Supplementary Material for "SSplain: Sparse and Smooth Explainer for Retinopathy of Prematurity Classification"

A. Evaluation metrics

We define three different metrics to evaluate SSplain on the ROP dataset, in addition to the post-hoc balance accuracy metric. "Connected Components" evaluates smoothness. The "Curvature" and "Dilation" metrics assess whether the explainer detects tortuous and dilated regions, which are features that clinicians find important [11].

A.1. Connected components

With this metric, we aim to measure the smoothness of explanation maps, focusing particularly on the smoothness of vessel structures at each insertion step s. To achieve this, we extract graphs from the image $\mathbf{X}$ and from images during the insertion process $\mathbf{X}_s$, and identify the length of the largest connected component in each graph. This length is the number of nodes in the largest connected component.

We generate undirected graphs from images by extracting horizontal and vertical edges based on adjacent pixel values. Specifically, horizontal edge pairs h_{edges} and vertical edge pairs v_{edges}, which define the start and end positions of edges, are given by:

$$\begin{aligned} h_{edges} &= \{[(j,k),(j+1,k)] | X_{j,k} \neq X_{j+1,k}\}, \\ v_{edges} &= \{[(j,k),(j,k+1)] | X_{j,k} \neq X_{j,k+1}\}. \end{aligned} \tag{7}$$

We create the graph using the NetworkX library [16] from these edge pairs. To identify the connected components, i.e., the sets of nodes that are connected to each other in the graph, we use the "connected_components" function from NetworkX. This gives us a list of all connected components. We then calculate the length of the largest connected component, which is the number of nodes in that component.

Finally, we compute the ratio between the length of the largest connected components from $\mathbf{X}_s$ and $\mathbf{X}$. A higher value of the largest connected component at s indicates that the inserted pixels create smooth vessel structures.

A.2. Curvature

Due to the link between retinal vessel tortuosity and ROP development, we evaluate the explainer's ability to capture tortuous regions at each insertion step s. We compute pixel-wise curvatures $\mathbf{C} \in \mathbb{R}_{+}^{h \times w}$ for $\mathbf{X}$ using the Tubular Curvature Filter Method [37], where higher pixel-wise curvature values indicate greater tortuosity. Sunger *et al*. [37] calculates pixel-wise curvature values by measuring the directional rate of change in the eigenvectors. For more details on the curvature calculation, please refer to the work by Sunger *et al*. [37].

Then, $\cos(\mathbf{X}_s, \mathbf{C}) = \frac{\mathbf{X}_s \cdot \mathbf{C}}{\|\mathbf{X}_s\| \|\mathbf{C}\|}$ computes the similarity. A higher value at s shows that the inserted pixels capture tortuous regions, and doing this at an earlier s indicates that the explainer assigns greater importance to these regions.

A.3. Dilation

We utilize the Average Segment Diameter (ASD) feature proposed by Ataer *et al*. [3] to assess how well the explainer captures dilation features at each insertion step s. ASD computes the total number of pixels in a vessel branch divided by the curve length. Specifically,

$$\text{ASD}(x_{\text{branch}}) = \frac{N(x_{\text{branch}})}{L_c(x_{\text{branch}})}\,, \tag{8}$$

where $N(x_{\text{branch}})$ is the pixel count on the vessel segment x_{branch}, and $L_c(x_{\text{branch}})$ is the length of x_{branch} given in Eq. (9).

$$L_c(x_{\text{branch}}) = \int_a^b \|\mathbf{c}'(t)\|\,\mathrm{d}t\,, \tag{9}$$

parameterized by $\mathbf{c}(t) : [a,b] \subset \mathbb{R} \mapsto \mathbb{R}^2$. We follow the tracing process defined by Ataer *et al*. [3] and Tian *et al*. [41] to extract vessel branches x_{branch}. Their tracing algorithm consists of two steps. First, they find sample points on vessel center-lines using a principal curve-based method. Then, they apply a minimum spanning tree algorithm to obtain tree structures and extract vessel branches.

Finally, we evaluate $\cos(\mathbf{D_s}, \mathbf{D})$, where $\mathbf{D} \in \mathbb{R}^{b \times 1}$ contains the ASD features from the original image $\mathbf{X}$ and $\mathbf{D}_s$ contains the features from the images during the insertion process $\mathbf{X}_s$. Here, b is the number of vessel branches.

We extract x_{branch} from the original image $\mathbf{X}$ to calculate both $\mathbf{D}$ and $\mathbf{D}_s$, so only the number of pixels on each branch changes with changing $\mathbf{X}_s$. Similar to the curvature metric, a higher value at an earlier s indicates that the explainer assigns greater importance to dilated vessel regions.

B. Ablation study

The problem formulation in Eq.(1) from Section 3.1 includes a loss function $l(\cdot)$ (we use the cross-entropy function), a total variation function $\nu(\cdot)$, and S_1 and S_2 constraints with the support S_0 on pixels of vessels. Here, we present an ablation study on the ROP dataset to investigate the effects of sparsity, using S_1 with the ℓ_0 norm and the total variation function, while always including the cross-entropy loss and the S_2 constraint (having mask values within $[0,1]$). Figure 6 shows that using this formulation improves performance, resulting in lower deletion accuracy, higher insertion accuracy and higher max connected component ratio.

