearson)

	gaussblur	cutmix	randcrop	colorjitter	affine	eq	elastic
gaussblur	1.00	0.43	0.57	0.52	0.51	0.54	0.59
cutmix	0.43	1.00	0.50	0.55	0.56	0.45	0.50
randcrop	0.57	0.50	1.00	0.51	0.54	0.53	0.55
colorjitter	0.52	0.55	0.51	1.00	0.57	0.54	0.56
affine	0.51	0.56	0.54	0.57	1.00	0.51	0.55
eq	0.54	0.45	0.53	0.54	0.51	1.00	0.56
elastic	0.59	0.50	0.55	0.56	0.55	0.56	1.00

Figure 5.14 – Boxplot for overlap rate with threshold $Y = 5$, showing the distribution of the metric values over the test set for each augmentation.

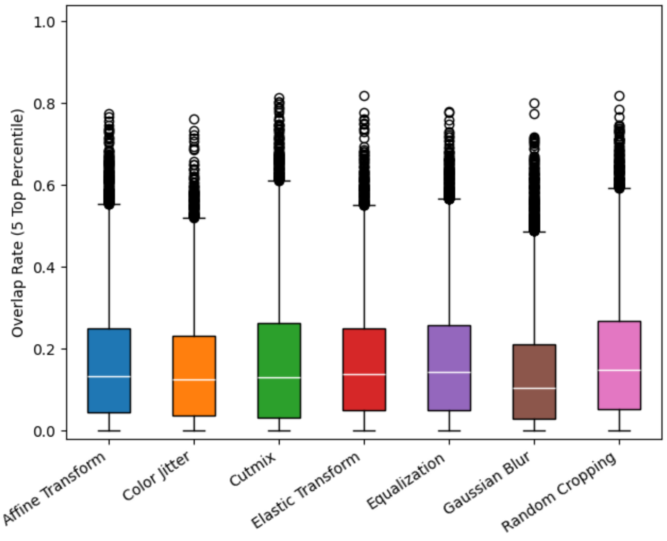
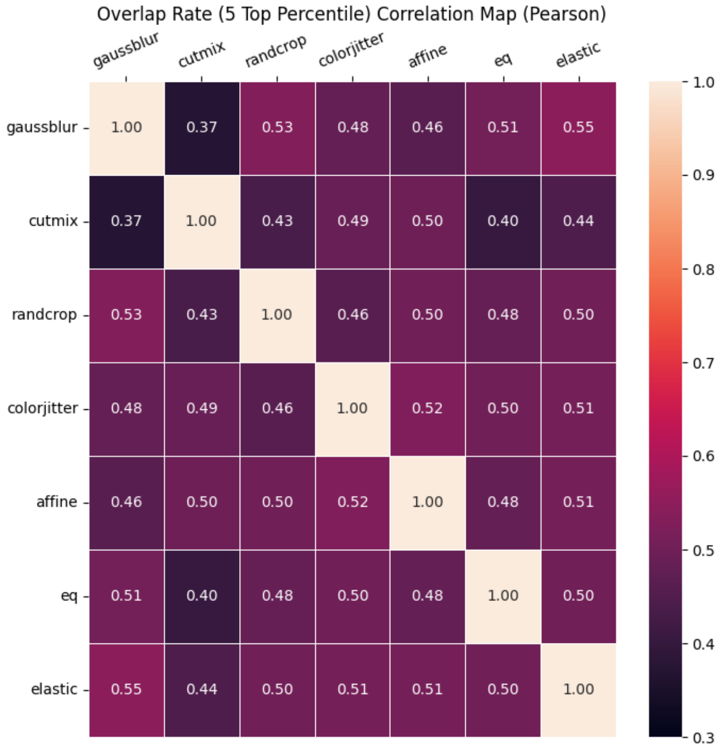


Figure 5.15 – Correlation map for overlap rate with threshold $Y = 5$, showing the correlation of the metric between augmentations over the test set.



	gaussblur	cutmix	randcrop	colorjitter	affine	eq	elastic
gaussblur	1.00	0.37	0.53	0.48	0.46	0.51	0.55
cutmix	0.37	1.00	0.43	0.49	0.50	0.40	0.44
randcrop	0.53	0.43	1.00	0.46	0.50	0.48	0.50
colorjitter	0.48	0.49	0.46	1.00	0.52	0.50	0.51
affine	0.46	0.50	0.50	0.52	1.00	0.48	0.51
eq	0.51	0.40	0.48	0.50	0.48	1.00	0.50
elastic	0.55	0.44	0.50	0.51	0.51	0.50	1.00

but it is worth noting as it quantitatively confirms that when considering the most significant minutiae of an image, augmented models consider different image regions as highly important towards their classification compared to the baseline.

When compared to the Top 20% correlation map (Figure 5.11), we can also see that the Top 5% overlap rate correlation map (Figure 5.15) has lower correlation values across augmentations, with the majority of them being at or below 0.5. This observation may indicate that not only do augmented models look at different regions to the baseline model, but how these important regions overlap with the baseline may also differ between augmentations. This observation is supported by the fact that if all augmented model CAMs were to, on average, differ from the baseline ones in the same ways, we could expect their correlation levels to be significantly higher, which does not happen. This observation could be made qualitatively by inspecting many different samples of CAMs, but through this method, we can confirm that this analysis holds throughout the entirety of the dataset in a quantitative manner without the need for a human to compare the maps individually.

5.2.3 Pearson and Spearman correlations

In this section, we will present and discuss boxplot and correlation map results across augmentations for the Pearson and Spearman correlation metrics. Figure 5.16 shows the boxplot for Pearson correlation, and Figure 5.17 shows the correlation map for the metric across augmentations. Figure 5.18 shows the distribution plot for Spearman correlation, and Figure 5.19 shows its correlation map.

For the Spearman and Pearson metric distributions (Figure 5.18, Figure 5.16), we notice a moderate correlation between the baseline and augmented CAMs. Note that while a large portion of the population presents correlations between baseline and augmented CAMs above 0.5 for both the Pearson (Figure 5.16) and Spearman (Figure 5.18) boxplots, the median line for all augmentations is below that value, indicating only moderate correlation. We also note that a significant amount of augmented model CAMs have an inverse correlation to baseline CAMs, which indicates that, for those images, when the activation for a region rises in baseline, the activation for that region lowers in the augmented model, and vice versa. One notable observation from the correlation map for Pearson across augmentations (Figure 5.17) is that all augmentations have lower Pearson correlation metric values across augmentations on average compared to their Spear-

Figure 5.16 – Boxplot for Pearson Correlation Coefficient, showing the distribution of the metric values over the test set for each augmentation.

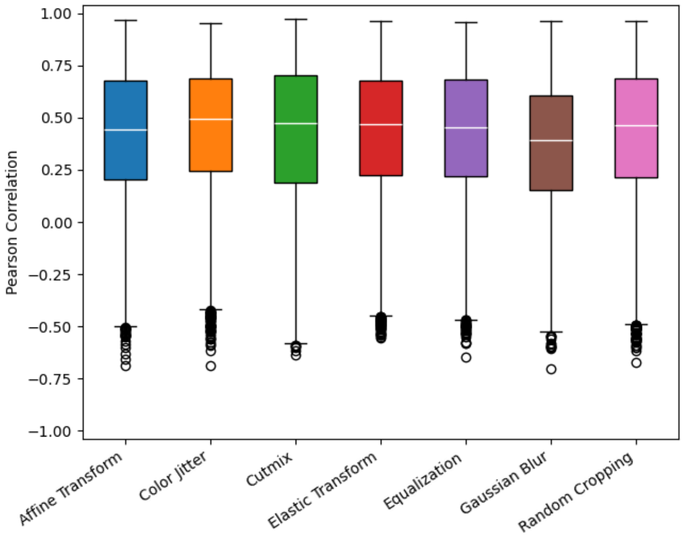


Figure 5.17 – Correlation map for Pearson Correlation Coefficient, showing the correlation of the metric between augmentations over the test set.

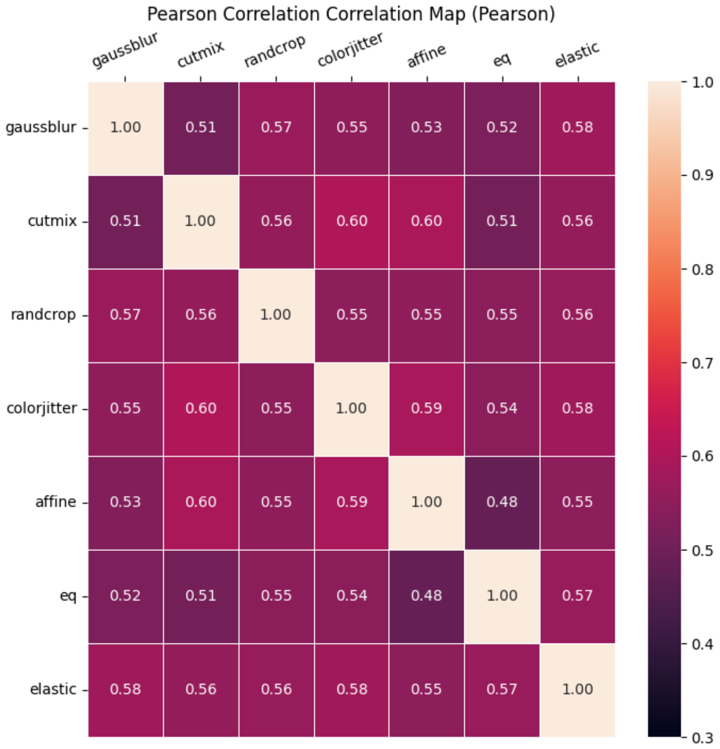


Figure 5.18 – Boxplot for Spearman Correlation Coefficient, showing the distribution of the metric values over the test set for each augmentation.

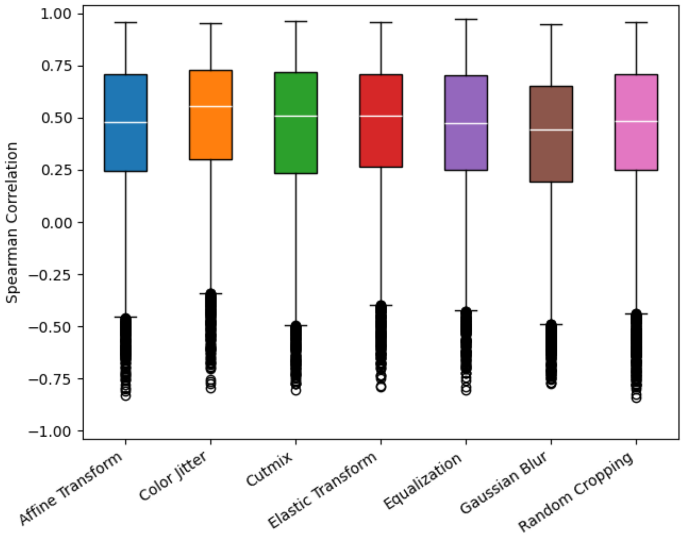
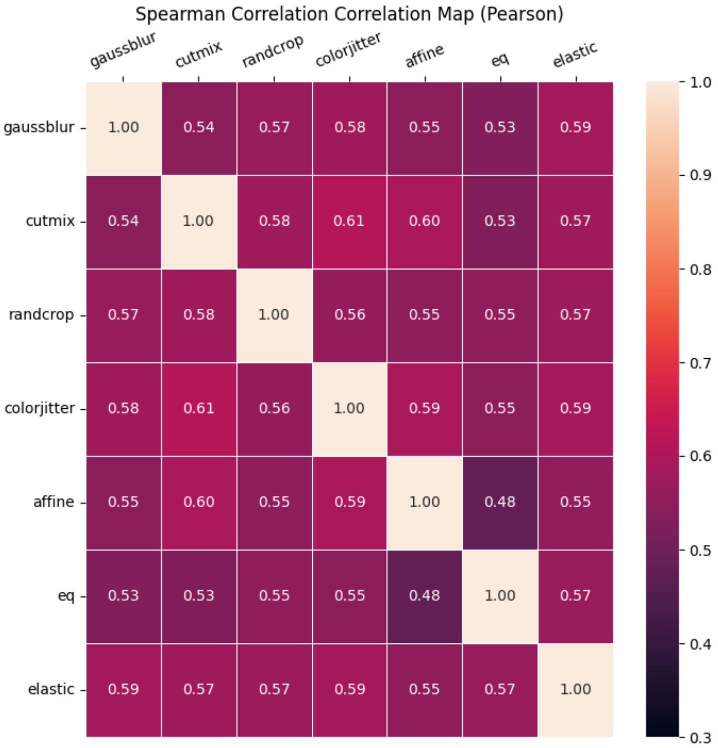
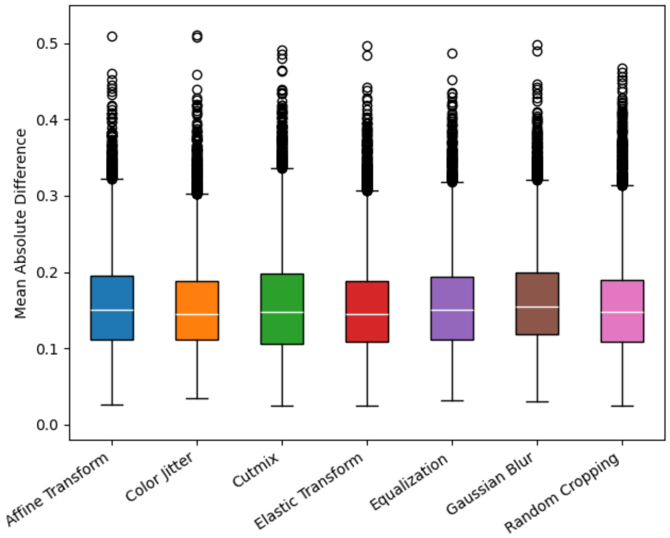


Figure 5.19 – Correlation map for Spearman Correlation Coefficient, showing the correlation of the metric between augmentations over the test set.



	gaussblur	cutmix	randcrop	colorjitter	affine	eq	elastic
gaussblur	1.00	0.54	0.57	0.58	0.55	0.53	0.59
cutmix	0.54	1.00	0.58	0.61	0.60	0.53	0.57
randcrop	0.57	0.58	1.00	0.56	0.55	0.55	0.57
colorjitter	0.58	0.61	0.56	1.00	0.59	0.55	0.59
affine	0.55	0.60	0.55	0.59	1.00	0.48	0.55
eq	0.53	0.53	0.55	0.55	0.48	1.00	0.57
elastic	0.59	0.57	0.57	0.59	0.55	0.57	1.00

Figure 5.20 – Boxplot for Mean Absolute Difference, showing the distribution of the metric values over the test set for each augmentation.



man correlation metrics (Figure 5.19). This can also be evidenced in the, on average, lower values between the Pearson metric distributions (Figure 5.16) and their Spearman metric counterparts (Figure 5.18). This observation indicates that the correlation these augmented model CAMs have to baseline is more alike in monotonicity than in terms of linear relationship.

5.2.4 Mean Absolute Difference and Mean Squared Difference

In this section, we share the results for the MAD e MSD metrics. Figure 5.20 shows the boxplot of MAD, while Figure 5.21 shows the correlation map for MAD. For MSD, Figure 5.22 shows its boxplot and Figure 5.23 shows its correlation map.

We can discern from the MSD (Figure 5.22) and MAD (Figure 5.20) boxplots that when the CAMs differ between baseline and augmentation, the average magnitude of the pixel value difference follows similar trends throughout augmentations. None of the augmentations show a significant increase or decrease in their metric distributions compared to the others. One observation to note is that all the augmentation pairs in the correlation maps across augmentations (Figure 5.21 for MAD, Figure 5.23 for MSD) present moderate correlations (around 0.5 and 0.6), which implies that the magnitude of difference between CAM and baseline maps is correlated throughout the augmentations. Considering both observations, they indicate that, in general, the mean magnitude of pixel activation

Figure 5.21 – Correlation map for Mean Absolute Difference, showing the correlation of the metric between augmentations over the test set.

