# ULIME: Uniformly weighted Local Interpretable Model-agnostic Explanations for Image Classifiers

by

Minhaj Uddin Ansari

A thesis submitted to the
School of Computing
in conformity with the requirements for
the degree of Master of Science

Queen's University
Kingston, Ontario, Canada
May 2022

# Abstract

With the rapid development of complex machine learning models, there is uncertainty on how these models truly work. Their black-box nature restricts experts from evaluating models solely on standard numerical metrics, which may result in a model performing seemingly well on a dataset but for the wrong reasons. It is particularly critical to have transparency in medical image analysis due to the importance those decisions have. This is because health care practitioners are hesitant to trust machines when there is no clear reason for the machine to use the appropriate logical steps for a decision, and most importantly, human life is at stake. Regulatory bodies, such as the U.S. Federal and Drug Administration, have recently started pushing guidelines to make AI medical devices more transparent so that they are safer to use.

To address this challenge, many researchers have proposed interpretability methods to generate explanations for the behavior of machine learning models. One popular interpretability method is Local Interpretable Model-agnostic Explanations (LIME), which can explain any black-box model trained on any type of data. Although LIME is a powerful tool, the explanations generated by it depend on how the hyper-parameters of the algorithm are set. If little care is given to the hyper-parameters, the resulting explanations can be meaningless.

In this thesis, we first address the impact of a core component in LIME – weights.

The weights impact the quality and variability of explanations generated by LIME. Since weights are determined by the distance metric, we compare the explanations of LIME for three cases; when weights are calculated using (1) an unnormalized distance metric, (2) a normalized distance metric (LIME authors default), and (3) set uniformly to a value of one. Through theoretical analysis and experimentation with different data sets and models, we demonstrate that using the unnormalized distance metric results in poor explanations, whereas the normalized metric and uniform weights give comparable high-quality explanations. We, therefore, propose ULIME (Uniform LIME), a variant of LIME that forgoes the weighted notion of locality by setting all weights to one. Our motivation behind ULIME is the simplification of the existing LIME algorithm by removing the weighting step, and consequently, removal of the distance metric hyper-parameter that is strongly coupled with explanation quality.

Secondly, we address the impact of superpixelization techniques on explanation intuitiveness – a method used to partition the image for explanation purposes. We develop a domain-specific superpixelization method that aids in generating more intuitive explanations and emphasize our results on two medical imaging datasets. Although LIME can be configured for any type of input and model, for this thesis, we restrict our investigation to image classifiers.

Our contributions aim to improve the current state-of-the-art LIME for image interpretability. We simplify LIME by removing the weighting step, reducing the possibility of error due to the distance hyper-parameter, and introducing the concept of domain-specific superpixelization verified by a radiologist. Taken together, our approach improves the quality of explanations, and, thereby, provides a mechanism

for physicians and patients to place greater trust in the results of ML models.

# Acknowledgments

This research was funded by the Vector Institute in Ontario, Canada, and the National Cancer Institute. This work was supported in part NCI R01 CA233888 to be in compliance with federal regulations.

First and foremost, I am extremely grateful to my supervisors, Dr. Amber Simpson and Dr. Christian Muise for their invaluable advice, continuous support, and patience during my master's study. I would also like to thank every member of the Mu Lab and the Simpson Lab. For the sake of brevity, I would specifically like to thank Dr. Jacob Peoples from the Simpson Lab for being one of the first ones to set my research direction.

Back in Pakistan, my family supported me as if we were all in this together. My dad, Dr. ShahabUddin Ansari, my mom, Dr. Aisha Sultana, and my sister, Zhara Sultana, were always there to give me the moral support and motivation needed to continue my unprecedented research journey, and I am eternally grateful for their unconditional love.

Finally, during the COVID pandemic, everyone was restricted to work remotely. During those trying days, my labmate, Katy Scott, was always there to start my PC whenever I lost remote access. Therefore, I would like to extend my gratitude to her and wish her the best in her future endeavors.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

The ability of a model to make predictions by learning has automated the traditional process of manually formulating rules from data. However, unlike the clear understanding, one had of manual logic, Machine Learning (ML) models are more difficult to understand. This difficulty grows with the complexity of the model, and at some point, it becomes unreasonable to find the meaning behind the many parameters involved in making a decision. At that point, we can only treat the model as a black box, i.e. algorithms that cannot be explained due to their inherent complexity, and blindly trust the computer to use the right information for decision making. Although blind trust is appropriate for some applications, medical settings have implications on human health and are therefore governed by regulatory bodies including the United States Food and Drug Administration (FDA) and Health Canada. Any bias in prediction can lead to catastrophic consequences, and because biases are usually difficult to locate, it may be the case that patients receive suboptimal treatment before the bias is detected. Even when the bias is found, the high complexity of current ML models only makes it more difficult to fix immediately. Therefore, regulatory bodies

such as the FDA have started incorporating regulations for AI systems in their framework, and soon AI manufacturers will be required to ensure their medical devices are transparent and safe to use [17].

One might then consider making less complex models for the same task. Presently, there is a trade-off between performance and complexity. As the machine learning model becomes more complex, the machine is able to capture more sophisticated behavior [22]. For example, in certain applications, a simple machine learning algorithm such as linear regression is not suitable for classification compared to the more complex convolution neural network (CNN) [24]. Going deeper, a single layer CNN might not be as effective as using a multi-layer residual network. One could then opt for decreasing the overall complexity to improve interpretability, however, decreasing a model's complexity is undesirable for many critical tasks where the need for performance or accuracy is high. Therefore, intentionally decreasing complexity is not always viable.

An alternative is to understand how a machine learning model works and/or what input features it uses to discriminate between different types of input. This would boost the end user's trust [20], since they could better understand the circumstances leading to a model's decision. In the clinical setting, a physician may more readily accept the machine's decision if the machine presents reasons as to why it thought a certain way, and those reasons are comparable to what the physician uses. For example, if the presence of a disease is exhibited by an aberration on the surface of a tissue, the physician would expect the machine to use those regions as an explanation for the disease prediction. To make machine learning models explainable, researchers have recently started working on certain methods, called interpretability methods. Many

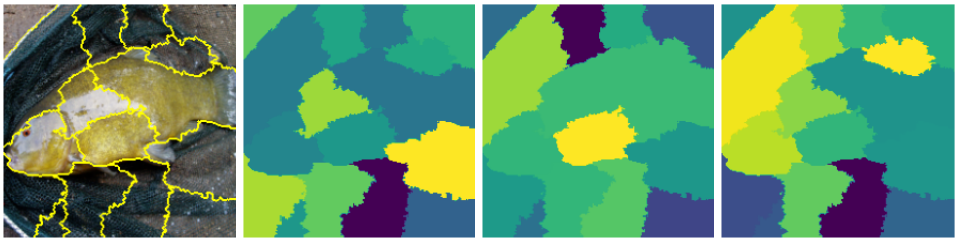
types of interpretability methods have been introduced in the past decade, and we focus on one of the most popular recent ones called Local Interpretable Model-Agnostic Explanations (LIME) [34]. LIME helps deep learning practitioners explain any machine learning classifier by highlighting the important input features responsible for a prediction. For example, the authors of LIME trained a model on biased images of wolves and huskies and showed that for predicting images containing wolves, the model was considering the snowy background rather than the animal. This caused the model to confuse a husky with a wolf when an image of a husky in the snow was shown. Furthermore, ten out of twenty-seven human subjects trusted the "bad" model before examining the explanations, and after, the number reduced to three [34]. Therefore, without the aid of an interpretability method, it would be difficult to catch biases early on due to the impracticality of sifting through thousands of images used for training the model.

The significance of interpreting a model is evident, but this gives rise to another important question. Can we trust the *explanations* generated by an interpretability method? If the explanations are inconsistent or do not capture the true behavior of a model, this not only defeats the purpose of having an explanation but is counterproductive to establishing user trust. Even worse, if the user ends up trusting an erroneous explanation, this can lead to unnecessary wastage of resources trying to fix a perfectly fine model [36]. Therefore, along with establishing trust in a model, establishing trust in the *explanations* generated by an explanation method is equally important.

## 1.1 Problem

The explanations generated by an explanation method need to be correct to properly understand a model. The term "correct" can encompass many definitions, but before that, we briefly introduce the concept of superpixels so that the definition of the term is more clear. Superpixels are a preset collection of pixel groups. Each pixel group contains pixels of similar intensity. To illustrate superpixels, Figure 1.1 shows a fish image divided into 16 superpixels. LIME indicates the superpixels responsible for the model prediction, e.g. in this case, LIME might point to the superpixels containing the fish's body. A correct explanation needs to (1) be consistent for that input, (i.e. if the input and model are kept fixed, the explanations that show which superpixels are important should not differ after every new iteration of the explaining method) and (2) highlight *only* the superpixels responsible for a models decision. If the explanations differ in every iteration of the explaining method, or the superpixels are poorly designed such that individual superpixels contain both essential and unessential regions (hence we cannot know for sure which region inside the superpixel was actually important for the model), this will result in uncertainty in the correctness of the explanations.

Bringing this concept to the context of LIME, although LIME is certainly capable of generating reasonable explanations, we observed variations in LIME explanations in each run when certain distance metrics were used as hyper-parameters. These variations were in the form of varying importance of superpixels of an image. For example, if LIME was being used for identifying which superpixels of an image were responsible for a model's prediction, and the distance metric was not chosen carefully, in one run, LIME would point to one superpixel of the image but in another run, it



(a) (b) (c) (d)

Figure 1.1: The fish image is divided into 16 superpixels. Variations in the LIME explanations are apparent in the heatmaps when the L2 distance is used for calculating the weights.

would point to a different superpixel of the image. We show an example in Figure 1.1. Therefore, the explanations were dependent on the choice of the distance metric.

Secondly, the choice of superpixelization also affected the quality of explanations. If the image was segmented randomly, some superpixels of the image would be highlighted as important only because they contained a small fraction of the image responsible for the prediction even though the majority of the superpixel was insignificant. These two problems raise a potential roadblock to explaining ML models in healthcare, given the sensitive nature of the application.

## 1.2 Objective

This thesis has two primary objectives. The first objective is to investigate the impact of a core component in the operation of LIME – the weighting step (determined by the choice of the distance metric) – on the explanations generated. We contrast this with our approach to disregard weights entirely in LIME, called ULIME (Uniform LIME), for the purposes of making LIME simpler and less error-prone. The second objective is to investigate the effect of superpixelization on the explanations generated by deep

(a) (b) (c)

Figure 1.2: (a): Superpixelized image of a Labrador. (b)-(c): Heatmaps of the explanations using L2 LIME -vs- ULIME averaged across 10 LIME runs, 300 perturbations per run. Because of the L2 distance, the heatmaps of L2 LIME (b) are more dispersed compared to the heatmaps of ULIME (c).

learning models trained on medical imaging datasets. To examine the explanations, we collaborate with a Radiologist from Memorial Sloan Kettering Cancer Center (New York, NY, USA). These two objectives build upon our main motivation of bridging the gap of trust between intelligent machines and patients/physicians.

## 1.3 Summary of our Approach and Findings

In our experiments, we used a variety of image-based benchmarks and ran LIME with two traditional distance configurations as well as our augmented version ULIME which does not rely on any distance configuration. We found that our proposed approach, ULIME, provided highly stable results comparable with the one distance configuration (used by LIME authors), and does so without the added complexity of a user-specified distance hyper-parameter.

We further evaluated all methods on two real-world radiology imaging datasets, the liver, and the pancreatic tumor datasets, and our observations on previous datasets

still upheld. However, on these medical imaging datasets, the standard superpixelization technique failed to focus on the proper parts of the input image, leading to sub-par explanations. To remedy this, we implemented a domain-specific superpixelization for medical tumors, leading to better insights into what the model was looking for in an image.

## 1.4 Contributions

We have three key contributions to improve the reliability and understanding of LIME:

- A modification of LIME for image classifiers called ULIME, based on uniformly setting all weights to one is proposed. This is done to make LIME simpler and less error-prone since the choice of distance metric can significantly impact the explanation quality.
- The performance of ULIME is extensively tested on different models and data sets. We compare ULIME with two other versions of LIME (including one used by the authors). Our empirical and qualitative metrics show that the weighted locality is not necessary for the application of LIME on image classifiers.
- We introduce a domain-specific superpixelization technique for medical images and show the improvement in explanation intuitiveness compared to the standard method. The improvement in explanations is verified by a radiologist.

## 1.5 Organization of Thesis

Chapter 2 explains the inner workings of LIME and goes over existing literature related to our work. Chapter 3 discusses the datasets and models used, our hypothesis

on the impact of weight values on LIME explanations, and experimentation to test our hypothesis. The experimentation and results section in Chapter 4 demonstrates the impact of using our modified LIME on various public datasets. In Chapter 5, we explore our modified LIME in the medical domain and introduce a novel super-pixelization technique that improves the intuitiveness of the explanations. Chapter 6 concludes with useful insights we gained from our results and questions for future research.

# Chapter 2

# Literature Review

## 2.1 What are interpretability methods?

The term interpretability is not rigorously defined, however many authors have attempted to contribute to the definition of the term. Doshi-Valez [13] defines interpretability as the *"ability to explain or to present in understandable terms to a human."* Similar to the previous definition, Miller [29] defines interpretability as *"the degree to which a human can understand the cause of a decision"*. From these two definitions, we realize the purpose of interpretability; to enhance human understanding of why a machine learning model reached a certain prediction. Note that although the term interpretability is used interchangeably with explainability, there is a subtle difference between both. Explainability focuses on the internal processes of a model, for instance, why a certain set of weight values give better accuracy. Another term correlated with explainability is algorithm transparency. Algorithm transparency means how the algorithm creates a model using the data, i.e. for Convolution Neural Networks, the algorithm learns various filters in each layer [30]. The more transparent a machine learning algorithm is, the more explainable it will be. Interpretability, on the

other hand, can identify reasons for a model's behavior but that does not necessarily entail understanding the internals of the model [25]. In this thesis, we will use both terms interchangeably, however, LIME is precisely an interpretability method.

## 2.2 Interpretability Methods for Image Models

There are many different types of interpretability methods, however, the goal of all methods is identical, i.e. to explain a machine learning model's working. Some interpretability methods involve visualizing the internals of a model such as using deconvolution neural networks [47], observing images that are optimized to stimulate filters [32], and creating and visualizing filters that fire on specific object parts [48]. Other methods are attribution based and they classify pixels that either positively and/or negatively influence a prediction such as GRAD-CAM [38], Grad-CAM++ [9], Score-CAM [44], Ablation-CAM [12], and Eigen-CAM [31]. Finally, there are other methods that perturb the input using different techniques and observe how those changes affect the output, therefore creating a causal relationship between the input and output. Such methods include SHAP (SHapley Additive exPlanations) [27], LIME [34], and one that focuses on designing *meaningful* perturbations for creating an interpretation of a black box model [16]. Perturbation-based methods usually involve running the black-box model multiple times on perturbed samples of the input to find the prediction accuracy. Samples that result in lower accuracy generally indicate a lack of the presence of the object being predicted. Since only the input needs to be modified, they are much easier to implement and are more intuitive to individuals who have no prior background in machine learning.

