What Does LIME Really See in Images?

Damien Garreau [1] **Dina Mardaoui** [2]

Abstract

The performance of modern algorithms on certain computer vision tasks such as object recognition is now close to that of humans. This success was achieved at the price of complicated architectures depending on millions of parameters and it has become quite challenging to understand how particular predictions are made. Interpretability methods propose to give us this understanding. In this paper, we study LIME, perhaps one of the most popular. On the theoretical side, we show that when the number of generated examples is large, LIME explanations are concentrated around a limit explanation for which we give an explicit expression. We further this study for elementary shape detectors and linear models. As a consequence of this analysis, we uncover a connection between LIME and integrated gradients, another explanation method. More precisely, the LIME explanations are similar to the sum of integrated gradients over the superpixels used in the preprocessing step of LIME.

1. Introduction

Deep neural networks and deep convolutional neural networks (CNN) in particular have changed the way computers look at images (Schmidhuber, 2015). Many specific tasks in computer vision such as character recognition and object recognition are now routinely achieved by personal computers with human-like accuracy. The success of these algorithms seems partly due to the great complexity of the models they encode, the most recent relying on hundreds of layers and millions of parameters.

While the accuracy is often the only relevant metric for practitioners, there are numerous situations where one is not satisfied if the model is making good predictions for the wrong reasons. We would like to know *why* the model makes a particular prediction. Responding to this emerging need, many *interpretability methods* have appeared in the last five years. Among them, *model agnostic* methods aim to provide to the user meaningful insights on the inner working of a specific algorithm without making any specific assumption on the architecture of the model. We refer to Adadi and Berrada (2018); Guidotti et al. (2018) and Linardatos et al. (2021) for recent review papers.

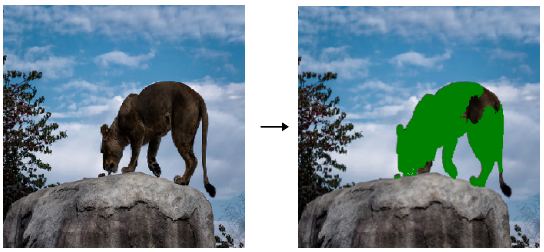

Figure 1. Explaining a prediction with LIME. In this example, the function to be explained f is the likelihood, according to the InceptionV3 network, that the input image ξ contains a lion. After a run of LIME with default parameters, the top five positive coefficients are highlighted in the right panel.

In this paper, we study the image version of LIME (Local Interpretable Model-agnostic Explanations, Ribeiro et al., 2016). Let us recall briefly how it operates: in order to explain the prediction of a model f for an example ξ, LIME

1. decomposes ξ in d superpixels, that is, small homogeneous image patches;
2. creates a number of new images $x_1, \dots, x_n$ by *randomly turning on and off* these superpixels;
3. queries the model, getting predictions $y_i = f(x_i)$;
4. builds a local weighted surrogate model $\hat{\beta}_n$ fitting the y_is to the presence or absence of superpixels.

Each coefficient of $\hat{\beta}_n$ is associated to a superpixel of the original image ξ and, intuitively, the more positive the more important the superpixel is for the prediction at ξ according to LIME. Generally, the user visualizes $\hat{\beta}_n$ by highlighting

[1]Université Côte d'Azur, Inria, CNRS, LJAD, France [2]Polytech Nice. Correspondence to: Damien Garreau <damien.garreau@univ-cotedazur.fr>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

the superpixels associated to the top positive coefficients (usually five, see Figure 1).

The central question underlying this work is that of the soundness of LIME for explaining simple models: before using LIME on deep neural networks, are we sure that the explanations provided make sense for the most simple models? Can we guarantee it theoretically?

Contributions. Our contributions are the following:

- when the number of perturbed examples is large, the interpretable coefficients **concentrate with high probability around a vector** β that depends only on the model and the example to explain;
- we provide an **explicit expression for** β, from which we gain some reassurance on LIME. In particular, the explanations are **linear** in the model;
- for simple **shape detectors**, we can be more precise in the computation of β and we show that **LIME provides meaningful explanations** in that case;
- we can also compute β for **linear models**. The limit explanation takes a very simple form: β_j is **the sum of coefficients multiplied by pixel values on each superpixel**;
- as a consequence, we show experimentally that for models that are sufficiently smooth with respect to their inputs, the outputs of LIME are similar to the sum over superpixels of **integrated gradients**, another interpretability method.

Related work. While some weaknesses of LIME are well-known, in particular its vulnerability adversarial attacks (Slack et al., 2020), investigating whether the produced explanations do make sense is still an ongoing area of research. (see for instance Narodytska et al. (2019)). The present work follows the line of ideas initiated by Garreau and von Luxburg (2020a;b) for the tabular data version of LIME and later extended to text data by Mardaoui and Garreau (2021). In particular, our main result and its proof are similar to the theory laid out in these papers. The interesting differences come from the sampling procedure of LIME for images: there is no superpixel creation step in the text and tabular data version of the algorithm. Therefore, the exact expression of the limit explanations and the associated conclusions differ.

Organization of the paper. We start by presenting LIME for images in Section 2. Section 3 contains our main results, which are further developed for simple models in Section 4. Finally, we investigate the link between LIME and integrated gradients in Section 5.