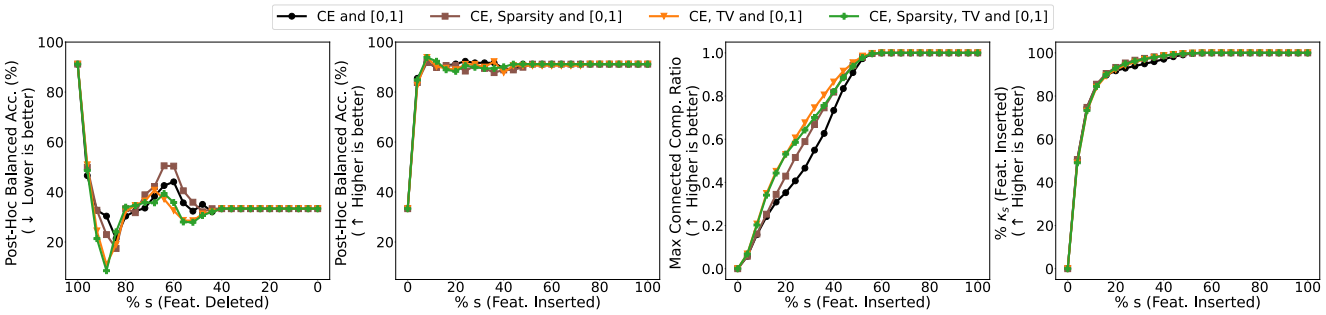


Figure 6. Ablation study for sparsity (using ℓ_0 constraint) and total variation (tv) constraints on the ROP dataset. From left to right, we report the average of: Post-hoc balanced accuracy with deletion (lower is better) and insertion (higher is better) of pixels with the highest attribution scores, connected components ratio during the insertion process, and sparsity s versus normalized sparsity κ_s during the insertion process. Note that S_0 and S_2 constraints are always applied. Using all components in the problem formulation: cross-entropy loss, sparsity constrain S_1, total variation loss, and the $[0, 1]$ (S_2) constraint together improves performance on all metrics.

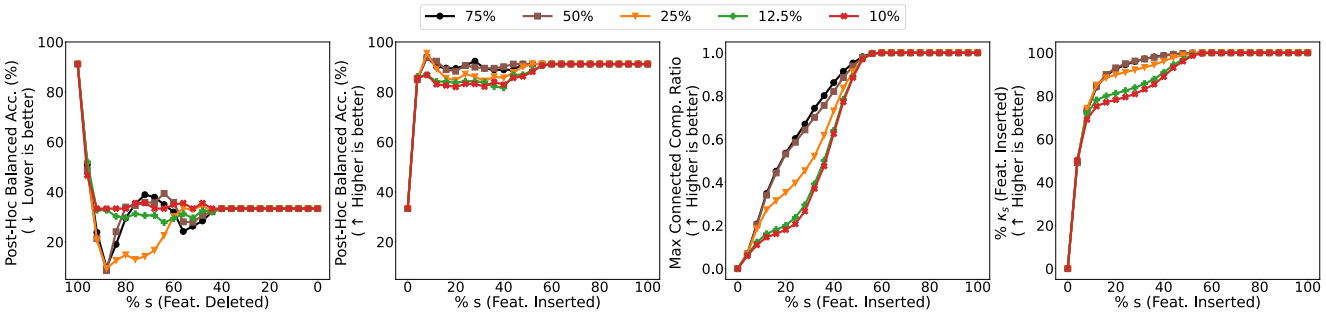


Figure 7. Comparison of different α for SSplain (S_1 with ℓ_0 constraint): $75\%, 50\%, 25\%, 12.5\%$ and 10% of $\|\mathbf{X}\|_0$ on the ROP dataset. From left to right, we report the average of: Post-hoc balanced accuracy with deletion (lower is better) and insertion (higher is better) of pixels with the highest attribution scores, connected components ratio during the insertion process, and sparsity s versus normalized sparsity κ_s during the insertion process. Making the sparsity constraint stricter (α smaller than 50% of $\|\mathbf{X}\|_0$) results in performance loss.

C. Sparsity area limit α for SSplain

In Eq.(1), S_1 enforces the sparsity constraint, which is implemented using either the ℓ_0 or ℓ_1 norm. In Figure 7, we use SSplain with the ℓ_0 norm on the ROP dataset to compare different sparsity levels (upper bounds for the number of non-zero mask values): $75\%, 50\%, 25\%, 12.5\%$ and 10% of $\|\mathbf{X}\|_0$. We show that making this constraint stricter (by decreasing α in S_1) results in performance loss, i.e., higher deletion accuracy, lower insertion accuracy, a lower rate of connected components, and lower κ_s.

D. Competing method details

We compare SSplain with seven state-of-the-art explainers using the following settings in both our post-hoc performance analyses and visualizations on the ROP dataset:

- **Gradient-based:**
  1. *Saliency [35]:* The Saliency method is defined as the absolute value of the partial derivative of the y-th output neuron with respect to the input image, where y is the target label.
  2. *Guided Grad-CAM [32]:* It calculates the positive gradients of the target class with respect to the feature maps of a specific convolutional layer. One can apply the Guided Grad-CAM method to any layer. We use the output of the last Inception layer ("inception 5b").
  3. *Integrated Gradients [36]:* It calculates the integral of the gradients along the path from a baseline input, such as a black image, to the input image.
  4. *DeepSHAP [27]:* DeepSHAP uses Shapley values from parts of the model to compute Shapley values for the whole model by propagating DeepLIFT [34]'s multipliers.
- **Perturbation-based:**
  1. *KernelSHAP [27]:* It uses the LIME method to calculate Shapley values, which evaluates each feature's or input pixel's contribution to the prediction by considering all possible combinations, i.e., analyzes how the prediction changes when features are added to combinations. We use 200 perturbation samples.
  2. *LIME [30]:* It perturbs the image and trains a simple linear model with the given target label t to learn fea-

ture importances using the perturbed images and their corresponding model predictions. We use 200 perturbation samples for this process.

3. *Occlusion [52]:* The Occlusion method measures feature importance by analyzing changes in the model's predictions when applying binary masks to image regions. The method uses a patch-wise mask that slides over the image. We use a patch size of 16×16 with 4×4 strides.

- **Incorporating sparsity or smoothness or both:**
  1. *Input×Gradient [33]:* It calculates attribution scores by element-wise multiplication of the input with the gradients (Saliency scores) with respect to the target label y.
  2. *Extremal Perturbation [13]:* It defines a ranking-based area loss and uses a Gaussian kernel to incorporate smoothness to learn explanation maps by perturbing the input image. This perturbation involves either blurring or replacing pixel values with a constant value. The loss function includes preservation-based, deletion-based, or hybrid loss functions. Also, this method needs to start from an initial mask, which is initialized with all values set to one. For the ROP dataset, we use a Gaussian blur as the perturbation method, a "hybrid" loss with 400 iterations, 0.05 smoothing factor, and a target area to 50% of $\|\mathbf{X}\|_0$.