## 2.3 Why interpretability methods in healthcare?

AI is gaining momentum in the healthcare sector as a means to speed up disease diagnosis, assist clinicians in decision making, outperform human capability in certain tasks [3], and finally reduce cost and resources required to deliver quality care. However, even with the rapid success of AI technology, there is still widespread skepticism for its implementation in medical applications. Unlike humans, machines cannot be held accountable for any wrongdoing and are only lightly regulated. Additionally, modern AI is highly complex and easily influenced by data biases. This includes racial, ethnic, and gender bias issues that regulatory bodies such as the FDA still struggle to overcome. There has not been any formal understanding of the relationship between the weights involved in the decision-making processing. This is also impractical given the large number of weights involved in some models. Therefore, if any progress is to be achieved in the healthcare sector, time and effort will be required to design and improve methods that explain the reasoning behind a machine's decision.

From a clinicians' perspective [41], AI models should be explainable so that they can be used to justify a decision. This justification should be backed by proper medical context, and if the model does fall short of accuracy, the under-performance should be explained. If a proper justification is provided, this will boost the clinician's trust in the model, since they already trust the underlying justifications proven through years of medical science research. The clinicians also express the need for the model to repeatedly perform well in the diagnosis of a disease for continuous usage, i.e. explanations should indicate consistency for a similar type of input data.

Google recently conducted an experiment [5] across eleven clinics in Thailand to

detect diabetic retinopathy (DR) from patient eye images. They observed that the quality of eye images varied significantly between clinics. Some clinics had a dedicated room they would turn dark to enlarge the pupil before taking a picture, whereas other clinics took pictures in lit rooms, leading to reduced image quality. The deep learning model was only trained on high-quality images resulting in performance issues when the nurses tried to feed in dark or blurry images. The bad performance of the model on blurry images led nurses to believe they could concatenate two good halves of the same patient to create a complete image of the retina, but deep learning models were not trained to handle composite images, therefore the performance did not improve. The nurses were not at fault because they did not understand *how the model really worked*.

From the patients' perspective, Cadario *et al.* [7] reports that patients were generally hesitant to use AI for their health checkups because they did not understand what AI was, and believed that only medical doctors could perform the job properly. Similarly, in another study [26], researchers found that patients were less likely to trust AI because they thought their diseases were unique to them, and a standard algorithm was incapable of understanding their circumstances. They were more willing to consult a physician even if the risk involved was greater than taking the aid of a machine. Recently, a study [35] was carried out in the span of 5 months where the authors investigated patient views on AI in healthcare. They received positive responses from the participants, however, those responses were contingent on the prior that AI had been well tested and evaluated before deployment to the public. In all three cases, an interpretability method was needed to show patients that AI can complement a human decision, and in the future, even replace doctors for certain

tasks.

## 2.4 LIME Definitions

Since the field of explainability is nascent, all methods are equally pervasive in the research community. Therefore, amongst the many methods that we could have explored, we choose to research one popular method, LIME. In this section, we detail the working of LIME, these are important terms in the discussion of LIME: local, global, model agnostic, and model-specific:

*Local explanation methods* are only faithful around a particular prediction and therefore cannot be generalized for the entire model. In order to generate local explanations for an instance, a model is trained on a subset of the data acquired by zooming into the region that surrounds the instance.

*Global explanation methods*, on the other hand, explain a prediction by examining the whole model. This ensures that the explanation is consistent with the entire data the model has been trained on. However, sometimes the decision boundary of the model is complex making it difficult to extract an explanation.

*Model-agnostic explanation methods* are independent of the model that needs to be explained. They are capable of identifying the reason behind a particular prediction across different types of models since they do not need access to the model's internals. The model-agnostic approach involves fitting an interpretable model on the input features and output predictions regardless of the complexity of the underlying mechanism in the model. Additionally, independent explanations give the flexibility of explaining the model in a variety of ways. The explanation for a prediction presented to a doctor can conveniently be set to differ from the one presented to a

statistician.

*Model-specific explanation methods* are methods that are coupled with the weights and architecture of a model. Therefore they depend on a particular model to explain its working. Due to this constraint, they are not as portable as model-agnostic methods [8], although some model-specific methods can work across models of a similar class such as Grad-CAMS that primarily work on CNN's regardless of the CNN architecture.

### LIME Specifics

For an instance $x$ that needs to be explained, where $x$ can be a tabular data point, text, or an image, the first step is to convert $x$ from its original space $z \in R^d$, to $x'$, the space of the interpretable model $z' \in \{0,1\}^{d'}$.

Multiple perturbations $p'_i$ are generated around $x'$ in the interpretable space. In tabular LIME, the perturbations are random points, whereas in textual and image LIME, the perturbations are randomly initialized binary arrays. The elements of the binary array indicate the presence or absence of parts of the instance $x$ in the original space. In the case of $x$ being text, each perturbation is a random inclusion and exclusion of words, whereas in the case for $x$ being an image, $x$ is first segmented into $N$ *superpixels*, where a superpixel is a small portion of an image, and then the superpixels are randomly included and excluded.

The black box classification values are obtained for each perturbation in the original space. The perturbation and associated classification values are used for training a weighted interpretable model taken from a class $g \in G$ of interpretable models. The interpretable model can be any linear model. In the linear model, the input features

are the perturbations and the labels are the classification values. The distance, $D$, between the instance $x'$ (binary array of all ones) and the perturbation $p'_i$ is used to calculate the weight, $w_i$, for a particular perturbation. $\sigma$ is the width of the local region, i.e. the size of the region considered to explain an instance. Greater $\sigma$ will decrease the locality of the explanation. The distance metric $D$ is treated as a hyper-parameter. Therefore, it can be substituted with any metric such as the Cosine distance, the L1, the L2, etc.

$$w_i = \exp(-D(x', p'_i)^2/\sigma^2) \tag{2.1}$$

Perturbations closer to $x'$ are assigned larger weights, so that the model is *locally faithful*. The loss function is the weighted squared difference between the black box $f$, classification values $f(p_i)$, in the original space and the interpretable model $g \in G$, prediction values $g(p'_i)$, in the interpretable space for all perturbations.

$$L(f, g, w) = \sum_{i=1}^{N} w_i(f(p_i) - g(p'_i))^2 \tag{2.2}$$

Once trained, each coefficient of the interpretable model $g \in G$ indicates the importance of an input feature. Algorithm 1 shows the implementation of LIME.

## 2.5 LIME Stability

David Alvarez-Melis *et al.* [2] performed robustness tests on various interpretability methods. Their goal was to study how slight changes in the input affected the explanations. The authors found that out of all interpretability methods, LIME was the least robust method. A stability analysis was also carried out by [14] on text

**Algorithm 1** LIME

**Input** $Perturbations$
**Output** $Interpretable\ Model\ g$
**for** $pert$ in $Perturbations$ **do**
    $Weights.append(WeightEq(Distance(allOnes, pert)))$
**end for**
**for** $pert$ in $Perturbations$ **do**
    $Labels.append(BlackBox(OriginalSpace(pert)))$
**end for**
$Interpretable\ Model\ g = LinearModel(Weights, Perturbations, Labels)$
**return** $Interpretable\ Model\ g$

data, and out of six local model agnostic interpretability methods, LIME was the least stable.

One of the reasons for explanation instability is the perturbation step [30, 4, 46, 45] that involves random sampling of perturbations around the instance to be explained. To solve this problem, the authors [46] decided to forgo perturbations in their implementation of LIME, called DLIME, and instead use the training data. They first clustered similar data together and then used the k-nearest neighbors algorithm to find the cluster of data points from the training set most similar to an instance that needed to be explained. Since the clusters had the same data for every LIME run, the explanations were constant. Although this approach solved the problem of the variable explanation, three separate problems arose. Firstly, the data set size had to be large enough to accommodate clusters suitably sized for training an interpretable model. Secondly, even if a large data set was available, it might be the case that clusters similar to the test instance were not available in the data. Finally, if the black-box model was overfitting the training data, this can cause the interpretable model to explain a new instance based on the classification values of training data

the black-box model had already memorized, leading to incorrect explanations.

Beyond perturbations, some authors suggested that the weighting process also leads to explanation instability. The authors of ALIME [45] replaced the weighting method with an autoencoder that calculated distances in the latent space and showed that ALIME was able to achieve higher stability compared to LIME. However, the autoencoder needs to first be trained on the training data. This can lead to other challenges if the data set is either very large or unavailable due to privacy reasons such as health datasets involving patient confidentiality.

The authors of C-LIME [1] also used weight values of one for tabular LIME, only after they changed the method of generating perturbations from random to Gaussian distribution. They specifically stated that since they've changed to Gaussian distribution, the samples are already assured to have close proximity to the instance that needed explaining. In our work, we show that regardless of how far an image perturbation is (i.e. regardless of the distribution), weight values of one are comparable to weight values computed using normalized distance metrics.

An improvement in stability was also observed using a Bayesian version of LIME, BayLIME [49]. BayLIME used prior beliefs to improve stability, however, the constraint not only lies in the usage of resources for obtaining prior beliefs but also making sure the prior beliefs are derived from appropriate sources.

Finally, it is crucial to stress that C-LIME, ALIME, and DLIME have only been applied to tabular data so far. Therefore, our work is separate because, we explore the image version of LIME, which has an entirely different interpretable space. We only cite these authors for the reported variance and modification in weight values.

## 2.6 LIME Explanation Metrics

Generating explanations for the black box is not enough: the explanations need to be evaluated to verify that they correctly explain the black-box model. For this purpose different metrics have been introduced. LEAF (Local Explanation evAluation Framework) focuses on five metrics for tabular LIME [4]:

- **Conciseness:** Measures the number of non-zero weights required to explain an instance $x$. The smaller the value, the more easier and intuitive the explanation is to understand.
- **Local fidelity:** Measures how well the interpretable model $g \in G$ approximates the prediction of the black box model for an instance $x$.
- **Local concordance:** An amalgamation of local fidelity and conciseness. It measures local fidelity *under the constraint of conciseness.*
- **Reiteration similarity:** Uses the Jaccard Index to measure the similarity of features between two or more interpretable models for the same instance $x$.
- **Prescriptivity:** Measures how well an explanation can be used to induce changes in the instance $x$ in a certain direction leading to new output predictions.

Visani *et al.* [43] proposed two metrics - VSI (Variable Stability Index) and CSI (Coefficient Stability Index). They first called LIME $r$ times on an instance and then checked the (1) similarity of the features (VSI) comparable to the reiteration similarity metric discussed earlier and (2) similarity of the coefficient value for the

same variables (CSI) between each pair of interpretable models. For stability, both metrics need to be high.

Yin *et al.* [45] used mean standard deviation to measure change in the coefficient values across 10 LIME runs for an instance $x$. Greater changes in the values corresponded to less stable explanations.

The LnO method, originally named MoRF (Most Relevant First), was proposed by [37] where they first placed a grid on the image, each box sized 9x9, and then iteratively removed the boxes based on importance. Using this technique, they observed the AOPC (Area Under the MoRF Perturbation Curve) to evaluate the quality of different heat mapping techniques. Based on heatmap quality, they were also able to assess the deep neural network performance. A similar method, termed as *deletion metric*, was also used by Zhao *et al.* [49] to evaluate explanation quality.

## 2.7 Experimentation Notation

For each image $i \in I$ belonging to dataset $I$, we first segment $i$ into $N$ superpixels, where $N$ depends on the image and the segmentation algorithm. We then run LIME $r = 10$ times on perturbation quantity $p \in P$, where $P$ is the set of perturbation quantities we use in our experiments, e.g. in our experiments, the set contains 12 perturbation quantities $P = \{10, 50, 100,200, 300, 500, 700, 1000, 1500, 2000, 2500, 3000\}$, and the $p$ perturbations are randomly selected in each run. This gives us $r * |P|$ explanations per image $i$, each explanation having $N$ coefficients. We run our experiments three times for each dataset, once for L2 LIME, once for Cosine LIME, and once for ULIME. This means that $|I| * |P| * r$ explanations are generated by L2 LIME, Cosine LIME, and ULIME. In our experiments, we use three metrics, one

qualitative; heatmaps, and two quantitative; LnO and the combined variance.

## 2.8 Experimentation Metrics

### 2.8.1 Heatmaps

Heatmaps are a data visualization technique that represents values as colors. The intensity of the color indicates the magnitude of the value. Visualizing values with heatmaps makes it easier for users to capture information in the data. When the interpretable model is trained, every coefficient has a certain value that specifies the importance of the superpixel it is associated with. The sum of the coefficients can vary, and do not necessarily sum to one. Therefore, to compare the explanations between LIME/ULIME runs, coefficients are normalized between 0 and 1 in each explanation by deducting the minimum coefficient value and then dividing by the difference between the maximum coefficient value and the minimum coefficient value. Once normalized, heatmaps are used to visualize the explanations of L2 LIME, Cosine LIME, and ULIME. Figure 2.1 shows the heatmaps of explanations generated by ULIME explaining why a black box model predicted brain tumor. We use the Viridis colormap where the bright yellow color is closer to 1 and the dark purple color is closer to 0.

### 2.8.2 Leave n Out (LnO)

In LnO, instead of placing a grid as done by [37] (originally named MoRF (Most Relevant First)), we find the set of black-box classification values for an explanation by removing *k=1 to N* superpixels of an image instance one at a time, without replacement, and then passing the image through the black box model each time.

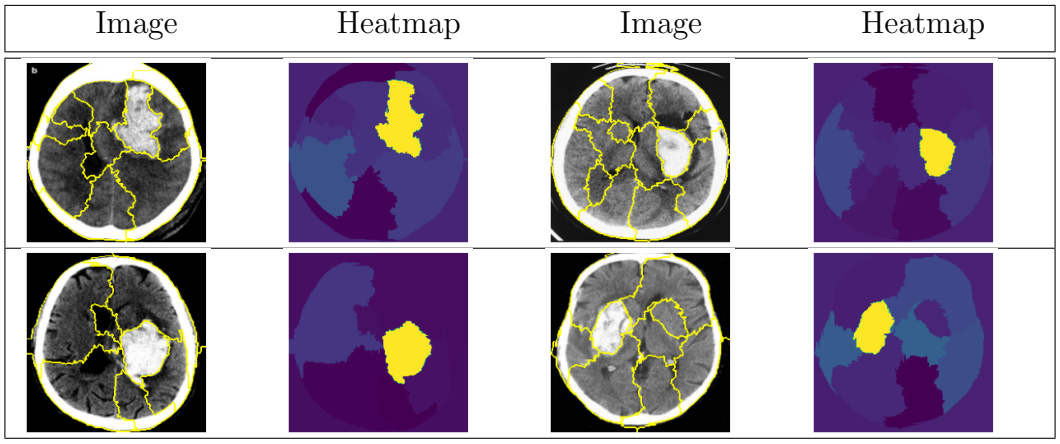



Figure 2.1: Heatmaps visually explain why ULIME predicted brain tumor for each image.