2. LIME for Images

From now on, we consider a model $f : [0,1]^D \to \mathbb{R}$ as well as a fixed example to explain $\xi \in [0,1]^D$. Hence D denotes the number of pixels of the images on which f operates. In practice, the inputs of f are always 2- or 3-dimensional arrays. Of particular interest, grayscale images are usually encoded as $h \times w$ arrays, whereas RGB images are $h \times w \times 3$, with each channel corresponding to a primary color. We will see that it does not make a difference and our results can be read *channel-wise* if there is more than one color channel.

2.1. Superpixels

The first step of the LIME operation is to split ξ into *superpixels*. These are contiguous patches of the image that share color and / or brightness similarities. We refer to Figure 2 for an illustration. In the text version of LIME, the counterpart of this superpixel decomposition is a local dictionary where each interpretable feature is a unique word of the text, whereas in the tabular version a complicated discretization procedure is needed.

By default, LIME uses the *quickshift* algorithm to produce these superpixels (Vedaldi and Soatto, 2008). In a nutshell, quickshift is a mode-seeking algorithm that considers the pixels as samples over a 5-dimensional space (3 color dimensions and 2 space dimensions).

For any $1 \leq k \leq d$, we denote the kth superpixel associated to ξ by J_k. Therefore, the d subsets $J_1, \ldots, J_d$ form a partition of the pixels, that is,

$$J_1 \cup \cdots \cup J_d = \{1, \ldots, D\} \quad \text{and} \quad J_k \cap J_\ell = \emptyset \ \ \forall k \neq \ell \,.$$

Note that, even though the superpixels are generally contiguous patches of the image, we do not make this assumption.

2.2. Sampling

As we have seen in Section 1, one of LIME's key ideas is to create *new examples* from ξ by randomly replacing some superpixels of the image. By default, these chosen superpixels are replaced by the mean color of the superpixel, a procedure that we call *mean replacement*. It is also possible to choose a specific color as a replacement image. We demonstrate the sampling procedure in Figure 2 as well as the two possible choices for the replacement image.

Let us be more precise and let us assume that ξ is fixed and $J_1, \ldots, J_d$ are given. The first step of the sampling scheme is to compute the replacement image $\overline{\xi} \in [0,1]^D$. If a given color c is provided, then $\overline{\xi}_u = c$ for all $1 \leq u \leq D$. If no color is provided, then the mean image is computed: for any superpixel J_k, we define $\overline{\xi} \in [0,1]^D$ by

$$\forall u \in J_k, \quad \overline{\xi}_u = \frac{1}{|J_k|} \sum_{u \in J_k} \xi_u \,. \tag{1}$$

Figure 2. Sampling procedure of LIME for images. The image to explain, ξ, is first split into d superpixels (*lower left corner*, here $d = 72$). A replacement image $\overline{\xi}$ is computed, which is by default the mean of ξ on each superpixel (*top row*), see Eq. (1). This replacement image can also be filled uniformly with a pre-determined color (*bottom row:* replacement with the color black). Then, for each new generated example x_i with $1 \leq i \leq n$, the superpixels are randomly switched depending on the throw of d independent Bernoulli random variables with parameter $1/2$. Thus LIME creates n new images where key parts of ξ disappear at random.

Of course, if the input images have several channels, the mean is computed on each channel.

Then, for each $1 \leq i \leq n$, LIME samples a random vector $z_i \in \{0,1\}^d$ where each coordinate of z_i is i.i.d. Bernoulli with parameter $1/2$. Each $z_{i,j}$ corresponds to the activation ($z_{i,j} = 1$) or inactivation ($z_{i,j} = 0$) of superpixel j. We call the z_is the *interpretable features*. To be precise, for any given $i \in \{1, \ldots, n\}$, the new example $x_i \in [0,1]^D$ has pixel values given by

$$\forall u \in J_j, \quad x_{i,u} = z_{i,j}\xi_u + (1 - z_{i,j})\overline{\xi}_u \,. \tag{2}$$

Again, if ξ has several color channels, Eq. (2) is written channel-wise. Note that ξ corresponds to the vector $\mathbf{1} = (1, \ldots, 1)^\top$ (all the superpixels of the image are activated).

2.3. Weights

Of course, the new examples x_i can be quite different from the original image. For instance, if most of the $z_{i,j}$ are zero, then x_i is close to $\overline{\xi}$. Some care is taken when building the surrogate model, and new examples are given a positive weight π_i that takes this proximity into account. By default, these weights are defined by

$$\forall 1 \leq i \leq n, \quad \pi_i := \exp\left(\frac{-d_{\cos}(\mathbf{1}, z_i)^2}{2\nu^2}\right), \tag{3}$$

where $\nu > 0$ is a positive *bandwidth parameter* equal to 0.25 by default and $d_{\cos}$ is the *cosine distance*. Namely,

$$\forall u, v \in \mathbb{R}^d, \quad d_{\cos}(u, v) := 1 - \frac{u^\top v}{\|u\| \cdot \|v\|} \,.$$

We see that $d_{\cos}(z_i, \mathbf{1})$ takes near zero values if most of the superpixels are activated, and values near 1 in the opposite scenario, as expected.

