IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 10, 2021, accepted March 1, 2021, date of publication March 9, 2021, date of current version March 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3064838

These *do not* Look Like Those: An Interpretable Deep Learning Model for Image Recognition

GURMAIL SINGH[1] AND KIN-CHOONG YOW[2], (Senior Member, IEEE)
[1]Department of Computer Science, University of Regina, Regina, SK S4S 0A2, Canada
[2]Faculty of Engineering and Applied Sciences, University of Regina, Regina, SK S4S 0A2, Canada

Corresponding author: Kin-Choong Yow (Kin-Choong.Yow@uregina.ca)

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), funding reference number DDG-2020-00034. Cette recherche a été financée par le Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG), numéro de référence DDG-2020-00034.

ABSTRACT Interpretation of the reasoning process of a prediction made by a deep learning model is always desired. However, when it comes to the predictions of a deep learning model that directly impacts on the lives of people then the interpretation becomes a necessity. In this paper, we introduce a deep learning model: *negative-positive prototypical part network* (NP-ProtoPNet). This model attempts to imitate human reasoning for image recognition while comparing the parts of a test image with the corresponding parts of the images from known classes. We demonstrate our model on the dataset of chest *X*-ray images of Covid-19 patients, pneumonia patients and normal people. The accuracy and precision that our model receives is on par with the best performing non-interpretable deep learning models.

INDEX TERMS Covid-19, pneumonia, image recognition, *X*-ray, prototypical part.

I. INTRODUCTION

The importance of deep learning algorithms stems from the fact that they are capable of solving many social and economic problems. However, most of the deep learning algorithms work as a black-box, because they lack the transparency of the reasoning process of their predictions. Hence, the lack of interpretability/transparency of the reasoning process of such deep learning models has become a key issue for whether we can trust predictions that are coming from these models. There have been cases where incorrect data fed into black box models have gone unnoticed, leading to unfairly long prison sentences (e.g., prisoner Glen Rodriguez was denied parole due to an incorrect COMPAS score) [22] and [36]. Our research question is to find an interpretable method to do image classification so that we can tell why an image is classified in a certain way. In this paper, we introduce an interpretable deep learning model: *negative-positive prototypical part network* (NP-ProtoPNet). NP-ProtoPNet is closely related to *prototypical part network* (ProtoPNet) [3] model, but it is strikingly different from ProtoPNet. Our model considers both positive reasoning and negative reasoning to classify images, whereas ProtoPNet emphasizes only on positive reasoning. In our work, prototype/prototypical part of an image will mean a patch of the image, and we will use the words patch, prototype and prototypical part interchangeably. To classify an image, ProtoPNet model dissects the (test) image, and compare its prototypical parts with some patches of images from each class of images. Then ProtoPNet accepts a class as the true class of the test image if some patches of the test image match with some prototypical parts of images from the class [3]. We call this type of reasoning a positive reasoning. In addition to positive reasoning, our model also uses the negative reasoning process of rejecting the classes whose images do not have any prototypical part that matches with any prototypical part of the test image. The idea of using negative reasoning process is similar to the idea of solving a multiple choice question, where it becomes helpful to rule-out the options that are surely not an answer of the question. We demonstrate our model on the dataset of *X*-ray images of Covid-19 patients, pneumonia patients and normal people. The accuracy and precision that our model receives is on par with the best performing non-interpretable deep learning models.

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Wei.

II. LITERATURE REVIEW

Some techniques have been developed to interpret the convolution neural networks, such as: posthoc interpretability analysis and attention-based interpretability. In *posthoc* analysis of a deep learning model, one interprets a trained convolution

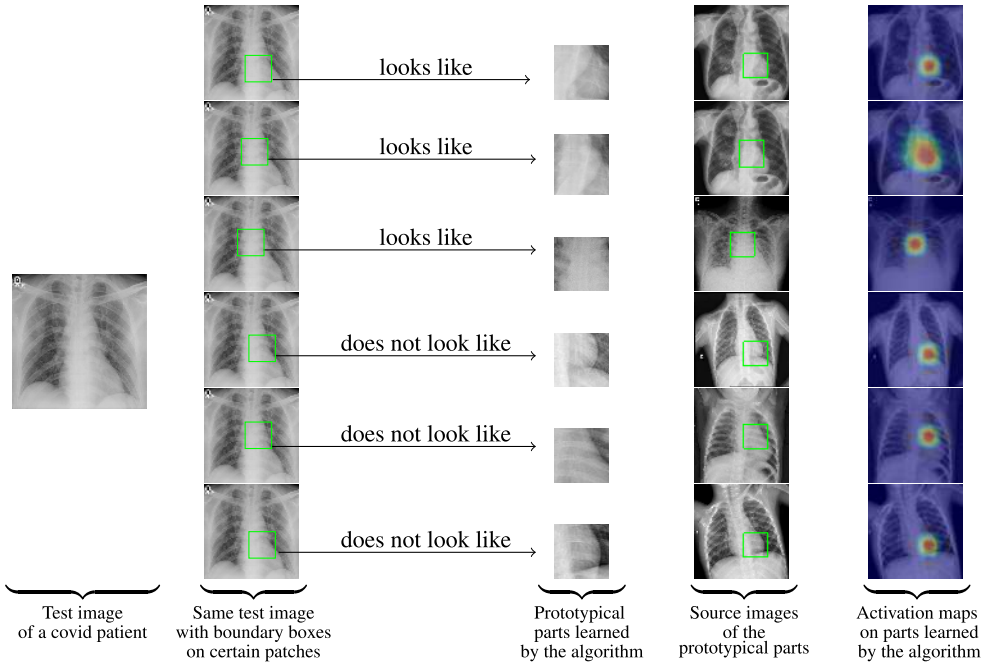

FIGURE 1. An *X*-ray image of a Covid-19 patient and how its parts look like or do not look like some learned prototypical parts of *X*-ray images of Covid-19 patients, normal persons and pneumonia patients.

neural network after the predictions with the fitting explanations of the reasoning behind classifications made by the model. Posthoc analysis is made with various techniques, such as: activation maximization [5], [13], [20], [24], [30], [35], [43], deconvolution [45], and saliency visualization [27], [30], [32], [33]. However, these posthoc visualization methods fail to explain the actual reasoning process behind the classifications made by the models.

Many models have been developed that build attention-based interpretability, such as: class activation maps and part-based models. An attention-based interpretability model attempts to highlight the parts of an input image on which the network focuses [6], [10], [11], [15], [26], [29], [34], [42], [46]–[49]. Nevertheless, these models have serious drawback of not pointing out the prototypical parts that are similar to patches of the image on which the model focuses.

Li *et al.* [21], proposed a deep learning architecture that builds case-based reasoning into a neural network. Then Chen *et al.* [3] along with the authors of the above paper made a considerable improvement in their model ProtoPNet, whereby the network makes prediction by comparing image patches with the learned prototypes.

Typically a person has some physical features similar to his parents and even to his siblings. We can say with some certainty that a person belongs to a given family if we remember the faces of the person's family members, because we can compare eyes, nose, lips and some other physical features of the person with those features of his family members. Similarly, to classify an image we might focus on parts of the image and compare them with prototypical parts of images from a given class. This type of human reasoning is accommodated by the ProtoPNet model, where comparison of image parts with learned prototypes is integral to the reasoning process of the model.