*k=1* is the most important superpixel indicated by the highest coefficient value of the interpretable model. Starting from the whole image with zero superpixels removed all the way to an empty image, we observe a decreasing trend in accuracy as $k$ increases. The intuition is that if the top superpixels are truly discriminating features in the image, the removal of superpixels should result in a quicker accuracy decline. Figure 2.2 shows the LnO for 10 explanations generated by running L2 LIME, Cosine LIME, and ULIME 10 times on ImageNet animal images at 500 perturbations. As can be seen, our ULIME is almost identical to Cosine LIME and focuses on more important superpixels. Therefore the accuracy decline is much quicker when those are removed compared to L2 LIME. Variations in the L2 LIME's explanations can also be observed from the more dispersed LnO's.

There have been some concerns regarding the use of the LnO method. [21] pointed out that the decrease in accuracy may be either due to loss of information or by the change in distribution and instead proposed removing superpixels and then retraining (ROAR). The end result was the same, i.e. a decrease in accuracy was observed but

the decrease was less steep. We believe that the loss of information is the main cause since LIME fundamentally works by training an interpretable model on perturbations of the image. If distribution change was a major issue, we would not expect LIME to work for images at all. Secondly, we reverse the LnO method by removing the least important superpixels first and note that the accuracy is not affected until the removal of the most important superpixel. [50] raises the concern that both approaches (MoRF and ROAR) are unable to work in the scenario that two superpixels are equally important in an image. This is because the removal of the first superpixel does not impact the accuracy, therefore the first superpixel will be considered unimportant by both methods. We agree with them and point out that *we are not using LnO for the purpose of finding the most important superpixel.* Rather, LnO is used to calculate the difference between the accuracy decrease of LIME and ULIME *after* the importance of features has been identified by LIME and ULIME.

For each dataset, we acquire $|I| * |P| * r$ LnO values for L2 LIME $L_{L2}$, Cosine LIME $L_{Cos}$, and ULIME $L_U$. Each element in $L$ is a list of accuracies from *k=1 to N*. In order for the explanations of ULIME to be superior to L2 LIME, and comparable to Cosine LIME, the difference $L_U - L_{L2}$ should be positive indicating that $L_U$ shows a faster decrease in accuracy than $L_{L2}$ and the difference between $L_U - L_{Cos}$ should be near zero indicating that both explanations are similar. Since we are dealing with entire datasets, we investigate two aggregate measures, Average Difference, and Minimum Difference.

In Average Difference, we compute the average LnO values of the explanations, $L'_{L2}$ for L2 LIME, $L'_{Cos}$ for Cosine LIME, and $L'_U$ for ULIME explanations, using Algorithm 2. In Algorithm 2, line 3 goes through all the images of the dataset, line

**Algorithm 2** LnO Average

1: **Input** $L$
2: **Output** $L_a$
3: **for** $i = 1$ to $len(Dataset\ I)$ **do**
4: **for** $j = 1$ to $len(Perturbations\ P)$ **do**
5: **for** $r = 1$ to $runs$ **do**
6: $L_{sum} = L_{sum} + L_{(i,j,r)}$ (element-wise list addition)
7: **end for**
8: $L_p.append(L_{sum}/runs)$ (element-wise list division)
9: **end for**
10: $L_a.append(L_p)$
11: **end for**
12: **return** $L_a$

4 goes through all perturbation quantities for each image, and line 5 goes through all runs per perturbation quantity. Line 6 sums up all the LnO value arrays (*k=1 to* $N$ elements per array) element-wise across all runs and line 8 finds the element-wise average. The average is appended for all perturbations and then appended for all images.

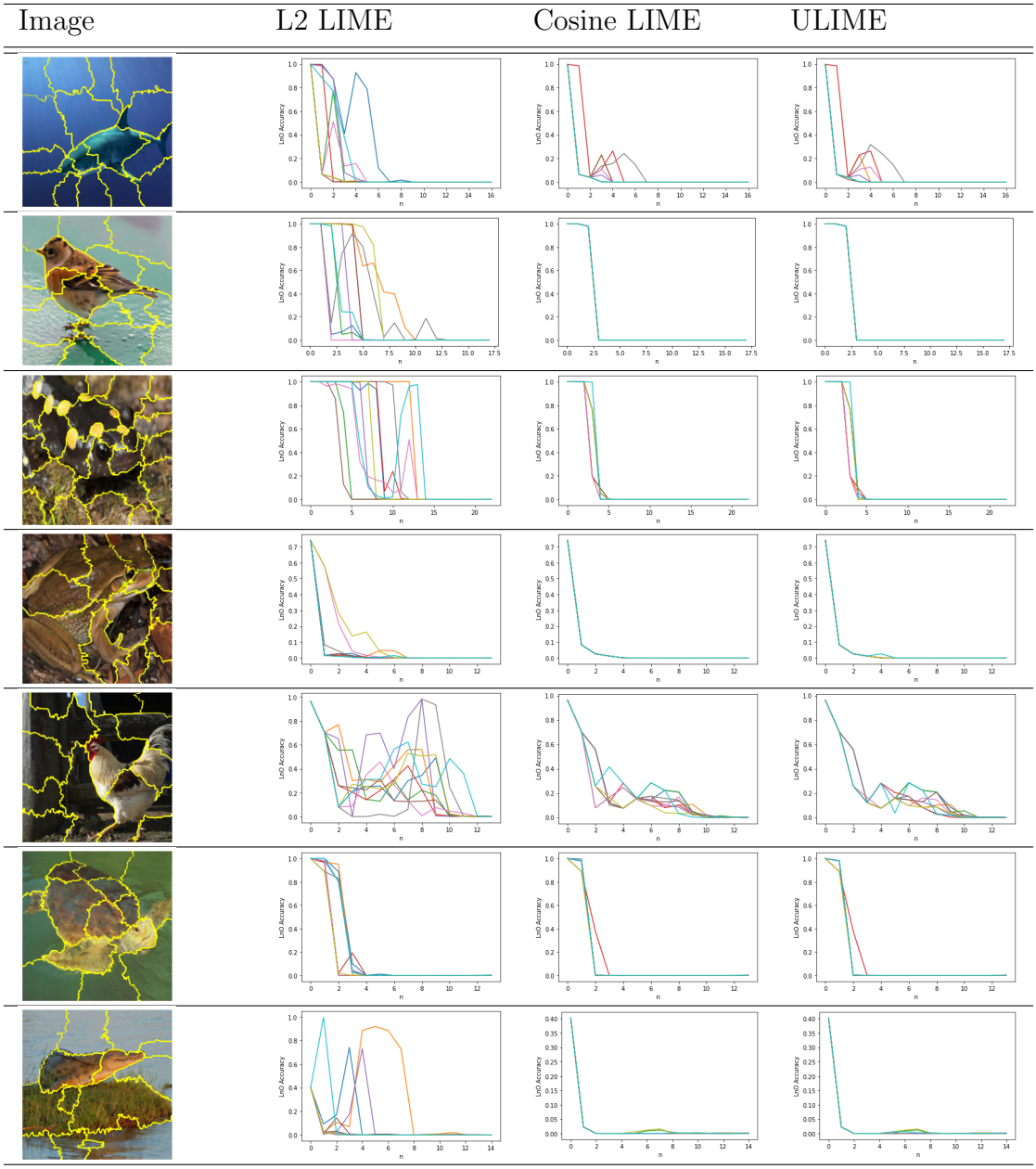



Figure 2.2: ImageNet Animals: LnO Comparison of L2 LIME, Cosine LIME and ULIME explanations for images of animals across 10 runs. LnO's of Cosine LIME and ULIME are similar, and better than L2 LIME.

We then find the difference between the averaged LnO values of L2 LIME/Cosine

**Algorithm 3** LnO Average Difference

1: **Input** $L_a,\ L'_a$
2: **Output** $L_d$
3: **for** $i = 1$ to $len(Dataset\ I)$ **do**
4: **for** $j = 1$ to $len(Perturbations\ P)$ **do**
5: $L_{diff} = L_{a(i,j)} - L'_{a(i,j)}$
6: $L_{id}.append(L_{diff})$
7: **end for**
8: $L_d.append(L_{id})$
9: **end for**
10: **return** $L_d$

**Algorithm 4** LnO Average Difference Per Perturbation

1: **Input** $L_d$
2: **Output** $L_v$
3: **for** $j = 1$ to $len(Perturbations\ P)$ **do**
4: $sum = 0$
5: **for** $i = 1$ to $len(Dataset\ I)$ **do**
6: $sum = sum + SUM(L_{d(i,j)})$
7: **end for**
8: $mean = sum/len(Dataset\ I)$
9: $L_v.append(mean)$
10: **end for**
11: **return** $L_v$

LIME and Cosine LIME/ULIME, $L_d = L_a - L'_a$, where $L_a = L'_{L2/Cos}$ and $L'_a = L'_U$ using Algorithm 3. Again line 3 goes through all the images and line 4 goes through all perturbation quantities. In line 5, we find the difference between the averaged LnO across 10 runs. The difference is also a list that is appended for each perturbation in line 6, and then for each image in line 8. Finally, we calculate the single average value, $L_v$, for each perturbation across the whole dataset using Algorithm 4.

In Minimum Difference, instead of taking the average, we find the minimum LnO values, $L_{(min)L2}$ for L2 LIME, $L_{(min)Cos}$ for Cosine LIME, and $L_{(min)U}$ for ULIME

explanations and compute the difference $L_d = L_m - L'_m$, where $L_m = L_{(min)L2/Cos}$ and $L'_m = L_{(min)U}$. We observe a positive value when the minimum LnO value of ULIME is subtracted from L2 LIME, meaning that on average, ULIME is better than L2 LIME in generating the *best* explanation. The minimum difference between ULIME and Cosine LIME is close to zero, showing that both methods are comparable.

### 2.8.3 Combined Variance (CV)

The purpose of calculating the combined variance is to find out how much the explanations vary between the LIME runs. First the variance for each normalized coefficient, $C_k$ (with $k$=*1 to* $N$), is calculated across the $r$ L2, Cosine, and ULIME runs per perturbation quantity using Equation 2.3. $\overline{C_k}$ is the mean of a particular coefficient $k$ across the $r$ runs per perturbation quantity:

$$Var(C_k) = \frac{\sum_{j=1}^{r}(C_{kj} - \overline{C_k})^2}{N} \tag{2.3}$$

The combined variance for dependent variables is calculated using Equation 2.4 [6] by adding the sum of variances for each $C_k$ with the sum of co-variances between pairs of $C_k$:

$$Var(C) = \sum_{j=1}^{N} Var(C_j) + 2 * \sum_{j<k} Cov(C_k, C_j) \tag{2.4}$$

The dependent version of combined variance is necessary since the coefficients are dependent on one another, i.e. if one coefficient is important, the others will be less

so. A similar metric has been used by [45], where instead of finding the combined variance, they used the mean standard deviation.

For generalizing across a whole dataset, the combined variance is averaged across all images for each perturbation quantity $p \in P$ just like we did for the LnO calculation.

# Chapter 3

# Impact of Weights on LIME

## 3.1 Impact of the Distance Metric on Weights

The goal of this chapter is to investigate the impact of weight values on LIME explanations, and the effect of different distance metrics on the weight values. As discussed in the literature, the weights are calculated using Equation 2.1. In the weights equation, a major problem arises when the distance metric used to calculate distances between the original image and the perturbations is not normalized, i.e., the distance can go to infinity as the dissimilarity increases. This is exactly the case with the L2 distance metric used in L2 LIME. Since the domain of L2 is $d \in [0, \infty)$, and the values of the weights are calculated using an exponential function, therefore the values exponentially approach zero with slight increases in the distance. Even though the authors of LIME use the exponential function to keep weights between [0, 1], the extremely small values of weights end up negatively influencing the quality and variability of explanations. When we normalize L2 or use any other distance metric between $d \in [0, 1]$, the minimum value a weight can acquire is 1.13e-7 only when distance is a maximum of 1 which is rare. On average, the distance stays at 0.5

leaving weight values around 0.018. Below, we show an example of how Cosine and L2 distance metrics affect the weight values. The distance is calculated between an image with all superpixels ON, with perturbations of the images P1, P2, and P3.

$$Image = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$$

$$P1 = [0, 0, 1, 1, 1, 1, 1, 1, 1, 1] \quad Cos = 0.11 \quad W_{Cos} = 0.91, \quad L2 = 1.4 \quad W_{L2} = 1e-7$$

$$P2 = [0, 0, 0, 0, 1, 1, 1, 1, 1, 1] \quad Cos = 0.23 \quad W_{Cos} = 0.67, \quad L2 = 2 \quad W_{L2} = 1e-14$$

$$P3 = [0, 0, 0, 0, 0, 0, 1, 1, 1, 1] \quad Cos = 0.37 \quad W_{Cos} = 0.34, \quad L2 = 2.4 \quad W_{L2} = 1e-21$$

As can be seen, when the perturbations get more dissimilar, the L2 values shoot beyond one, making the weights extremely small. This does not happen with Cosine (or any other metric) that limits the distance to 1. In the next section, we theoretically explore why lower weight values negatively affect the explanations, and if weights are really necessary for LIME image.

## 3.2 The Theory Behind LIME

The intuition of LIME is very simple, and so is the underlying math. In LIME, the goal is to train a simpler model, such as a linear model, to predict the predictions of the underlying black-box model that needs to be explained. However, unlike the image input fed into the black-box model, the linear model learns from binary vectors (the perturbations) with superpixels either ON or OFF as features $X$ and the black box prediction values for those perturbations as labels $y$. Ultimately, LIME ends up fitting a loss function on the data, but before that, the data has to be preprocessed.

Preprocessing depends on the weight values.