Even though the ROP images are in grayscale, the model accepts images with three channels, and some methods extract attribution for each channel. For visualization, we calculate the maximum of the channel-wise values for Saliency, Guided Grad-CAM, Integrated Gradients, Input×Gradient, LIME, DeepSHAP and KernelSHAP. For evaluating connected components, curvature, and dilation, we report the channel-wise mean of these methods.

We use two different libraries to apply the competing methods to the ROP dataset. We use the Captum [21] library for all of the competitor methods except the Extremal Perturbation method. For the Extremal Perturbation method, we use TorchRay [13].

E. Additional Dataset and Model Details

In this section, we apply SSplain to the MNIST [22] and FMNIST [46] datasets. Similar to the ROP dataset, the background of the images consists of black pixels (zero-valued pixels), and we can define support S_0 as in Section 3.1 using the non-zero pixels. To obtain the trained black-box model for both of these datasets, we train the LeNet-5 [23] using the Adam optimizer [20] with a learning rate of 0.001, a weight decay of 0.0001, over 50 epochs, and a batch size of 32, with early stopping.

We compare SSplain with nine explainers. *Gradient-based:* Saliency [35], Guided Grad-CAM [32], Integrated Gradients [36] and DeepSHAP [27]. We apply Guided Grad-CAM to the last CNN layer. *Perturbation-based:* LIME [30], KernelSHAP [27] and Occlusion [52]. KernelSHAP and LIME employ 200 input samples. Occlusion uses 1×1 windows. *Incorporating sparsity or smoothness or both:* Input×Gradient [33] and Extremal Perturbation [13]. Extremal Perturbation uses a Gaussian blur, a "hybrid" loss with 200 iterations, 0.0001 smoothing factor, and a target area to 25% of $\|\mathbf{X}\|_0$ with having pixel-wise masks. It initializes mask values to be all ones.

To evaluate the performance of explainers, we use post-hoc accuracy for the insertion and deletion processes, and we also evaluate sparsity s and normalized sparsity κ_s during the insertion process. We do not use the domain-specific metrics for ROP, such as curvature and dilation.

E.1. MNIST

MNIST dataset [22] has 70,000 28×28 greyscale images (split into 60,000 for training and 10,000 for testing), each corresponding to one of the digits from 0 to 9. The trained model has 98.53% accuracy on the test set. We perform our post-hoc explainability analysis on 500 sample images from the test set. For SSplain, we initialize masks with values proportional to the corresponding pixel intensity between 0 and 1, as follows: $\mathbf{M} = \frac{\mathbf{X}}{|\mathbf{X}|_\infty}$. We use the ℓ_0 norm with α in S_1 set to 25% of $\|\mathbf{X}\|_0$. We use the cross-entropy loss function for $l(\cdot, \cdot)$ and run 20 iterations of ADMM for each image. We utilize the Adam optimizer [20] to solve problem (4a) with a learning rate of 0.1, $\rho = 0.01$ and $\lambda = 10^{-3}$.

E.2. FMNIST

FMNIST dataset [46] consists of 70,000 28×28 greyscale images (split into 60,000 for training and 10,000 for testing), each corresponding to fashion products from 10 different classes. The trained model achieves 88.67% accuracy on the test set. We conduct our post-hoc explainability analysis on 500 sample images from the test set. For SSplain, we initialize masks with values proportional to the corresponding pixel intensity between 0 and 1, as follows: $\mathbf{M} = \frac{\mathbf{X}}{|\mathbf{X}|_\infty}$. We use the ℓ_0 norm with α in S_1 set to 25% of $\|\mathbf{X}\|_0$, the cross-entropy loss function for $l(\cdot, \cdot)$ and run 20 iterations of ADMM for each image. We use the Adam optimizer [20] to solve problem (4a) with a learning rate of 0.1, $\rho = 0.01$ and $\lambda = 10^{-4}$.

F. Visualization

F.1. ROP

Figure 8 shows the explanation maps from SSplain (with ℓ_0) and comparative methods (Saliency, Input×Gradient, Guided Grad-CAM, Occlusion, and Extremal Perturbations) for sample images from the ROP dataset. We can see that SSplain gives more importance to tortuous and dilated vessel regions, while other methods fail to capture these re-