Recently, some deep learning/machine learning models have been developed to classify the *X*-ray images of Covid-19 patients, normal people and pneumonia patients, see [1], [7], [16], [17], [19], [25], [28], [44]. A survey article is also written that summarizes the research works related to deep learning applications on COVID-19 medical image processing [2]. The survey paper also points out the issues related to deep learning implementation on COVID-19 medical image processing, including lack of reliable and adequate data. Probably, the models in the above papers are developed with an intention to reduce the menace of the pandemic Covid-19 by developing alternate method for detection of the disease from *X*-ray images while there was shortage of test kits all over the world. Unfortunately, the models developed in these papers are not interpretable, unlike NP-ProtoPNet. However, transparency of a deep learning model is important if it directly impacts on the lives of people. In this work, we developed an interpretable model and experimented it over the same dataset of medical images that is used in the above Covid-19 related papers.

III. WORKING PRINCIPLE OF THE NP-ProtoPNet

For an *X*-ray image as in Figure 1, ProtoPNet model is able to identify several parts of the image where it thinks that *this* patch of the image *looks like that* prototypical part of an image of some class. It makes its prediction based on a weighted combination of the similarity scores between patches of the test image and the prototypes (prototypes that

are learned by the model during the training process) of images from all classes [3]. Similarity scores between patches of a test image and the learned prototypes are obtained with minimization of the $L2$ (Euclidean) distances between patches of the test image and the learned prototypes. However, in addition to the positive reasoning of ProtoPNet, our model NP-ProtoPNet rejects the classes by identifying several parts of the image where it thinks that any of *these* parts of the image *do not look like those* prototypical parts of images from incorrect classes (classes other than the true class of the image), and makes its prediction based on a weighted combination of the similarity scores between parts of the test image and the learned prototypes. The similarity scores corresponding to the correct class and incorrect classes are assigned the weight of $+1$ and -1, respectively. In Figure 1, we see that the model detects some patches of the test image that *do not look like* the learned prototypical parts of some classes. The novelty of our model NP-ProtoPNet is that it considers the negative connection between similarity scores and incorrect classes as well as positive connection between similarity scores and correct class, whereas ProtoPNet emphasizes only on positive connection with correct class. The consideration of the negative connections of similarity scores with incorrect classes accompanying positive connections with correct class helped our model to improve its performance in some metrics, such as: accuracy, precision, recall and F1-score, see Table 1.

IV. METHODOLOGY

In this section, we explain our architecture *negative-positive prototypical parts network,* NP-ProtoPNet. We trained and evaluated our network on the dataset of frontal chest X-ray images of Covid-19 patients [8], pneumonia patients and normal persons [18]. The dataset of chest X-ray images from Kaggle database [18] has 3875 and 1341 training images of pneumonia patients and normal persons, respectively. Also, the dataset has 390 and 234 test images of pneumonia patients and normal persons, respectively. The other database [8] has 930 medical images of Covid-19 patients. These medical images include, frontal chest X-ray images, CT-scan images, side X-ray images and few obscure (completely black or white) images. Among the medical images of Covid-19 patients, we selected only 748 frontal chest X-ray images. As compared to the number of chest X-ray images of pneumonia patients and normal persons, the number of chest X-ray images of Covid-19 patients was much lesser. Therefore, to balance the data, a copy of the chest X-ray images of Covid-19 patients was included to form a dataset of 1496 images. The 1496 frontal chest X-ray images of Covid-19 patients were divided into train and test sets of 1248 and 248 images, respectively. All these images form the three classes, labeled as: Covid, Normal and Pneumonia. We resized the images to dimension 224×224. A detailed explanation of our model is provided below with the classification of a new X-ray image and explanations of the reasoning process of its prediction.

A. NP-ProtoPNet ARCHITECTURE

An overview of the architecture of our NP-ProtoPNet model is given in Figure 2. Our network consists of a regular convolutional neural network f, whose parameters are collectively denoted by w_{conv}, followed by a prototype layer g_p and a fully connected layer h with weight matrix w_h and no bias. The prototype layer g_p is a generalized convolution layer, [9], [23]. For the regular convolutional network f, our model uses the convolutional layers from models such as VGG-16, VGG-19 [31], ResNet-34, ResNet-152 [12], DenseNet-121 and DenseNet-161 [14] (initialized with filters pretrained on ImageNet [4]). Our model does not have any additional convolution layer unlike ProtoPNet network. In this work, we call the models VGG-16, VGG-19, ResNet-34, ResNet-152, DenseNet-121 and DenseNet-161 base/baseline models.

For an input image x, the convolutional layers of a base model extract useful features $f(x)$ to use for prediction. The shape of the convolutional output $f(x)$ is $7 \times 7 \times D$, where D is the number of the output channels. The number of output channels of the convolution layers of VGG-16, VGG-19, Resnet-34, ResNet-152, DenseNet-121 and DenseNet-161 are 512, 512, 512, 2048, 1024 and 2208, respectively. Since the prototype layer of our model is attached immediately after the regular convolution layers, the depth of each prototype is the same as that of the convolution output. Our network learns m prototypes $P = \{p_j\}_{j=1}^{m}$, whose shape is $1 \times 1 \times D$, where $m = 30$. For each class of X-rays, we choose 10 prototypes, and the selection of the number of prototypes is arbitrary. These prototypes should capture sufficient relevant patches for identifying images of a class. Since the depth of each prototype is the same as that of the convolutional output but the height and the width of each prototype is smaller than those of the whole convolutional output, each prototype will be used to represent some prototypical activation pattern in a patch of the convolutional output, which in turn will correspond to some prototypical image patch in the original pixel space [3]. Hence, each prototype p_j can be viewed as the latent representation of some prototypical part of some X-ray image.

The prototype p_1 in Figure 2 corresponds to some patch of the chest X-ray image of a Covid-19 patient, and the prototype p_{11} and p_{30} correspond to some patch of the chest X-ray image of a normal person and a pneumonia patient, respectively. Given a convolutional output $z = f(x)$, the j-th prototype unit g_{p_j} in the prototype layer g_p computes the squared $L2$ distances between the j-th prototype p_j and all patches of z that have the same shape as p_j, and inverts the distances into similarity scores. The result is an activation map of similarity scores whose value indicates how strong a prototypical part is present in the image. The most activated patch of an image is the patch of the image that gives the highest similarity score. This activation map preserves the spatial relation of the convolutional output, and can be upsampled to the size of the input image to produce a heat map

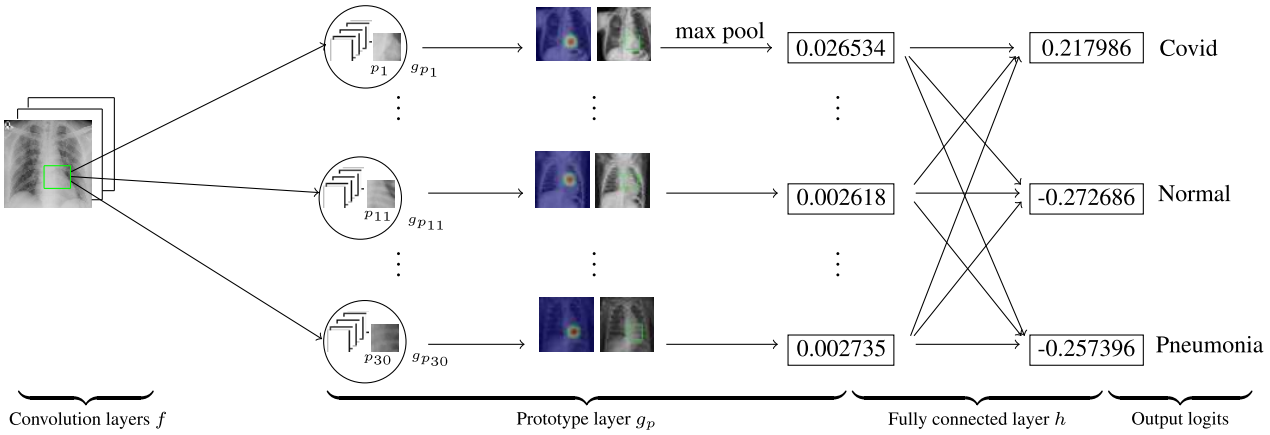

FIGURE 2. NP-ProtoPNet architecture.

that identifies which part of the input image is most similar to the learned prototype [3]. The activation map can also be used to draw rectangles on the source images to specify the parts that correspond to prototypes. The activation map of similarity scores produced by each prototype unit g_{p_j} is then reduced using global max pooling to a single similarity score, which can be understood as how strongly a prototypical part is present in some patch of the input image. Then in the dense layer h, the similarities scores produced by the prototype layer g_p are multiplied with the weight matrix w_h to produce logits, which are further normalized with softmax to get the predictions for a given test image belonging to the classes under consideration.