In the preprocessing stage, available in sklearn library's linear model fit function, two operations are applied to the $X$ features and $y$ labels, mean centering (also known as zero centering) and weight rescaling. The purpose of mean centering is to shift the data so that the mean lies at zero, thereby equalizing the influence of all features. This is done by subtracting the features of each data point in $X$ and $y$ in Equation 3.2 with the weighted mean of the entire feature column shown in Equation 3.1 (the equation for $y$ is the same). This gives us $X'$ and $y'$. $N$ is the number of features, i.e. the number of superpixels.

$$mean_j = \frac{\sum_{i=1}^{samples} x_{ij} w_i}{\sum_{i=1}^{samples} w_i} \quad for \quad j = 1, 2, ...N \tag{3.1}$$

$$X' = \sum_{i=1}^{samples} x_{ij} - mean_j \quad for \quad j = 1, 2, ...N \tag{3.2}$$

The purpose of rescaling is to apply the influence of weights. This is done by first taking the square root of the weights. The square rooted weight values are placed in the diagonal of a sparse diagonal matrix, and a dot product is taken between the resulting matrix and the features $X'$ and labels $y'$. This gives us $X'_r$ and $y'_r$. We pass $X'_r$ and $y'_r$, our preprocessed data, to a least-squares function.

$$X'_r = sparse_{diagonal}(\sqrt{(weights)}) \cdot X' \tag{3.3}$$

Inside the loss function of the linear model, such as the one for Linear Regression used in this thesis, the least-squares function is calculated using Equation 3.4 where $X$ is an $m * n$ matrix and $y$ is a $m * 1$ matrix. The goal is to find the vector $c$

by minimizing the sum of squared differences $(Xc - y)^2$ between the predicted and the original labels. The smaller the difference, the better the regression line fit on the data. From 3.4, we derive a matrix equation to calculate the coefficients using Equation 3.5.

$$Xc = y \tag{3.4}$$

$$\begin{aligned} X^T Xc &= X^T y \\ (X^T X)^{-1} X^T Xc &= (X^T X)^{-1} X^T y \\ c &= (X^T X)^{-1} X^T y \end{aligned} \tag{3.5}$$

### 3.3 Weight Influence on LIME

Since throughout the data preprocessing step and the loss function, *products* are calculated between different variables, this leads to larger variance in the coefficient values of the trained linear model when using the L2 distance for weights. This is because the weights vary significantly in negative powers of 10 with a slight change in perturbations, as shown in the example above. This, in turn, influence the values of $X'_r$ and $y'_r$. Therefore, the explanations (i.e. coefficients) end up being different for the same instance, black-box model, and LIME settings, because the random perturbations strongly influence the weights. To illustrate variation between explanations, Figure 3.1 (a) shows the superpixelization of the photocopier image. This image was fed to an Inception-V3 model pre-trained on the ImageNet dataset and the model correctly predicted it to be a photocopier. (b) and (c) illustrates heatmaps for explanations of two L2 LIME runs using 100 perturbations. In explanation (b),

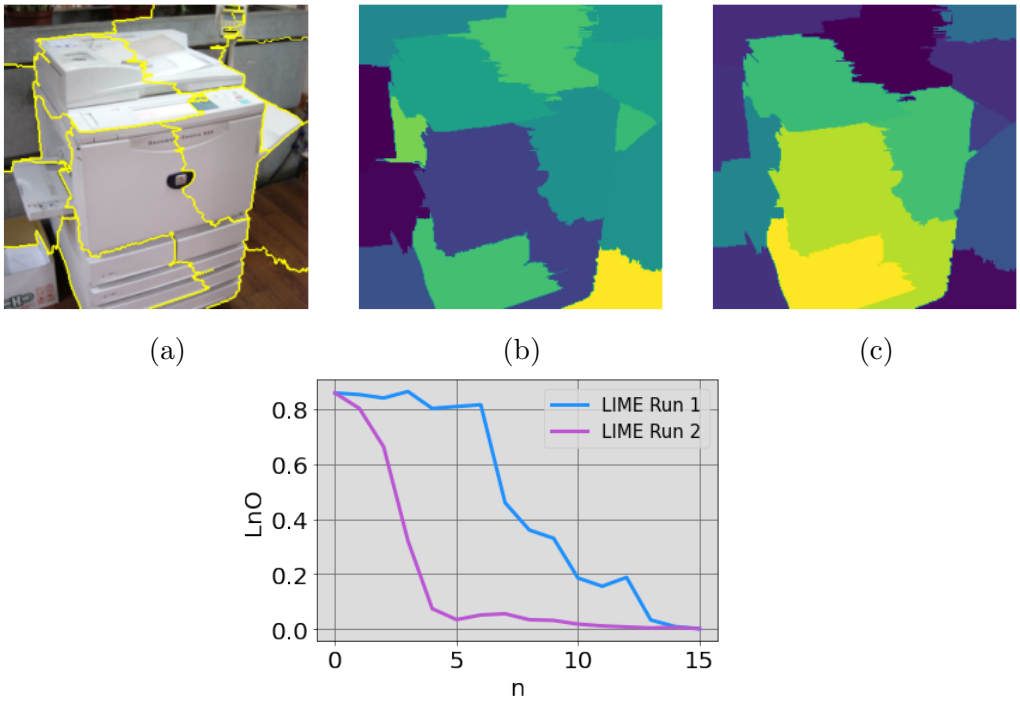



Figure 3.1: (a) Segmented image of a photocopier. (b)-(c) Heat maps for two explanations generated by two L2 LIME runs, 100 perturbations per run. (d) LnO value graphs for the two explanations. Since the explanation in (b) does not focus on the photocopier, its LnO takes longer to decline compared to the LnO of explanation in (c).

the most important superpixel is at the right corner of the image but in explanation (c), the most important superpixel is part of the left corner of the photocopier. We can clearly see a difference (i.e variation) between explanations, even though all we did was rerun L2 LIME on the same image. Similarly, the LnO's also differ for both explanations. Figure 3.1 (d) shows the LnO values graphed for the two L2 LIME explanations. We see that for the second LIME, removing a smaller number of superpixels leads to a quicker decline. Therefore we classify the second explanation to be better than the first. We generally note that the LnO values are related to the

quality of explanations. If the explanations do not prioritize important superpixels, as seen in the heatmap (b) where instead of focusing on the photocopier, the most important superpixel focuses on a corner. the LnO values take longer to decline, as seen in LnO graph (d) for LIME Run 1.

When L2 was replaced with cosine distance, the random perturbations had a smaller influence on the weights. Therefore the weights were better stabilized leading to lesser explanation variation between multiple cosine LIME runs.

### 3.3.1 Impact of LIME Perturbations

Many authors have identified the cause of variable explanations to be the random perturbation step. In the perturbation step, perturbed samples are randomly generated around the instance that needs to be explained and an interpretable model is trained on those samples as input features where the labels are the black box classification values for each sample. Since the perturbations are random, this could potentially cause variations in the explanations. To investigate this, we fixed the number of perturbations and ran L2 LIME, Cosine LIME, and ULIME multiple times on an input image, randomly generating perturbations on each run. We noticed that for Cosine and ULIME, the randomness of perturbations had little impact on the stability of the coefficient values if the perturbation quantity was kept reasonably large. This was not the case for L2 LIME. Due to the L2 distance, we noticed high variations between the explanations for different numbers of perturbations while everything else was kept the same, i.e. the input image, LIME settings, and underlying black-box model. In those variations, some explanations did not even highlight the important features responsible for the model's prediction.

In an attempt to reduce variations, we slowly incremented the perturbation quantity, rerunning L2 LIME, Cosine LIME, and ULIME multiple times after each increment. Increasing perturbations resulted in more consistent explanations across multiple Cosine LIME and ULIME runs but, again, not for L2 LIME. For both Cosine LIME and ULIME, beyond a certain number of perturbations (on average 500), the explanations were consistent between runs regardless of how randomly each perturbation was chosen. Therefore we agree with the premise that random perturbations are responsible for initial variance. However, we find this issue insignificant for LIME image beyond a certain number of perturbations, because (1) the underlying image being perturbed is the same, and (2) the linear model has seen enough perturbation permutations for that image to generate consistent coefficients. Finally, we point out that 500 perturbations are enough for stable explanations when the number of superpixels is kept around 25 (as done in our experiments). Increasing the superpixels will require more perturbations for stabler explanations. We leave the topic of superpixel quantity and its relation to explanation stability for future research.

### 3.3.2 Impact of LIME Weights

We then explored the step after perturbations, the weighting step. The L2 distance is used to measure similarity. It is calculated in the interpretable space, i.e. between the binary array of the perturbations and the instance. The greater the difference, the smaller the weight assigned to the perturbation, since the perturbation is supposed to be less similar to the instance making it less local. Initially, we were convinced that this method failed to capture the similarity between the original image and the perturbed images for two reasons.

(a) (b) (c)

Figure 3.2: (a): Ice cream image segmented into 24 superpixels. Pretrained InceptionV3 predicts the image 99% ice cream. (b): Removal of 5 superpixels leads to a 6% prediction. (c): Removal of 19 superpixels maintains 99% prediction.

Firstly, if the perturbation had many zeros but the zeroed superpixels did not contain the discriminating features, the distance would be greater therefore this perturbation would be assigned a smaller weight even though it is more similar to the instance. This is illustrated in Figure 3.2. In the Figure, the ice cream image has been segmented into 24 superpixels. The distance of the third image was greater than the second since more superpixels have been zeroed out, however, to the model, the third image was more similar to the first in terms of the actual object it is trying to predict. Hence, the actual object's size and location in the image space were disregarded when performing calculations in a binary space.

Secondly, the area of superpixels was overlooked in the interpretable space. If a perturbation had many zeroed superpixels, but they only covered a small portion of the image, the distance between the original instance and the perturbation would be greater compared to a perturbation having only one zeroed superpixel that covered most of the image. Again, this problem was similar to the first and was a direct consequence of using binary space to calculate distances.

However, even though both reasons were logically sound, we observed that in Cosine LIME and ULIME, the linear model still learned the proper coefficients from the binary vectors. This opened up two intuitions. Either, *on average*, the number of superpixels excluded did indicate the loss of similarity, hence assigning lower weights was reasonable, *OR*, the linear model was sufficiently learning from the perturbations and black box prediction values alone, and therefore, weights were unnecessary.

There have been prior mathematical analyses of LIME from a theoretic perspective by Garreau [18], [28] and [19] however the objective of those papers is to explore the current implementation of LIME with no intention to investigate how changes in the weighting step impact the performance of LIME. The first two papers focus on tabular LIME but as noted in the conclusion of [28], image LIME is more complex than tabular LIME. In the case of tabular data, dissimilarity can be calculated in the Euclidean space whereas, for Image LIME, we show that the euclidean L2 distance fails to generate good explanations. In the third paper, the authors explicitly state that there is no principled way of choosing kernel width $\sigma$, as also done by authors of [49]. In our observations, we did however note that only for L2 LIME, the value of $\sigma$ depended on the number of superpixels used to segment the input image. For a higher number of superpixels, the value of $\sigma$ also needed to be increased so that the weights in the weight Equation 2.1 did not end up becoming zero.

## 3.4 Our Modification: ULIME

Rather than impose a weighted view of locality on the perturbations, we propose ULIME where we set all weights uniformly to one and let the interpretable model learn from perturbations as features and the class accuracy as labels. Since the linear

model will eventually learn which features are important, and which are not, based on the magnitude of class accuracy alone, we hypothesize that it is unnecessary to enforce any extra weights.

When we changed the weights to one, we observed that the explanations generated by ULIME were comparable to Cosine LIME. This is because the values of $X'_r$ and $y'_r$ for both Cosine LIME and ULIME were similar hence, the coefficient values calculated using Equation 3.5, for all the images tested on, were also similar. Therefore, for image LIME, we concluded that the weighting step was not really necessary. Adding weight calculations only made the algorithm more complex and error-prone due to an extra distance hyper-parameter (i.e. choosing the wrong distance metric such as the L2 resulted in poorer explanations). Note that removing the weighting step did not really change the computation time, since the running of the black box model on all the perturbations was still the main bottleneck of the algorithm. An interesting problem for future research could be to find quicker ways to generate perturbations and corresponding labels to decrease LIME computation time.

Figure 3.3 shows a comparison between 3 L2 LIME, Cosine LIME, and ULIME runs for the same input image, same black-box model, and for 500 perturbations. In this case, we've trained our custom CNN model on the MedNIST dataset.

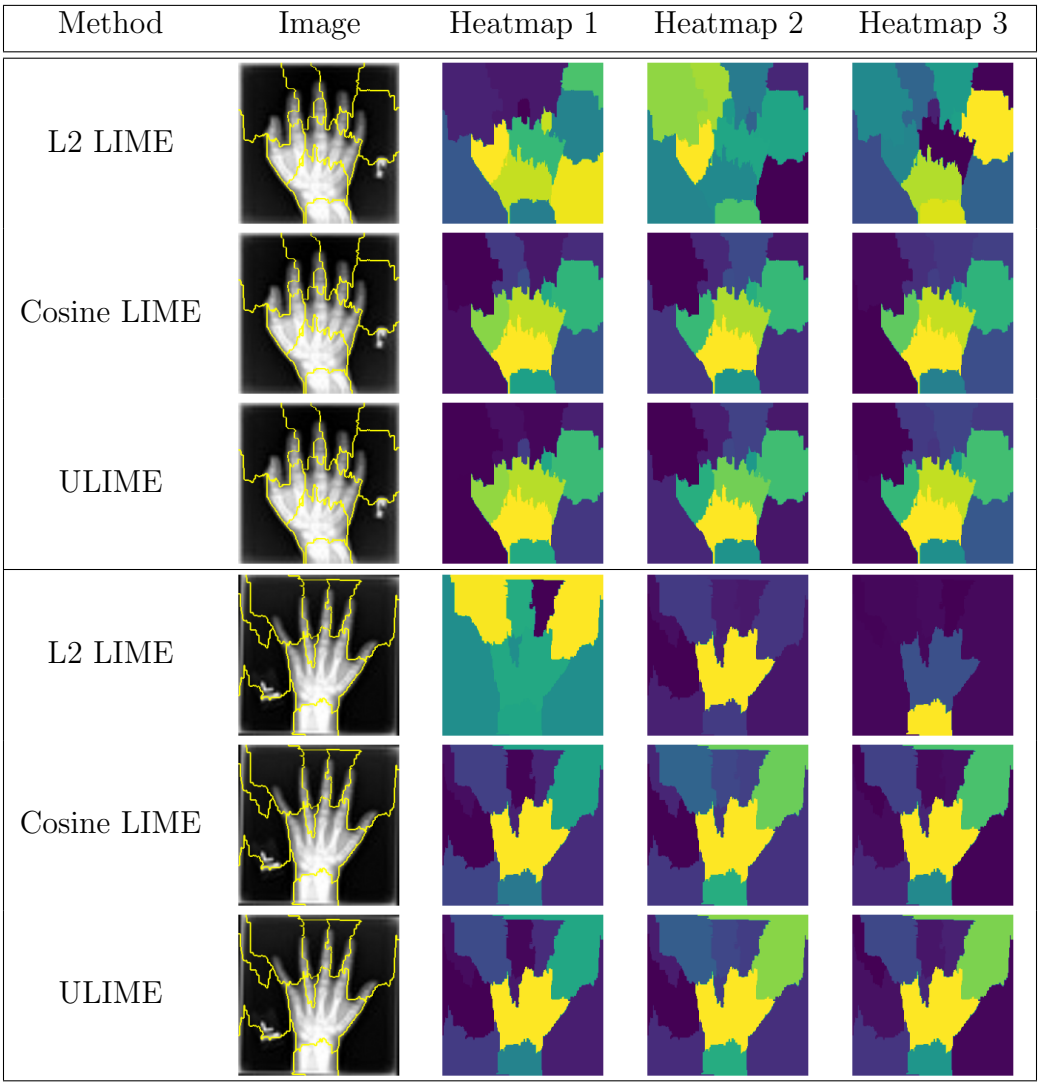



Figure 3.3: Explanation variation comparison between L2 LIME, Cosine LIME, and ULIME for a model trained on MedNIST data. In this case, the model is predicting a hand X-Ray. Less variance is observed in the explanations of Cosine LIME and ULIME (which are also very similar) compared to the explanations of L2 LIME.