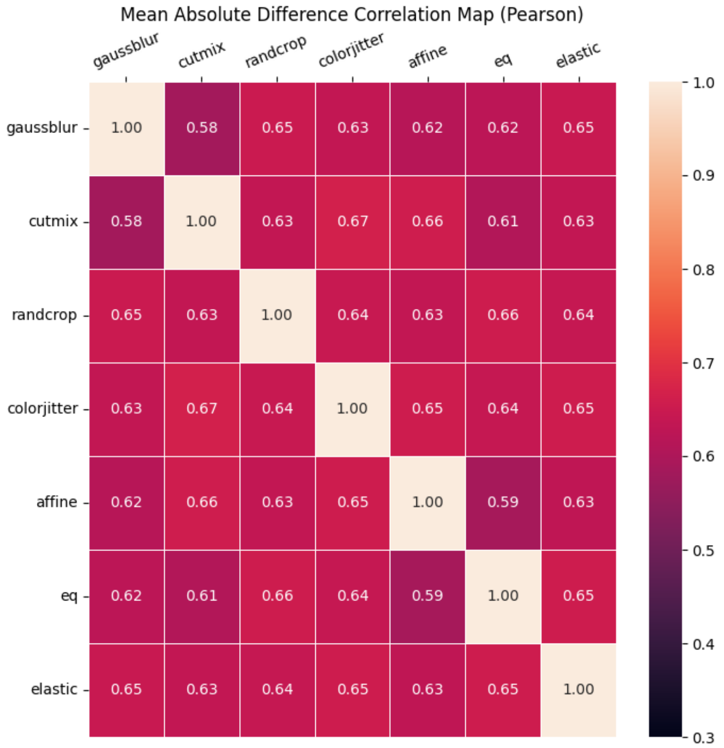


Figure 5.22 – Boxplot for Mean Squared Difference, showing the distribution of the metric values over the test set for each augmentation.

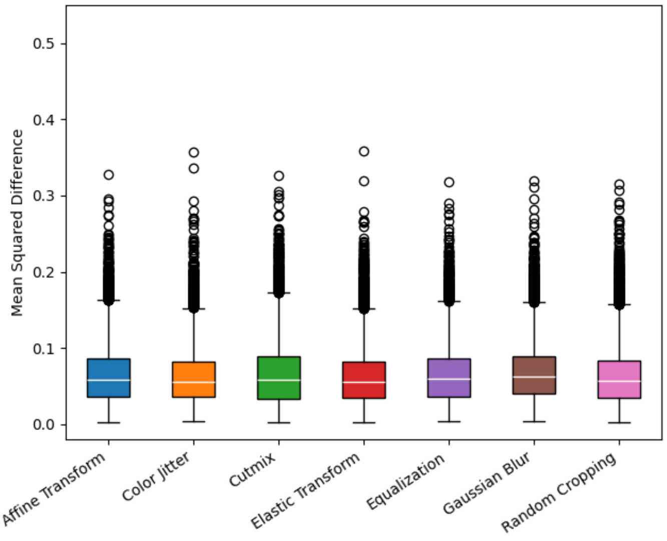
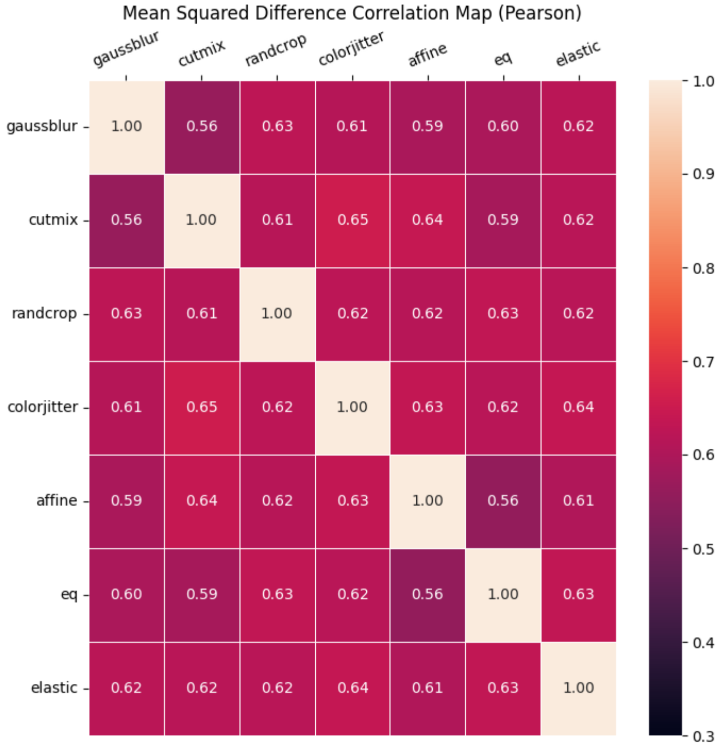


Figure 5.23 – Correlation map for Mean Squared Difference, showing the correlation of the metric between augmentations over the test set.



	gaussblur	cutmix	randcrop	colorjitter	affine	eq	elastic
gaussblur	1.00	0.56	0.63	0.61	0.59	0.60	0.62
cutmix	0.56	1.00	0.61	0.65	0.64	0.59	0.62
randcrop	0.63	0.61	1.00	0.62	0.62	0.63	0.62
colorjitter	0.61	0.65	0.62	1.00	0.63	0.62	0.64
affine	0.59	0.64	0.62	0.63	1.00	0.56	0.61
eq	0.60	0.59	0.63	0.62	0.56	1.00	0.63
elastic	0.62	0.62	0.62	0.64	0.61	0.63	1.00

differences between baseline and augmented model CAMs is not high across all augmentations for each test image. Moreover, considering these two metrics for each image, we note that these values covary positively and with moderate force between augmentation pairs.

5.2.5 Class Prediction Kullback-Leibler Divergence

Lastly, we display the results for the Class-KLD metric. Figure 5.24 shows the boxplot for the metric and Figure 5.25 shows its correlation map.

Class-KLD proved a hard metric to interpret. Due to the large amount of outliers, the visual density of the outliers, and most of the prediction metric values falling very close to one another, the boxplot (Figure 5.24) becomes convoluted and hard to analyze. An observation we can make from the boxplot is that most of the values for all augmentations tend to fall within the 0 to 1 range, which indicates that the class predictions made by the augmented models do not stray very far from those of the baseline models on average.

We note that although augmentation methods alter the importance attributed by

Figure 5.24 – Boxplot for Class-KLD, showing the distribution of the metric values over the test set for each augmentation.

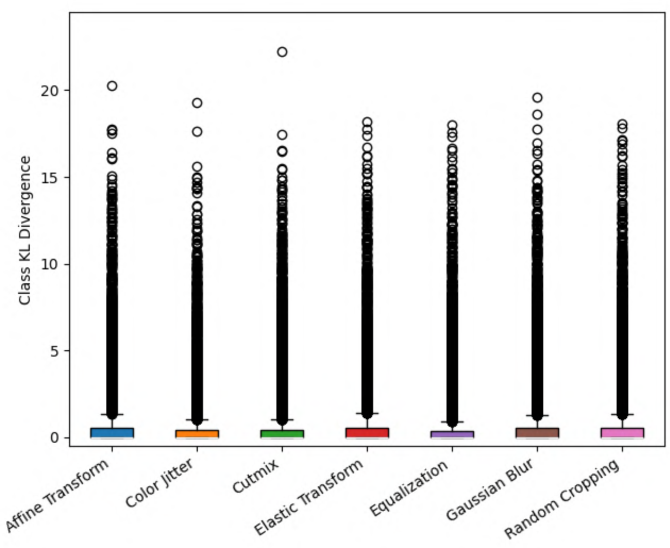
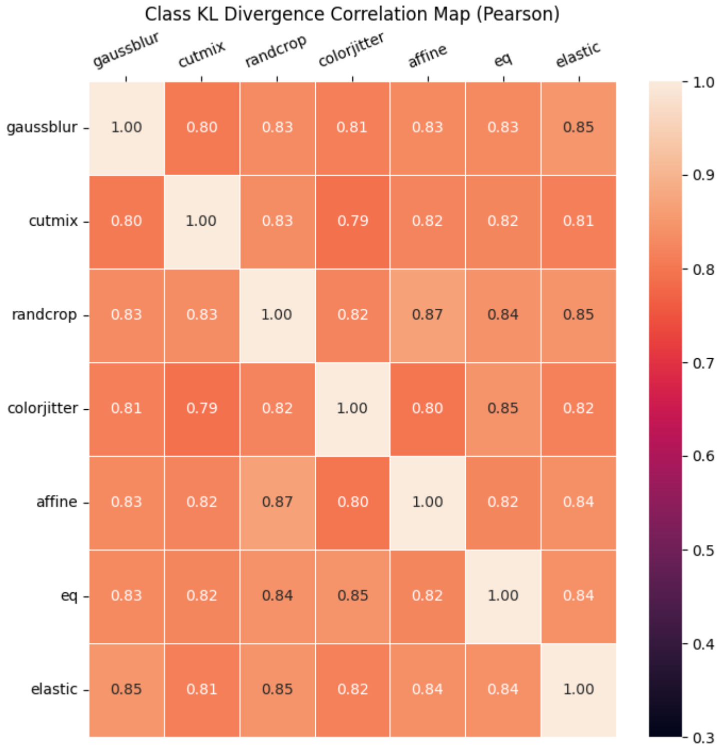


Figure 5.25 – Correlation map for Class-KLD, showing the correlation of the metric between augmentations over the test set.



the last convolutional layer to the image pixels, the resulting models generate classification probability distributions that, in general, have a Class-KLD value in a relatively small range (even if the outliers are numerous). To examine this fact and the previous observation, it is important to remember that the final convolutional layer of our chosen architecture is followed by classification layers. This aspect of EfficientNet B0 suggests that these classification layers can approximate the resulting probabilities of the network in a relatively similar way between baseline and augmented models, even with the differences observed in the final convolutional layer through some of the CAM metrics (such as the overlap rate with a threshold of $Y = 5$ as seen in Figure 5.14 and Figure 5.15). We also note that this behavior holds across all augmentations for Class-KLD due to the high correlation values between all augmentation pairs (Figure 5.25).

5.2.6 General Analysis

As we can see from the metric distribution boxplots (Figure 5.10, Figure 5.12, Figure 5.14, Figure 5.16, Figure 5.18, Figure 5.20, Figure 5.22, Figure 5.24), although they enable general analyses of the behavior of all augmentations, their individual results are similar across augmentations. This fact makes it challenging to discern meaningful differences between the behaviors of the augmentations by visual analysis of the boxplots alone; this is partly due to the boxplot distribution analysis not lending itself to image-by-image metric comparisons between the augmentations, but allowing only aggregate analyses. The augmentations behave very similarly to one another in these aggregate comparisons, and the correlation maps do not immediately provide much direct insight either. We also investigated segmenting the metric results for each augmentation, segmenting them by cases where both baseline and augmented models predicted the target label correctly, cases where both models classified the image incorrectly, and cases where one model predicted correctly while the other predicted an incorrect class. We did not identify any tendencies that could help us further differentiate behavior between augmentations through this method. A discussion about and some examples of these investigations can be found in Appendix C.

In the course of this work, we also conducted qualitative explorations into the CAMs across augmentations. The purpose was to gain a better idea of what the extremes for each metric would look like but they ultimately did not contribute insights to the analyses we made in our experiment, even if they do provide a good visualization of what differences in CAMs can look like. Some of the results we obtained can be seen in Appendix A.

Although hard to analyze, the differences in correlation across augmentations imply that augmentations present different behaviors. For example, the correlation for overlap rate with higher granularity (Figure 5.15) is significantly lower than for lower granularity (Figure 5.11). This observation indicates that this aspect of the relationship between augmentations follows less of a linear relationship than MAD (Figure 5.21), MSD (Figure 5.23), Pearson (Figure 5.17), and Spearman (Figure 5.19) correlation maps would indicate. In fact, as mentioned earlier, when focusing on general differences between CAMs, the correlation between augmented models and baseline is higher, but when focusing on the most significant regions for each augmentation, their correlation score is lower, which could indicate different patterns of behavior at scale.

We note that when analyzing the correlation maps in absolute terms, we observe that the correlation values are not significantly different across pairs. MAD's correlation map (Figure 5.21), for example, has a minimum correlation between augmentations of 0.58 and a maximum of 0.67, which means the range of variation between correlation values is low. Additionally, in most metrics, the correlation maps score above 0.5, indicating moderate correlations between augmentations for those metrics.

Even though the absolute correlation values are not remarkably different, another analysis we can attempt is a relative one. By evaluating which pairs of augmentations are the most and least correlated across all metrics, supposing the existence of behavior profiles, then specific augmentation pairs should consistently score highly, while others would consistently score lower. To measure this, we count the number of times an augmentation pair is among the most strongly or most weakly correlated pair of augmentations across the correlation maps for each metric. For each CAM metric, if we count the four most and least correlated pairs for each metric's correlation map across augmentations and then aggregate the number of times those pairs appear across all metrics' correlation maps, we will have a relative analysis of the correlations. Following these steps, we get Table 5.1 for the count of strongest pairs, and Table 5.2 for the count of weakest pairs.

Table 5.1 – Frequency table counting how many times each pair of augmentations appeared in the top four most strongly correlated augmentation pairs across all metrics' correlation maps

Color Jitter - Affine Transform	6
Cutmix - Affine Transform	6
Cutmix - Color Jitter	5
Gaussian Blur - Elastic Transform	5
Color Jitter - Elastic Transform	3
Gaussian Blur - Random Cropping	2
Color Jitter - Equalization	1
Gaussian Blur - Elastic Transform	1
Random Crop - Affine Transform	1
Random Crop - Equalization	1
Random Crop - Elastic Transform	1

Table 5.2 – Frequency table counting how many times each pair of augmentations appeared in the top four most weakly correlated augmentation pairs across all metrics' correlation maps

Gaussian Blur - Cutmix	8
Cutmix - Equalization	7
Affine Transform - Equalization	5
Gaussian Blur - Affine Transform	3
Cutmix - Elastic Transform	2
Cutmix - Random Cropping	2
Gaussian Blur - Equalization	2
Cutmix - Color Jitter	1
Cutmix - Affine Transform	1
Color Jitter - Affine Transform	1

Analyzing the table for strongest correlations (Table 5.1), we can tell that Cutmix-Affine appears in the top 4 most strongly correlated for 6 out of 8 metrics, Cutmix-Color Jitter appears 5 out of 8 times, and Color Jitter-Affine Transform appears 6 out of 8 times, with Gaussian Blur-Elastic Transform appearing 7 out of 8 times as well. This data points to Cutmix, Affine Transform, and Color Jitter having stronger correlations across the trio, possibly suggesting the existence of a sort of cluster of augmentation behaviors and that Gaussian Blur and Elastic Transform constitute another cluster or profile of impact. Analyzing the most weakly correlated pairs (Table 5.2), Gaussian Blur-Cutmix and Cutmix-Equalization are the least correlated pairs, with Gaussian Blur-Cutmix appearing 8 out of 8 times, Cutmix-Equalization appearing 7 out of 8, and Affine Transform-Equalization 5 out of 8 times. This observation supplements the earlier observation by suggesting that this Cutmix cluster and the Gaussian cluster have the weakest correlation among them and additionally implying that Equalization may be its own cluster due to it appearing in the four most weakly correlated pairs with both Gaussian and Cutmix 3 and 7 times out of 8, respectively.