An important remark is that the weights π_i **depend only on the number of inactivated superpixels**. Indeed, conditionally to z_i having exactly s elements equal to zero, we have $z_i^\top \mathbf{1} = d - s$ and $\|z_i\| = \sqrt{d - s}$. Since $\|\mathbf{1}\| = \sqrt{d}$, using Eq. (3), we deduce that $\pi_i = \psi(s/d)$, where we defined

$$\forall t \in [0,1], \quad \psi(t) := \exp\left(\frac{-(1 - \sqrt{1-t})^2}{2\nu^2}\right). \tag{4}$$

2.4. Surrogate Model

The next stage of LIME is to build a *surrogate model*. More precisely, LIME builds a linear model with the interpretable features z_i as input and the model predictions $y_i := f(x_i)$ as responses. This linear model, in the default implementation, is obtained by (weighted) ridge regression (Hoerl and Kennard, 1970). Formally, the outputs of LIME for model f and image ξ are given by

$$\hat{\beta}_n^\lambda \in \operatorname*{arg\,min}_{\beta \in \mathbb{R}^{d+1}} \left\{ \sum_{i=1}^n \pi_i (y_i - \beta^\top z_i)^2 + \lambda \|\beta\|^2 \right\}, \tag{5}$$

where $\lambda > 0$ is a regularization parameter. We call the coordinates of $\hat{\beta}_n^\lambda$ the *interpretable coefficients*. By convention, the 0th coordinate of $\hat{\beta}_n^\lambda$ is the intercept of the model. Some feature selection procedure can be used: we do not consider such extensions in our analysis and keep to the default implementation, which is ridge.

Another important remark is the following: as in the text and tabular cases, LIME uses the default setting of `sklearn` for the regularization parameter, that is, $\lambda = 1$. Hence the first term in Eq. (5) is roughly of order n and the second term of order d. Since we experiment in the large n regime

($n = 1000$ is default) and with images split up in ≈ 100 superpixels, we are in a situation where $n \gg d$. Therefore, **we can consider that $\lambda = 0$ in our analysis and still recover meaningful results**. We will denote by $\hat{\beta}_n$ the solution of Eq. (5) with $\lambda = 0$, that is, ordinary least-squares.

The final step of LIME for images is to display the superpixels associated to the top *positive* coefficients of $\hat{\beta}_n^\lambda$ (usually five, see Figure 1). Part of what makes the method attractive to the practitioner is the ease with which one can read the results from one run of LIME just by looking at the highlighted part of the image. Note that it is also possible to highlight the superpixels associated to the top *negative* coefficients in another color, to see which parts of the image have a *negative* influence on the prediction.

3. Main Results

In this section we present our main results. Namely, the concentration of $\hat{\beta}_n$ around β^f (Section 3.1) and the expression of β^f as a function of f and other quantities (Section 3.2).

3.1. Concentration of $\hat{\beta}_n$

When the number of new samples n is large, we expect the empirical explanations provided by LIME to stabilize. Our first result formalizes this intuition.

Theorem 1 (Concentration of $\hat{\beta}_n$). *Assume that f is bounded by a constant $M > 0$ on $[0,1]^D$. Let $\epsilon > 0$ and $\eta \in (0,1)$. Let d be the number of superpixels. Then, there exists $\beta^f \in \mathbb{R}^{d+1}$ such that, for every*

$$n \gtrsim \max(M, M^2)\epsilon^{-2} d^7 \mathrm{e}^{\frac{4}{\nu^2}} \log \frac{8d}{\eta} \,,$$

we have $\mathbb{P}(\|\hat{\beta}_n - \beta^f\| \geq \epsilon) \leq \eta$.

We refer to the appendix for a complete statement (we omitted numerical constants and the intercept for clarity). Intuitively, Theorem 1 means that when n is large, $\hat{\beta}_n$ stabilizes around β^f. Thus we can focus on β^f to study LIME. The main limitation of Theorem 1 is the dependency on d and ν: the control that we achieve on $\|\hat{\beta}_n - \beta^f\|$ is quite poor whenever d is too large or ν is too small. Note also that $\hat{\beta}_n$ is given by the *non-regularized* version of LIME.

Theorem 1 is quite similar to Theorem 1 in Garreau and von Luxburg (2020b) and Theorem 1 in Mardaoui and Garreau (2021), which are essentially the same result for the tabular data and the text data version of LIME. The rate of convergence is slightly better here, but this seems to be an artifact of the proof and we do not think that one should sample less when dealing with images.

3.2. Expression of β^f

In this section we obtain the explicit expression of β^f. Before doing so, we need to introduce additional notation. From now on, we introduce the random variable $z \in \{0,1\}^d$ such that $z_1, \ldots, z_n$ are i.i.d. samples of z; it is the only source of randomness in the sampling and all expectations are taken with respect to it. We denote by π and x the associated weights and examples.

Definition 1 (α coefficients). Define $\alpha_0 := \mathbb{E}[\pi]$ and, for any $1 \leq p \leq d$, $\alpha_p := \mathbb{E}[\pi z_1 \cdots z_p]$.