The images indicate hand X-Rays. It can be seen that LIME L2 explanations vary much more than ULIME or Cosine LIME. Both Cosine LIME and ULIME are very similar.

# Chapter 4

# Experimentation and Results

## 4.1 Dataset

We used five image datasets, two private datasets, and three publicly available datasets to evaluate our methodology. Out of the five datasets, two datasets were non-medical, ImageNet [11] and CIFAR-10 [23], and three datasets were medical, MedNIST [15], Liver Tumor dataset, and the Pancreatic Tumor dataset. We will detail the last two datasets in the next chapter. ImageNet has over 14 million images with an average size of 469x387 belonging to 1000 different classes organized according to the WordNet hierarchy. CIFAR-10 has 60,000 images of size 32x32 belonging to 10 classes, 6000 images per class and MedNIST has 58,954 images of size 64x64 belonging to six classes; 10000 in AbdomenCT, 8954 in BreastMRI, 10000 in ChestCT, 10000 in Chest X-Ray (CXR), 10000 in Hand and 10000 in HeadCT gathered from multiple sources - from TCIA, the RSNA Bone Age Challenge, and the NIH Chest X-ray dataset.

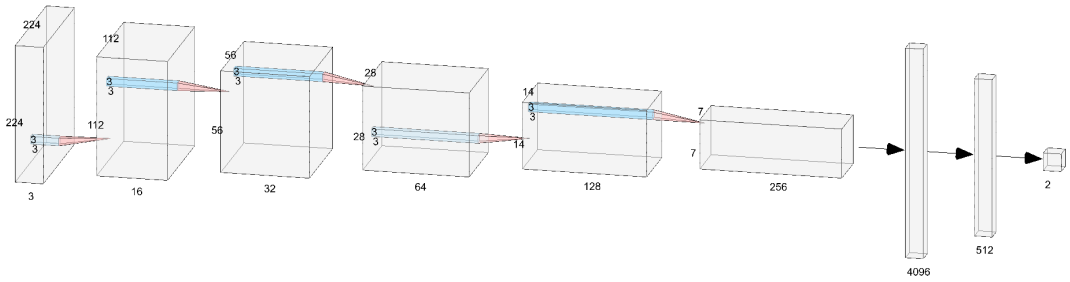



Figure 4.1: Our custom CNN model trained on the MedNIST dataset.

## 4.2 Experiments

We took advantage of publicly available pre-trained models for ImageNet and CIFAR-10. We imported the pre-trained version of Inception-v3 [40], trained on ImageNet images of size 224x224, from the Torchvision library and pre-trained version of VGG-16 [39], trained on CIFAR-10 images from an open-source library [33]. For MedNIST, we trained our own custom model separately on both datasets, resizing images to 224x224, and achieved 93% test accuracy. Figure 4.1 shows the architecture of our custom CNN model. Each convolution is followed by a batch-norm and max-pooling layer which has not been shown in the image.

Our goal was to apply L2 LIME, Cosine LIME, and ULIME on all datasets. The purpose of applying L2 LIME was to show the impracticality of using the L2 distance for calculating weights. We then showed that using a distance metric that normalizes distances between [0, 1], such as the cosine distance, improved the quality and lessened variability of explanations. Finally, we showed that setting weights to one gave explanations that were equal in quality to explanations obtained using normalized distances, therefore concluding that weights were unnecessary for LIME images. For Inception-v3, we used 100 images belonging to the first 100 ImageNet

classes, one image per class. For VGG-16, we used 150 images belonging to the 10 CIFAR-10 classes, 15 images per class. For our custom models, we used 120 images, 20 per class for MedNIST. All images were selected randomly.

The quickshift segmentation algorithm [42] was used to superpixelize all images. The parameters of the algorithm; ratio, the width of Gaussian kernel, and distance cut-off was fixed for each data set given in table 4.1.

| Dataset | Ratio | Gaussian Kernel | Distance cut-off |
|---|---|---|---|
| ImageNet | 0.2 | 6 | 200 |
| CIFAR-10 | 0.5 | 1 | 50 |
| MedNIST | 0.1 | 5 | 100 |
| Pancreas | 0.1 | 6 | 100 |
| Liver | 0.1 | 6 | 100 |

Table 4.1: Parameters for the Quickshift Algorithm.

We chose these values after testing on multiple images, with the difference in values likely resulting from the difference in image sizes of different datasets. The width of the local region was set to $\sigma = 0.25$, a value set by the authors of LIME for images.

In our experiments, we set the number of runs per perturbation to $r$=10 and our perturbations quantities to contain 12 perturbations $P = \{10, 50, 100, 200, 300, 500, 700, 1000, 1500, 2000, 2500, 3000\}$. Due to computational resources, we limit the perturbation quantity to 3000 per image. The coefficients of the explanations were normalized and used to plot heatmaps, calculate LnO's and combined variance. Even though we observed little influence of random perturbations on the explanations, we still fixed a seed value of 123 so that, 1. our results were reproducible, and 2. the perturbations were fixed across all runs for all three LIME versions.

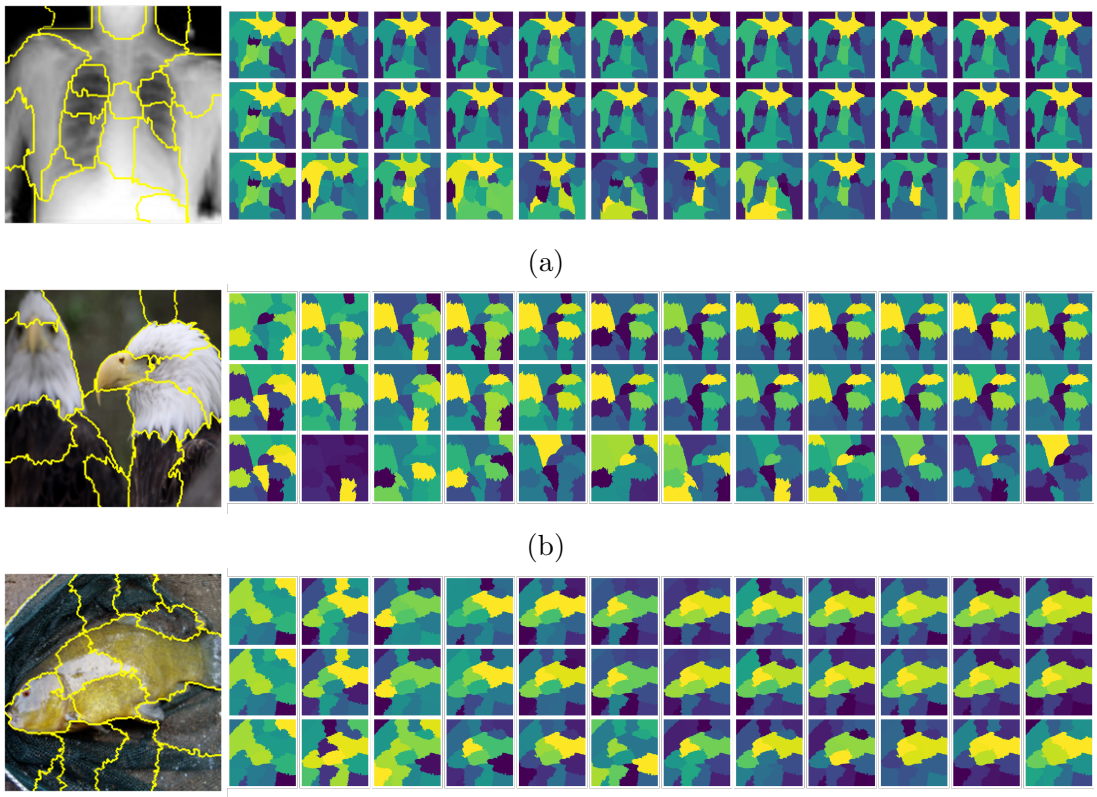
Experiments were run on a Windows 10 Intel Core-i5 9th generation PC having a 6GB NVIDIA GeForce GTX 1660 GPU and Google Colab Pro environment having a 16GB NVIDIA P100 GPU. Jupyter Lab was used for all code written in Python version 3.7.7. The LIME algorithm was taken from the author's official GitHub page: *https://github.com/marcotcr/lime*. Our code can be found here: *https://github.com/ansariminhaj/ULIME*

## 4.3 Results

### 4.3.1 Comparison of Heatmaps

Heatmaps were plotted for the explanations of images belonging to each dataset. The heatmaps showed that L2 LIME had inconsistent explanations and that both ULIME and LIME Cosine explanations stabilized at the same number of perturbations. Therefore, with the help of heatmaps, we visually justified the equivalence of ULIME and Cosine LIME.

Figure 4.2 shows the heatmaps of the explanations generated by Cosine LIME (top row (a)-(c)), ULIME (middle row (a)-(c)), and L2 LIME (bottom row (a)-(c)) for all perturbation quantity $P$ taken from a single run. Each heatmap represents a unique $p \in P$ in ascending order up to the last column, i.e. 3000 perturbations. It can be visually observed that the heatmaps for explanations generated by ULIME and Cosine stabilize at the same time (in the first 300 perturbations for all four images) in contrast to L2 LIME (does not stabilize even at 3000 perturbations for three out of four images).



(a)

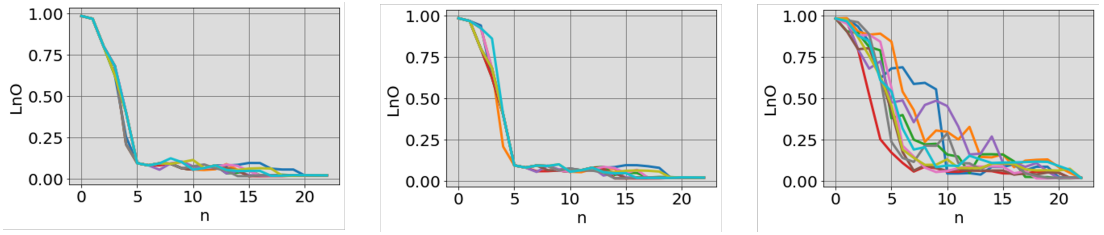
(b)

(c)

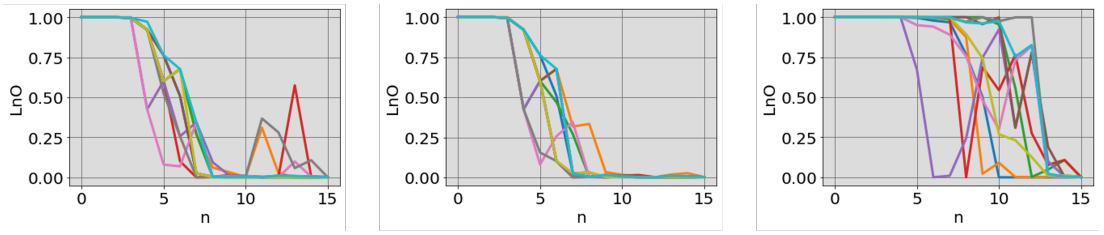
Figure 4.2: For each image (a) Chest X-ray, (b) Eagle and (c) Fish, the top row is for Cosine LIME, the middle row is for ULIME and the bottom row is for L2 LIME. From the left, (a-c) start with the segmented image followed by heatmaps for each explanation per perturbation quantity $P$.

### 4.3.2 Comparison of LnO

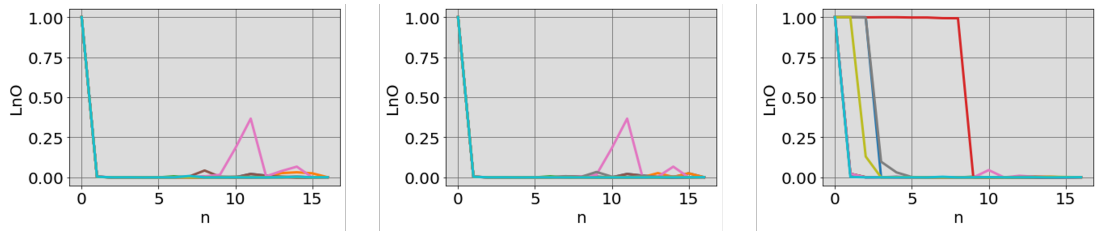
The LnO of both ULIME and Cosine LIME explanations declined at the same rate, whereas the LnO of L2 LIME took more time and was also more fluctuating. In other words, both ULIME and Cosine LIME did an equal job capturing the most important parts of the input image. Figure 4.3 (a)-(c) shows the LnO values of 10 explanations of Cosine LIME, ULIME, and L2 LIME. These explanations were obtained by running each LIME version 10 times on the images in Figure 4.2 at 500 perturbations per run.




(a) Chest X-ray




(b) Eagle




(c) Fish

Figure 4.3: The LnO's across 10 runs at 500 perturbations for the (a) Chest X-ray, (b) Eagle and (c) Fish images shown in Figure 4.2. From the left, the first column shows LnO's for Cosine LIME, the middle column shows LnO's for ULIME, and the rightmost column shows LnO's for L2 LIME.

To generalize our observations across entire datasets, Figure 4.4 shows both differences, $L_a - L'_a$ (Average Difference) and $L_m - L'_m$ (Minimum Difference), between the LnO's of Cosine LIME and ULIME and the LnO's of L2 LIME and ULIME. As can be seen, the difference between Cosine LIME and ULIME is near zero for both differences showing that both techniques are similar. For the difference between L2 LIME and ULIME, the graph is all positive for the average difference indicating that *on average*, the LnO of explanations generated by ULIME decline quicker compared to the ones generated by L2 LIME. In the case of minimum difference, the difference

is either positive or insignificantly negative. The lower values for initial perturbations may be due to the high variance in the explanations for those perturbations, which is observable in our heatmaps.