Entries of kth row of a weight matrix w_h are connections between similarity scores and logit of kth class, see Section *V* for an example of a weight matrix. We fix entries of the matrix w_h to ± 1, unlike the weight matrix of ProtoPNet model whose entries are initially set to $+1$ and -0.5. Therefore, we set the incorrect class connection to -1 instead of -0.5, unlike ProtoPNet. As mentioned above, our model equally considers both the positive reasoning and negative reasoning to classify an image. Hence, our model does not attempt to make the negative connections -1 to 0 during the training process, again unlike ProtoPNet.

The similarity scores between most activated patches of the test image and prototypes of classes Covid, Normal and Pneumonia form a column matrix a, see Section V. The logits for the classes Covid, Normal and Pneumoina are obtained by multiplying the first, second and third row of w_h with the matrix a, see Section V. In Figure 2, the test image is an X-ray image of a Covid-19 patient. The prototypes p_1 to p_{10}, p_{11} to p_{20} and p_{21} to p_{30} are the parts of X-ray images of Covid-19 patients, normal persons and pneumonia patients, respectively. The similarity scores between patches of the test image and prototypes p_1, p_{11} and p_{30} are 0.026534, 0.002618 and 0.002735, respectively. The full list of similarity scores that we obtain from our experiments is given in the matrix a, see Section V. Therefore, the logits for Covid, Normal and Pneumoina classes are 0.217986, -0.272692 and -0.257398, respectively.

B. MATHEMATICAL FORMULATION AND TRAINING ALGORITHM

In our model, mathematically, the prototype unit g_{p_j} computes the following:

$$g_{p_j}(z) = \max_{\tilde{z} \in \text{ patches}(z)} \log\left(\frac{||\tilde{z} - p_j||_2^2 + 1}{||\tilde{z} - p_j||_2^2 + \epsilon}\right). \tag{1}$$

The above equation (1) tells us that if $\tilde{z}$ is the closer latent patch to p_j then $||\tilde{z} - p_j||_2$ is smaller. Therefore, the function g_{p_j} is monotonically decreasing with respect to $||\tilde{z} - p_j||_2$. Hence, if the output of the j-th prototype unit g_{p_j} is large, then there is a patch in the convolutional output that is very close to the j-th prototype in the latent space, and this in turn means that there is a patch in the input image that has a similar patch to what the j-th prototype represents [3].

The training of our model is divided into two steps: (1) stochastic gradient descent (SGD) of layers before the last layer; (2) projection of prototypes. At the first step, we aim to learn a meaningful latent space, where the most important patches for classifying images are clustered around semantically similar prototypes of the images' true classes, and the clusters that are centered at prototypes from different classes are well-separated. To achieve this goal, we jointly optimize the convolutional layers' parameters w_{conv} and the prototypes $P = \{p_j\}_{j=1}^m$ in the prototype layer g_p using SGD, while keeping the last layer weight matrix w_h fixed. Let $D = [X, Y] = \{(x_i, y_i)\}_{i=1}^n$ be the set of training images. The optimization problem we aim to solve here is:

$$\min_{P, w_{\text{conv}}} \frac{1}{n} \sum_{i=1}^{n} \text{CrsEnt}(h \circ g_P \circ f(x_i), y_i) + \lambda_1 \text{Clst} + \lambda_2 \text{Sep}, \tag{2}$$

where Clst and Sep are defined by

$$\text{Clst} = \frac{1}{n} \sum_{i=1}^{n} \min_{j: p_j \in P_{y_i}} \min_{z \in \text{patches}(f(x_i))} ||z - p_j||_2^2; \tag{3}$$

$$\text{Sep} = -\frac{1}{n} \sum_{i=1}^{n} \min_{j: p_j \notin P_{y_i}} \min_{z \in \text{patches}(f(x_i))} ||z - p_j||_2^2. \tag{4}$$

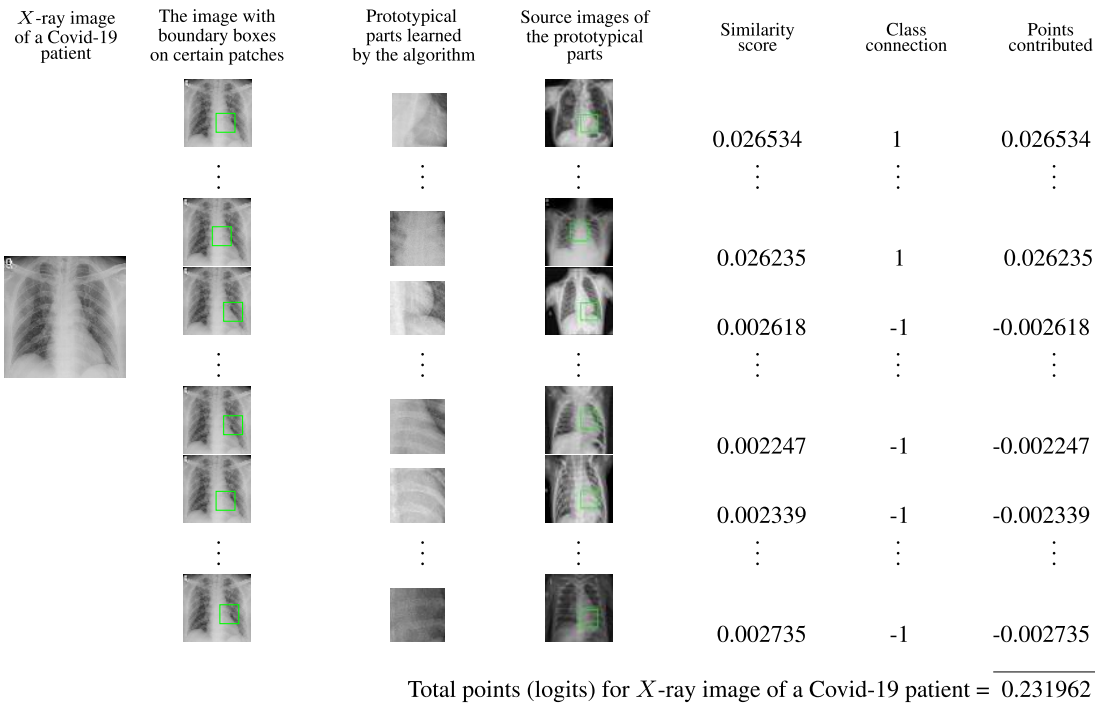

FIGURE 3. The reasoning process of our network in deciding the class of an *X*-ray image.