Intuitively, when ν is large, α_p corresponds to the probability that exactly p superpixels of ξ are turned *on*. Since the sampling scheme of LIME for images is completely symmetrical as well as the definition of the weights, we see that this probability does not depend on the exact set of indices, hence the definition of the α coefficients. We show in appendix that the expected covariance matrix of problem (5) can be written with the first three α coefficients. Though Definition 1 is identical to Definition 3 in Mardaoui and Garreau (2021), the exact expression of the α coefficients is different in this case since the sampling procedure differs.

Proposition 1 (Computation of the α coefficients). *Let $d \geq 2$ and $0 \leq p \leq d$. For any $\nu > 0$, it holds that*

$$\alpha_p = \frac{1}{2^d} \sum_{s=0}^{d} \binom{d-p}{s} \psi\left(\frac{s}{d}\right) \,,$$

where ψ is defined as in Eq. (4).

We prove Proposition 1 in the appendix. From the α coefficients, we then form the normalization constant

$$c_d := (d-1)\alpha_0\alpha_2 - d\alpha_1^2 + \alpha_0\alpha_1 \,,$$

and the σ coefficients:

Definition 2 (σ coefficients). For any $d \geq 2$ and $\nu > 0$, define

$$\begin{cases} \sigma_1 & := -\alpha_1 \,, \\ \sigma_2 & := \frac{(d-2)\alpha_0\alpha_2 - (d-1)\alpha_1^2 + \alpha_0\alpha_1}{\alpha_1 - \alpha_2} \,, \\ \sigma_3 & := \frac{\alpha_1^2 - \alpha_0\alpha_2}{\alpha_1 - \alpha_2} \,. \end{cases}$$

We show in appendix that the *inverse of the expected covariance matrix* associated to problem (5) can be expressed with the help of the σ coefficients and c_d. With these notation in hand, we have:

Proposition 2 (Expression of β^f). *Under the assumptions of Theorem 1, we have $c_d > 0$ and, for any $1 \leq j \leq d$,*

$$\beta_j^f = c_d^{-1}\Big[\sigma_1 \mathbb{E}[\pi f(x)] + \sigma_2 \mathbb{E}[\pi z_j f(x)] + \sigma_3 \sum_{\substack{k=1 \\ k \neq j}}^{d} \mathbb{E}[\pi z_k f(x)]\Big] \,.$$

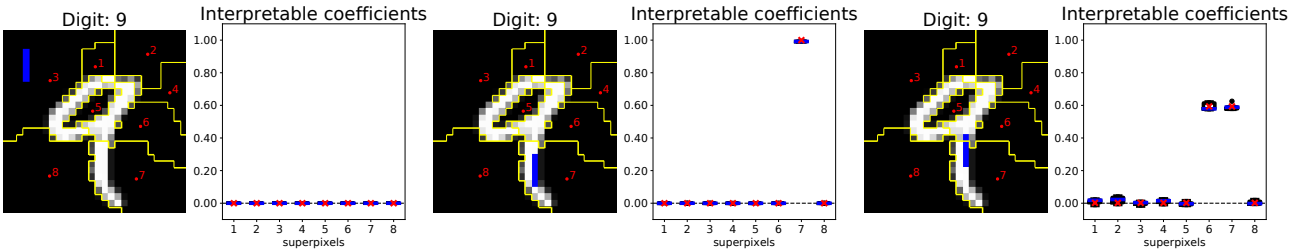

Figure 3. In this figure, we show how the theoretical predictions of Proposition 3 compare to practice. We considered a digit from the MNIST dataset (LeCun et al., 1998). The function to explain takes value 1 if all pixels marked in blue have value greater than $\tau = 0.5$, 0 otherwise. In each case, we ran LIME with $n = 1000$ examples, default regularization $\lambda = 1$ and zero replacement. We repeated the experiment five times, which gave the boxplot corresponding to the empirical values of the interpretable coefficients for each superpixel. The red crosses correspond to the predictions given by Proposition 3. We see that when the shape is split among p superpixels, each one receives a coefficient approximately equal to $1/2^{p-1}$.

We provide a detailed proof of Proposition 2 as well as the expression of the intercept β_0^f in the appendix. Let us note that Proposition 2 is quite similar to Eq. (6) in Garreau and von Luxburg (2020b) and Eq. (9) in Mardaoui and Garreau (2021). We will see that many properties of β^f are similar to the tabular and text case. Let us now present some immediate consequences of Proposition 2.

Linearity of explanations. As in the tabular and the text setting, the mapping $f \mapsto \beta^f$ is **linear**. Thus for any model that can be decomposed as a sum, the explanations provided by LIME are **the sum of the explanations of individual models**. This is true up to noise coming from the sampling (quantified by Theorem 1) and a small error due to the regularization, which is not taken into account in Theorem 1.