### 4.3.3 Comparison of Variance

The combined variance of both ULIME and Cosine LIME explanations were comparable. Therefore, both ULIME and Cosine LIME generated consistent explanations

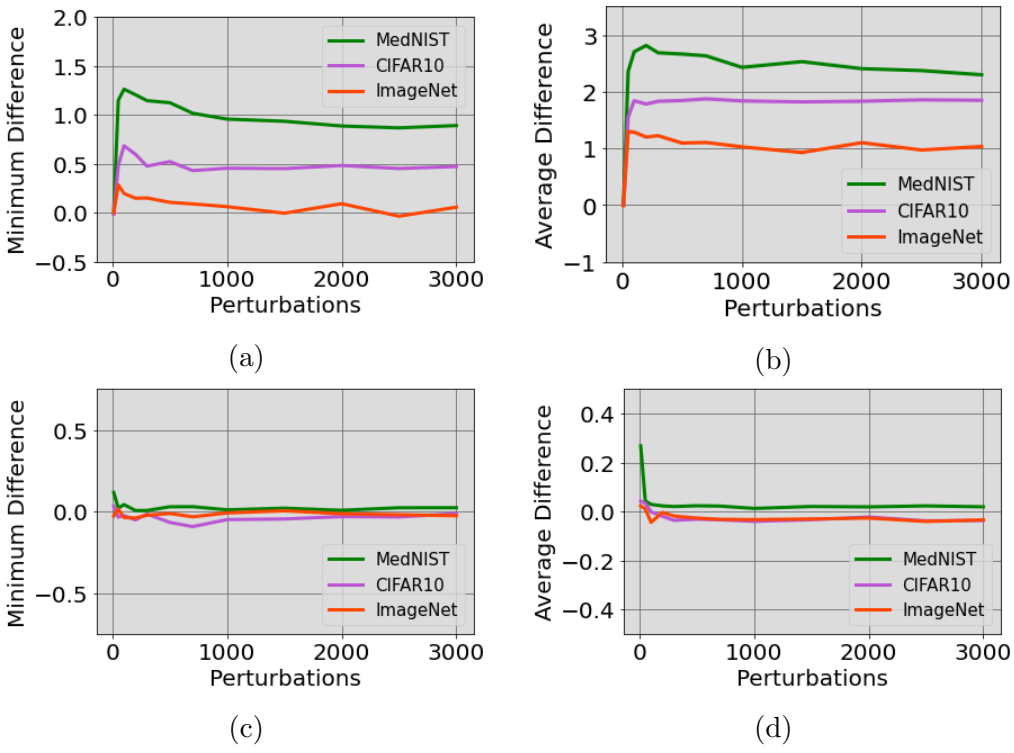
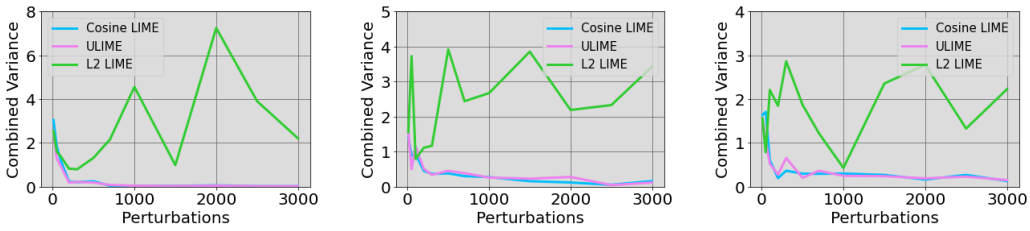


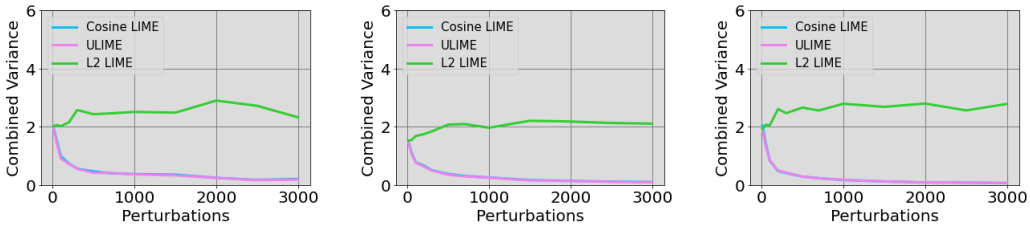

Figure 4.4: (a) Minimum difference between LnO's of L2 LIME and ULIME. (b) Average difference between LnO's of L2 LIME and ULIME. (c) Minimum difference between LnO's of Cosine LIME and ULIME. (d) Average difference between LnO's of Cosine LIME and ULIME. Differences have been averaged across entire datasets (CIFAR-10, ImageNet, and MedNIST).




(a) Chest X-ray (b) Eagle (c) Fish

Figure 4.5: Comparison of combined variance for the segmented images (a) Chest X-ray, (b) Eagle, and (c) Fish shown in Figure 4.2.

across 10 runs per perturbation quantity. This consistency increased as the perturbations were increased because the linear model learned from more data. In the case of L2 LIME, the combined variance showed no sign of decrease when the perturbations were increased. To illustrate this concept, Figure 4.5 compares the combined variance between L2 LIME, Cosine LIME, and ULIME for images in Figure 4.2.




(a) Imagenet (b) CIFAR-10 (c) MedNIST

Figure 4.6: Comparison of the combined variance of L2 LIME, Cosine LIME and ULIME per dataset (ImageNet, CIFAR-10, and MedNIST).

Figure 4.6 shows the average combined variance between coefficients for L2 LIME, Cosine LIME, and ULIME explanations for all three datasets. In both cases, it can be seen that L2 LIME explanations take longer to stabilize compared to ULIME and Cosine LIME explanations. Within 3000 perturbations, there is no indication of a decrease in the combined variance for the explanations generated with L2 LIME. However, for ULIME and Cosine LIME, an instant decrease in combined variance can

be observed for the three datasets.

# Chapter 5

# Experimenting ULIME with CT Datasets

In this chapter, we investigate the application of Cosine LIME, L2 LIME, and ULIME on two CT image datasets, one containing liver tumors, and the other pancreatic tumors. The primary goal of this chapter is to understand how different superpixelization techniques affect the quality of the explanation and the role of interpretability methods in establishing clinicians' trust in deep learning models trained on real-world healthcare datasets. A continuing goal is to compare L2 LIME, Cosine LIME, and ULIME, and to investigate if weights are really necessary for LIME image. This Chapter is not meant to serve as a complete book on applying domain-specific superpixelization on medical images, since this is a vast topic. We introduce the concept, propose our methodology, and leave further investigations for future research in this direction.

## 5.1 Datasets

Our liver tumor dataset contains 335 patient liver CT image segmentation. The tumor images belong to two classes; HCC (Hepatocellular Carcinoma) and MCRC

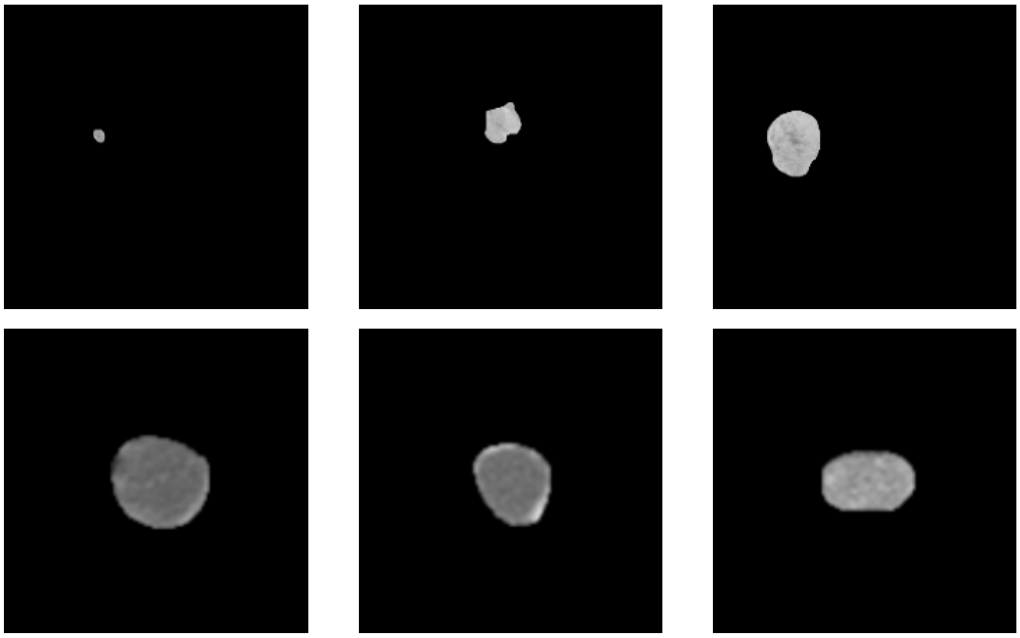


Figure 5.1: Top Row: Three liver tumor slices. Bottom Row: Three pancreas tumor slices.

(Metastatic Colorectal Cancer). Using one tumor per patient gave us 155 HCC tumors and 180 MCRC tumors. Our pancreatic mass dataset contains 529 patient pancreas CT image segmentation. The tumor/mass images belong to three classes; IPMN (Intraductal Papillary Mucinous Neoplasms), PNET (Pancreatic Neuroendocrine Tumor), and PDAC (Pancreatic Ductal Adenocarcinoma). Using one tumor/mass per patient, gave us 103 IPMN, 165 PNET, and 261 PDAC.

## 5.2 Preprocessing

The preprocessing is similar for both CT datasets. We first create a mask from each tumor CT image, by setting all pixels outside the tumor to zero and inside the tumor to one. We then dilate the mask using a 5x5 kernel so that the mask can capture features beyond the tumor, such as the tumor boundary. This is because in

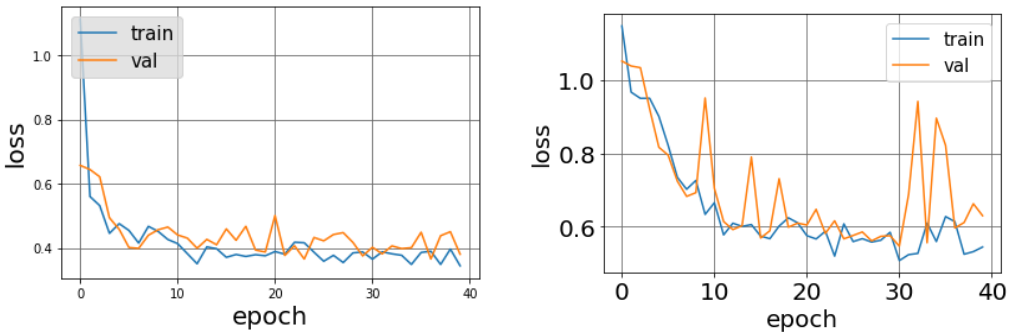
some cases, tumor boundary has clinical significance in the prognosis of liver disease [10]. Afterward, we create new CT images by multiplying the corresponding slices of the tumor mask with the liver (liver tumor) and pancreas (pancreatic tumor) segmentation effectively extracting the tumor plus extra boundaries surrounding the tumor. Finally, we extract the largest tumor plus boundary slice from each CT image by counting the total number of pixels that are greater than -1000 (background). To ensure the features of the tumor were visible, Hounsfield units from -135 to 215 were linearly mapped to gray levels from (0 to 1)/(0 to 255) when converting to PNG. For pancreatic tumors only, we magnify the tumor by further cropping it to 120x120 pixels from the .PNG images. These images, shown in Figure 5.1, were used to train our 2D CNN model.

## 5.3 Training

For the liver dataset, our train size was 285, 132 HCC, and 153 MCRC, and our test size was 50, 23 HCC, and 27 MCRC. For the pancreatic dataset, our train size was 398, 78 IPMN, 196 PDAC, and 124 PNET and our test size was 131, 25 IPMN, 65 PDAC, and 41 PNET. For both datasets, we used the same 2D-CNN model as used previously for the brain CT slice dataset in Figure 4.1, however, for the liver tumor dataset, the model was slightly modified by removing the last convolution layer. The number of epochs for training was set to 40. The loss function was the cross-entropy loss. We used the Adam optimizer with a learning rate set to 0.0002 and weight decay set to 1e-4. The weights were saved throughout the training process whenever the validation loss reached a new minimum.

## 5.4 Results

Figure 5.2 shows the train and validation loss for both models. The liver model achieved 80% validation accuracy and the pancreatic model achieved 81% accuracy. Table 5.1 and 5.2 shows the performance metrics and confusion matrix for the liver model and table 5.3 and 5.4 shows the performance metrics and confusion matrix for the pancreas model. Since our main objective was to apply different superpixelization techniques and compare L2 LIME, Cosine LIME, and ULIME, therefore we did not further tune the models, and instead focused on the interpretability analysis.




(a) (b)

Figure 5.2: Loss graphs of models trained on (a) liver tumor slices and (b): pancreatic tumor/mass slices.

| Class | Precision | Recall | F1-Score |
|---|---|---|---|
| HCC | 0.76 | 0.83 | 0.79 |
| MCRC | 0.84 | 0.78 | 0.81 |

Table 5.1: Liver model's Precision, Recall and F1-Score.

| | HCC | MCRC |
|---|---|---|
| HCC | 19 | 4 |
| MCRC | 6 | 21 |

Table 5.2: Liver model's Confusion Matrix.

| Class | Precision | Recall | F1-Score |
|---|---|---|---|
| IPMN | 0.96 | 0.96 | 0.96 |
| PDAC | 0.83 | 0.80 | 0.81 |
| PNET | 0.70 | 0.73 | 0.71 |

Table 5.3: Pancreas model's Precision, Recall and F1-Score.

| | IPMN | PDAC | PNET |
|---|---|---|---|
| IPMN | 24 | 1 | 0 |
| PDAC | 0 | 52 | 13 |
| PNET | 1 | 10 | 30 |

Table 5.4: Pancreas model's Confusion Matrix.

## 5.5 Model Interpretability using LIME and ULIME

We tested L2 LIME, Cosine LIME, and ULIME on both models to see what features it looks for when making a prediction. Figure 5.5 shows the heatmaps for each LIME version. As can be seen, in some cases, L2 LIME does focus on the tumor but there are more variations in the explanations compared to ULIME and Cosine LIME. On the other hand, both ULIME and Cosine have similar heatmaps. Figure 5.4 shows the average and minimum difference between LnO's of L2 LIME, ULIME, and Cosine LIME explanations. In both metrics, the difference between ULIME and L2 LIME is mostly greater than zero meaning that ULIME shows a quicker accuracy decline compared to L2 LIME. For Cosine LIME and ULIME, the difference is near zero. In

the Liver difference specifically, we omit the MCRC class, because, regardless of the superpixelization technique, the model looks at the background to make its prediction. This makes sense since HCC tumors are typically much larger than MCRC tumors so the model may be using the size difference as a discriminator between the two classes.

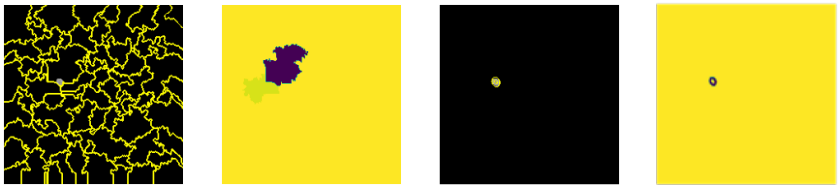


Figure 5.3: The model focuses on the background for MCRC tumors.