6 CONCLUSION

In this work, our goal was to investigate a possible methodology to work with CAMs at scale for the purpose of data augmentation impact analysis. By proposing a generic series of steps and then applying them to a specific set of configurations, we have shown an initial exploration of how one may utilize this methodology, as well as the type of results it may be able to generate.

Metric distribution plots and correlation plots, for our specific selection of metrics, seem apt for generating data about the augmentations as a group. We find that for overlap rate, the metric values get lower as the augmented models become more sensitive to different regions of the test images. The models also do not correlate strongly with each other through their overlap rate, and the correlation diminishes with an increase in region importance. Through MSD and MAD, we note that the average difference between augmented and baseline CAMs tends to follow the same distribution for all augmentations. They also present a moderate correlation (roughly in the range of 0.5 to 0.6) between augmentations and indicate that the augmentations are alike in the magnitude of differences between augmented and baseline CAMs. The Pearson and Spearman metrics show a median correlation at or below the moderate correlation zone for all augmentations, meaning more than half of the CAMs do not have evidence of significant correlation or similarity between augmented and baseline model CAMs, but neither do they have evidence of low or insignificant correlation. They have some correlation, but it is hard to extract information from this observation alone. Lastly, although Class-KLD metric distributions struggle to yield observations by themselves, when combined with the Class-KLD correlation maps they suggest that the dense classification layers of a model are able to approximate the image classification probability distribution in a similar way across augmentations and test images, even if the final convolutional layer contributes different importances for image regions between augmentations.

Although these techniques may be able to produce observations on the behavior of augmentations as groups, they struggle to generate observations about the behaviors of individual data augmentation methods. We note that there are indications of clusters of techniques with higher correlations with one another, but the absolute differences in the correlation values are small. It may be the case that the set of metrics we chose may not fully capture some significant aspect of the differences between CAM or that the data augmentation techniques themselves have a harder-to-measure impact on model behavior than other aspects of model training. Additional work could also be conducted with our experiment to analyze and gauge possible impacts of image bias in the model CAMs induced by the CIFAR-10 dataset, as we did not test the models against other datasets.

The methodology presented in this work provides an initial, generic, scalable series of steps that allow for quantitative exploration of behavioral aspects in the relationship between data augmentation techniques. Impacts on model behavior, in general, are not trivial to analyze. Although our work struggled to investigate the effects of data augmentation on an individual level, the steps delineated in this work constitute a method usable with other metrics, other augmentation methods, other datasets, and for purposes other than analyzing the impact of data augmentation. The purpose of the methodology is to provide a modifiable way to investigate model behavior through CAMs that also provides adaptability to examine other sources of impact different from data augmentation.

Future works could expand on this work and its methodology by focusing on different aspects of it. Utilizing other analysis techniques over the metrics, such as clustering analysis techniques, we may consider characteristics of the augmentation methods as features to investigate groupings of methods with similar characteristics. Another type of future work would be to use datasets that have some ground truth for important image regions, allowing the analysis of differences between CAMs and ground truth. Other works could explore different convolutional architectures and datasets, other techniques to determine the importance of pixels and investigate the impact of applying multiple data augmentation techniques used in conjunction instead of individually, comparing CAMs between augmented models directly instead of through a baseline, or other analysis metrics to measure CAM similarities. These investigations can reveal different insights into the relationships between important image regions and different aspects of the learning process for CNNs.

REFERENCES

ANGELOV, P.; SOARES, E. Towards explainable deep neural networks (xdnn). **Neural Networks**, Elsevier, v. 130, p. 185–194, 2020.

BYLINSKII, Z. et al. What do different evaluation metrics tell us about saliency models? **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 41, n. 3, p. 740–757, 2018.

CAO, C. et al. A survey of mix-based data augmentation: Taxonomy, methods, applications, and explainability. **arXiv preprint arXiv:2212.10888**, 2022.

CHEN, P. et al. Gridmask data augmentation. **arXiv preprint arXiv:2001.04086**, 2020.

CUBUK, E. D. et al. Autoaugment: Learning augmentation strategies from data. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.: s.n.], 2019. p. 113–123.

FACELI, K. et al. Inteligência artificial: uma abordagem de aprendizado de máquina. 2021.

FAN, F.-L. et al. On interpretability of artificial neural networks: A survey. **IEEE Transactions on Radiation and Plasma Medical Sciences**, IEEE, v. 5, n. 6, p. 741–760, 2021.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep learning**. [S.l.]: MIT press, 2016.

HAAR, L. V.; ELVIRA, T.; OCHOA, O. An analysis of explainability methods for convolutional neural networks. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 117, p. 105606, 2023.

HE, K. et al. Deep residual learning for image recognition. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 770–778.

HUANG, G. et al. Densely connected convolutional networks. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2017. p. 4700–4708.

HUSSAIN, M.; BIRD, J. J.; FARIA, D. R. A study on cnn transfer learning for image classification. In: SPRINGER. **Advances in Computational Intelligence Systems: Contributions Presented at the 18th UK Workshop on Computational Intelligence, September 5-7, 2018, Nottingham, UK**. [S.l.], 2019. p. 191–202.

INOUE, H. Data augmentation by pairing samples for images classification. **arXiv preprint arXiv:1801.02929**, 2018.

JIANG, P.-T. et al. Layercam: Exploring hierarchical class activation maps for localization. **IEEE Transactions on Image Processing**, IEEE, v. 30, p. 5875–5888, 2021.

KHALIFA, N. E.; LOEY, M.; MIRJALILI, S. A comprehensive survey of recent trends in deep learning for digital images augmentation. **Artificial Intelligence Review**, Springer, p. 1–27, 2022.

KRIZHEVSKY, A.; HINTON, G. et al. Learning multiple layers of features from tiny images. Toronto, ON, Canada, 2009.

LI, W. et al. Data augmentation for hyperspectral image classification with deep cnn. **IEEE Geoscience and Remote Sensing Letters**, IEEE, v. 16, n. 4, p. 593–597, 2018.

LIU, S.; DENG, W. Very deep convolutional neural network based image classification using small training sample size. In: IEEE. **2015 3rd IAPR Asian conference on pattern recognition (ACPR)**. [S.l.], 2015. p. 730–734.

LIU, X.; WANG, X.; MATWIN, S. Interpretable deep convolutional neural networks via meta-learning. In: IEEE. **2018 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2018. p. 1–9.

MITCHELL, T. Introduction to machine learning. **Machine learning**, McGraw-hill New York, v. 7, p. 2–5, 1997.

MUHAMMAD, M. B.; YEASIN, M. Eigen-cam: Class activation map using principal components. In: IEEE. **2020 international joint conference on neural networks (IJCNN)**. [S.l.], 2020. p. 1–7.

MURPHY, K. P. **Machine learning: a probabilistic perspective**. [S.l.]: MIT press, 2012.

NANNI, L. et al. Comparison of different image data augmentation approaches. **Journal of imaging**, MDPI, v. 7, n. 12, p. 254, 2021.

O'GARA, S.; MCGUINNESS, K. Comparing data augmentation strategies for deep image classification. Technological University Dublin, 2019.

OLAH, C. et al. The building blocks of interpretability. **Distill**, v. 3, n. 3, p. e10, 2018.

PATTERSON, J.; GIBSON, A. **Deep learning: A practitioner's approach**. [S.l.]: " O'Reilly Media, Inc.", 2017.

PEREZ, L.; WANG, J. The effectiveness of data augmentation in image classification using deep learning. **arXiv preprint arXiv:1712.04621**, 2017.

PULS, E. d. S.; TODESCATO, M. V.; CARBONERA, J. L. An evaluation of pre-trained models for feature extraction in image classification. In: **ICEIS**. [S.l.: s.n.], 2024. p. 123–135.

RADHAKRISHNAN, A. et al. Patchnet: interpretable neural networks for image classification. **arXiv preprint arXiv:1705.08078**, 2017.

RATNER, A. J. et al. Learning to compose domain-specific transformations for data augmentation. **Advances in neural information processing systems**, v. 30, 2017.

REZATOFIGHI, H. et al. Generalized intersection over union: A metric and a loss for bounding box regression. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.: s.n.], 2019. p. 658–666.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach (4th Edition)**. Pearson, 2020. ISBN 9780134610993. Available from Internet: <http://aima.cs.berkeley.edu/>.

SANTOS, F. A. O. et al. On the impact of interpretability methods in active image augmentation method. **Logic Journal of the IGPL**, Oxford University Press, v. 30, n. 4, p. 611–621, 2022.

SCHULZ, K. et al. Restricting the flow: Information bottlenecks for attribution. **arXiv preprint arXiv:2001.00396**, 2020.

SELVARAJU, R. R. et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2017. p. 618–626.

SHORTEN, C.; KHOSHGOFTAAR, T. M. A survey on image data augmentation for deep learning. **Journal of big data**, SpringerOpen, v. 6, n. 1, p. 1–48, 2019.

TAN, M.; LE, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In: PMLR. **International conference on machine learning**. [S.l.], 2019. p. 6105–6114.

TANG, J.; SHARMA, M.; ZHANG, R. **Explaining the Effect of Data Augmentation on Image Classification Tasks**. [S.l.]: Stanford University, 2020.

TODESCATO, M. V. et al. Multiscale context features for geological image classification. In: **ICEIS (1)**. [S.l.: s.n.], 2023. p. 407–418.

TODESCATO, M. V. et al. Multiscale patch-based feature graphs for image classification. **Expert Systems with Applications**, Elsevier, v. 235, p. 121116, 2024.

UDDIN, A. et al. Saliencymix: A saliency guided data augmentation strategy for better regularization. **arXiv preprint arXiv:2006.01791**, 2020.

WANG, H. et al. Score-cam: Score-weighted visual explanations for convolutional neural networks. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops**. [S.l.: s.n.], 2020. p. 24–25.

WON, S.; BAE, S.-H.; KIM, S. T. Analyzing effects of mixed sample data augmentation on model interpretability. **arXiv preprint arXiv:2303.14608**, 2023.

YANG, S. et al. Image data augmentation for deep learning: A survey. **arXiv preprint arXiv:2204.08610**, 2022.

ZHANG, Y. et al. A survey on neural network interpretability. **IEEE Transactions on Emerging Topics in Computational Intelligence**, IEEE, v. 5, n. 5, p. 726–742, 2021.

ZHOU, B. et al. Learning deep features for discriminative localization. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 2921–2929.

APPENDIX A — CAM EXAMPLES

In this section, we share some of the CAMs we obtained after training. To generate these qualitative analyses of CAM images, we analyzed the metric values of all augmentations to find the test images whose CAM metrics had the highest mean or standard deviation across augmentations. These grid comparisons between CAMs help us visualize the extremes for each metric, as well as have a better idea of what CAM differences might visually look like.

Figure A.1 shows the CAM comparison for the image with the highest mean for overlap rate (20), Figure A.2 for highest standard deviation for overlap rate (20), Figure A.3 for highest mean for overlap rate (10), Figure A.4 for the highest standard deviation for overlap rate (10), Figure A.5 for highest mean for overlap rate (5), Figure A.6 for the highest standard deviation for overlap rate (5), Figure A.7 shows this comparison for the image with highest mean Pearson Correlation, Figure A.8 does so for the highest standard deviation for Pearosn Correlation, Figure A.9 for highest mean Spearman Correlation, Figure A.10 for highest standard deviation for Spearman Correlation, Figure A.11 shows the same comparison for the image with the highest mean MAD, Figure A.12 does so for the highest standard deviation for MAD, Figure A.13 shows the comparison for highest mean MSD, Figure A.14 does so for the highest standard deviation for MSD, Figure A.15 shows the comparison for highest mean Class-KLD, and Figure A.16 shows the CAM comparison for the image with highest standard deviation for the Class-KLD metric.

Figure A.1 – Grid comparing the model CAMs for the test image that produced the highest mean for the overlap rate (20) metric across all test images and augmentations.

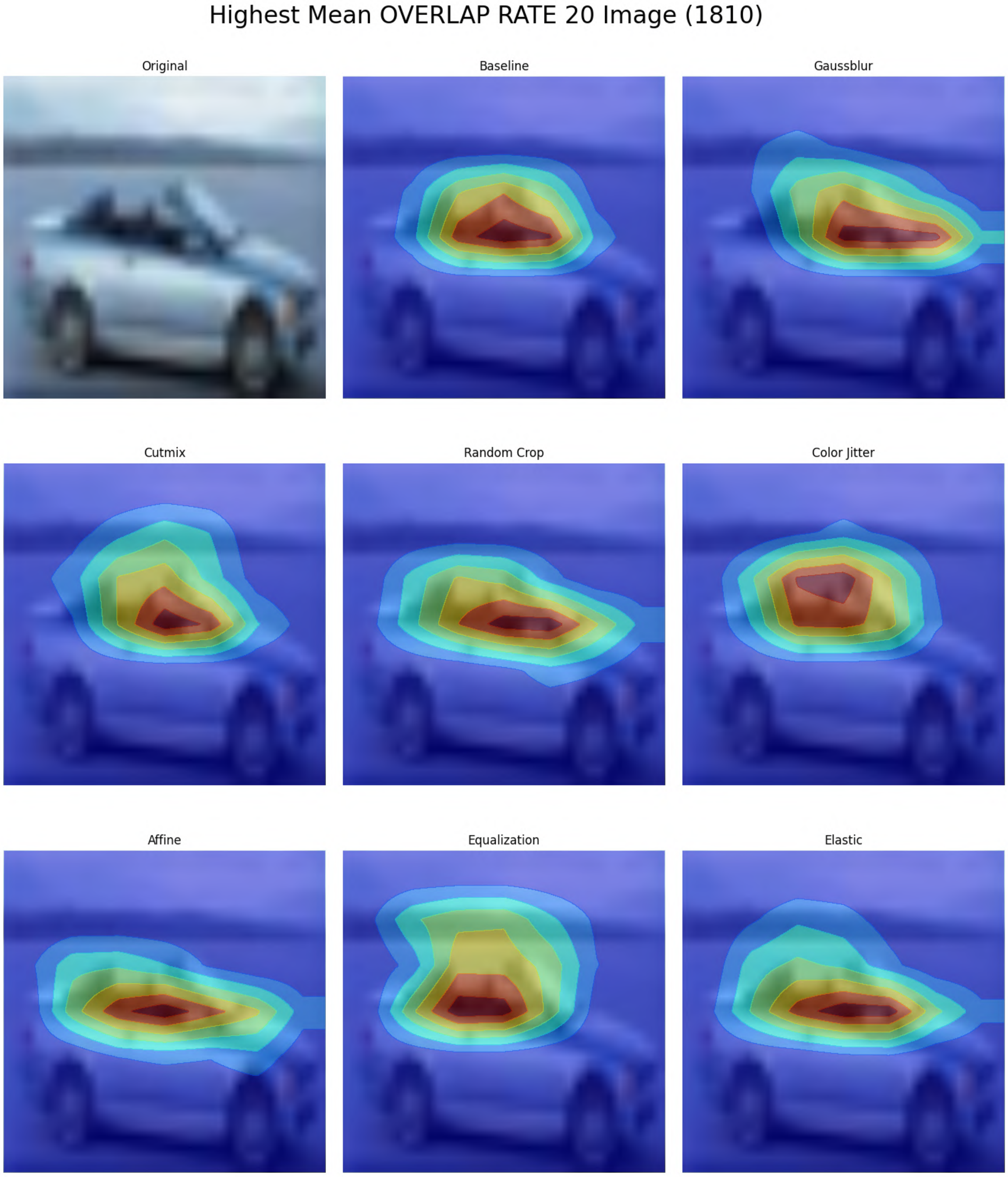


Figure A.2 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the overlap rate (20) metric across all test images and augmentations.