The increase in the cross entropy (CrsEnt) leads to misclassifications on the training data, see the optimization problem (2). The minimization of the cluster cost (Clst) encourages each training image to have some latent patch that is close to at least one prototype of its own class, while the minimization of the separation cost (Sep) encourages every latent patch of a training image to stay away from the prototypes not of its own class [3], see the above equations (3) and (4). We set the coefficient of the cluster cost λ_1 and the coefficient of the separation cost λ_2 equal to 0.8.

Let $w_h^{(k,j)}$ be the (k,j)-th entry in weight matrix w_h that corresponds to the weight connection between the output of the j-th prototype unit g_{p_j} and the logit of a class k. Let P_k be the collection of prototypes of a class k. Given a class k, we set $w_h^{(k,j)} = 1$ for all j with $p_j \in P_k$ and $w_h^{(k,j)} = -1$ for all j with $p_j \notin P_k$. More importantly, during the training process, our model does not make these negative connections 0; whereas in ProtoPNet model $w_h^{(k,j)} = -0.5$ for all j with $p_j \notin P_k$, and even these negative values do not remain fixed as ProtoPNet attempts to make negative connections 0. Note that, similarity scores are nonnegative numbers. Therefore, the positive connection between a prototype of class k and the logit of class k means that similarity to a prototype of class k should increase the predicted probability that the image belongs to class k. On the other hand, if $j \neq k$ then the negative connection between a prototype of a class j and logit of class k means that similarity to a prototype of class j should decrease class k's predicted probability. By fixing the last layer h in this way, we can force the network to learn a meaningful latent space because if a latent patch of a class k image is too close to a prototype of class j, it will decrease the predicted probability that the image belongs to class k and increase the cross entropy in the training objective [3].

At the second step, to be able to visualize the prototypes as training image patches, we project/push each prototype p_j onto the nearest latent training patch from the same class as that of p_j. In this way, we can conceptually equate each prototype with a training image patch [3]. Mathematically, for prototype p_j of class k, i.e., $p_j \in P_k$, we perform the following update: $p_j \longleftarrow \min_{z \in Z_j} ||z - p_j||_2$, where

$$Z_j = \{\tilde{z} : \tilde{z} \in \text{patches}(f(x_i))\ \forall i \text{ s.t. } y_i = k\}.$$

C. SELECTION OF AN IMAGE PATCH AS A PROTOTYPE

The image patch of an image x that is highly activated by a protype p_j is used for the visualization of p_j. Such a patch of an image x on which p_j has the strongest activation can be found by forwarding x through a trained NP-ProtoPNet and upsampling the activation map produced by the prototype unit g_{p_j} (before maxpooling) to the size of the image x. Then we can visualize p_j with the smallest rectangular patch of x that encloses pixels whose corresponding activation value in the upsampled activation map from g_{p_j} is at least as large as the 93rd-percentile of all activation values [3].

V. EXPLANATION OF THE REASONING PROCESS

Consider the three classes from where the images are taken (listed in lexicographical order): Covid, Normal and Pneumonia. Among the six base models that we have used for

our experiments, we choose the base model VGG-19 for the experiments explained in this paper.

In Figure 3, the leftmost column has a test image, an X-ray image of a Covid-19 patient. In the second column, the same test images has rectangular boxes that enclose the pixels that correspond to the pixels bounded by the boxes on the source images of the prototypes in the same row and fourth column. So, the bounded patches of the images in the second column correspond to the bounded patches of the images in the fourth column. The third column has prototypes obtained from the corresponding source images in the fourth column. The fourth column consists of source images of the prototypes from the same rows. The bounded patch of the source images are actually the parts from where the prototypes are projected/pushed. In the fifth column, similarity scores are calculated as described in Section IV-A. The similarity scores are listed in the column matrix a, see below. The ith entry of matrix a is the similarity score between ith prototype and a patch of the test image. Therefore, the entries 1 to 10, 11 to 20 and 21 to 30 of matrix a are the similarity scores of the prototypes of Covid class, Normal class and Pneumonia class, respectively. In the sixth column, the class connections (weights) are given. The (k, j)-th entry $w_h^{(k,j)}$ of weight matrix w_h is the weight connection between the output of the j-th prototype unit g_{p_j} and the logit of class k. Therefore, the weight matrix

$$w_h = \begin{bmatrix} 1 & \dots & 1 & -1 & \dots & -1 & -1 & \dots & -1 \\ -1 & \dots & -1 & 1 & \dots & 1 & -1 & \dots & -1 \\ -1 & \dots & -1 & -1 & \dots & -1 & 1 & \dots & 1 \end{bmatrix}$$

where horizontal dots $\dots$ represent missing eight columns that are equal to the neighboring columns. The weights corresponding to first (Covid), second (Normal) and third (Pneumonia) classes are listed in the first, second and third row of the weight matrix w_h. The seventh column has the points contributed to the Covid class that are obtained by multiplying similarity scores with the corresponding weights. The logit for Covid class is the total of the points contributed to the Covid class. Another words, the logit for the Covid class is obtained by multiplying first row of the weight matrix w_h and the similarity score matrix a. Similarly, the logits for Normal class and Pneumonia classes are obtained by multiplying the second row and third row of w_h with the matrix a, respectively. Therefore, the logits (up to six decimal places) for Covid, Normal and Pneumonia classes are 0.217986, -0.272686 and -0.257396, respectively.

In Figure 4, the test image belongs to the Normal class. For the test image of the Normal class, the similarities scores (that we obtained from our experiments) between the test image and prototypes (up to six decimal places) for the Covid class, Normal class and Pneumonia class are entries of the matrix b as given below. Therefore, logits for Covid class, Normal class and Pneumonia class are obtained by multiplying the first, second and third row of the weight matrix w_h with the similarity score matrix b, respectively. To save space, we do not write the weights in the figure, but the weights are given in the weight matrix w_h. Therefore, the logits for Covid class, Normal class and Pneumonia class are -0.145744, 0.046204 and -0.085446, respectively.

In Figure 5, the test image belongs to the Normal class. For the test image of the Normal class, the similarities scores (up to six decimal places) are entries of the matrix c as given below. Therefore, the logits for Covid class, Normal class and Pneumonia class are obtained by multiplying the first, second and third row of the weight matrix w_h with the similarity score matrix c, respectively. Similar to Figure 4, to save space, we do not write the weights in the figure. The logits for Covid class, Normal class and Pneumonia class are -0.546797, -0.280923 and 0.221254, respectively.

The following are the similarity score matrices a, b and c for the test images used in figures 4, 5 and 6, respectively.

$$a = \begin{bmatrix} 0.026534 \\ 0.026643 \\ 0.026534 \\ 0.026235 \\ 0.026584 \\ 0.026634 \\ 0.026574 \\ 0.026534 \\ 0.026534 \\ 0.026235 \\ 0.002618 \\ 0.002618 \\ 0.002618 \\ 0.001633 \\ 0.001633 \\ 0.001633 \\ 0.001633 \\ 0.001536 \\ 0.001536 \\ 0.002247 \\ 0.002339 \\ 0.002741 \\ 0.002805 \\ 0.002805 \\ 0.002770 \\ 0.002749 \\ 0.002736 \\ 0.002835 \\ 0.002835 \\ 0.002735 \end{bmatrix}, \quad b = \begin{bmatrix} 0.002089 \\ 0.001858 \\ 0.001858 \\ 0.001858 \\ 0.002038 \\ 0.002038 \\ 0.002038 \\ 0.001948 \\ 0.001948 \\ 0.001948 \\ 0.045742 \\ 0.011448 \\ 0.011448 \\ 0.007843 \\ 0.007843 \\ 0.007843 \\ 0.007843 \\ 0.005195 \\ 0.005195 \\ 0.005195 \\ 0.004872 \\ 0.004982 \\ 0.005007 \\ 0.005007 \\ 0.005007 \\ 0.004987 \\ 0.004977 \\ 0.004977 \\ 0.004977 \\ 0.004977 \end{bmatrix}, \quad c = \begin{bmatrix} 0.002089 \\ 0.001858 \\ 0.001858 \\ 0.001858 \\ 0.002038 \\ 0.002038 \\ 0.002038 \\ 0.001948 \\ 0.001948 \\ 0.001948 \\ 0.045742 \\ 0.011448 \\ 0.011448 \\ 0.007843 \\ 0.007843 \\ 0.007843 \\ 0.007843 \\ 0.005195 \\ 0.005195 \\ 0.005195 \\ 0.004872 \\ 0.004982 \\ 0.005007 \\ 0.005007 \\ 0.005007 \\ 0.004987 \\ 0.004977 \\ 0.004977 \\ 0.004977 \\ 0.004977 \end{bmatrix}.$$