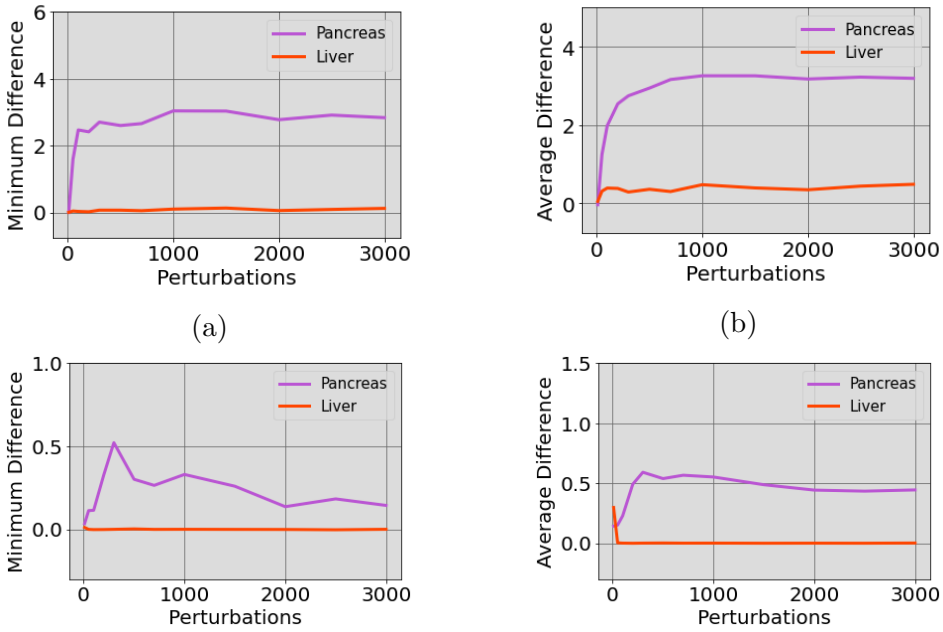



Figure 5.4: (a) Minimum difference between LnO's of L2 LIME and ULIME. (b) Average difference between LnO's of L2 LIME and ULIME. (c) Minimum difference between LnO's of Cosine LIME and ULIME. (d) Average difference between LnO's of Cosine LIME and ULIME. Differences have been averaged across entire datasets (Liver Tumor and Pancreatic Tumor).

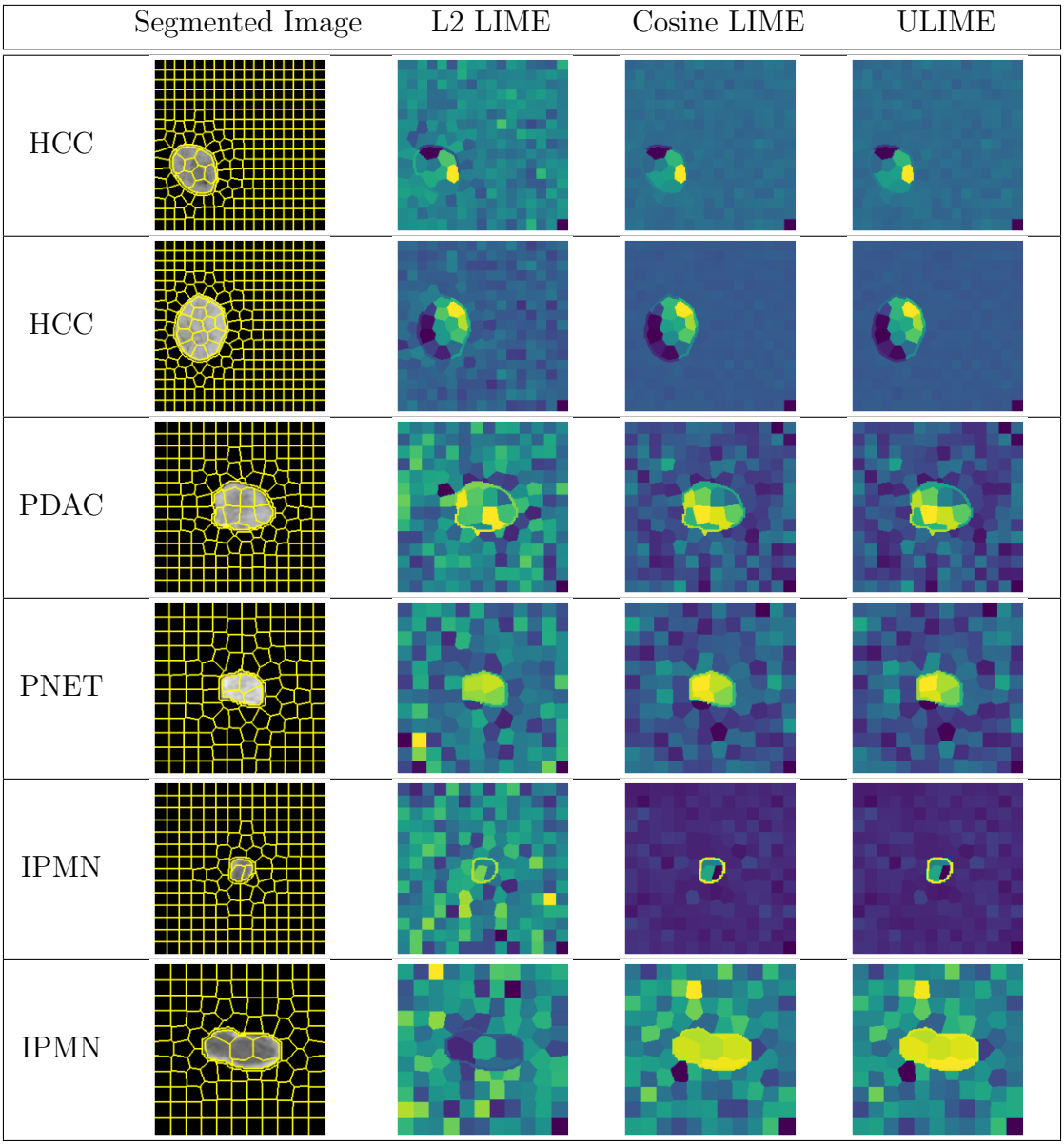
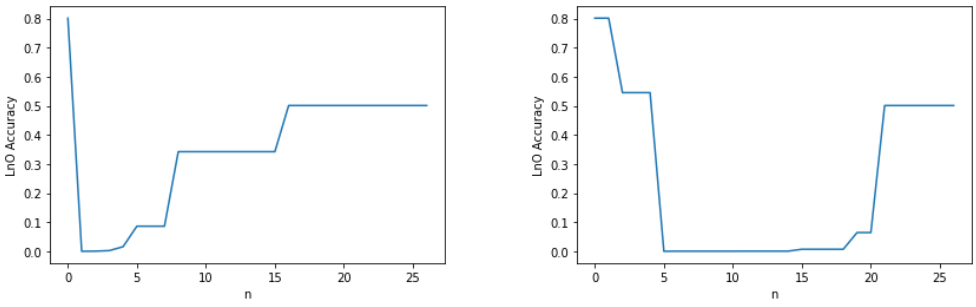



Figure 5.5: Explanation comparison between L2 LIME, Cosine LIME and ULIME for models trained on liver and pancreatic tumor data. Cosine and ULIME explanations are very similar, and both are better than L2 LIME's explanations.

We do understand that in this scenario, a simpler technique such as a linear regression applied to white pixel count may be equally effective in differentiating

between the two tumors, but we trained a CNN model solely for the purposes of testing ULIME.

One limitation of LnO is that for a smaller number of classes, as the superpixels are removed and replaced with zeros, the model ends up assigning a class to the completely zeroed background. For example, in Figure 5.6, two LnO's have been shown for a PDAC image. The left LnO is for ULIME and the right LnO is for L2 LIME. Ideally, when the tumor superpixels are removed and replaced with zeros one by one, the accuracy should decline and not raise up again, but since the model has to ultimately assign a classification value to the completely zeroed image, it ends up going up to 0.5. Therefore, even though ULIME explanation shows quicker accuracy decline, instead of the difference of L2 LIME LnO - ULIME LnO being large, it remains small. Different techniques can be used to solve this but we ended up taking only a fourth of the LnO's (The part that actually indicated information loss) for those particular classes. Note that this issue exists minimally when the number of classes is high since the zeroed image is more likely to get distributed to all classes.



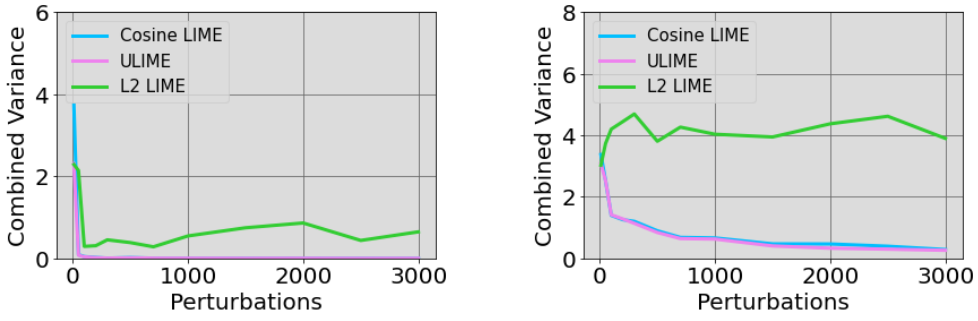

(a) (b)

Figure 5.6: LnO limitation shown for PDAC. (a): ULIME. (b): L2 LIME

## 5.6 Impact of Superpixelization on Explanations

Superpixelization has a direct effect on the intuitiveness of an explanation. If the superpixelization is not done based on the domain of the image, the explanation can end up highlighting unimportant parts of the image. This may lead to a user believing the problem exists in the model. Domain-specific superpixelization means segmenting the image into discreet parts where each part contains a single important region of the image. The importance of the region has to be determined by an expert in that domain. If we take an image of an animal, superpixelization should divide the body into separate parts so that the explanation can highlight a particular part, e.g. the ears of a cat. If the superpixel contains information beyond one body part, it becomes unclear what part the model was looking for.

Figure 5.8 shows three types of pancreatic tumors. The tumors have been superpixelized using two techniques; the author's default Quickshift algorithm and our modified SLIC algorithm. For liver tumors, radiologists usually observe the boundaries to predict the class of a tumor. Therefore, we modify our superpixelization




(a) (b)

Figure 5.7: Variance of (a) liver tumor explanations and (b) pancreatic tumor explanations

method so that it does a better job segmenting the boundary and the internals of the tumor, giving more insight into what a model is looking for inside and around the tumor itself. Since the boundary information is important for the IPMN, we can immediately see an improvement in the explanation quality when we separate the boundary from the tumor's internals.

In our modified SLIC, we do not make changes to the algorithm itself, rather we modify the input image before SLIC is applied. We do this by first converting the tumor to a binary mask and eroding it using a 5x5 kernel making it slightly smaller. Then we invert the erosion so that the tumor region is zero and the background is one, and apply the binary AND operation between the original binary mask and the eroded inverted mask. This gives us the boundary of the tumor only, with everything else black. Afterward, we apply the SLIC algorithm to the boundary, thereby separating the boundary from the tumor internals, and play around with the parameters to further subdivide the separated boundary and internals. Finally, we overlay those superpixels back on top of the original tumor. Please note that this is one of the many ways domain-specific superpixelization can be achieved. We initiate domain-specific superpixelization in our thesis by introducing one method, specifically for the liver tumors where boundary information is important, and leave a broader study for future research.

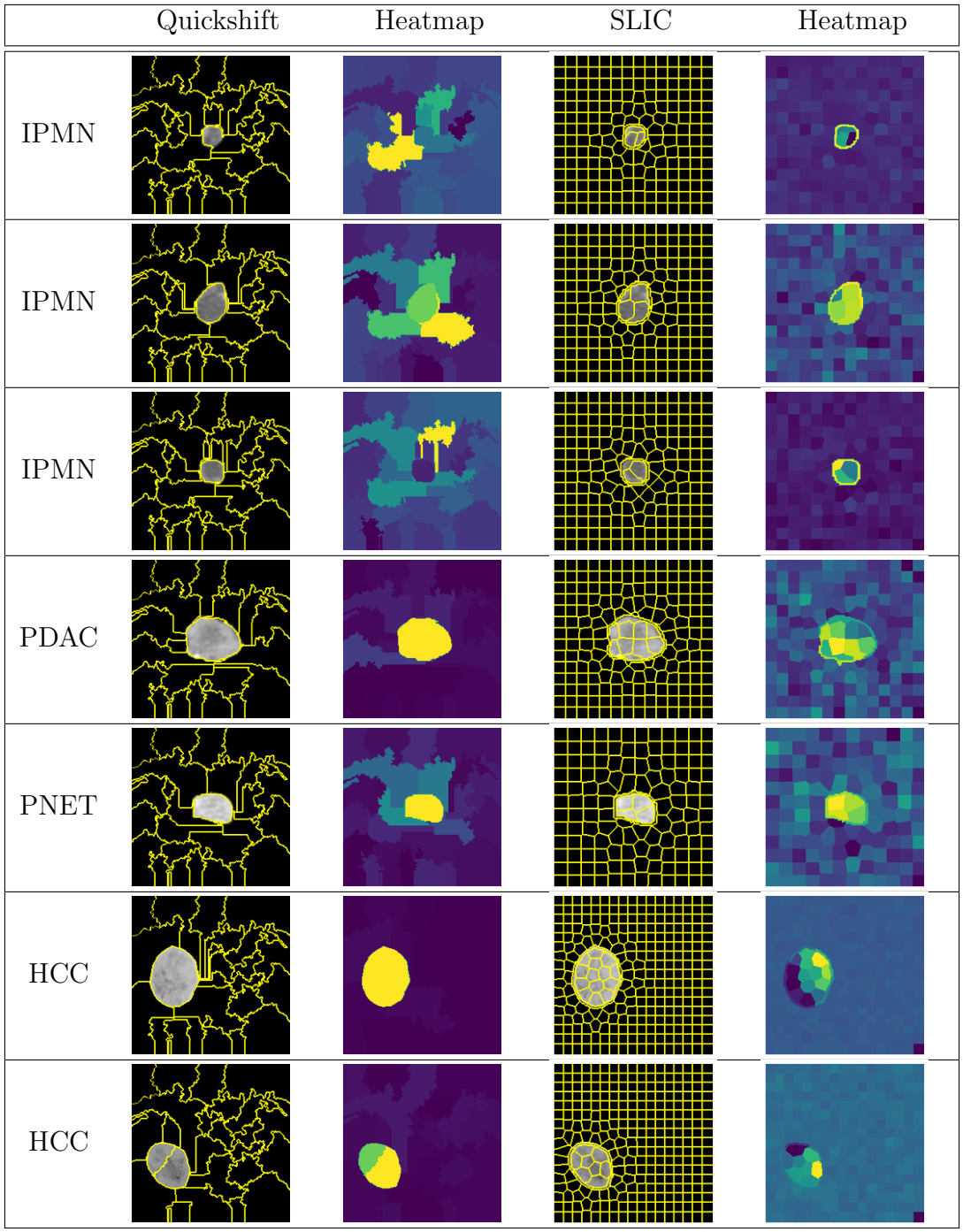



Figure 5.8: Domain-specific superpixelization and its impact on explanations.

## 5.7 Interpretable AI Compared to Radiologist Interpretation

As part of our research, we showed a radiologist from Memorial Sloan Kettering Cancer Center the results of ULIME on several tumor types and requested feedback. The radiologist observed the explanations and added his comments in Figure 5.9 and Figure 5.10. He emphasized the need for deep learning models to be interpretable so that physicians can observe what parts of an image the model looks at, which will help understand if those areas are reasonably strong indicators for a particular disease. Finally, he proposed using LIME for the segmentation of blood vessels, i.e. segmenting the most important superpixels from the image. We believe that using LIME for segmentation can be a unique application and leave it for future research.