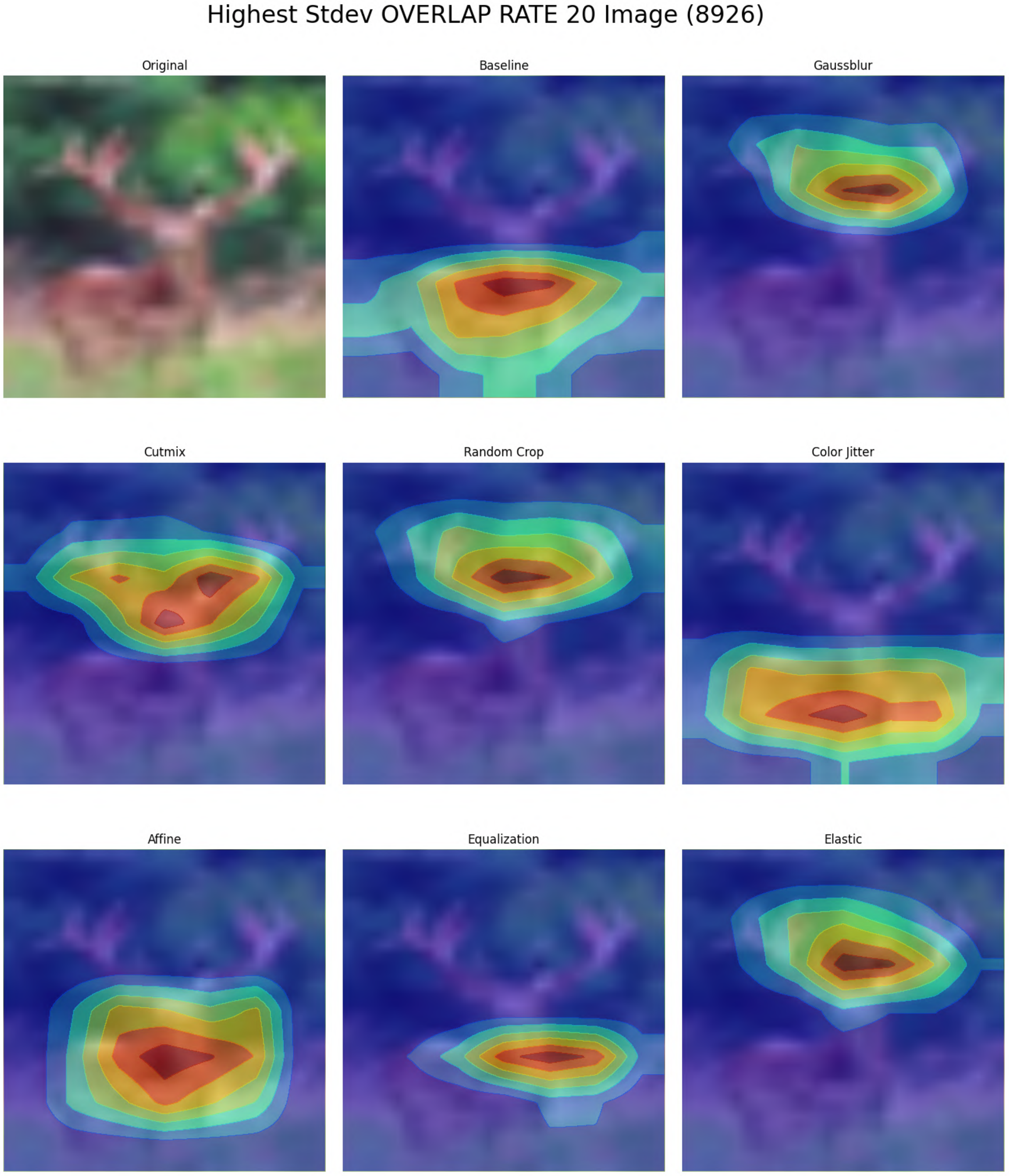


Figure A.3 – Grid comparing the model CAMs for the test image that produced the highest mean for the overlap rate (10) metric across all test images and augmentations.

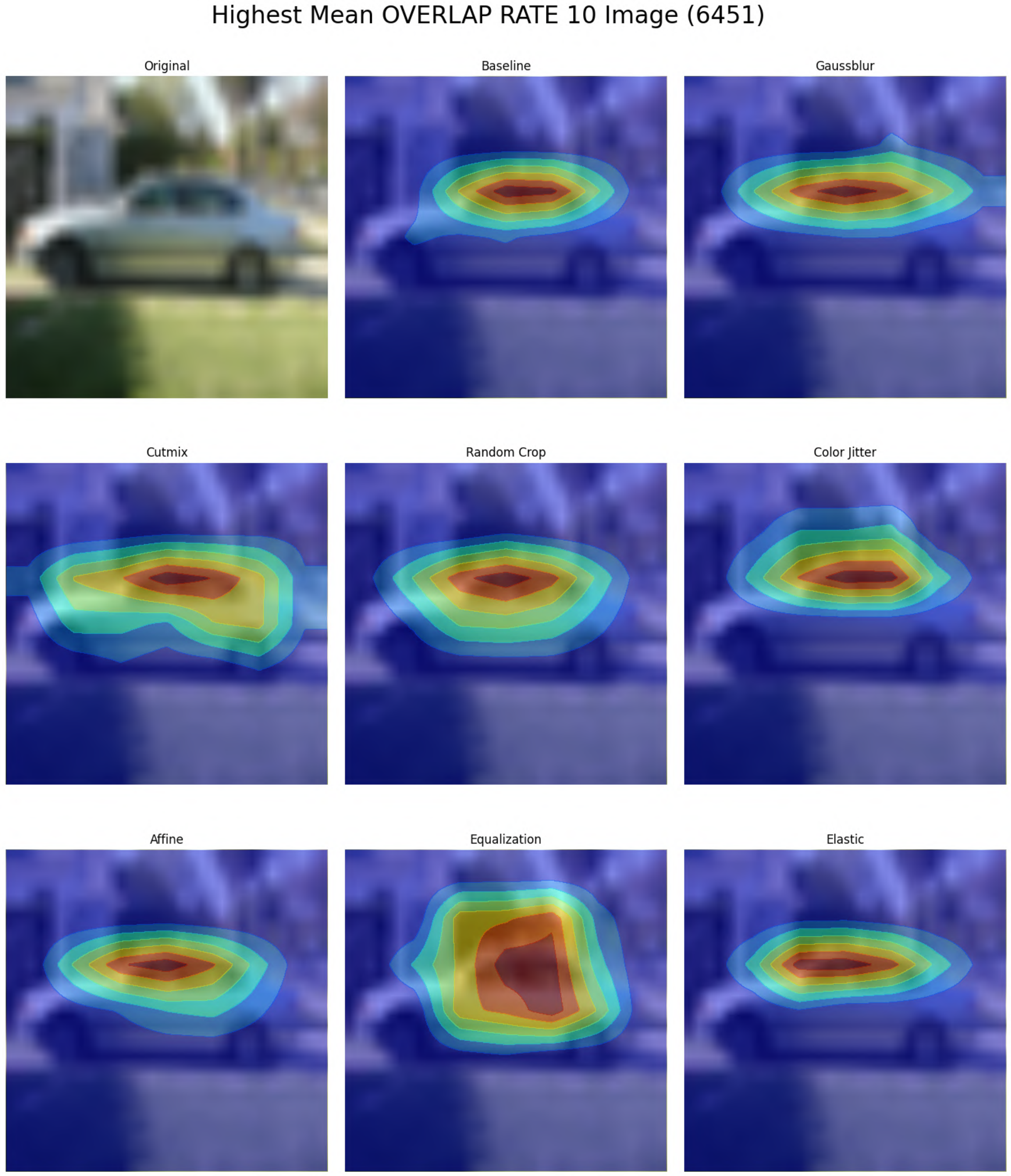


Figure A.4 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the overlap rate (10) metric across all test images and augmentations.

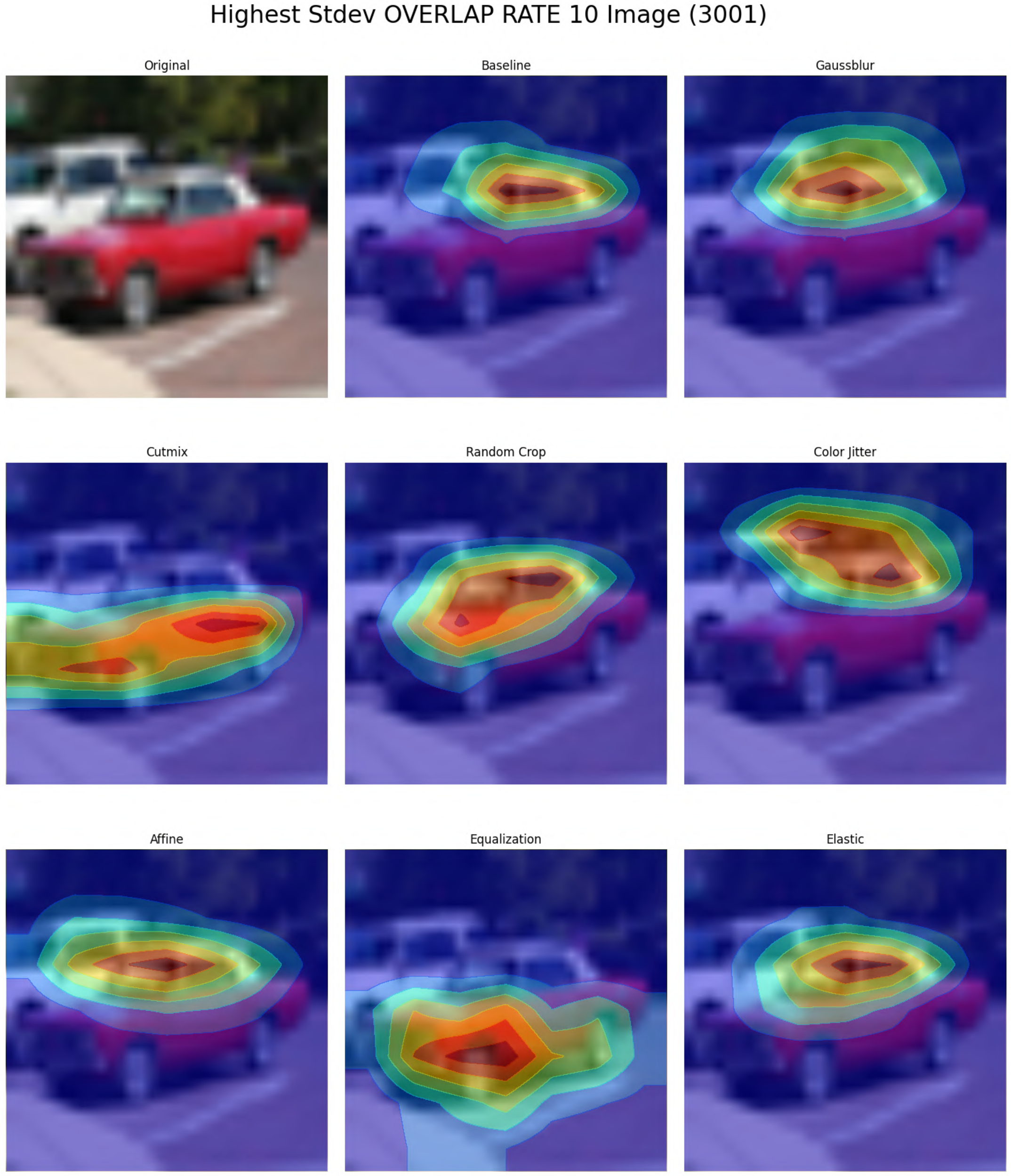


Figure A.5 – Grid comparing the model CAMs for the test image that produced the highest mean for the overlap rate (5) metric across all test images and augmentations.

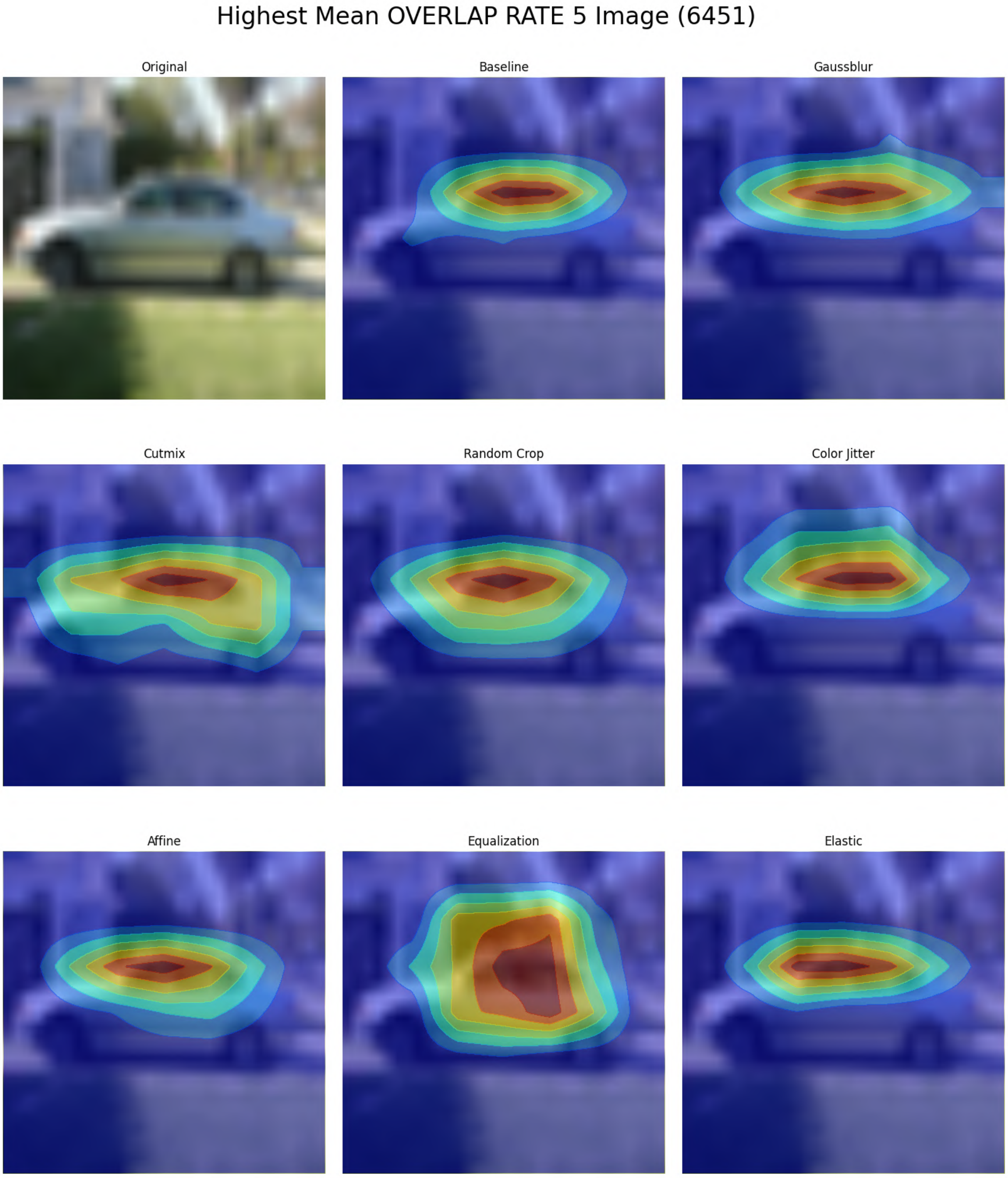


Figure A.6 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the overlap rate (5) metric across all test images and augmentations.