Large bandwidth. Because of the weights π and their complex dependency in the bandwidth ν, it can be difficult to make sense of Proposition 2 in the general case. It is somewhat easier when $\nu \to +\infty$. Indeed, we show in the appendix that $c_d \to 1/4$, $\sigma_1 \to -1/2$, $\sigma_2 \to 1$, and $\sigma_3 \to 0$. Moreover, $\pi \to 1$ almost surely. Therefore, by dominated convergence, the expression of β^f simplifies to

$$\left(\beta_\infty^f\right)_j = 2\left(\mathbb{E}\left[f(x)|z_j = 1\right] - \mathbb{E}\left[f(x)\right]\right) , \qquad (6)$$

for any $1 \leq j \leq d$. In other words, the explanation provided by LIME is proportional to the difference between the mean value of the model conditioned to superpixel j being activated and the mean value of the model for x sampled as explained previously. It seems that Eq. (6) encompasses a desirable trait of LIME for images: when the bandwidth is large, the interpretable coefficient for superpixel j takes large positive values if **in the vicinity of ξ, the model takes significantly larger values when this superpixel is present in the image**. Of course, presence or absence of a given superpixel depends on the replacement scheme. Eq. (6) hints that the explanation for superpixel j could be near zero if $\overline{\xi}$ is close to ξ on J_j, whereas J_j is actually important for the prediction.

4. Expression of β^f for Simple Models

In this section, we get meaningful insights on the explanations provided by LIME for simple models.

4.1. Shape Detectors

We start with a very simple model: a *fixed shape detector* for gray scale images. To this extent, let $\mathcal{S} := \{u_1, \ldots, u_q\}$ be a set of q distinct pixels indices, the *shape*. Let $\tau \in (0, 1)$ be a positive threshold. We define the associated shape detector

$$\forall x \in [0,1]^D, \quad f(x) := \prod_{u \in \mathcal{S}} \mathbf{1}_{x_u > \tau} . \qquad (7)$$

Readily, $f(x)$ takes the value 1 if the pixels of shape $\mathcal{S}$ are lit up, and 0 otherwise.

It is possible to compute β^f in closed-form in this case. In fact, the result does not depend on the exact shape to be detected, but rather on how it intersect the LIME superpixels. Let us define

$$E := \{j \in \{1, \ldots, d\} \text{ s.t. } J_j \cap \mathcal{S} \neq \emptyset\} ,$$

the set of superpixels intersecting the shape $\mathcal{S}$. We separate this set in two parts,

$$E_+ := \{j \in E \text{ s.t. } \overline{\xi}_j > \tau\}, \text{ and } E_- := \{j \in E \text{ s.t. } \overline{\xi}_j \leq \tau\}.$$

Intuitively, E_+ (resp. E_-) contains superpixels that are brighter than τ on average (resp. darker). We also need to define

$$\mathcal{S}_+ := \{u \in \mathcal{S} \text{ s.t. } \xi_u > \tau\}, \text{ and } \mathcal{S}_- := \{u \in \mathcal{S} \text{ s.t. } \xi_u \leq \tau\}.$$

Namely, $\mathcal{S}_+$ (resp. $\mathcal{S}_-$) contains pixels in the shape that have a value greater (resp. smaller) than the threshold.

We now make the following assumption:

$$\forall j \in E_+, \quad J_j \cap \mathcal{S}_- = \emptyset . \qquad (8)$$

Intuitively, Eq. (8) means that there is no superpixel intersecting the shape such that the average value of the superpixel activates our detector without the detector being activated. This could happen for instance if the shape intersection with the superpixels is very dark but pixels around are much brighter. It is a reasonable assumption since superpixels are quite homogeneous in color and shape. Moreover, in the case of grayscale images with zero replacement, Eq. (8) is always satisfied since $\tau > 0$. Nevertheless, we provide a result that does not rely on Eq. (8) in the appendix, which specializes into:

Proposition 3 (Computation of β^f, shape detector). *Let f be defined as in Eq. (7). Assume that Eq. (8) holds and let $p := |E_-|$. Then, if there exists $j \in E_-$ such that $J_j \cap \mathcal{S}_- \neq \emptyset$, $\beta^f = 0$. Otherwise, for any $j \in E_-$,*

$$\beta^f_j = c_d^{-1} \{\sigma_1\alpha_p + \sigma_2\alpha_p + (p-1)\sigma_3\alpha_p + (d-p)\sigma_3\alpha_{p+1}\}$$

and for any $j \in \{1, \dots, d\} \setminus E_-$,

$$\beta^f_j = c_d^{-1}\{\sigma_1\alpha_p + \sigma_2\alpha_{p+1} + p\sigma_3\alpha_p + (d-p-1)\sigma_3\alpha_{p+1}\}$$

The accuracy of Proposition 3 is demonstrated in Figure 3. Note that we use grayscale images for clarity (it is easier to define *brightness* in this case), but Proposition 3 could be adapted for RGB images.

Note that Proposition 3 is reminiscent of Proposition 3 in Mardaoui and Garreau (2021). This is not a surprise, since both these results study LIME for models with analogous structures. However, the result differ since one has to consider the intersections of the superpixels in the present case. We now make two remarks, still focusing on grayscale images with zero replacement for simplicity: in that case, E_+ is always empty since τ is positive.

Shape included in a single superpixel. In the simple case where the shape is detected and is totally included in superpixel k, then it is straightforward from the definitions of the α coefficients and c_d that $\beta^f_k = 1$ and $\beta^f_j = 0$ otherwise. This is a good property of LIME for images since superpixel k is the only relevant superpixel in this particular situation (see Figure 3).

