Project Sequence

Appropriate Dataset

Model Selection

Explainability Methods- SHAP, GradCam++, LIME for visual explainability

Combining Grad-CAM++ and SHAP as proposed in the Ensemble Image Explainable AI (XAI) Algorithm for Severe Community-Acquired Pneumonia and COVID-19 Respiratory Infections.

Ensemble Methods

Evaluation methods

Comparisons

Neural Networks, while offering really high accuracy, still mainly have black box operation. This black box nature prevents the trust of their healthcare adoption, which can be mitigated using explainability techniques.

Model

Different deep learning models are available on Keras. This include VGGNet, ResNet, InceptionV3 and Xception. InceptionV3 was selected for this experiment. The model introduces a network within a network to improve accuracy and efficiency. This model adopts multi-level feature extraction adapting well to different input sizes and complexities. There is an Xception model further increasing the capabilities of InceptionV3. However, its performance is better on bigger datasets; 350 million images.

## Explainable techniques

### SHAP

SHAP is a common method to explain the prediction of instance x by computing he contribution of each feature to the prediction. The SHAP method is an extension of the Shapley values to infinite player games for differentiable models using integrated gradients methods.

Assuming the input features to be independent and we have approximated the model using a linear function between the background data sample and the current input, the SHAP is given by computing the expected gradients. The gradient explainer then integrates the expected gradients of all interpolations between foreground and background samples.

The expected gradients are used to reformulate the integral and combine the expectation with sampling reference values from the background dataset to approximate the SHAP values. Thus we have a single expected gradient that converges towards the

### LIME

Locally Interpretable Model-Agnostic Explanation approximates complex models locally by an interpretable model that explains the prediction of a particular instance of interest.

To apply LIME method, following procedure is followed:

Firstly, a suitable intreable representation of the the instanc of interest is selected. Usually, it is superpixels which refer to the continuous patches of similar pixls. Thus, the interpretable prediction of image is the binary vector with 1’s indicating original superpixels and 0’s indicating grayed super pixels.

Then, disturb the interpretable representation and take out a sample. The sample will contain some 1’s for super pixels and few 0’s indicating greyed-out pixels.

Apply the original model to perturbed images and generate predictions.

Now fit the interpretable model to the proximity weighted sampled images and to the predictions generated in the step before.

Now, using the interpretable table, draw conclusions about the relevance of each interpretable component.

LIME is resource intensive process. Also the presence of noise in the input results in the instability of the explanation.

### Grad\_cam++

Grad-CAM makes the CNN-based model more transparent by visualizing the regions with high resolution details for producing more robust predictions. It performs this by visualization of the final feature map A^k using averaged gradient score as weights. The last layer of convolutional layer acts as the features of the classification model.

Possible shortcomings include the failure to localize the objects if there are multiple occurrences of the same objects. Moreover, only a portion of localization of objects due to the unweighted average of partial derivatives.

Grad-CAM++ improves on the limitations of Grad-CAM by introducing a measure of importance to each pixel in the feature contributing to the final CNN. Therefore, if there are multiple instances of the same object, all the spatially relevant regions are equally highlighted.

### Ensemble XAI

Existing Ensemble methods utilize Grad-Cam++ and SHAP to enhance the interpretability of models. These two methods are based on gradient descent, but each one has its own merits and demerits.

To implement this, an ensemble XAI based kernel ridge regression is appled to normalized SHAP and Grad-cam++ mapping layer to identify discriminative regions.

Generating grad-cam++ and SHAP heatmaps using the base model and for the ground truth for each image.

Grad-cam is bad with finer details.

**Ensemble XAI (SHAP + Grad-CAM++):** This ensemble method is described as applying Kernel Ridge regression to normalized values from Grad-CAM++ and SHAP heatmaps to generate a mapping layer identifying discriminative regions7.... It works by extracting and combining the **high contributed pixel features**9. By learning from annotations, it can assign weights to specific pixel features, potentially identifying important finer details within regions of interest9

LIME uses superpixels (contiguous patches of similar pixels) as its interpretable representation. Its superpixel-based explanations are noted to have large variance and can be linked to areas outside the lung, making it less competent in localization effectiveness compared to methods like Ensemble XAI and SHAP/Grad-CAM++. This suggests that its explanations might be at a coarser granularity than pixel-level methods or those that better localize fine features.

## Evaluation methods

The evaluation criteria for the performance of the method are as follows:

Decision Impact Ratio measures the impact of omitting critical regions as identified by the interpretable method using percentage change in decision.

Confidence Impact Ratio measures the percentage drop in confidence as the important regions identified by the interpretation method are omitted.

I built and evaluated an explanation ensemble that learns a pixel level fusion from Grad CAM, LIME, and SHAP, then I stabilized thresholding and precision with calibration and spatial priors.