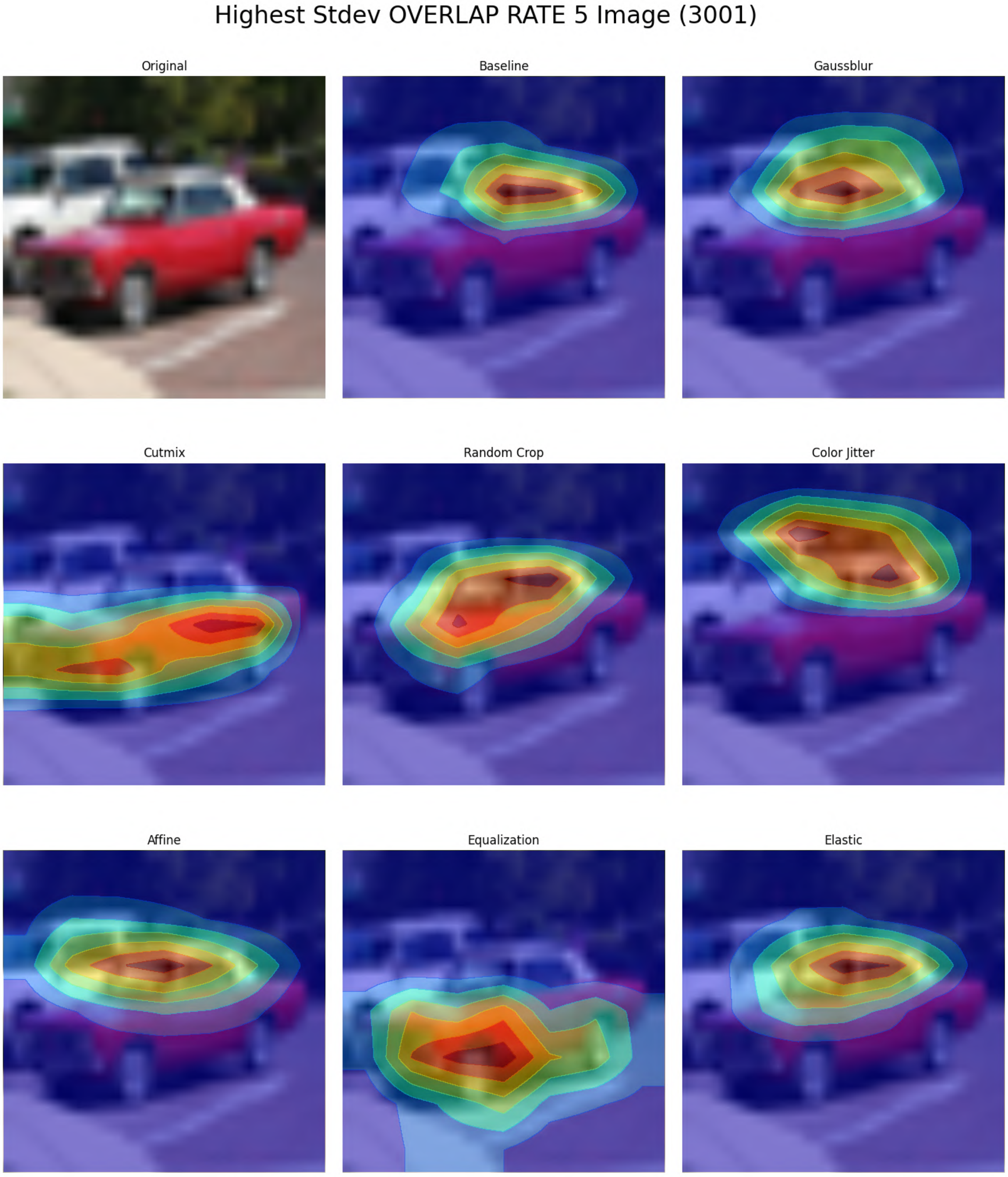


Figure A.7 – Grid comparing the model CAMs for the test image that produced the highest mean for the Pearson Correlation metric across all test images and augmentations.

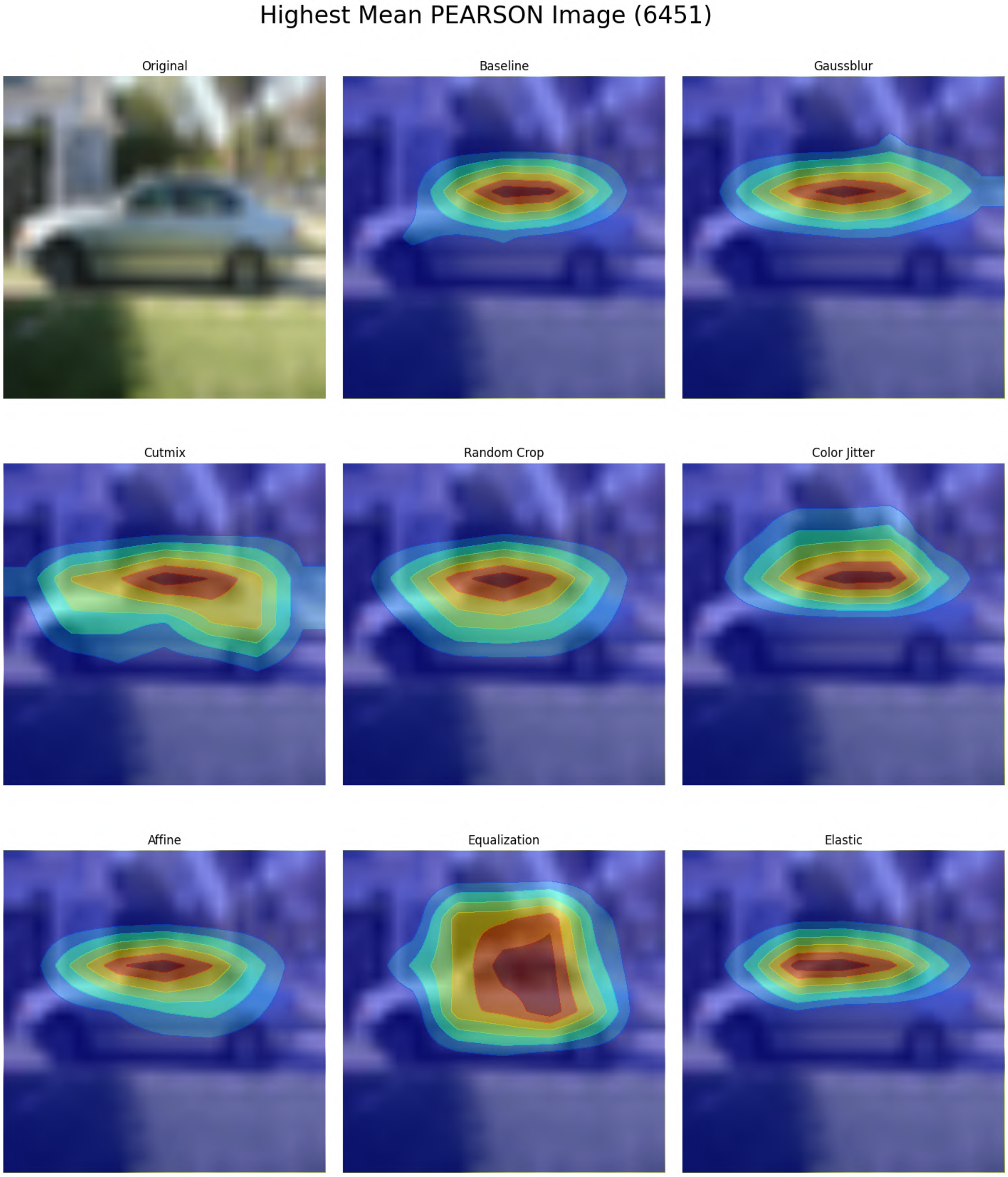


Figure A.8 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the Pearson Correlation metric across all test images and augmentations.

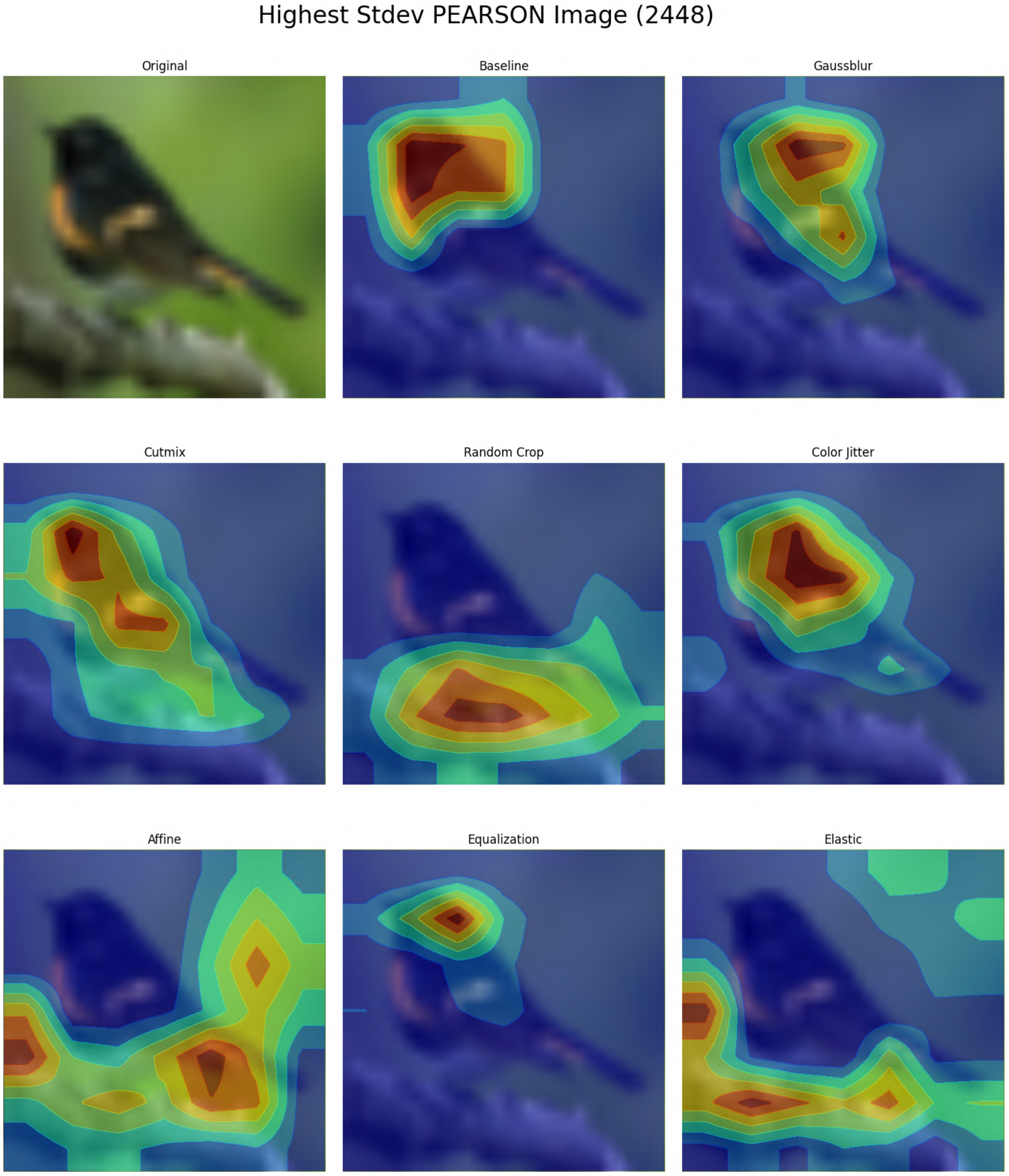


Figure A.9 – Grid comparing the model CAMs for the test image that produced the highest mean for the Spearman Correlation metric across all test images and augmentations.

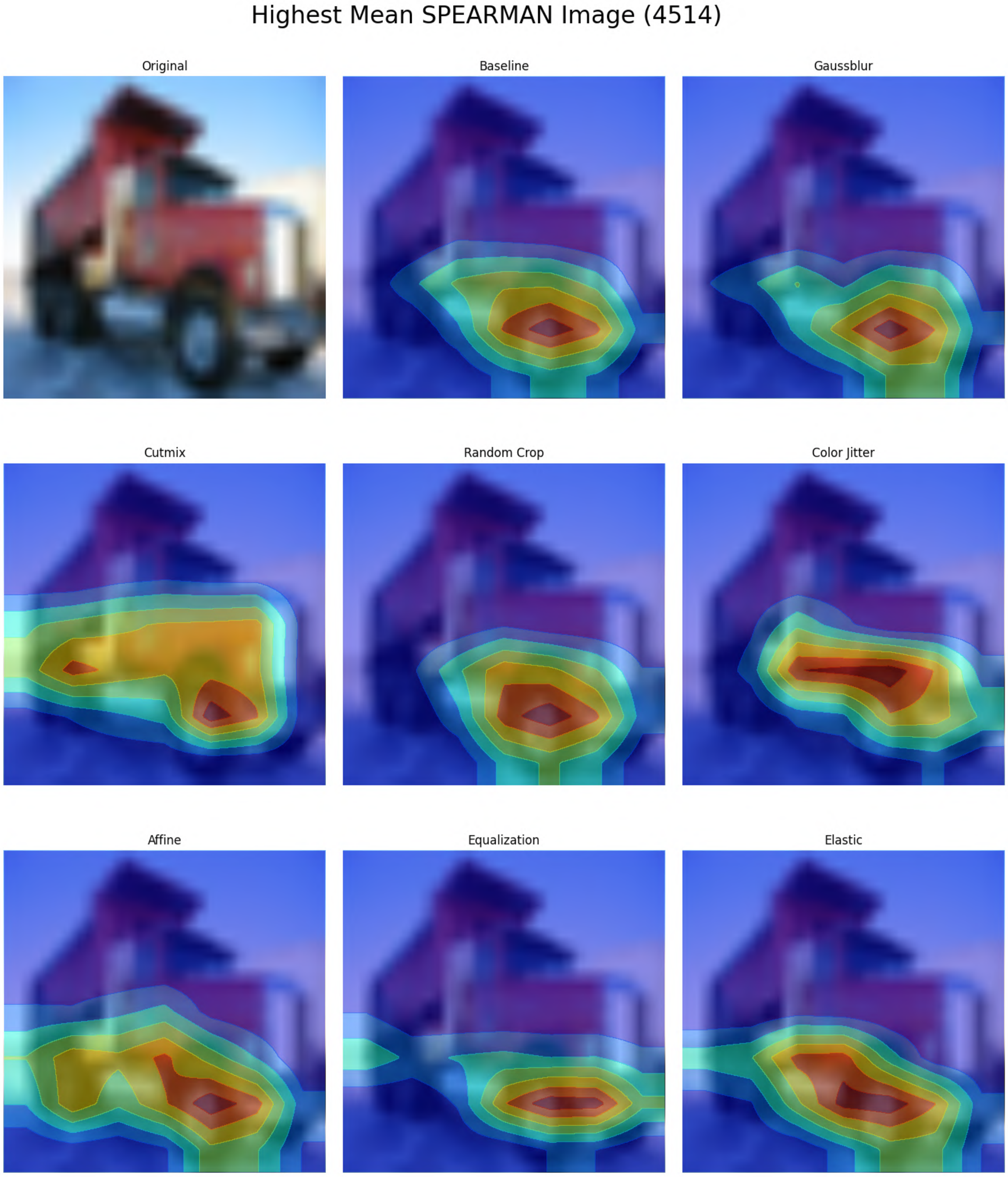


Figure A.10 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the Spearman Correlation metric across all test images and augmentations.