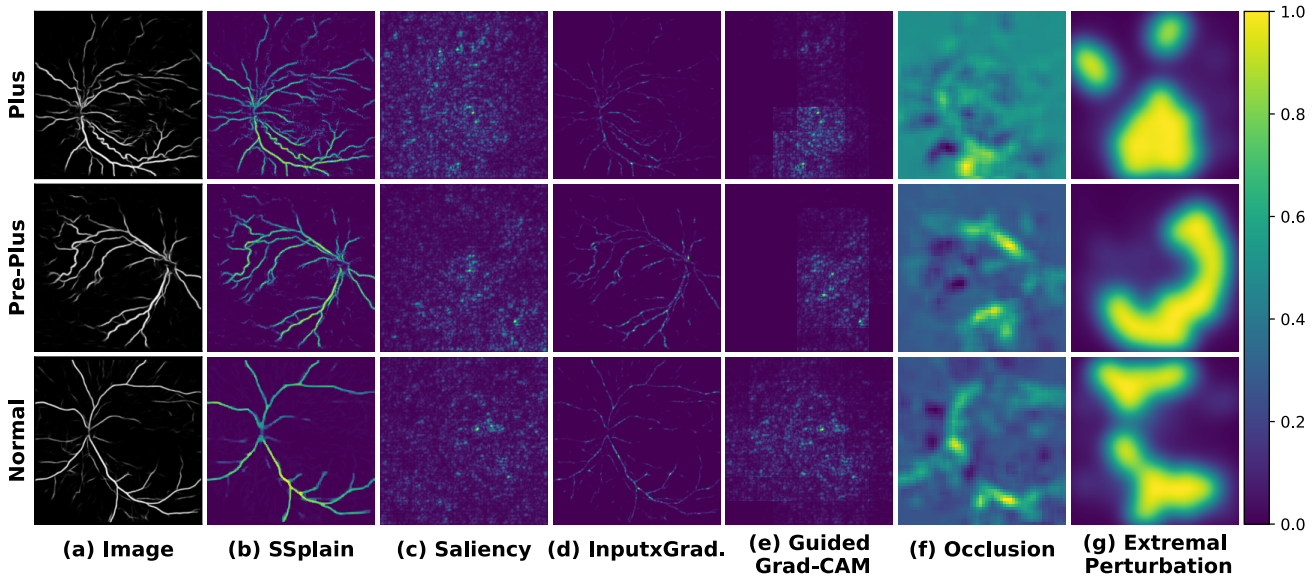


Figure 8. Comparison with other explainability methods using fundus images from the ROP dataset by visualizing the attribution scores. From left to right: (a) images, explanation maps generated using (b) SSplain (S_1 with ℓ_0 constraint), (c) Saliency, (d) Input×Gradient, (e) Guided Grad-CAM, (f) Occlusion and (g) Extremal Perturbation. We observe that SSplain effectively preserves image structures and assigns more importance to pixels with high tortuosity and dilation while maintaining smoothness. Except Input×Gradient, other methods treat images as a whole and assign importance to pixels not associated with vessels.

gions. Except for Input×Gradient, the other methods assign importance to background pixels because they treat the images as a whole.

F.2. MNIST and FMNIST

Figure 9 shows the explanation maps from SSplain (with ℓ_0) and comparative methods (Saliency, Input×Gradient, Guided Grad-CAM, Occlusion, and Extremal Perturbations) for sample images from the MNIST and FMNIST datasets. SSplain effectively preserves sparse image structures while other methods assign importance to background pixels.

G. Sparsity constraints (ℓ_0 vs ℓ_1)

G.1. SSplain with ℓ_0 and ℓ_1 constraints

SSplain can integrate both the ℓ_0 and ℓ_1 norms in its S_1 constraint. In Section 4.3, we showed that even though both variations work well, the ℓ_0 variation performs slightly better for the deletion and insertion post-hoc accuracy analyses. Figure 10 displays the explanation maps for both SSplain-0 and SSplain-1 on the sample ROP images, showing no significant difference between them.

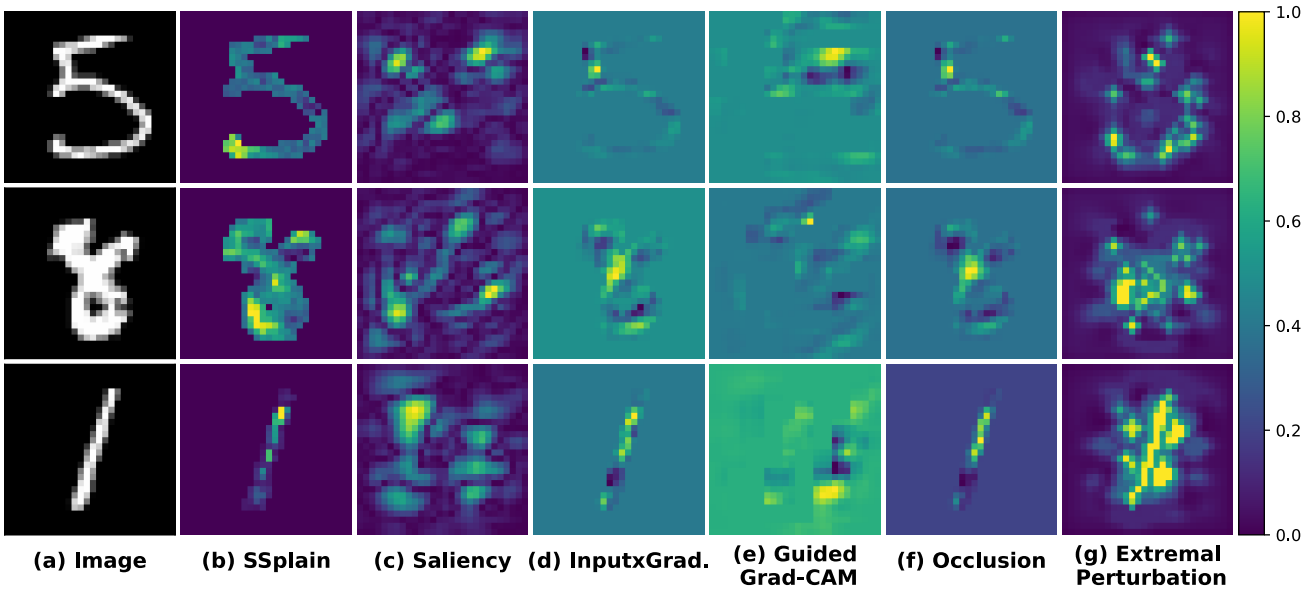
G.2. SSplain vs Extremal Perturbations

In this section, we compare SSplain with the ℓ_0 norm constraint in S_1 and Extremal Perturbations methods, which use the ℓ_1 constraint. Although SSplain has the S_0 constraint, for this experiment, we remove this constraint and assign initial mask valuesof all pixels to be ones in both methods to ensure a fair comparison. Both methods run for 400 iterations with a target area of 5% and a learning rate of 0.01. SSplain uses $\rho = 0.01$ and $\lambda = 0.0001$. The Extremal Perturbation method uses a "preservation" loss with a 0.0001 smoothing factor and employs pixel-wise masks. It perturbs the image by replacing pixel values with a constant value of zero. Figure 11 shows that SSplain still creates explanations on the vessel pixels and does not assign importance to the background. It still gives more importance to tortuous and dilated vessel regions. However, the extremal perturbation method gives importance to the background, resulting in significantly less sparse masks.