VI. THE METRICS TO MEASURE THE PERFORMANCE

To compare the performance of our model with ProtoPNet and baseline models, we used some most commonly used metrics, such as: accuracy, precision, recall and F1-score. We also used confusion matrix to the describe the performance of our model with the six baseline models.

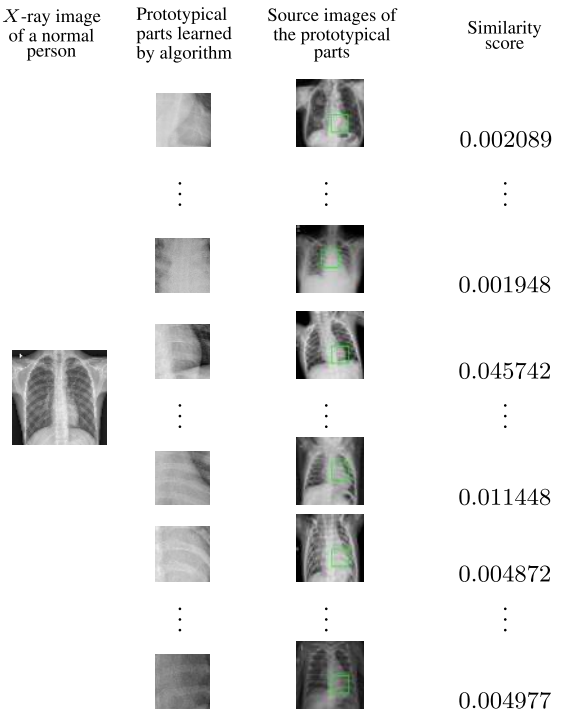

FIGURE 4. The reasoning process in deciding the class of an *X*-ray.

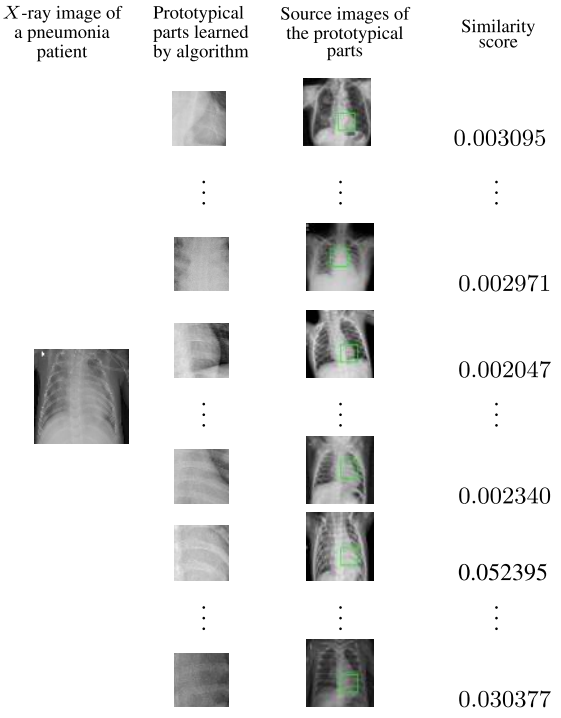

FIGURE 5. The reasoning process in deciding the class of an *X*-ray image of a normal person. image of a pneumonia patient.

True positive (*TP*) is the number of items correctly labelled as belonging to the positive class, that is, the items are predicted to belong to a class when they actually belong to that class. True negative (*TN*) measures the proportion of

Actual class

Predicted Class	Covid	Normal	Pneumonia
Covid	246	7	0
Normal	2	105	3
Pneumonia	0	122	387

FIGURE 6. Confusion matrix with baseline VGG-16.

negatives that are correctly identified as negatives, see [41]. False positive (*FP*) is the number of items incorrectly labelled as belonging to the positive class, that is, the items are predicted to belong to a class when they actually do not belong to that class. False negative (*FN*) is the number of items incorrectly labelled as not belonging to the positive class, that is, the items are predicted to not belonging to a class when they actually belong to that class, see [38].

Accuracy is the proportion of correct predictions (both true positives and true negatives) among the total number of cases examined [37], that is:

$$\text{Accuracy} = \frac{\text{number of correct predictions}}{\text{total number of cases}}.$$

Precision is the proportion of true positive predictions among the positive (true positive and false positive) predictions [38], that is:

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

Recall is the proportion of true positive predictions among the true positive predictions and false negative predictions [38], that is:

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

F1-score is the harmonic mean of precision and recall [39], that is:

$$\text{F1-score} = \frac{2}{\text{Precision}^{-1} + \text{Recall}^{-1}}.$$

Therefore,

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

A *confusion matrix* is a table that is used to describe the performance of a classification model on a set of test data for which the true values are known [40]. The confusion matrices for three classes Covid, Normal and Pneumonia are given in Section VII. For further discussion on the metrics and confusion matrices readers are directed to [37]–[41] and references therein.

VII. THE PERFORMANCE DESCRIPTION WITH CONFUSION MATRICES

In this section, we describe the performance of our model in some metrics. In figures 6 through 11, the confusion matrices give the visualization of the performance of our model with the baseline models VGG-16, VGG-19, ResNet-34, ResNet152, DensNet-121 and DenseNet-161, respectively.

Let C be any of the following six confusion matrices. Let $C[i][j]$ be the (i, j) entry of the matrix C. Then $C[0][0]$ is the TP for the Covid class. Also, $C[0][1] + C[0][2]$ and $C[1][0] + C[2][0]$ are the FP and FN for the Covid class. As mentioned in Section IV-A, the number of test images for the classes Covid, Normal and Pneumonia are 248, 234, 390, respectively. Therefore, the total number of test images is 872. For the baseline model VGG-16, the number of correct predictions of our model are $738(= 246 + 105 + 387)$, see Figure 6. Therefore, the accuracy for our model with the baseline model VGG-16 is 84.63% The FP and FN for our model with the baseline model VGG-16 are 7 and 2, respectively, see Figure 6. Thus, the precision and recall for our model with the baseline model VGG-16 are 0.97 and 0.99, see Section VI for the formulas. Hence, F1-score for our model with the baseline model VGG-16 is 0.98. Similarly, all the metrics (accuracy, precision, recall and F1-score) for our model with the other baseline models (VGG-19, ResNet-34, ResNet-152, DenseNet-121 and DenseNet-161) can be calculated from the confusion matrices depicted in Figures 7-11. However, these metrics are also given in Table 1.