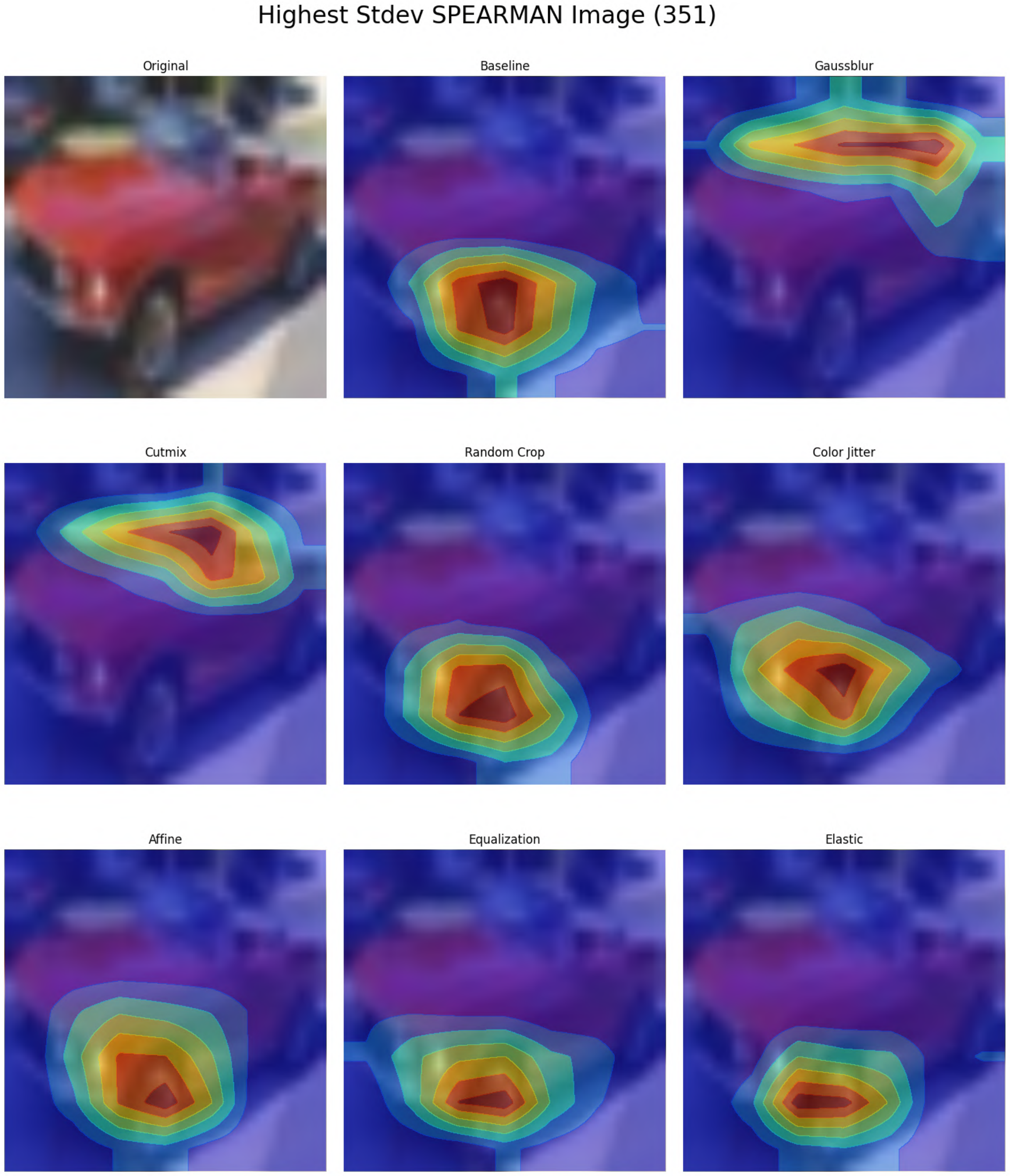


Figure A.11 – Grid comparing the model CAMs for the test image that produced the highest mean for the MAD metric across all test images and augmentations.

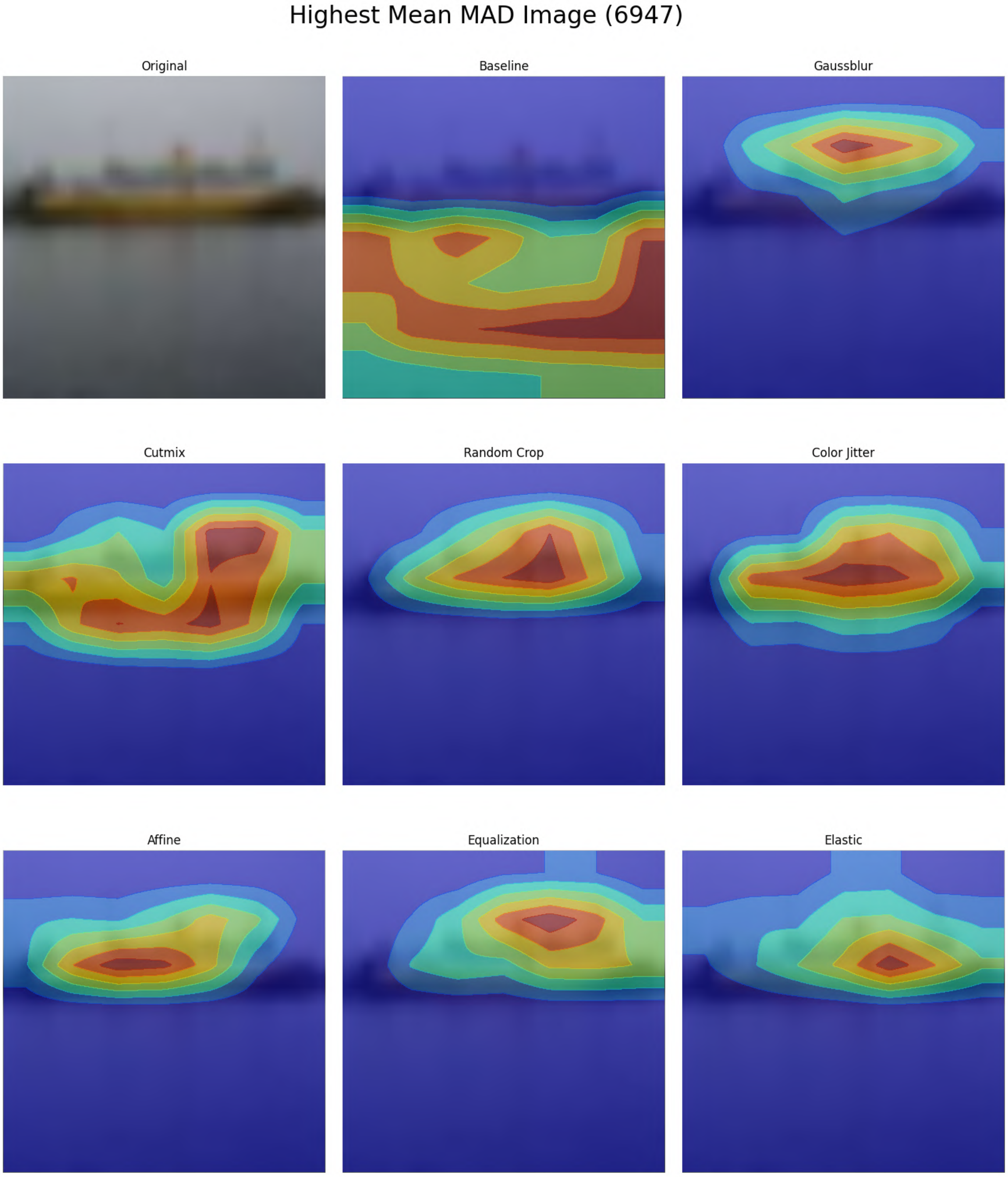


Figure A.12 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the MAD metric across all test images and augmentations.

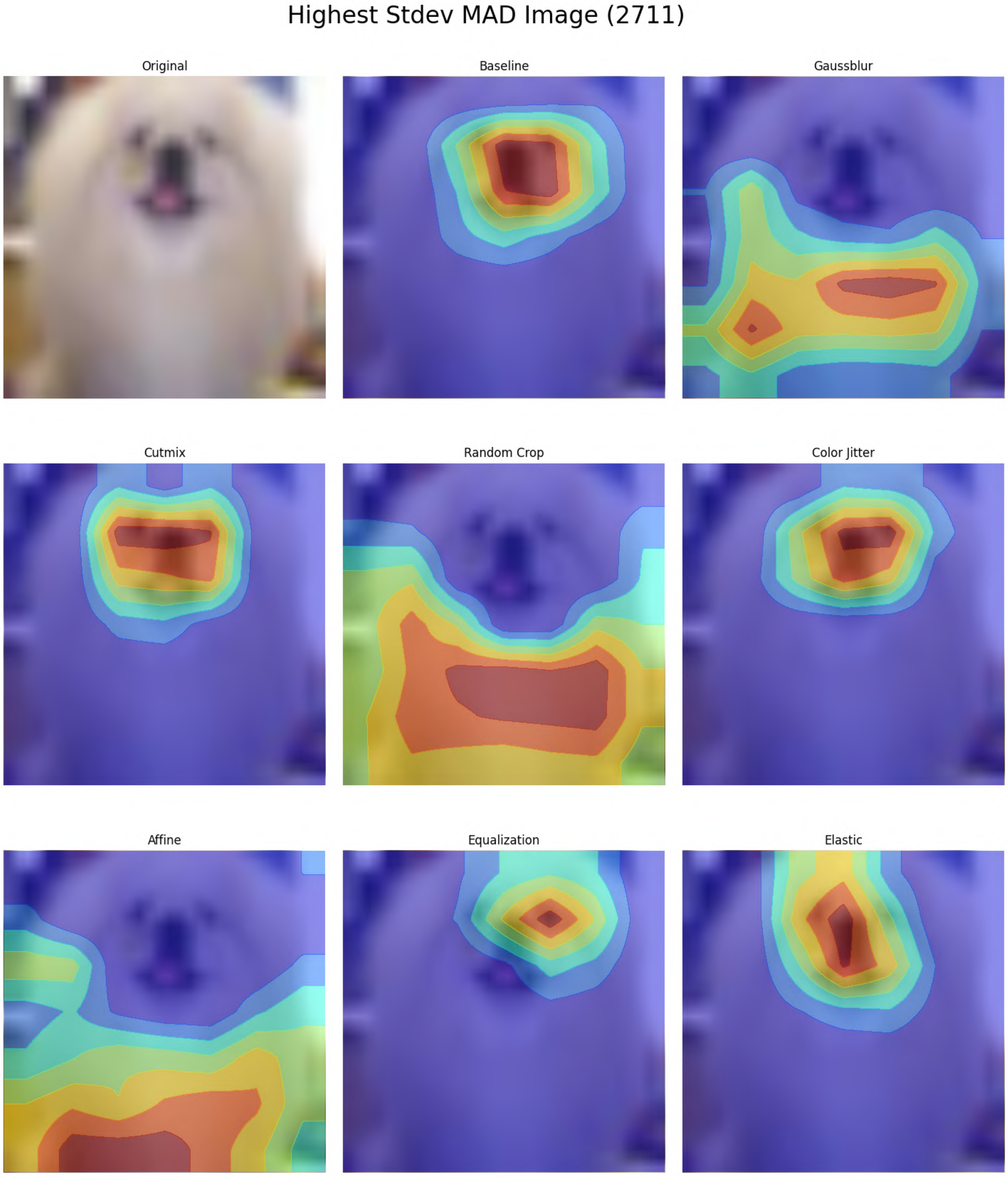


Figure A.13 – Grid comparing the model CAMs for the test image that produced the highest mean for the MSD metric across all test images and augmentations.

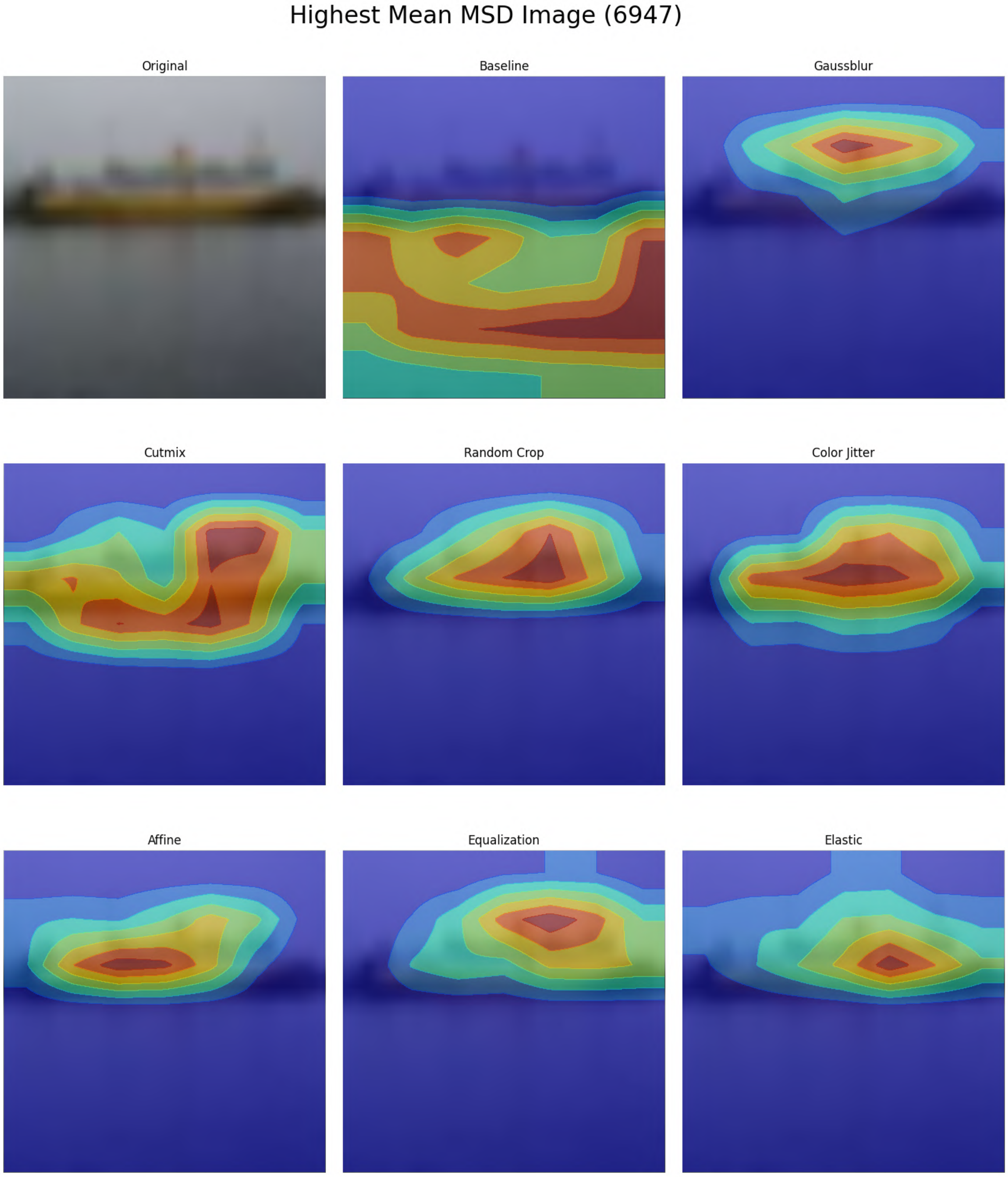


Figure A.14 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the MSD metric across all test images and augmentations.