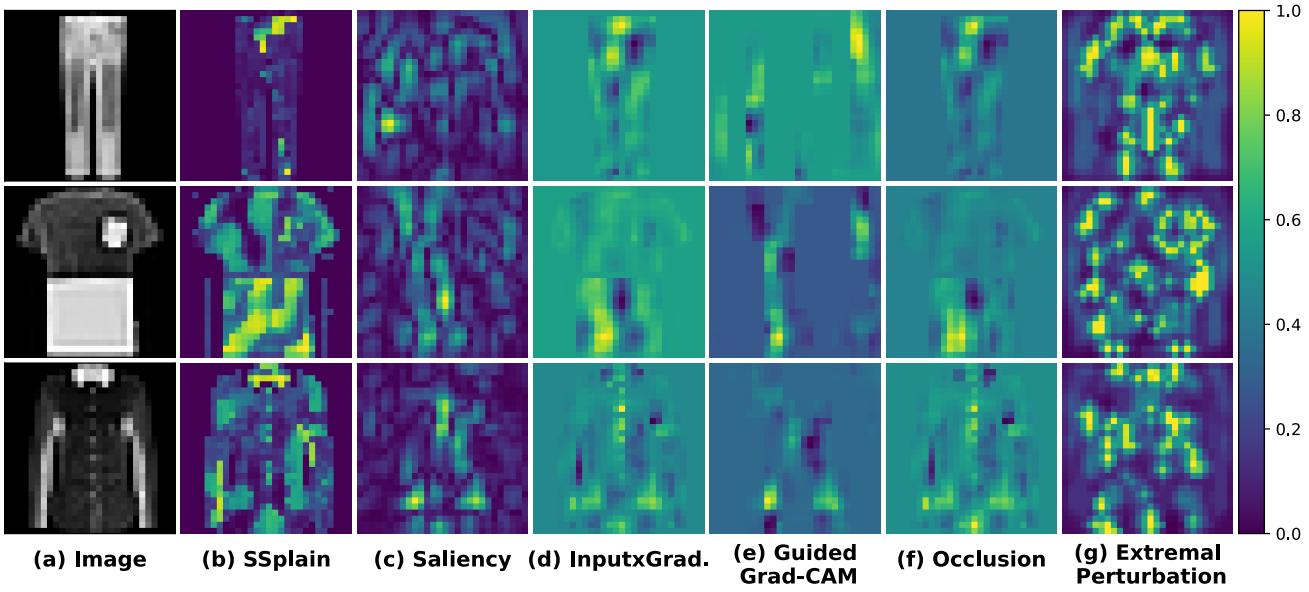
H. Execution Times

In Table 1, we report the execution times (mean ± standard deviation) for each dataset. We include only the execution times of input-perturbation-based approaches, which are KernelSHAP [27], LIME [30], Occlusion [52], and Extremal Perturbation [13], since SSplain is part of that explainability method family. The gradient-based method requires only a single backpropagation on the model and is faster than input-perturbation-based explainability methods. For the ROP dataset, execution times are averaged over 100 images, while for MNIST and FMNIST, they are averaged over 500 images.

Note that the runtime of all input perturbation methods



(a) MNIST visualizations.



(b) FMNIST visualizations.

Figure 9. Comparison of explainers on sample images from the MNIST and FMNIST datasets with attribution score visualization. From left to right: (a) images, explanation maps generated using (b) SSplain-0, (c) Saliency, (d) Input×Gradient, (e) Guided Grad-CAM, (f) Occlusion and (g) Extremal Perturbation. SSplain preserves sparse image structures while others assign importance to the background.

depends on the configurations of the explainers and the black-box model size. SSplain's runtime depends on the number of ADMM iterations, image size, and model size. Extremal Perturbations' runtime depends on the number of iterations, image size, patch size, and model size. KernelSHAP and LIME's runtime depends on the model size and the number of perturbation samples. Occlusion's runtime depends on the model size, image size, patch size, and stride. Please refer to Section 4.2 for the configurations of the explainers for the ROP dataset and Appendix E for the MNIST and FMNIST datasets.

I. Mask Initialization

We compare initializations for mask $\mathbf{M}$. In the main paper, we initialize masks with values proportional to the corre-

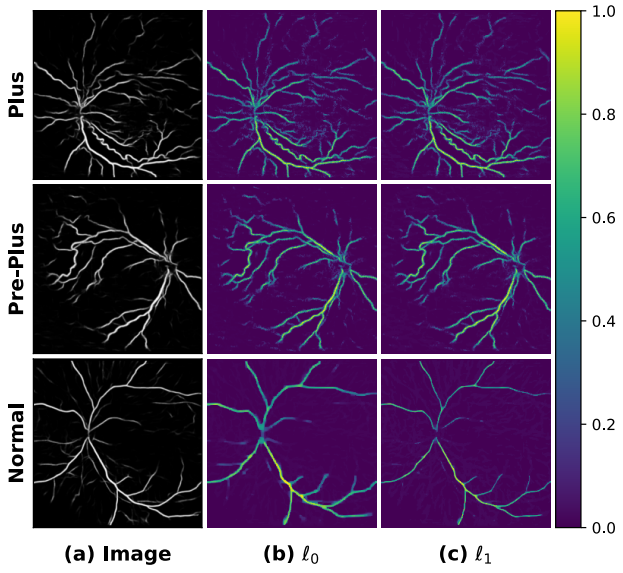


Figure 10. Comparison of the ℓ_0 and ℓ_1 norms in the sparsity constraint S_1 for the SSplain method, using fundus images from the ROP dataset to visualize the attribution scores. From left to right: (a) images, explanation maps generated using (b) SSplain with the ℓ_0 constraint, (c) SSplain with the ℓ_1 constraint. We initialize masks with values proportional to the corresponding pixel intensity between 0 and 1, as follows: $\mathbf{M} = \frac{\mathbf{X}}{|\mathbf{X}|_\infty}$. We run 50 iterations with a learning rate of 0.01, $\rho = 0.01$ and $\lambda = 10^{-5}$ using the Adam optimizer. We set the sparsity level to 50% of $\|\mathbf{X}\|_0$ for both cases. We do not observe any significant difference between these two variations in the S_1 constraint.

Table 1. Execution times (mean $\pm$ standard deviation) in terms of seconds for KernelSHAP [27], LIME [30], Occlusion [52], Extremal Perturbation [13], and SSplain on the ROP, MNIST, and FMNIST datasets. For the ROP dataset, execution times are averaged over 100 images, while for MNIST and FMNIST, they are averaged over 500 images. We report only the execution times of input-perturbation-based approaches, since SSplain belongs to this family of methods. Gradient-based methods require only a single backpropagation through the model and are generally faster than input-perturbation-based approaches. Note that these execution times depend heavily on the explainer configuration, such as the number of iterations and patch size, when applicable.