Shape included in several superpixels. The situation is slightly more complicated when $\mathcal{S}$ intersects several superpixels (that is, $p > 1$). Reading Proposition 3, we see that when ν is large, the coefficients for an intersecting superpixel is approximately $1/2^{p-1}$ and 0 otherwise (see Lemma 1 in appendix for a more precise statement). That is, the importance given by LIME is evenly divided between superpixels that contain part of the shape. Again, this makes sense since they are the only relevant superpixels in this case (see Figure 3). For instance, if only two superpixels are involved, then the weight is roughly halved. We can note that this halving occurs even if a very small portion of the shape falls into one of the two superpixels.

4.2. Linear Model

We now turn to *linear models*. That is, f is given by

$$\forall x \in [0,1]^D, \quad f(x) = \sum_{u=1}^{D} \lambda_u x_u \,, \tag{9}$$

with $\lambda_1, \dots, \lambda_D \in \mathbb{R}$ arbitrary coefficients. Note that we can adapt this definition if there is more than one color channel, by considering $3 \times D$ coefficients.

Let us compute β^f when f is linear. We show in the appendix the following:

Proposition 4 (Computation of β^f, linear case). *Assume that f is defined as in Eq. (9). Then, for any $1 \leq j \leq d$,*

$$\beta^f_j = \sum_{u \in J_j} \lambda_u \cdot (\xi_u - \overline{\xi}_u) \,.$$

When $\overline{\xi} = 0$, the coefficients take a very simple expression. Namely, the interpretable coefficient associated to superpixel J_j is **the sum of the coefficients of f multiplied by the pixel values on the superpixel**. If another replacement is chosen, then the *normalized* values of the pixel is taken into account in this product. This seems to make a lot of sense: let us say that the coefficients of f take large positive values on superpixel j. Then LIME will give a high interpretable coefficient to this superpixel, unless the pixel values are small (or very close to the replacement color if another replacement is chosen). This is particularly visible in Figure 4: the λ coefficients take high values in the lower right corner (left panel). But since the 5th superpixel contains only 0-values pixels (middle panel), the interpretable coefficient given by LIME for this superpixel cancels out (right panel).

It is also interesting to see that there is no bandwidth dependency in Proposition 4. In a sense, this is to be expected since LIME is doing averages that are scale invariant if the function to explain is linear.

Proposition 4 is similar in spirit to Eq. (12) in Mardaoui and Garreau (2021), where the interpretable coefficients provided by LIME for a linear model where also found to be (approximately) equal to the product of the coefficients of the linear model and the feature value.

5. Approximated Explanations

If f is regular enough, it can be written as in Eq. (9) in the vicinity of ξ. If this is the case, an interesting question in light of the results of the previous section is the following: are the explanations given by Proposition 4 close to the LIME explanations? To put it plainly, can we approximately recover the LIME coefficients by summing the partial derivatives of f at ξ over the superpixels? We will

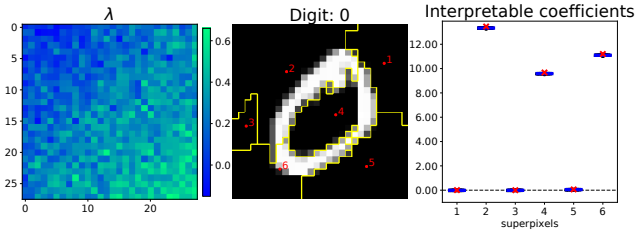

Figure 4. In this figure, we show how the theoretical predictions of Proposition 4 fare in practice. We consider a digit from the MNIST dataset. The function to explain is linear, with $\lambda_{i,j}$ proportional to $i + j$ with added white noise (heatmap in the *left panel*). We ran LIME 5 times with zero replacement and default parameters, the superpixels used are displayed in the *middle panel*. We see that our predictions match perfectly. As predicted, J_5 receives a coefficient equal to zero whereas f has high coefficients in this area, since the pixel values are equal to zero in this superpixel.

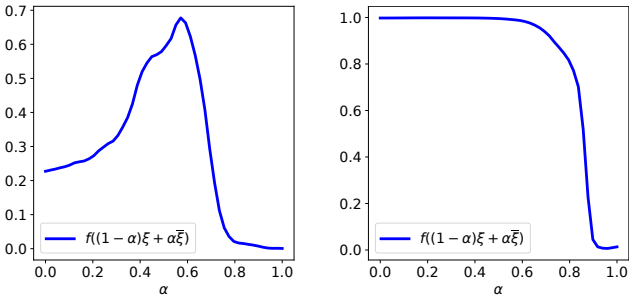

Figure 5. Predictions given by the InceptionV3 network (Szegedy et al., 2016) on a linear path between ξ ($\alpha = 0$) and $\bar{\xi}$ ($\alpha = 1$). *Left panel:* we see how the predicted values can vary along the path and why only considering the gradient at ξ or $\bar{\xi}$ may not be a good idea to build a linear approximation. *Right panel:* we see how the gradient can saturate when the network is very confident in the prediction.

see that the answer to this question is yes, with one caveat: simply taking the gradient does not always yield a satisfying linear approximation for complicated functions. We discuss linear approximations of an arbitrary function in Section 5.1 before investigating empirically in Section 5.2.