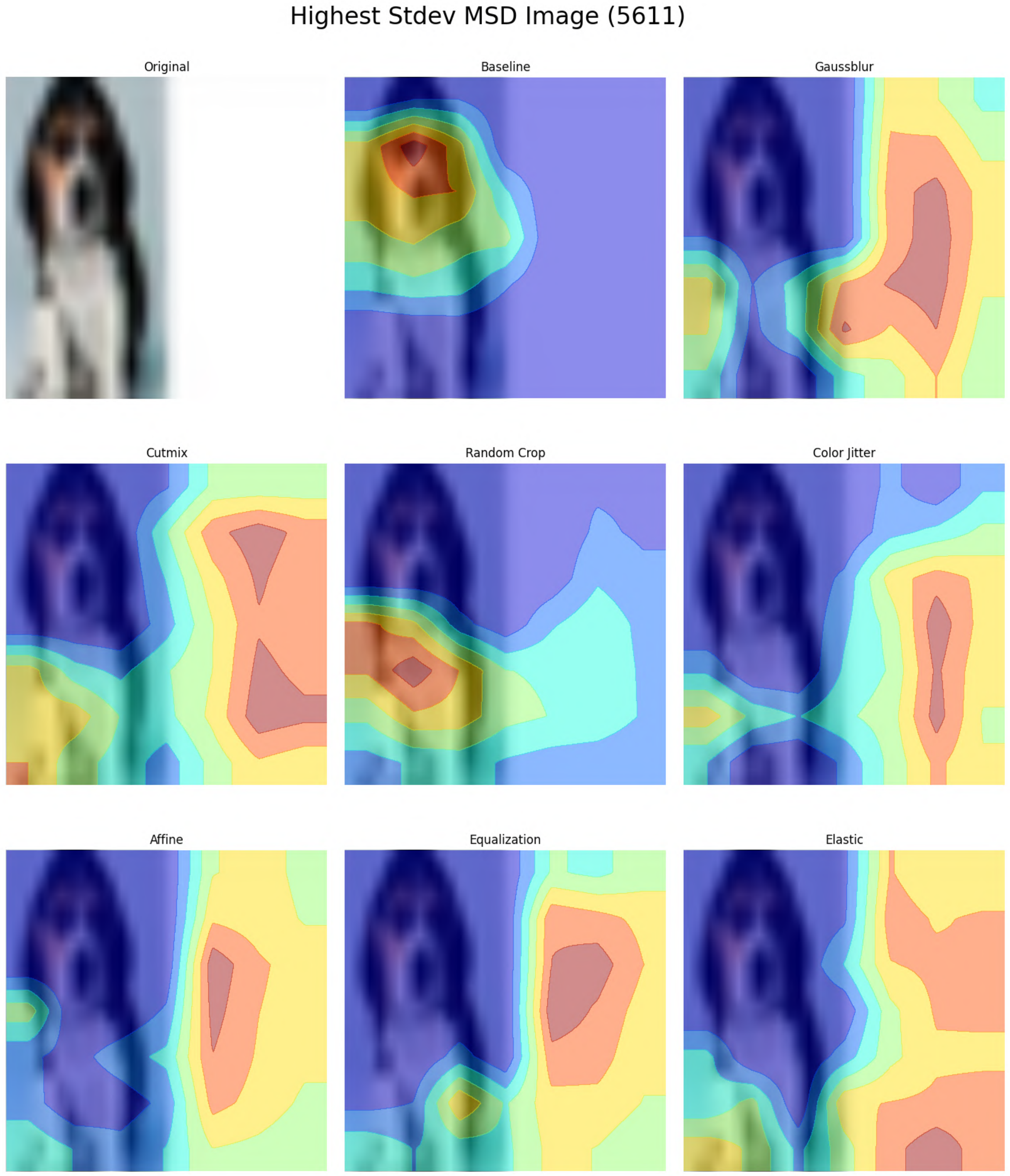


Figure A.15 – Grid comparing the model CAMs for the test image that produced the highest mean for the Class-KLD metric across all test images and augmentations.

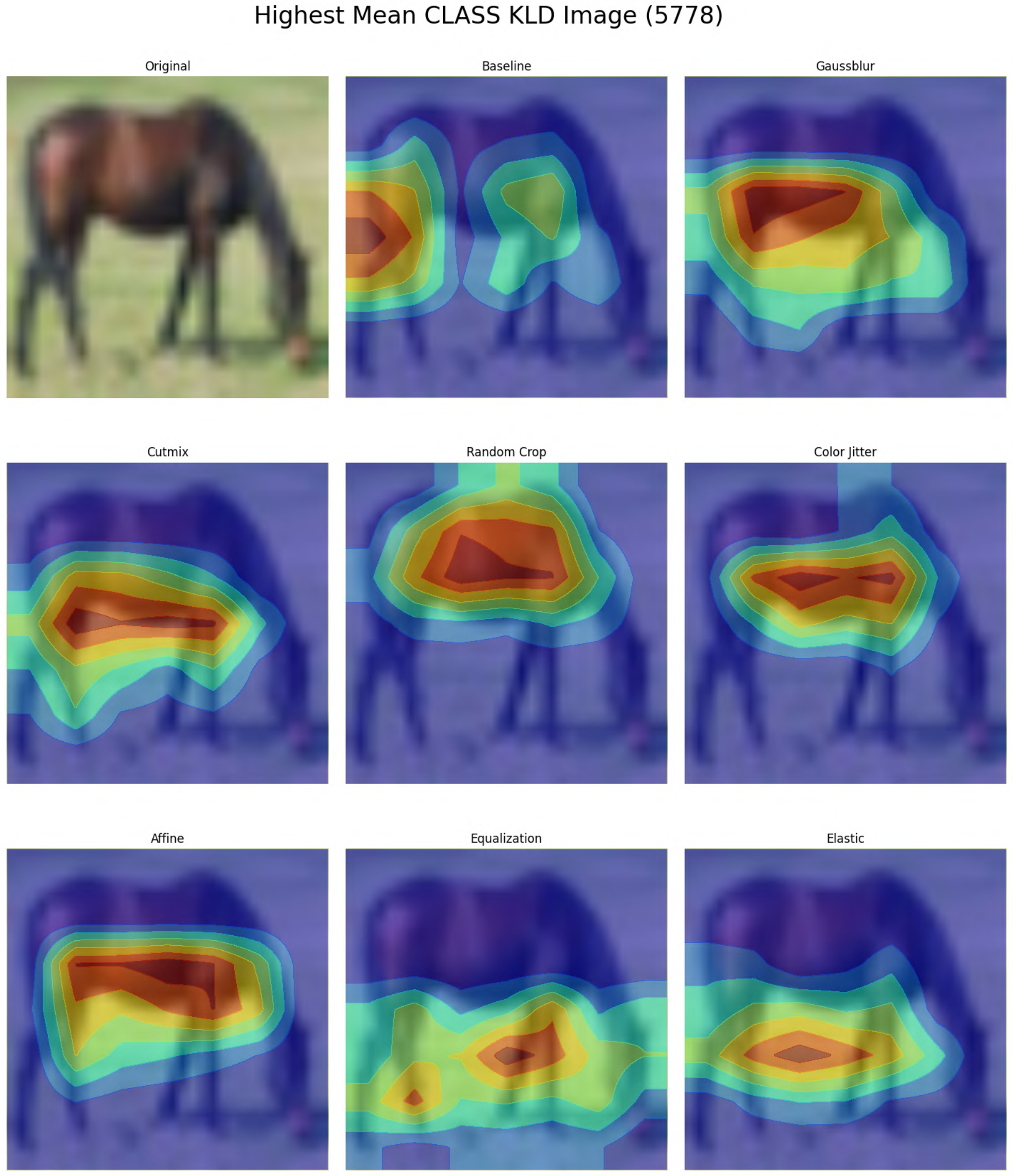
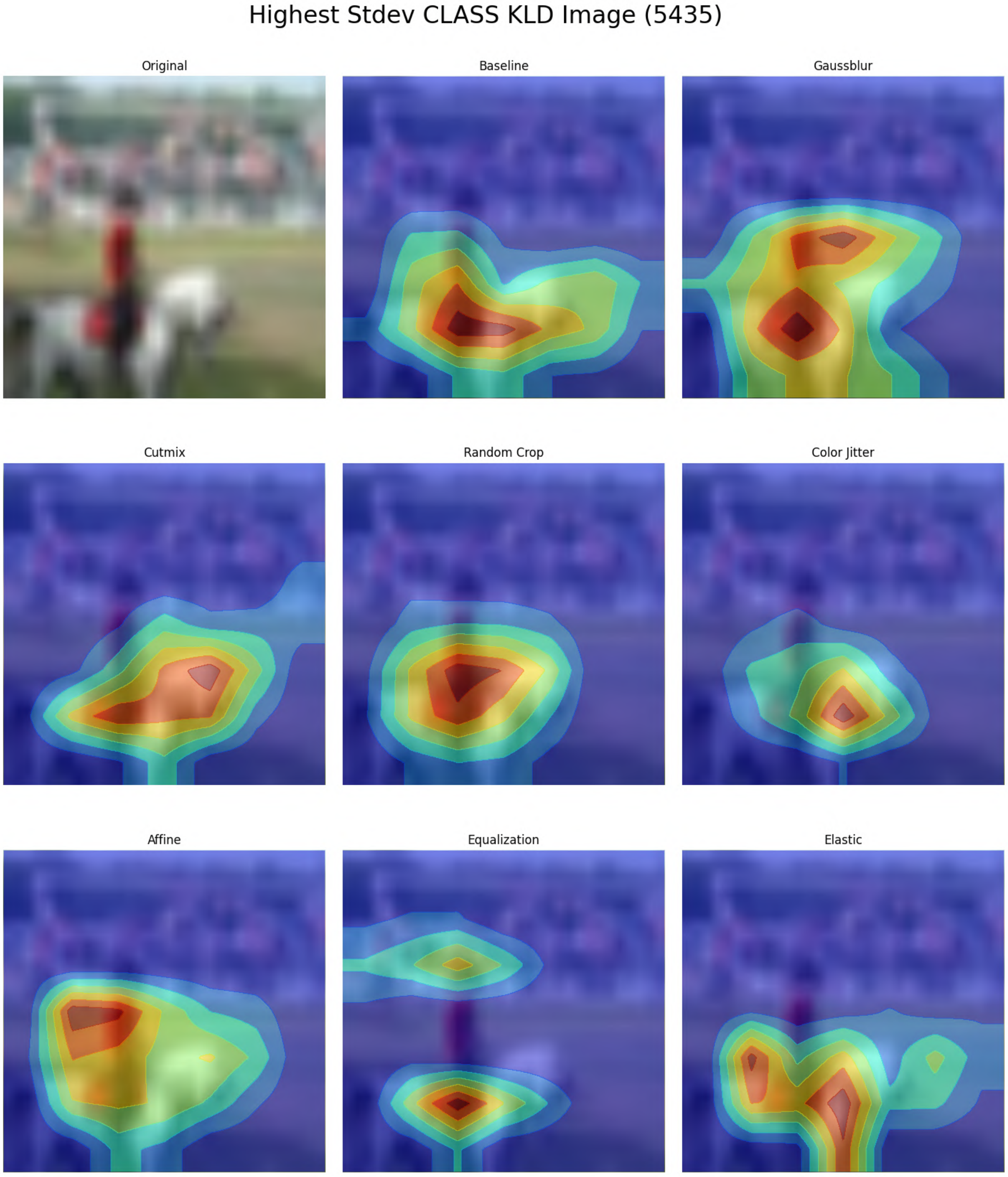


Figure A.16 – Grid comparing the model CAMs for the test image that produced the highest standard deviation for the Class-KLD metric across all test images and augmentations.



APPENDIX B — CONTROLLING UNDESIRED SOURCES OF IMPACT

As we are interested solely in analyzing the effects of data augmentation, part of our work consisted of controlling for undesired sources of impact during the construction of the augmented datasets and the training of our models, such that the impact differences we observed would come only from the data augmentation methods.

To accomplish this, we first needed to ensure that the algorithms from the neural network libraries we used behaved deterministically. To do so, we explicitly turned off non-deterministic behavior in both PyTorch and CUDA libraries, but to guarantee that other random generation functionality present in the augmentations behaved deterministically, we also manually seeded Numpy, Random, Torch, and Torchvision libraries wherever needed, including during the instantiation of the initial weights for the models. Additionally, when using Torch dataloaders to load batches of data from the datasets, the shuffle functionality was turned off, guaranteeing that models would load batches of images in the order they appear in the respective dataset.

After ensuring that these impact sources were controlled for, the next element that needed looking into was the construction of the augmented datasets and the ordering of the images therein. Although it would be impossible to control the image content of the datasets, as each augmentation will generate its own set of augmented images, it was important to control the order in which the images were ordered inside of the dataset.

As mentioned previously, every model uses the same train-validation split based on the training part of the CIFAR10 dataset. Of the 50,000 training images, 45,000 are used for training and 5,000 for validation of each epoch. Once we establish the baseline train dataset, an initial, purely augmented dataset is instantiated from the baseline dataset by generating one augmented image for every baseline image.

This process generates images in the order they appear in the baseline dataset, such that the 45,000 augmented images follow the same indexing order as the baseline, making it so that an image with index 42 in the augmented dataset will use the image with index 42 of the baseline dataset as the source for its augmentation.

For each augmentation method, we then concatenate the augmented images onto a copy of the baseline dataset for a total of 90,000 training images in this augmented dataset. After this, we shuffle the 90,000 images according to a specific, separate, stable seed. The seed used for this sampling is the same for every augmentation method, ensuring that the image index order will be the exact same across all augmented datasets. Note that the sampling seed is separate from the starting state seeds.

APPENDIX C — SEGMENTED RESULTS

Segmentation of the metric results was motivated by an issue we predicted could arise from the choice of using the predicted label for Grad-CAM instead of the ground truth label. Given that it would be reasonable to suppose each class has its own potentially distinct identifying features, if two different models predicted different labels for a given image, then it would be reasonable to expect their CAMs to be more dissimilar than they would be if they'd both predicted the same class. This case would be worrisome insofar as it could add hard-to-interpret data points into the CAM metric distributions. If, for example, two models predicted a large number of test images incorrectly, and both models made different predictions from one another at every step, then we could expect the metric results to be significantly different from each other, and it could become hard to analyze the differences in their patterns.

To measure the impact of these cases, we ran experiments segmenting our results into combinations of both models being correct, one model predicting correctly and the other being wrong, as well as both being wrong and found the impact of these cases on the metric distributions to be minimal for all metrics. As the boxplots for these results are numerous and do not show different patterns across metrics, we show a selection of these segmented boxplots for one specific metric and all augmentations.

Figure C.1 shows the segmented boxplot for affine transform method, Figure C.2 shows it for color jitter, Figure C.3 for cutmix, Figure C.4 for elastic transform, Figure C.5 for equalization, Figure C.6 for Gaussian blur, and Figure C.7 for random cropping

Figure C.1 – Segmented boxplot for Affine Transform and Overlap Rate (20), showing the distribution of the metric values across segmentations.