Methods	**ROP**	**MNIST**	**FMNIST**
KernelSHAP	29.57 $\pm$ 7.24	0.23 $\pm$ 0.07	0.24 $\pm$ 0.02
LIME	27.86 $\pm$ 2.36	0.21 $\pm$ 0.06	0.23 $\pm$ 0.03
Occlusion	114.60 $\pm$ 13.37	0.64 $\pm$ 0.05	0.64 $\pm$ 0.02
Extremal Perturbations	222.36 $\pm$ 11.27	15.13 $\pm$0.39	15.19 $\pm$0.49
SSplain	29.91 $\pm$ 2.13	1.58 $\pm$ 0.04	1.59 $\pm$ 0.02

sponding pixel intensity between 0 and 1, as follows: $\mathbf{M} = \frac{\mathbf{X}}{\|\mathbf{X}\|_\infty}$. The mask initialization depends on the use case, and for our purposes of optimizing a mask with sparsity and smoothness constraints, this intensity-based initialization is suitable. Alternatively, one can initialize masks as all ones assigned to the support of the image using S_0, where $M^0_{j,k} = 1$ for $M_{j,k} \in S_0$ with $S_0 = \{\mathbf{M} : \texttt{supp}(M_{jk}) \subseteq \texttt{supp}(X_{jk})\}$. $\texttt{supp}(\cdot)$ denotes the support (i.e., the non-zero pixels). We compare these two cases: intensity-based and uniform (all-ones) initialization on the ROP dataset.

Both methods use the S_0 constraint and the ℓ_0 constraint for sparsity in S_1. The intensity-based initialization case uses with 50 ADMM iterations, while the uniform initialization uses 400. This iteration difference is since the uniform case starts from a less sparse point. Both methods use $\rho = 0.01$, $\lambda = 0.00001$, a learning rate of 0.01 and a target area of 50% of α_0 in S_1 to 50% of $\|\mathbf{X}\|_0$.

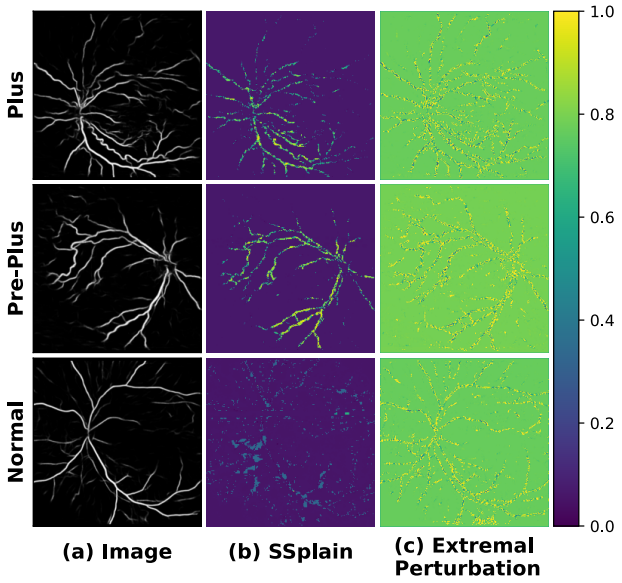


Figure 11. Comparison of SSplain-0 and Extremal Perturbation methods using fundus images from the ROP dataset by visualizing the attribution scores. From left to right: (a) images, explanation maps generated using (b) SSplain, Extremal Perturbation. Extremal Perturbation uses a "preservation" loss with 0.0001 smoothing factor, and apply pixel-wise masking. SSplain uses $\rho = 0.01$, $\lambda = 0.0001$ and does not use the S_0 constraint. Both methods use 400 iterations, a target area of 5%, a learning rate of 0.01 and initialize mask values to be all ones. SSplain captures the support of vessels, tortuous and dilated regions. It does not assign importance to the background, even without the S_0 constraint, whereas Extremal Perturbation does.

I.1. Visualizations

Here, we visually compare explainability maps from different mask initialization methods. Figure 12 shows that uniform initialization results in less sparse and less smooth explanations compared to intensity-based initialization on sample ROP images.

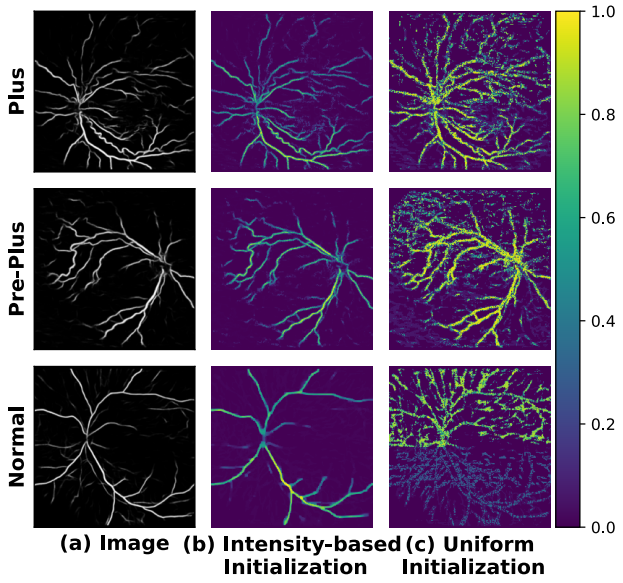


Figure 12. Comparison of SSplain-0 with intensity-based mask initialization versus uniform (all-ones) mask initialization using fundus images from the ROP dataset by visualizing the attribution scores. From left to right: (a) images, explanation maps generated using (b) SSplain-0 with intensity-based, and SSplain-0 with uniform mask initializations. The intensity-based initialization case uses with 50 ADMM iterations, while the uniform initialization uses 400. Both methods use $\rho = 0.01$, $\lambda = 0.00001$, a learning rate of 0.01 and a target area of 50% of α_0 in S_1 to 50% of $\|\mathbf{X}\|_0$. Uniform initialization results in less sparse and less smooth explanations compared to intensity-based initialization.