VIII. COMPARISON OF NP-ProtoPNet WITH BASELINE MODELS AND ProtoPNet MODEL

Both NP-ProtoPNet and ProtoPNet models can be constructed over convolution layers of many regular neural networks, such as: series of VGG, ResNet and DenseNet models. However, as mentioned in Section IV-A, we experimented our model over the X-ray datasets [8], [18] with the six baseline models: VGG-16, VGG-19 [31], ResNet-34, ResNet-152 [12], DenseNet-121 and DenseNet-161 [14]. ProtoPNet with the six baseline models is also experimented over the same dataset. We trained and tested NP-ProtoPNet and ProtoPNet for 500 epochs with the above mentioned baseline models. Also, the baseline models themselves were trained and tested for 500 epochs. In Table 1, the measure of the performance of each of the three models NP-ProtoPNet, ProtoPNet and the baseline models are given in the metrics: accuracy, precision, recall and F1-score.

The Table 1 is explained with the description of the performance of NP-ProtoPNet and ProtoPNet with only one baseline model VGG-16, though the performance of NP-ProtoPNet and ProtoPNet model with the other baseline models is also given in the Table 1.

For example, the measures of the performance of our model with the baseline model VGG-16 in the metrics accuracy, precision, recall and F1-score are 84.63, 0.97, 0.99 and 0.97, respectively. The accuracy, precision, recall and F1-score for ProtoPNet with baseline model VGG-16 are 79.73, 0.87, 0.92 and 0.89, respectively. The accuracy, precision, recall and F1-score of baseline model VGG-16 itself (Base only) are 82.45, 0.97, 0.98 and 0.97, respectively. The comparison of the performance of NP-ProtoPNet and ProtoPNet models over other baseline models (VGG-19, ResNet-34, ResNet-152, DenseNet-121 and DenseNet-161) is also given in Table 1.

The performance of our model in the metrics is better than the performance of ProtoPNet with all the baseline models, see Table 1. However, the performance of our model in accuracy is better than the performance of only two baseline models (Base only): VGG-16 and VGG-19.

Actual class

Predicted Class	Covid	Normal	Pneumonia
Covid	240	21	1
Normal	0	149	2
Pneumonia	8	64	387

FIGURE 7. Confusion matrix with baseline VGG-19.

Actual class

Predicted Class	Covid	Normal	Pneumonia
Covid	248	2	0
Normal	0	97	0
Pneumonia	0	135	390

FIGURE 8. Confusion matrix with baseline ResNet-34.

Actual class

Predicted Class	Covid	Normal	Pneumonia
Covid	246	2	0
Normal	0	119	1
Pneumonia	2	113	389

FIGURE 9. Confusion matrix with baseline ResNet-152.

Actual class

Predicted Class	Covid	Normal	Pneumonia
Covid	244	2	0
Normal	2	122	2
Pneumonia	2	110	388

FIGURE 10. Confusion matrix with baseline DenseNet-121.

Actual class

Predicted Class	Covid	Normal	Pneumonia
Covid	246	6	0
Normal	0	123	0
Pneumonia	2	105	390

FIGURE 11. Confusion matrix with baseline DenseNet-161.

The performance of the other baseline models ResNet-34, ResNet-152, DenseNet-121 and DenseNet-161 in the metrics is better than the performance of our model. Nevertheless, the accuracy of our model is at most 1.63% lesser than accuracy of these models. Since the baseline models are not interpretable as our model, the loss of accuracy is compensated with the interpretability.

IX. GRAPHICAL COMPARISON OF THE ACCURACIES

In this section, we graphically compare the accuracies of NP-ProtoPNet, ProtoPNet and the baseline models over the training epochs. In the figures 12-17, the curves of yellow, blue and brown colors represent the accuracies of NP-ProtoPNet, ProtoPNet and the baseline model, respectively. In each of these figures, the accuracies given by NP-ProtoPNet and ProtoPNet are compared with the same baseline model whose convolution layers are used to construct them. Thus, in figures 12 through 17, the accuracies given by NP-ProtoPNet and ProtoPNet are compared with the baseline models VGG16, VGG-19, ResNet-34, ResNet152, DenseNet-121 and DenseNet-161, respectively. Note that, the accuracy given by each model (NP-ProtoPNet, ProtoPNet

TABLE 1. Comparison of metrics of our model with ProtoPNet and baseline models.

Base	Metrics	NP-ProtoPNet+Base	ProtoPNet [1] + Base	Base only
VGG-16				
	accuracy	**84.63**	79.73	82.45
	precision	0.97	0.87	0.97
	recall	0.99	0.92	0.98
	F1-score	0.97	0.89	0.97
VGG-19				
	accuracy	**88.99**	81.78	82.68
	precision	0.91	0.84	0.96
	recall	0.96	0.95	0.98
	F1-score	0.93	0.89	0.97
ResNet-34				
	accuracy	84.29 ± 0.10	82.90 ± 0.21	$\mathbf{84.89 \pm 0.12}$
	precision	0.99	0.98	0.99
	recall	0.99	0.99	0.99
	F1-score	0.99	0.98	0.99
ResNet-152				
	accuracy	86.40 ± 0.12	83.60 ± 0.34	$\mathbf{88.03 \pm 0.24}$
	precision	0.99	0.87	0.98
	recall	0.99	0.89	0.99
	F1-score	0.99	0.88	0.98
DenseNet-121				
	accuracy	86.47 ± 0.11	85.90 ± 0.36	$\mathbf{88.41 \pm 0.12}$
	precision	0.99	0.98	0.99
	recall	0.98	0.98	0.99
	F1-score	0.98	0.98	0.99
DenseNet-161				
	accuracy	87.04 ± 0.21	86.58 ± 0.32	$\mathbf{88.42 \pm 0.23}$
	precision	0.97	0.95	0.99
	recall	0.99	0.96	0.99
	F1-score	0.97	0.96	0.99

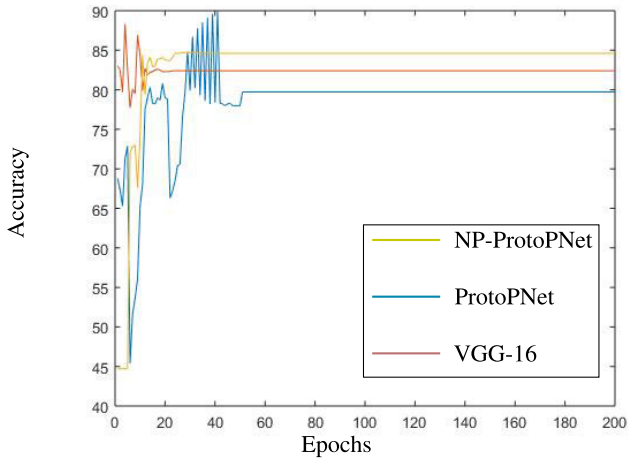

FIGURE 12. Accuracy comparison with baseline VGG-16.

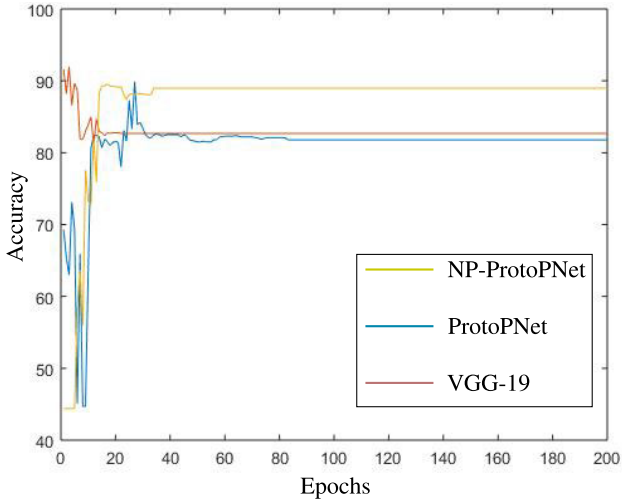

FIGURE 13. Accuracy comparison with baseline VGG-19.