5.1. Linear Approximation

The most natural linear approximation of a function is given by its Taylor expansion truncated at order one. Since we want to approximate $f(x)$, where x is somewhere between ξ and $\bar{\xi}$, we could write, for instance, that $f(x) \approx f(\xi) + \nabla f(\xi)^\top (x - \xi)$. There are two main objections in doing so in the present case. First, we do not expect f to be linear between ξ and $\bar{\xi}$, and taking just one gradient would lead to a poor approximation. We illustrate this behavior in Figure 5 by computing the predictions across a straight line between ξ and $\bar{\xi}$.

Second, it is a well-known phenomenon in modern architectures that the gradient of the model with respect to the input can *saturate* when the network is confident in the prediction for certain activation functions (see, for instance, Krizhevsky et al. (2012)). Since from our point of view f is a black-box model, we do not have information on the activation functions (in fact, we do not even assume that f is a neural network). Therefore gradients taken at ξ or $\bar{\xi}$ can be zero, giving us essentially no information on the behavior of f (see Figure 5).

For both these reasons, we build a linear approximation of f between ξ and $\bar{\xi}$ using the *averaged gradients on a linear path* between ξ and $\bar{\xi}$. Formally, we define

$$g_u := \int_0^1 \frac{\partial f((1-\alpha)\xi + \alpha\bar{\xi})}{\partial x_u} \, \mathrm{d}\alpha \tag{10}$$

the averaged gradient at pixel u. We approximate this integral by a Riemann sum, that is,

$$g_u^{\text{approx}} := \frac{1}{m} \sum_{k=1}^{m} \frac{\partial f((1 - \frac{k}{m})\xi + \frac{k}{m}\bar{\xi})}{\partial x_u} \, . \tag{11}$$

Subsequently, we approximate $f(x)$ by $(x - \bar{\xi})^\top g^{\text{approx}} + f(\bar{\xi})$. Applying Proposition 4 to this approximation we obtain the approximate explanations

$$\forall 1 \leq j \leq d, \quad \beta_j^{\text{approx}} = \sum_{u \in J_j} (\xi_u - \bar{\xi}_u) \cdot g_u^{\text{approx}} \, . \tag{12}$$

Inside the sum, we recognize the definition of *integrated gradients* between ξ and $\bar{\xi}$ (Sundararajan et al., 2017), another interpretability method. Eq. (12) therefore corresponds to **the sum of integrated gradients over superpixel** j.

5.2. Experiments

In this section, we show experimentally that LIME explanations are similar to the approximated explanations derived in the previous section. The code of all experiments is available at `https://github.com/dgarreau/image_lime_theory`

Setting. We first considered images from the CIFAR10 dataset (Krizhevsky et al., 2009), that is, 32×32 RGB images belonging to ten categories. For a subset of 1000 images of the test set, we computed the explanations given by LIME with default settings, with the exception of the kernel size used by the quickshift algorithm which we decreased to 1 to get wider superpixels. We compared these explanations with the approximated explanations of Section 5 for four different models. First, we started with a very simple one-hidden-layer neural network, trained to 35% accuracy. We then moved to VGG-like architectures (Simonyan and Zisserman, 2014), progressively increasing the number of

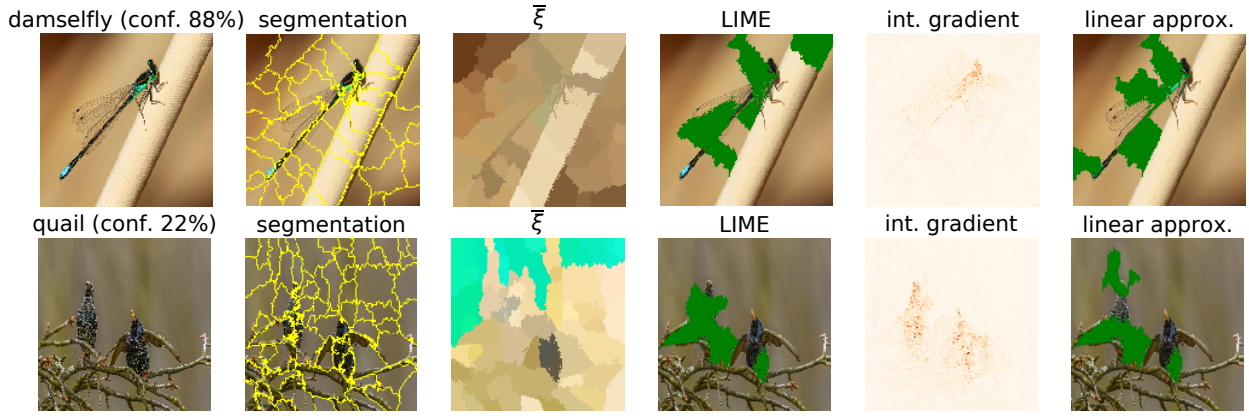

Figure 6. Comparing the explanations given by LIME *vs* approximate explanations obtained by summing the integrated gradient over the LIME superpixels. Here we explain the top predicted class for images of the ILSVRC2017 test data with the InceptionV3 network. In both cases, we showcase the top five positive coefficients. Qualitatively, the explanations obtained are quite similar, identifying close superpixels when they are not matching exactly.

blocks in the model (from 1 to 3). For each model, we considered the function corresponding to the most likely class for ξ. We then collected the indices of the superpixels associated to the top five and top ten positive average coefficients. For the sum of integrated gradients, we considered $m = 20$ steps in Eq. (11) as in Sundararajan et al. (2017). The results are presented in Table 1.