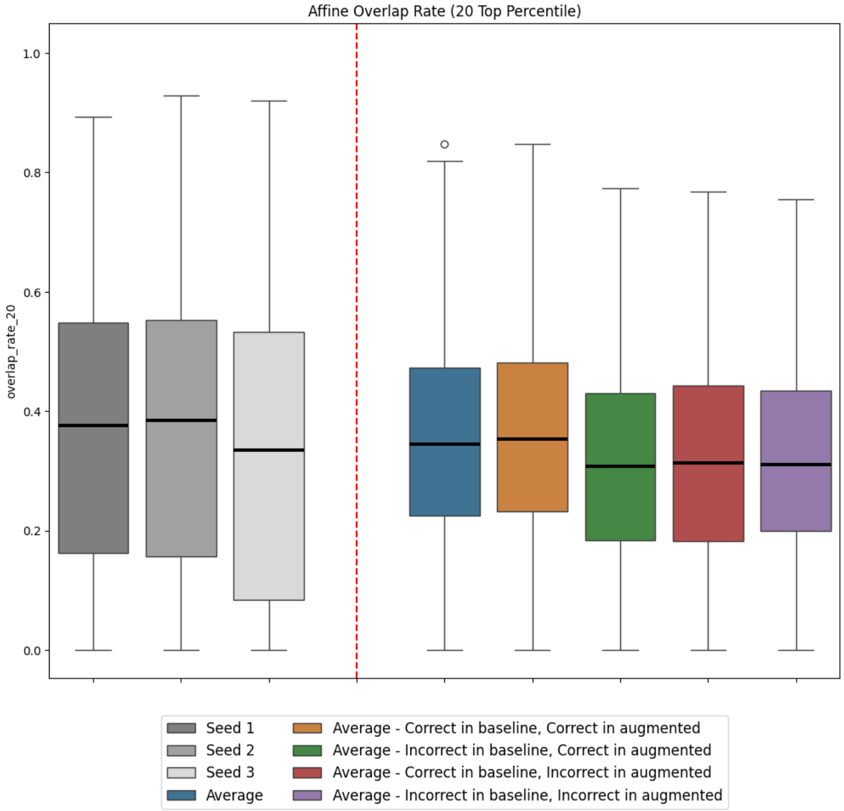


Figure C.2 – Segmented boxplot for Color Jitter and Overlap Rate (20), showing the distribution of the metric values across segmentations.

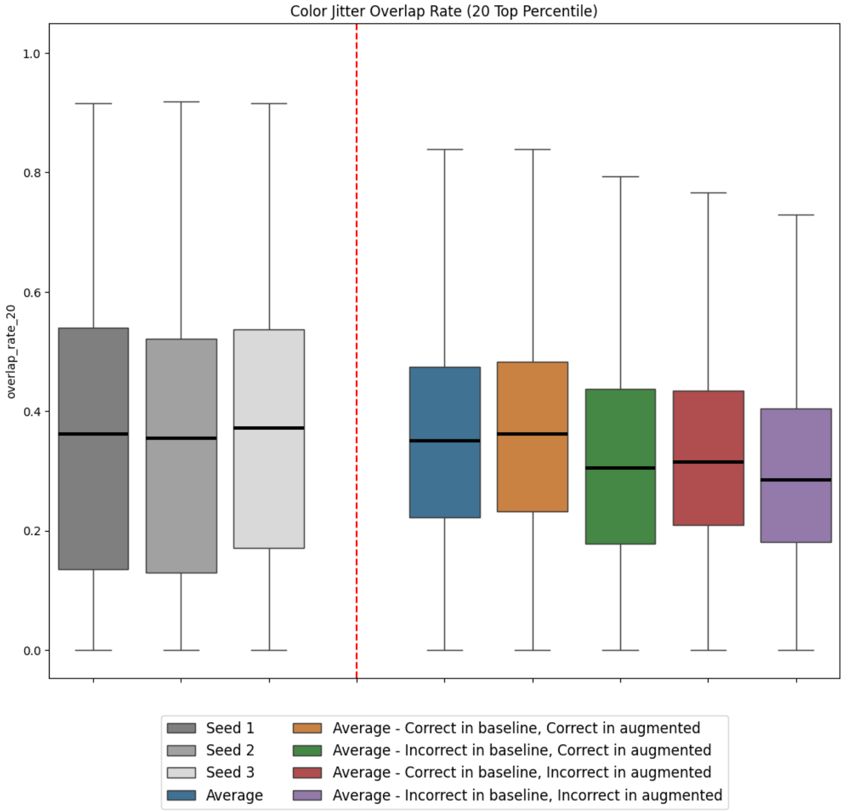


Figure C.3 – Segmented boxplot for Cutmix and Overlap Rate (20), showing the distribution of the metric values across segmentations.

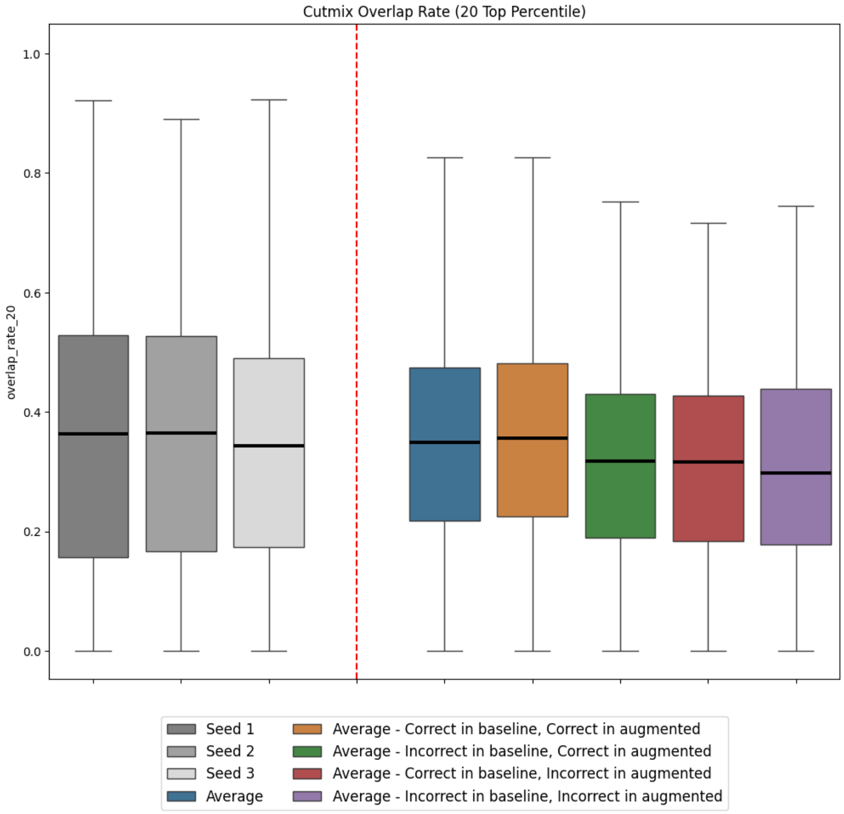


Figure C.4 – Segmented boxplot for Elastic Transform and Overlap Rate (20), showing the distribution of the metric values across segmentations.

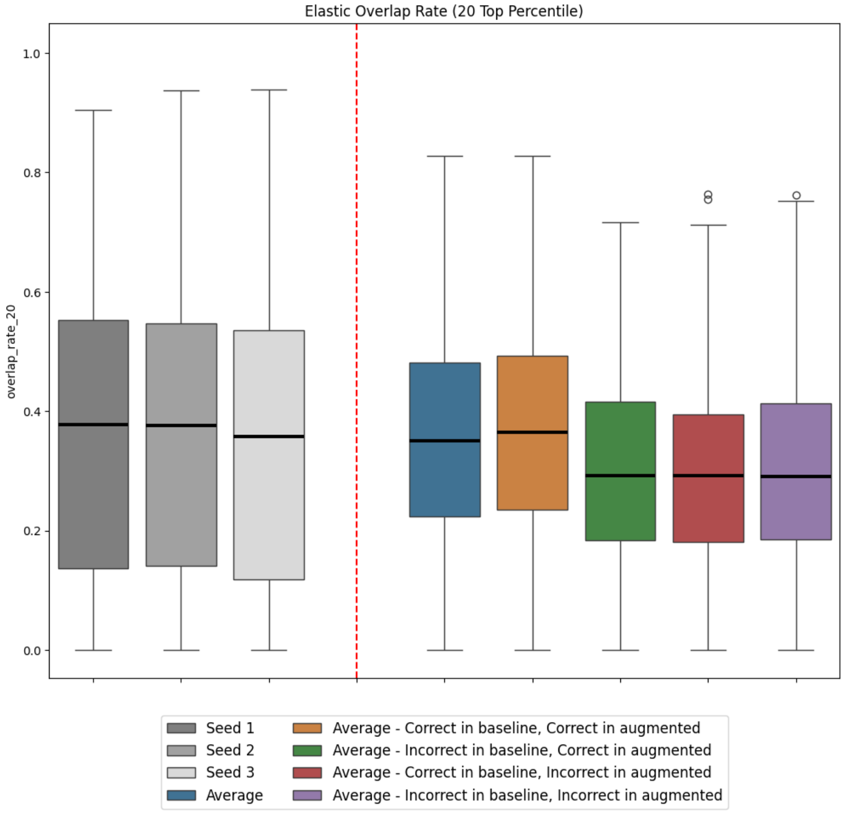


Figure C.5 – Segmented boxplot for Equalization and Overlap Rate (20), showing the distribution of the metric values across segmentations.

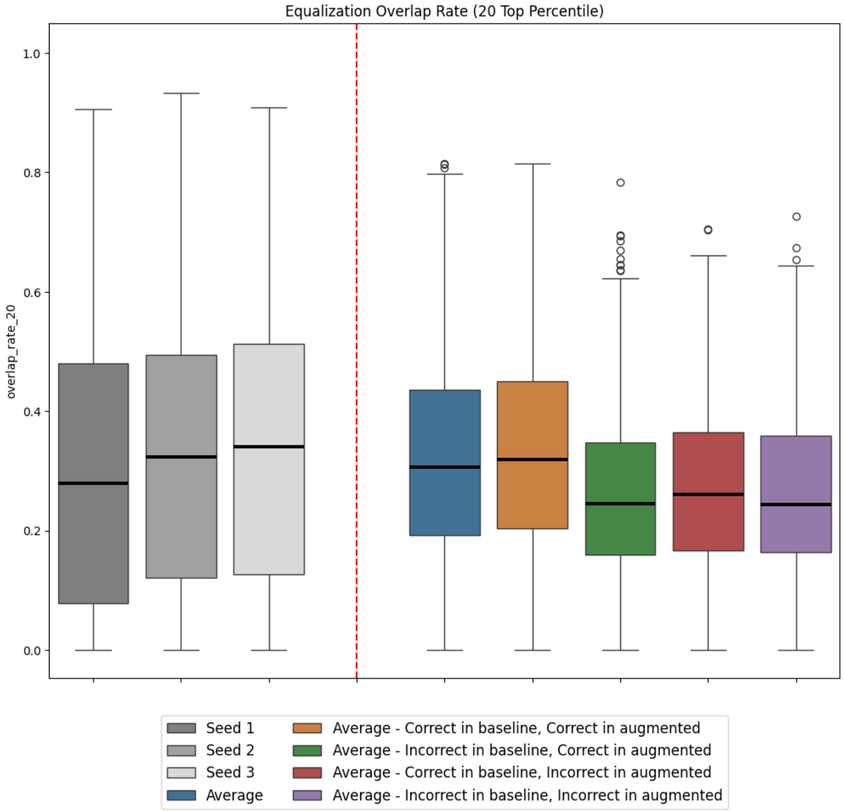


Figure C.6 – Segmented boxplot for Gaussian Blur and Overlap Rate (20), showing the distribution of the metric values across segmentations.

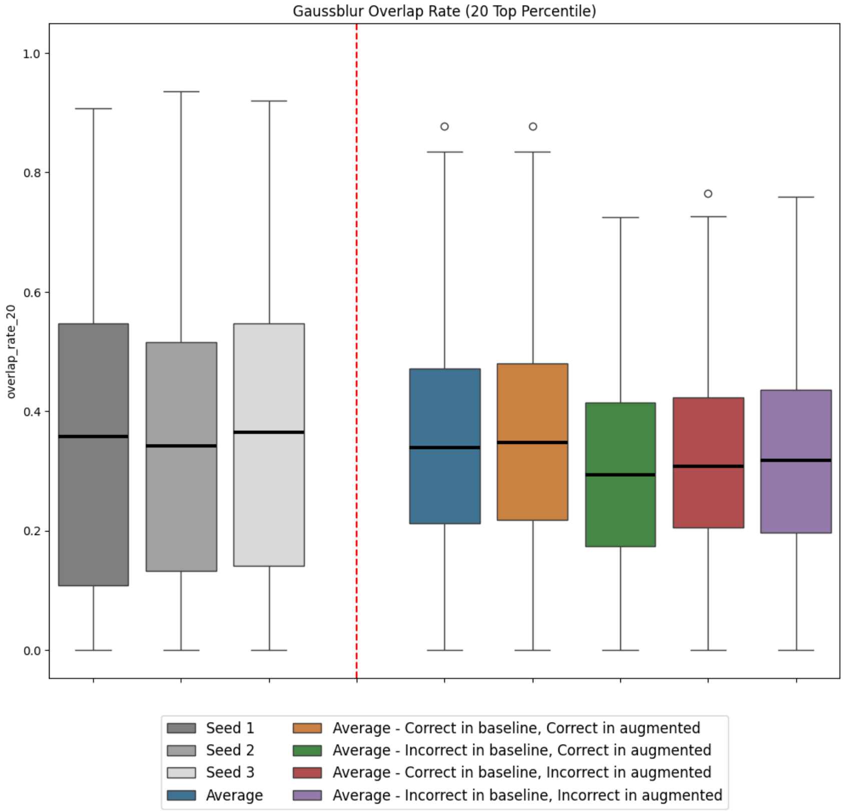
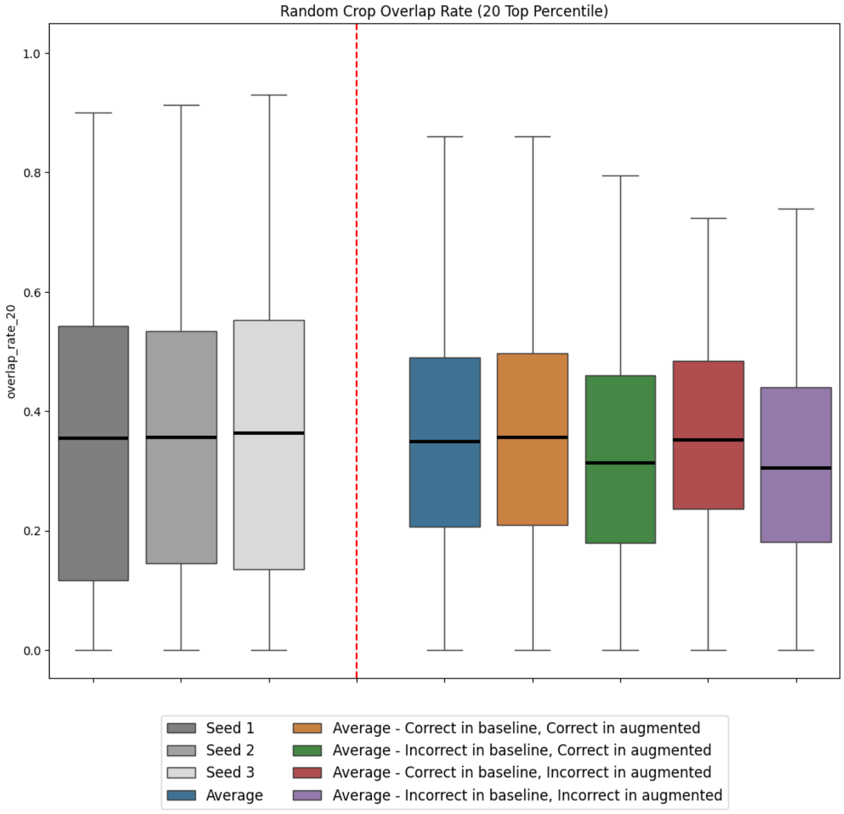


Figure C.7 – Segmented boxplot for Random Cropping and Overlap Rate (20), showing the distribution of the metric values across segmentations.


Random Crop Overlap Rate (20 Top Percentile)
overlap_rate_20
Seed 1
Seed 2
Seed 3
Average
Average - Correct in baseline, Correct in augmented
Average - Incorrect in baseline, Correct in augmented
Average - Correct in baseline, Incorrect in augmented
Average - Incorrect in baseline, Incorrect in augmented