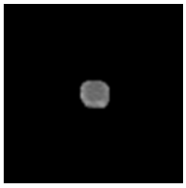
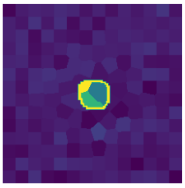
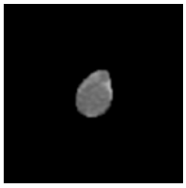
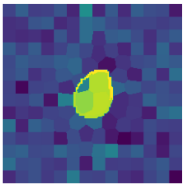
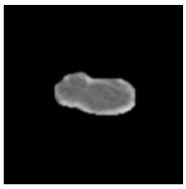
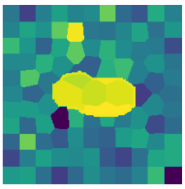
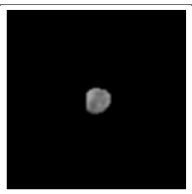
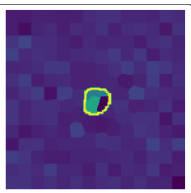
| Image | Heatmap |
|---|---|
| IPMN  |  |
| Yellow areas on LIME map emphasize thickened wall of cystic pancreatic lesion. A thickened wall is considered suspicious for malignancy on CT imaging, whereas a thin smooth wall would be considered benign. | |
| IPMN  |  |
| Nodular portion of the wall anteriorly on CT also correspond to yellow portions on LIME map. | |
| IPMN  |  |
| Nodular thickened wall along posterior wall correspond to yellow portion of the LIME map. | |
| IPMN  |  |
| Irregular thickened wall corresponding to yellow portion of the LIME map. In solid pancreatic malignancies, such as pancreatic adenocarcinoma (PDAC), the tumors will be more internally heterogeneous on CT compared to cystic neoplasms above. | |

Figure 5.9: Radiologist comments on our modified superpixelization tumor explanations for IPMN.

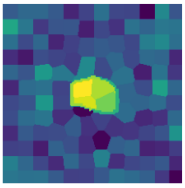
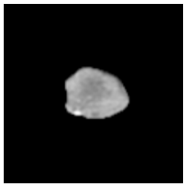
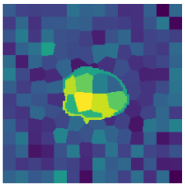
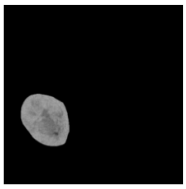
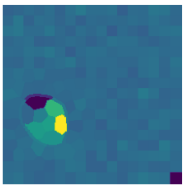
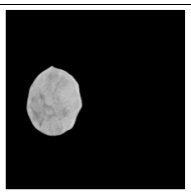
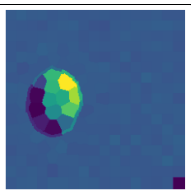
| Image | Heatmap |
|---|---|
| PNET  |  |
| Pancreatic neuroendocrine tumors (PNET) are more uniformly, hypervascular (bright) on CT. The LIME map shows relatively uniform green to yellow components of the map overlapping the tumor. | |
| PDAC  |  |
| The tumor is centrally necrotic (slightly darker in the center), but not cystic. The LIME map shows the most yellow portion, centrally. | |
| HCC  |  |
| Hepatocellular carcinoma (HCC) is a primary liver malignancy. The yellow portion of the LIME map overlaps with a slightly more heterogeneous enhancing portion of the tumor which contains a dark (nearly black) spot, which may be an area of fat, a feature that is relatively unique to HCC as a malignancy. | |
| HCC  |  |
| HCC with a mosaic appearance, a description that relates to the mixing of different histological features, including viable tumor, necrosis, and fibrosis. The yellow portion of the LIME map appears to coincide with a septa (a linear dark grey line), which is distinct in this tumor compared to pancreatic tumors above. | |

Figure 5.10: Radiologist comments on our modified superpixelization tumor explanations for PDAC, PNET, and HCC.

## 5.8 Superimposing Explanations on 3D Image

The explanations we see as heatmaps are 2D images for a particular tumor slice taken from a larger CT image. In order to overlay the explanations back to the CT image so that it's more convenient for radiologists to inspect using their software, such as 3D Slicer, we first find the index of the tumor slice that we extracted and create an empty 3D binary mask with the explanation placed exactly at that index. We then save the 3D binary mask as a new CT image, and then open the original liver and the binary mask in 3D Slicer. Afterward, we overlay both CT images together giving us the explanation of the tumor placed on the liver from which the tumor was extracted. Figure 5.11 illustrates the overlaying of the HCC tumor on the liver.

(a) (b)

Figure 5.11: Overlaying ULIME explanation for HCC tumor on the liver in 3D Slicer. (a): Explanation Overlay. (b): Original Image

## 5.9 Conclusion

In this chapter, we trained our CNN model on two CT image datasets, liver, and pancreatic tumor datasets. The model trained well on both datasets. A comparison of L2 LIME, Cosine LIME, and ULIME showed that ULIME explanations were comparable to Cosine LIME, and both were more stable and better qualitatively compared

to L2 LIME. Finally, we touched upon how different superpixelization techniques can improve the visual quality and give more insights into what the black-box model is looking at in an image.

# Chapter 6

# Conclusion

Interpretable machine learning is a relatively new field that has gathered interest due to the potential dangers of using black-box systems for critical applications. Regulatory bodies, such as the U.S. FDA, have announced guidelines for medical devices that use AI to be fully transparent to ensure physician/patient safety. To achieve full transparency, more research is still left to be done in this field, and our work contributes to this growing body of research. In this chapter, we summarize our contributions and propose directions for future research.

## 6.1 Summary

In this thesis, we focused on the fundamental principle of LIME: locality. The distance metric had a significant impact on valuing the similarity and dissimilarity between perturbations. When we chose the distance metric to be L2, we observed high variations in the explanations. Changing to a normalized Cosine metric, a default used by LIME authors, improved both the quality and stability of explanations. Since the perturbations and prediction labels already indicated which features were important, we removed the weighting step in our variant, ULIME, and found that both

ULIME and Cosine LIME demonstrated similar performance across all benchmarks, and both were better compared to L2 LIME. Therefore, our motivation for ULIME was to simplify the interpretability technique, since removing the weighting step removes a hyper-parameter that is responsible for the variation in explanations if chosen incorrectly.

We also focused on domain-specific superpixelization for tumor images as an improved preprocessing step. Using our modified superpixelization method, ULIME explanations for liver and pancreatic tumor images were better in quality than the standard. These explanations were further assessed by a domain expert (radiologist).

## 6.2 Future Work

There are several research questions worth considering for further research:

1. Are LIME explanations completely decoupled with the type of superpixelization algorithm used? If we were to segment an image using two different segmentation techniques, would the important superpixels in the first segmentation technique match with the second?

2. How precisely can an image be segmented into smaller regions as a preprocessing step? Is there a limit beyond which the explanation quality will start deteriorating? Related, how does the number of perturbations required to converge change with the granularity of the superpixels? More superpixels could mean the number of possible perturbations grows exponentially and the finest granularity would be just pixels.

3. Is there a better way to pick the perturbations for LIME? We randomly generated perturbations to train the linear model which significantly decreased the speed of the algorithm. Picking only the needed perturbations and/or finding a better way to generate labels for perturbations would improve LIME performance.

4. Are there other metrics that better reflect image interpretability? There remain many parameters in LIME for the user to decide, and we only touched on one (the distance metric). Future investigation may include exploring the significance of different types of interpretable models, interpretable model configuration, and a number of features to use for the interpretable model.

# Bibliography

[1] Sushant Agarwal, Shahin Jabbari, Chirag Agarwal, Sohini Upadhyay, Steven Wu, and Himabindu Lakkaraju. Towards the unification and robustness of perturbation and gradient based explanations. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 110–119. PMLR, July 2021.

[2] David Alvarez-Melis and T. Jaakkola. On the robustness of interpretability methods, 2018.

[3] Julia Amann, Alessandro Blasimme, Effy Vayena, Dietmar Frey, and Vince Madai. Explainability for artificial intelligence in healthcare: a multidisciplinary perspective. *BMC Medical Informatics and Decision Making*, 20:310, November 2020.

[4] Elvio Amparore, Alan Perotti, and Paolo Bajardi. To trust or not to trust an explanation: using LEAF to evaluate local linear XAI methods. *PeerJ Computer Science*, 7:e479, April 2021.

[5] Emma Beede, Elizabeth Baylor, Fred Hersch, Anna Iurchenko, Lauren Wilcox,

Paisan Ruamviboonsuk, and Laura M. Vardoulakis. A human-centered evaluation of a deep learning system deployed in clinics for the detection of diabetic retinopathy. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–12, New York, NY, USA, 2020. Association for Computing Machinery.

[6] J.K. Blitzstein and J. Hwang. *Introduction to Probability, Second Edition*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, 2019.

[7] Romain Cadario, Chiara Longoni, and Carey K. Morewedge. Understanding, explaining, and utilizing medical artificial intelligence. *Nature human behaviour*, 5(12):1636–1642, 2021.

[8] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8), 2019.

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

[10] Jie-Neng Chen, Ke Yan, Yu-Dong Zhang, Youbao Tang, Xun Xu, Shuwen Sun, Qiuping Liu, Lingyun Huang, Jing Xiao, Alan L. Yuille, Ya Zhang, and Le Lu. Sequential learning on liver tumor boundary semantics and prognostic biomarker mining. In Marleen de Bruijne, Philippe C. Cattin, Stéphane Cotin, Nicolas Padoy, Stefanie Speidel, Yefeng Zheng, and Caroline Essert, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, pages 764–774. Springer International Publishing, 2021.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[12] Saurabh Desai and Harish G. Ramaswamy. Ablation-cam: Visual explanations for deep convolutional network via gradient-free localization. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 972–980, 2020.

[13] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning, 2017.

[14] Radwa ElShawi, Youssef Sherif, Mouaz Al-Mallah, and Sherif Sakr. Interpretability in healthcare: A comparative study of local machine learning interpretability techniques. *Computational Intelligence*, 37(4):1633–1650, 2021.

[15] Bradley J. Erickson. Mednist. `https://www.dropbox.com/s/5wwskxctvcxiuea/MedNIST.tar.gz`, 2017. Accessed: 2021-08-22.

[16] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, October 2017.

[17] U.S. Federal Food and Drug Administration. Artificial intelligence and machine learning in software as a medical device. `https://www.fda.gov/medical-devices/software-medical-device-samd/artificial-intelligence-and-machine-learning-software-medical-device#regulation`, 2021.

[18] Damien Garreau and Ulrike von Luxburg. Explaining the explainer: A first theoretical analysis of lime. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1287–1296. PMLR, August 2020.

[19] Damien Garreau and Ulrike von Luxburg. Looking deeper into tabular lime, 2020.

[20] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 80–89, 2018.

[21] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. *A Benchmark for Interpretability Methods in Deep Neural Networks*, page 9737–9748. Curran Associates Inc., 2019.

[22] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, Donald Brown, Laura Id, and Barnes. Text classification algorithms: A survey. *Information (Switzerland)*, 10, April 2019.

[23] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021.

[26] Chiara Longoni, Andrea Bonezzi, and Carey K Morewedge. Resistance to Medical Artificial Intelligence. *Journal of Consumer Research*, 46(4):629–650, May 2019.

[27] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

[28] Dina Mardaoui and Damien Garreau. An analysis of lime for text data. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 3493–3501. PMLR, April 2021.

[29] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[30] Christoph Molnar. *Interpretable Machine Learning*. 2019.

[31] Mohammed Bany Muhammad and Mohammed Yeasin. Eigen-cam: Class activation map using principal components. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2020.

[32] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov,

and Shan Carter. An overview of early vision in inceptionv1. *Distill*, 2020. https://distill.pub/2020/circuits/early-vision.

[33] Huy Phan. Cifar10 pretrained. `https://github.com/huyvnphan/PyTorch_CIFAR10`, 2021. Accessed: 2021-08-15.

[34] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA*, pages 1135–1144, 2016.

[35] Jordan P. Richardson, Cambray Smith, Susan Curtis, Sara Watson, Xuan Zhu, Barbara Barry, and Richard R. Sharp. Patient apprehensions about the use of artificial intelligence in healthcare. *npj Digital Medicine*, 4(1):140, September 2021.

[36] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206–215, May 2019.

[37] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673, 2017.

[38] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from

deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[39] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, Conference Track Proceedings*, 2015.

[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, Los Alamitos, CA, USA, June 2016. IEEE Computer Society.

[41] Sana Tonekaboni, Shalmali Joshi, Melissa D. McCradden, and Anna Goldenberg. What clinicians want: Contextualizing explainable machine learning for clinical end use. In Finale Doshi-Velez, Jim Fackler, Ken Jung, David Kale, Rajesh Ranganath, Byron Wallace, and Jenna Wiens, editors, *Proceedings of the 4th Machine Learning for Healthcare Conference*, volume 106 of *Proceedings of Machine Learning Research*, pages 359–380. PMLR, August 2019.

[42] Andrea Vedaldi and Stefano Soatto. Quick shift and kernel methods for mode seeking. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, pages 705–718, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[43] Giorgio Visani, Enrico Bagli, Federico Chesani, Alessandro Poluzzi, and Davide Capuzzo. Statistical stability indices for LIME: Obtaining reliable explanations

for machine learning models. *Journal of the Operational Research Society*, pages 1–11, February 2021.

[44] H. Wang, Z. Wang, M. Du, F. Yang, Z. Zhang, S. Ding, P. Mardziel, and X. Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 111–119, Los Alamitos, CA, USA, June 2020. IEEE Computer Society.

[45] Hujun Yin, David Camacho, Peter Tino, Antonio J. Tallón-Ballesteros, Ronaldo Menezes, and Richard Allmendinger, editors. *Intelligent Data Engineering and Automated Learning – IDEAL 2019: 20th International Conference, Manchester, UK, November 14–16, 2019, Proceedings, Part II*, volume 11872 of *Lecture Notes in Computer Science*. Springer International Publishing, 2019.

[46] Muhammad Rehman Zafar and Naimul Khan. Deterministic local interpretable model-agnostic explanations for stable explainability. *Machine Learning and Knowledge Extraction*, 3(3):525–541, 2021.

[47] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 818–833, Cham, 2014. Springer International Publishing.

[48] Q. Zhang, Y. Wu, and S. Zhu. Interpretable convolutional neural networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8827–8836, Los Alamitos, CA, USA, June 2018. IEEE Computer Society.

[49] Xingyu Zhao, Wei Huang, Xiaowei Huang, Valentin Robu, and David Flynn. Baylime: Bayesian local interpretable model-agnostic explanations. In Cassio de Campos and Marloes H. Maathuis, editors, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 887–896. PMLR, July 2021.

[50] Yilun Zhou, Serena Booth, Marco Tulio Ribeiro, and Julie Shah. Do feature attribution methods correctly attribute features?, 2021.