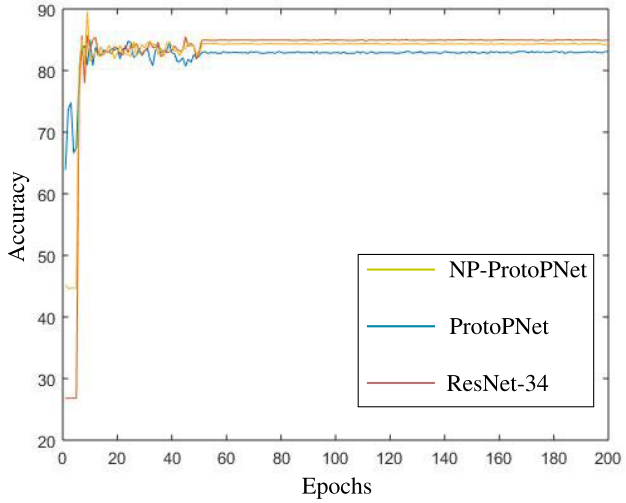

FIGURE 14. Accuracy comparison with baseline ResNet-34.

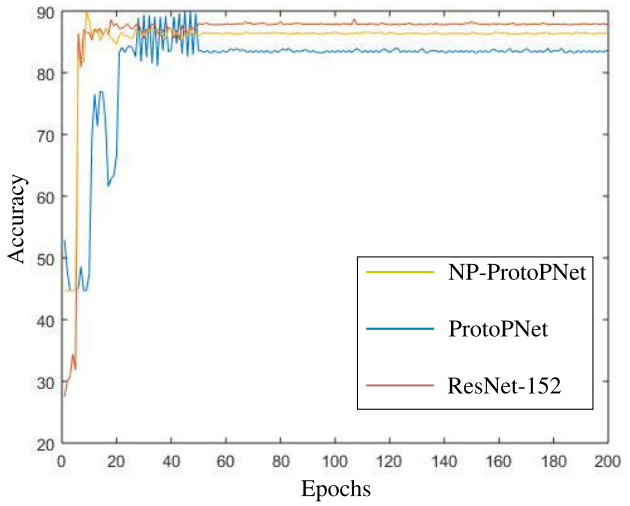

FIGURE 15. Accuracy comparison with baseline ResNet-152.

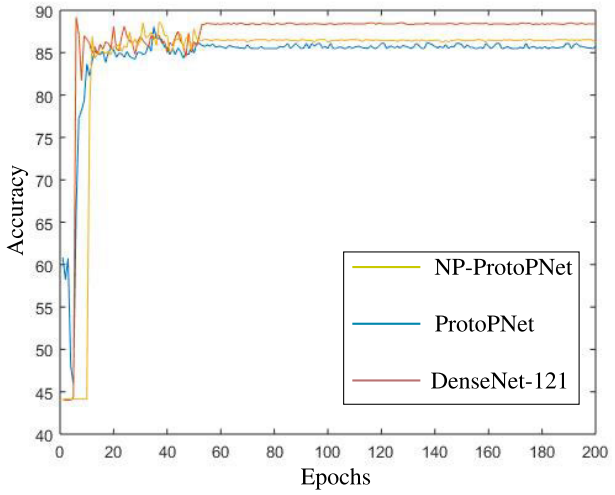

FIGURE 16. Accuracy comparison with baseline DenseNet-121.

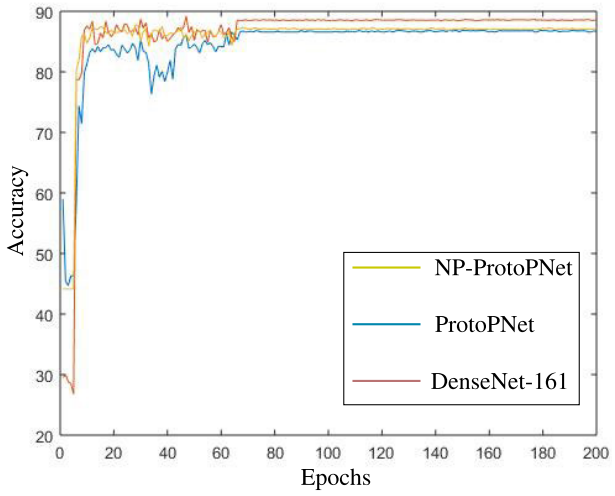

FIGURE 17. Accuracy comparison with baseline DenseNet-161.

and Baseline) stabilizes after about the same number of epochs, because they are constructed over the same baseline models. The comparison of the accuracies of the models is depicted in the figures 12-17 only for 200 epochs to make the

shape more clearer in the beginning. However, we trained and tested these models for 500 epochs.

In each of the above figures 12-17, the curve representing the accuracy given by NP-ProtoPNet is higher than the curve representing the accuracy given by ProtoPNet. Therefore, NP-ProtoPNet has better performance over ProtoPNet.

X. CONCLUSION

Our model is closely related to ProtoPNet model but strikingly different from the latter. ProtoPNet model imitate the positive reasoning of humans (*this* looks like *that*). However, our model equally uses the positive reasoning (*this* looks like *that*) and the negative reasoning (these *do not* look like those) to make predictions. The use of both positive and negative reasoning helped our model to improve its performance over ProtoPNet model. The performance of our model in the metric accuracy is better than the performance of ProtoPNet model while we experimented it on the dataset of chest *X*-ray images of Covid-19 patients [8], normal people and pneumonia patients [18]. The performance our model is on par with the non-interpretable models.

REFERENCES

[1] M. Z. C. Azemin, R. Hassan, M. I. M. Tamrin, and M. A. M. Ali, "COVID-19 deep learning prediction model using publicly available radiologist-adjudicated chest X-ray images as training data: Preliminary findings," *Int. J. Biomed. Imag.*, vol. 2020, pp. 1–7, Aug. 2020, doi: 10.1155/2020/8828855.

[2] S. Bhattacharya, P. K. R. Maddikunta, Q.-V. Pham, T. R. Gadekallu, S. R. Krishnan S, C. L. Chowdhary, M. Alazab, and M. J. Piran, "Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey," *Sustain. Cities Soc.*, vol. 65, Feb. 2021, Art. no. 102589.

[3] C. Chen, O. Li, C. Tao, A. Barnett, J. Su, and C. Rudin, "This looks like that: Deep learning for interpretable image recognition," in *Proc. 33rd Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2019, pp. 8930–8941.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[5] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, "Visualizing higher-layer features of a deep network," Dept. Comput. Sci. Oper. Res., Univ. Montreal, Montreal, QC, Canada, Tech. Rep. 1341, Jun. 2009.

[6] J. Fu, H. Zheng, and T. Mei, "Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4438–4446.

[7] J. P. Cohen, L. Dao, K. Roth, P. Morrison, Y. Bengio, A. F. Abbasi, B. Shen, H. K. Mahsa, M. Ghassemi, H. Li, and T. Duong, "Predicting COVID-19 pneumonia severity on chest X-ray with deep learning," *Cureus*, vol. 12, no. 7, p. e9448, Jul. 2020, doi: 10.7759/cureus.9448.