I.2. Post-hoc Analyses

Figure 13 shows a comparison of mask initialization on the ROP dataset. This evaluation uses post-hoc balanced accuracy, sparsity, number of connected components, and disease-related metrics of tortuosity and dilation. We observe that intensity-based mask initialization outperforms uniform mask initialization across all metrics.

J. Unsupervised Generation

Here, we follow an unsupervised approach where the target class y for generating explanations in Eq. 1 is the model's predicted class. Formally, $y = \arg\max_c f_c(\mathbf{X})$, where $f : \mathbb{R}^{h\times w} \to \mathbb{R}^c, \mathbf{X} \mapsto f(\mathbf{X})$ is the black-box model, with c the number of classes and $\mathbf{X} \in \mathbb{R}^{h\times w}$ the input image, with h and w denote the image height and width.

This section includes unsupervised explanation generations and their evaluation using accuracy-based metrics. SSplain and competing explainer setups are given in Sections 4.1 and 4.2 for the ROP dataset, and in Appendix E for the MNIST and FMNIST datasets.

J.1. Post-hoc Analyses

J.1.1. ROP Dataset

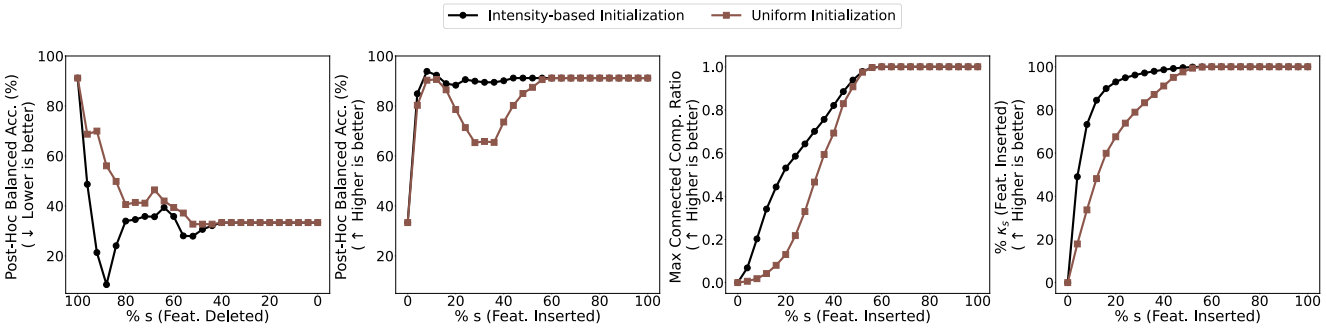
Figure 14 shows unsupervised explanation generation on the ROP dataset. Consistent with the supervised setup in Section 4.5, SSplain outperforms the competitors by achieving lower deletion, higher insertion accuracy, a higher connected components ratio, and higher normalized sparsity κ_s, while also stabilizing earlier than the other methods.

J.1.2. MNIST Dataset

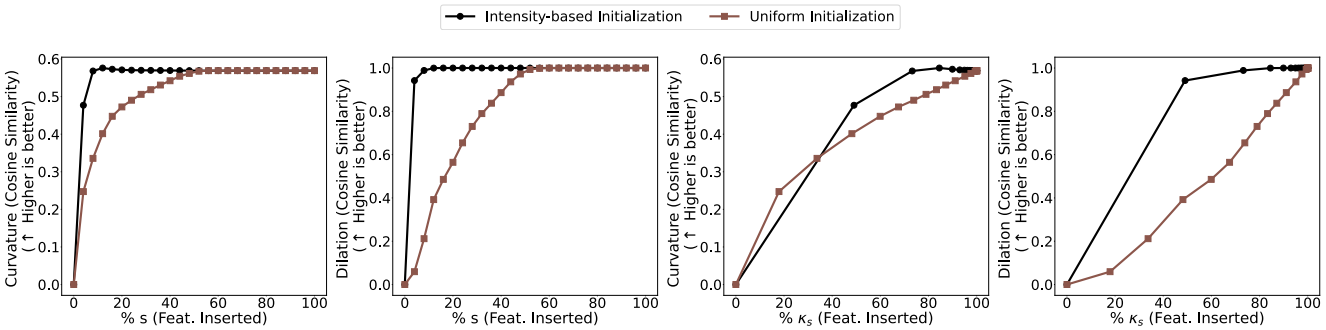
Figure 15 demonstrates unsupervised explanation generation across different analyses and metrics for the MNIST dataset. Similar to the supervised generation, SSplain-0 consistently outperforms the competing methods in all analyses, achieving lower deletion and higher insertion post-hoc accuracy, along with higher normalized sparsity κ_s. Moreover, SSplain reaches its maximum earlier than other methods and remains stable throughout the insertion process.

J.1.3. FMNIST Dataset

Figure 16 presents the performance of explainers in an unsupervised setup across different analyses and metrics for the FMNIST dataset. Consistent with the supervised setup in Section 4.7, although SSplain-0 excels in insertion-based accuracy and normalized sparsity κ_s analyses, Occlusion performs better than SSplain in deletion analysis. However, SSplain still stabilizes earlier than all competing methods.



(a) Accuracy, connected components, normalized sparsity κ_s metrics vs. sparsity s.



(b) Curvature and dilation analyses vs. sparsity s and normalized sparsity κ_s.

Figure 13. Comparison SSplain-0 with intensity-based mask initialization versus uniform (all-ones) mask initialization for post-hoc analyses on the ROP dataset. The intensity-based case uses 50 ADMM iterations, while the uniform case uses 400. Both use $\rho = 0.01$, $\lambda = 0.00001$, a learning rate of 0.01 and a target area of 50% of α_0 in S_1 to 50% of $\|\mathbf{X}\|_0$.(a) We report the average of: Post-hoc balanced accuracy with deletion (lower is better) and insertion (higher is better) of pixels with the highest attribution scores, connected components ratio with insertion, and sparsity s versus normalized sparsity κ_s during the insertion process. (b) Curvature and dilation similarity with the insertion process with respect to s and κ_s. Intensity-based mask initialization outperforms uniform mask initialization across all metrics.