We then moved to more realistic images coming from the test set of the 2017 large scale visual recognition challenge (LSVRC, Russakovsky et al., 2015). We used three pretrained models from the Keras framework: MobileNetV2 (Sandler et al., 2018), DenseNet121 (Huang et al., 2017), and InceptionV3 (Szegedy et al., 2016) with default input shape $(299, 299, 3)$. Again, we compared the LIME default explanations to the approximated explanations for 1000 of these images. Qualitative results are presented in Figure 6 for the InceptionV3 network, while Table 2 contains the quantitative results.

Metric. We compared the list of the top 5 and 10 positive coefficients *via* the *Jaccard sindex* (also know as Jaccard similarity coefficient), that is, the size of the intersection divided by the size of the union of the two lists. Hence a Jaccard index of 1 means perfect match between the identified superpixels whereas a Jaccard index of 0 complete disagreement. Note that the Jaccard similarity between a fixed set of size 5 (resp. 10) and a random subset of $\{1, \dots, 60\}$ is equal to 0.05 (resp. 0.06). Here, 60 is the observed average number of superpixels produced by the quickshift algorithms for the images at hand.

Results. Without being a perfect match, we observe a **substantial overlap between the LIME explanations and the approximated explanations** for all the models and datasets that we tried. This is particularly striking for simple

Table 1. Comparison between LIME and approximated explanations for CIFAR-10. For each model, we report JX, the Jaccard similarity between the top X positive coefficients.

Model	# param.	# layers	acc.	J5	J10
1-layer	330K	1	0.35	0.99	0.99
VGG1	1M	4	0.67	0.81	0.85
VGG2	600K	8	0.69	0.75	0.81
VGG3	550K	12	0.70	0.71	0.76

models. More precisely, the Jaccard similarities observed are several times higher than what a random guess would produce. This is surprising since we are considering a linear approximation of highly non-linear functions. As a matter of fact, the exact values of the interpretable coefficients are quite different. Nevertheless, they are sufficiently close so that the sets of superpixels identified by both methods are consistently overlapping.

We notice that this link seems to weaken when the models become too complex, while still a third of identified superpixels are common for InceptionV3. However, visual inspection reveals that the superpixels identified by both methods remain close from each other even when they are distinct (see Figure 6 and additional qualitative results in appendix).

We want to emphasize that, if the model is not smooth, the link between approximate explanations in the sense of Eq. (12) and LIME does not exist anymore. For instance, a random forest model based on CART trees has gradient equal to zero everywhere. Therefore, the integrated gradient is also zero, and $\beta_j^{\text{approx}} = 0$ for any j. We also want to point out that we did not evaluate β^{approx} as as an interpretability method. In particular, it could be the case that the associated

Table 2. Comparison between LIME and approximated explanations for LSVRC images.

Model	# param.	# layers	acc.	J5	J10
MobileNetV2	3.5M	88	0.90	0.43	0.54
DenseNet121	8.0M	121	0.92	0.42	0.44
InceptionV3	23.9M	159	0.94	0.35	0.36

explanations are of a lesser quality than LIME's.

Computation time. Setting aside the segmentation step, each run of LIME requires $n = 1000$ queries of the model, whereas the averaged gradient estimation requires $m = 20$ queries. In the favorable scenario where getting a gradient is as costly as a model query, computing the approximated explanations is much faster than LIME.

6. Conclusion

In this paper, we proposed the first theoretical analysis of LIME for images. We showed that the explanations provided make sense for elementary shape detectors and linear models. As a consequence of this analysis, we discovered that for smooth models the interpretable coefficients of LIME for images resemble to the sum of integrated gradients over the LIME superpixels.

As future work, we plan on tackling more complex models. A starting point is the study of polynomial functions: obtaining a statement analogous to Proposition 4 would open the door to more precise expression for the limit explanation depending on the higher derivatives of f.

Acknowledgments

This work was partly funded by the UCA DEP grant. The authors want to thank Ulrike von Luxburg for her insights in the writing phase of the paper.

References

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

D. Garreau and U. von Luxburg. Explaining the Explainer: A First Theoretical Analysis of LIME. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020a.

D. Garreau and U. von Luxburg. Looking Deeper into Tabular LIME. *arXiv preprint arXiv:2008.11092*, 2020b.

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

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

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.

P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis. Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy*, 23(1):18, 2021.

D. Mardaoui and D. Garreau. An Analysis of LIME for Text Data. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

N. Narodytska, A. Shrotri, K. S. Meel, A. Ignatiev, and J. Marques-Silva. Assessing heuristic machine learning explanations with model counting. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 267–278, 2019.

M. T. Ribeiro, S. Singh, and C. Guestrin. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018.

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2020.

M. Sundararajan, A. Taly, and Q. Yan. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *European Conference on Computer Vision*, pages 705–718, 2008.