[8] J. P. Cohen, P Morrison, and L. Dao. *COVID-19 Image Data Collection*. Accessed: Aug. 24, 2020. [Online]. Available: https://github.com/ieee8023/covid-chestxray-dataset

[9] K. Ghiasi-Shirazi, "Generalizing the convolution operator in convolutional neural networks," *Neural Process. Lett.*, vol. 50, no. 3, pp. 2627–2646, Dec. 2019.

[10] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[13] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 599–619.

[14] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[15] S. Huang, Z. Xu, D. Tao, and Y. Zhang, "Part-stacked CNN for fine-grained visual categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1173–1182.

[16] G. Jain, D. Mittal, D. Thakur, and M. K. Mittal, "A deep learning approach to detect covid-19 coronavirus with X-ray images," *Biocybernetics Biomed. Eng.*, vol. 40, no. 4, pp. 1391–1405, Oct. 2020.

[17] R. Jain, M. Gupta, S. Taneja, and D. J. Hemanth, "Deep learning based detection and analysis of COVID-19 on chest X-ray images," *Appl. Intell.*, vol. 51, pp. 1690–1700, Oct. 22021, doi: 10.1007/s10489-020-01902-1.

[18] Kaggle. *Chest X-Ray Images (Pneumonia)*. Accessed: Aug. 24, 2020. [Online]. Available: https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge

[19] R. Kumar, R. Arora1, V. Bansal, V. Sahayasheela, H. Buckchash, J. Imran, N. Narayanan, G. N. Pandian, and B. Raman, "Accurate prediction of COVID-19 using chest X-ray images through deep feature learning model with SMOTE and machine learning classifiers," *MedRxiv*, 2020, doi: 10.1101/2020.04.13.20063461.

[20] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th Annu. Int. Conf. Mach. Learn. (ICML)*, 2009, pp. 609–616.

[21] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," in *Proc. 32nd AAAI Conf. Artif. Intell. (AAAI)*, 2018, pp. 1–8.

[22] X. Liu, T. Xia, J. Wang, Y. Yang, F. Zhou, and Y. Lin, "Fully convolutional attention networks for fine-grained recognition," 2016, *arXiv:1603.06765*. [Online]. Available: http://arxiv.org/abs/1603.06765

[23] K. Nalaie, K. Ghiasi-Shirazi, and M.-R. Akbarzadeh-T, "Efficient implementation of a generalized convolutional neural networks based on weighted Euclidean distance," in *Proc. 7th Int. Conf. Comput. Knowl. Eng. (ICCKE)*, Oct. 2017, pp. 211–216.

[24] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 3387–3395.

[25] G. T. Reddy, S. Bhattacharya, S. S. Ramakrishnan, C. L. Chowdhary, S. Hakak, R. Kaluri, and M. P. K. Reddy, "An ensemble based machine learning model for diabetic retinopathy classification," in *Proc. Int. Conf. Emerg. Trends Inf. Technol. Eng. (IC-ETITE)*, Feb. 2020, pp. 1–6.

[26] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[27] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.

[28] A. Sharma, S. Rani, and D. Gupta, "Artificial intelligence-based classification of chest X-ray images into COVID-19 and other infectious diseases," *Int. J. Biomed. Imag.*, vol. 2020, pp. 1–10, Oct. 2020, doi: 10.1155/2020/8889023.

[29] M. Simon and E. Rodner, "Neural activation constellations: Unsupervised part model discovery with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1143–1151.

[30] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *Proc. Workshop 2nd Int. Conf. Learn. Represent. (ICLR Workshop)*, 2014, pp. 1–8.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–14.

[32] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "SmoothGrad: Removing noise by adding noise," 2017, *arXiv:1706.03825*. [Online]. Available: http://arxiv.org/abs/1706.03825

[33] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, vol. 70, 2017, pp. 3319–3328.

[34] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, 2013.
[35] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proc. 33rd Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 1747–1756.
[36] R. Wexler, "When a computer program keeps you in jail: How computers are harming criminal justice," *New York Times, Sect. A*, p. 27, Jun. 13, 2017.
[37] Wikipedia. *Accuracy and Precision*. Accessed: Feb. 5, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Accuracy_and_precision
[38] Wikipedia. *Precision and Reacall*. Accessed: Feb. 5, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Precision_and_recall
[39] Wikipedia. Accessed: Feb. 5, 2021. *F-Score*. [Online]. Available: https://en.wikipedia.org/wiki/F-score
[40] Wikipedia. *Confusion Matrix*. Accessed: Feb. 5, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Confusion_matrix
[41] Wikipedia. *Sensitivity and Specificity*. Accessed: Feb. 5, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Sensitivity_and_specificity
[42] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 842–850.
[43] J. Yosinski, J. Clune, T. Fuchs, and H. Lipson, "Understanding Neural Networks through Deep Visualization," in *Proc. Deep Learn. Workshop 32nd Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 1–13.
[44] T. Zebin and S. Rezvy, "COVID-19 detection and disease progression visualization: Deep learning on chest X-rays for classification and coarse localization," *Appl. Intell.*, vol. 51, pp. 1010–1021, Sep. 2020, doi: 10.1007/s10489-020-01867-1.
[45] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 818–833.
[46] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2014, pp. 834–849.
[47] H. Zheng, J. Fu, T. Mei, and J. Luo, "Learning multi-attention convolutional neural network for fine-grained image recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5209–5217.
[48] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2921–2929.
[49] B. Zhou, Y. Sun, D. Bau, and A. Torralba, "Interpretable basis decomposition for visual explanation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 119–134.

GURMAIL SINGH received the M.Sc. degree in mathematics from Panjab University Chandigarh, India, in 2003, the M.Phil. degree in mathematics from Madurai Kamaraj University, India, in 2007, and the Ph.D. degree in algebra from the University of Regina, Canada, in 2015.

From 2007 to 2009, he was a Lecturer with the BLM Girls College, Nawanshahr, India. After qualifying UGC-Net in 2009, he served as an Assistant Professor for two years at Khalsa College, Garhshankar, India. Since 2015, he has been working as a Research Associate with the Department of Computer Science, and the Department of Mathematics and Statistics. He has coauthored 12 articles.

KIN-CHOONG YOW (Senior Member, IEEE) received the B.Eng (Elect.) degree (Hons.) from the National University of Singapore, in 1993, and the Ph.D. degree from the University of Cambridge, U.K., in 1998.

From 1999 to 2005, he served as the Sub-Dean for computer engineering with Nanyang Technological University (NTU), Singapore, and served as an Associate Dean for admissions with NTU, from 2006 to 2008. In September 2018, he joined the University of Regina, where he is currently an Associate Professor with the Faculty of Engineering and Applied Science. Prior to joining the University of Regina, he was an Associate Professor with the Gwangju Institute of Science and Technology (GIST), Republic of Korea, from 2013 to 2018; a Professor with the Shenzhen Institutes of Advanced Technology (SIAT), China, from 2012 to 2013; and an Associate Professor with NTU, from 1998 to 2013. He has published over 90 top quality international journal and conference papers. His research interests include artificial general intelligence and smart environments. Artificial general intelligence (AGI) is a higher form of machine intelligence (or artificial intelligence) where the intelligent agent (or machine) is able to successfully perform any intellectual task that a human being can. He is also a member of the APEGS and ACM. He has served as a Reviewer for a number of premier journals and conferences, including the *IEEE Wireless Communications* and the IEEE TRANSACTIONS ON EDUCATION. He is also the Editor-in-Chief of the *Journal of Advances in Information Technology (JAIT)*. He has been invited to give presentations at various scientific meetings and workshops, such as ACIRS from 2018 to 2019; ICSPIC in 2018; and ICATME in 2021.