**Data and notation**

Inputs were three saliency maps per image, G for Grad CAM, L for LIME, S for SHAP, each H by W. Ground truth attention masks were Y in the same shape. N equals 44 images, H equals W equals 224.

**Preprocessing**

I normalized each map per image using percentile scaling to the interval zero to one. For stability I used the ninety ninth percentile as the scale, which limits the influence of extreme values and keeps the dynamic range comparable across methods and across images.

**Fusion baselines**

I provided two simple fusions for comparison. A weighted mean after per image normalization, using weights that emphasized LIME while keeping the other maps present. An uncertainty aware map defined as the per pixel mean divided by the standard deviation with a small additive constant, which downweights regions where the methods disagree.

**Stacked learner ensemble**

I trained a logistic regression that predicts a fused probability per pixel from six features. The three normalized maps, the per pixel mean across methods, the per pixel variance across methods, and an effect size feature defined as mean divided by the square root of variance plus a small constant. I downsampled the training grid with a stride to thin spatial correlation, then I used group cross validation with groups defined by image to avoid leakage across pixels from the same image. I tuned the regularization strength C on a small grid and used the saga solver with standardization in a pipeline.

**Class imbalance handling**

I constructed the training set with negative downsampling per image. For each image I kept all positive pixels, then sampled at most a fixed multiple of negatives relative to the number of positives. This reduces the dominance of background pixels and improves numerical conditioning without changing the evaluation distribution.

**Out of fold scoring and thresholding**

I produced out of fold probabilities for all sampled pixels, then I selected a decision threshold on these scores. The first attempt used a precision target through the precision recall curve. Scores were very concentrated near zero, so the precision target forced a threshold that yielded zero recall. I replaced this with a safe selector that searches thresholds on quantiles of the score distribution, enforces a minimum number of predicted positives, enforces nonzero recall, and otherwise falls back to the best F beta score. This avoids degenerate solutions on imbalanced data.

**Prior probability correction**

Negative downsampling changes the class prior in the training set. I corrected probabilities with a closed form prior shift update. If p is the model probability under the training prior pi train, and the true prevalence over all pixels is pi true, the calibrated probability is  
p∗=πtrueπtrainpπtrueπtrainp+1−πtrue1−πtrain(1−p)p^{\*} = \frac{\frac{\pi\_{\text{true}}}{\pi\_{\text{train}}} p}{\frac{\pi\_{\text{true}}}{\pi\_{\text{train}}} p + \frac{1 - \pi\_{\text{true}}}{1 - \pi\_{\text{train}}} (1 - p)}  
I applied this to both the out of fold scores for threshold selection and the full resolution probability maps for inference.

**Spatial post processing**

To raise precision without retraining the classifier I added two spatial priors at inference. A per image cap on the number of positive pixels that implements a top k rule on the probability map, where k equals a fixed fraction of H times W. A connected component filter that removes tiny isolated blobs under a chosen pixel count. Both operations reduce false positives by focusing on compact high score regions that are more consistent with focal lesions.

**Evaluation protocol**

I reported set level mean IoU, mean F1, mean precision, and mean recall by comparing the predicted binary masks to Y for each image and averaging the per image metrics. I also exposed the out of fold metrics for the sampled training pixels to show how the learner behaves under cross validation. I used precision recall analysis rather than ROC because the positive class is rare at the pixel level.

**Observed behavior**

Your initial stacked model achieved moderate recall and very low precision on out of fold pixels and on full image masks, which is consistent with an unfinetuned Xception backbone that yields diffuse saliency. The precision constrained threshold collapsed to zero recall, which indicated that the score distribution had very low separability. The safe selector restored nonzero recall, but precision stayed low. The calibration and caps are designed to trade a small amount of recall for meaningful precision gains, but they cannot fully compensate for a base classifier that does not separate classes.

**Practical reading of the threshold**

For the conservative fusion I defined the decision threshold per image as the k th largest value in the consensus map, where k equals the cap on the allowed number of positive pixels. This produces a clear control knob. Lower cap fraction yields higher threshold and higher precision, higher cap fraction yields lower threshold and higher recall.

**What to tune next**

If you can retrain the classifier, increase precision with focal loss, hard negative mining, or attention supervision that encourages activation inside Y. If you must keep the current backbone, tune three levers only. The precision target in the safe selector, the per image cap fraction, and the minimum component size. These three levers give a stable operating point on the precision recall curve and keep the procedure reproducible.

Notes

So gradcam used gradient dscent ot locate the high resolution areas of ht efeture map. This means localixzation of more than one similar object is not possible.

Grad-cam++ solves this problem by highlighting all the spacially important regions.

SHAP highlights the important tregions with respect ot SHAP value. Instance x to toal feqatuemap.

https://github.com/jacobgil/pytorch-grad-cam/blob/master/tutorials/CAM%20Metrics%20And%20Tuning%20Tutorial.ipynb