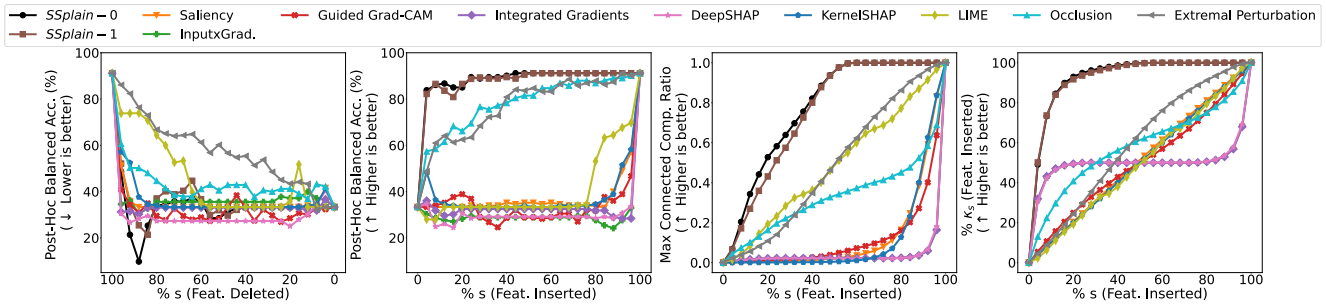


Figure 14. Comparison of explainers on the ROP dataset for unsupervised explaination generation: SSplain-0 (S_1 with ℓ_0 constraint), SSplain-1 (S_1 with ℓ_1 constraint), Saliency [35], Input×Gradient [33], Guided Grad-CAM [32], Integrated Gradients [36], DeepSHAP [27], KernelSHAP [27], LIME [30], Occlusion [52] and Extremal Perturbation [13]. We report the average of: Post-hoc balanced accuracy with deletion (lower is better) and insertion (higher is better) of pixels with the highest attribution scores, connected components ratio during the insertion process, and sparsity s versus normalized sparsity κ_s during the insertion process. Similar to the supervised explanation generation, SSplain consistently outperforms competitors and competitors give importance to the background.

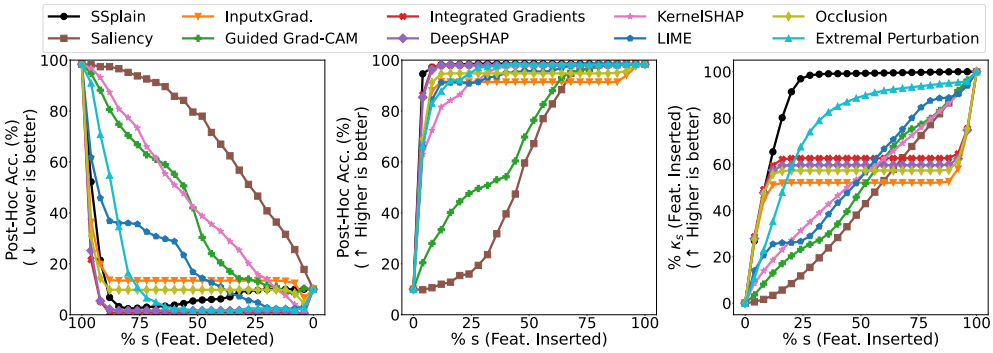


Figure 15. Comparison of explainers on the MNIST dataset for unsupervised explaination generation: SSplain-0 (S_1 with ℓ_0 constraint), SSplain-1 (S_1 with ℓ_1 constraint), Saliency [35], Input×Gradient [33], Guided Grad-CAM [32], Integrated Gradients [36], DeepSHAP [27], KernelSHAP [27], LIME [30], Occlusion [52] and Extremal Perturbation [13]. We report the average of: Post-hoc accuracy with deletion (lower is better) and insertion (higher is better) of pixels with the highest attribution scores, connected components ratio during the insertion process, and sparsity s versus normalized sparsity κ_s during the insertion process. Consistent with the supervised generation, SSplain-0 consistently outperforms the competing methods in all analyses, achieving lower deletion and higher insertion post-hoc accuracy, along with higher normalized sparsity κ_s.

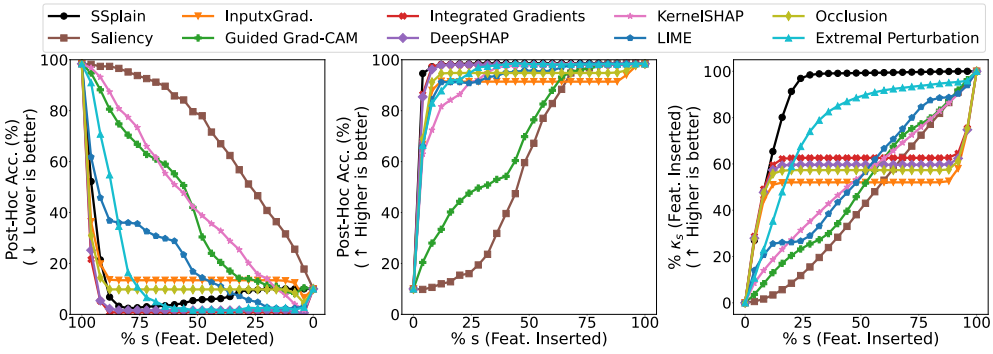


Figure 16. Comparison of explainers on the FMNIST dataset for unsupervised explaination generation: SSplain-0 (S_1 with ℓ_0 constraint), SSplain-1 (S_1 with ℓ_1 constraint), Saliency [35], Input×Gradient [33], Guided Grad-CAM [32], Integrated Gradients [36], DeepSHAP [27], KernelSHAP [27], LIME [30], Occlusion [52] and Extremal Perturbation [13]. We report the average of: Post-hoc accuracy with deletion (lower is better) and insertion (higher is better) of pixels with the highest attribution scores, connected components ratio during the insertion process, and sparsity s versus normalized sparsity κ_s during the insertion process. Consistent with the supervised generation, although SSplain-0 excels in insertion-based accuracy and normalized sparsity κ_s analyses, Occlusion performs better than SSplain in deletion analysis. However, SSplain still stabilizes earlier than all competing methods.