

size	Loss	Accuracy	Learning Rate	Batch Size	Epochs
24	0.01866 0.12054	0.73234 0.76057	0.001, 0.002, 0.003	100	90
128	0.09515	0.62292	0.002	128	5
224	0.10980	0.61768	0.003	64	

Table 1: PneumoniaMNIST

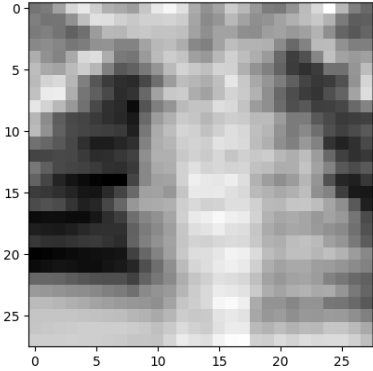
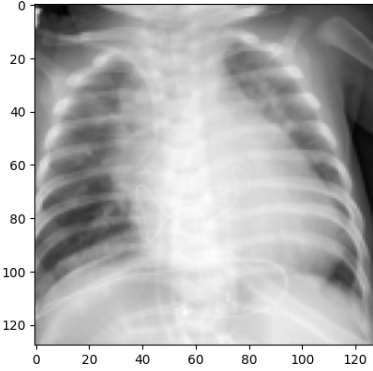

Size	Picture
24	 <p>Size: 28, Correctly Predicted Image: Ground Truth Label: 0, Predicted Label: 0</p>
128	 <p>Size: 128, Correctly Predicted Image: Ground Truth Label: 1, Predicted Label: 1</p>
224	 <p>Size: 224, Correctly Predicted Image: Ground Truth Label: 1, Predicted Label: 1</p>

Table 2: Sample Pictures

➤ Explanation Results of ML Model

Loss and Accuracy Trends

The above plot illustrates how loss and accuracy change over epochs for different learning rates. Each line represents a different learning rate.

Comparison of Learning Rates

- **Learning Rate 0.001:** The model achieves a stable accuracy around 78.10% with a decreasing trend in loss over epochs. However, it converges slowly.
- **Learning Rate 0.002:** With this learning rate, the model achieves a relatively lower accuracy compared to 0.001, fluctuating around 71% - 78%. Loss decreases quickly initially but fluctuates later.
- **Learning Rate 0.003:** This learning rate results in fluctuating accuracy around 70% - 76% and a decreasing trend in loss, albeit less stable compared to 0.001.

Discussion on Learning Rate Impact

- A lower learning rate (0.001) leads to a slower but more stable convergence with relatively higher accuracy.
- Higher learning rates (0.002 and 0.003) result in faster initial convergence but may lead to instability and fluctuation in accuracy and loss.

Observations and Recommendations

- Based on the results, a learning rate of 0.001 appears to be more suitable for this task, offering stable convergence and relatively higher accuracy.
- It's recommended to continue training with 0.001 learning rate for more epochs to observe if further improvement is possible.

Understanding the impact of hyperparameters like learning rate is crucial for optimizing model performance. In this case, a lower learning rate showed more promising results, indicating the importance of fine-tuning hyperparameters for specific tasks and datasets.

➤ **How we interpret the prediction results using ProtoPNet**

ProtoPNet interprets prediction results by identifying several parts of an image that resemble learned prototypes of a particular class. It then makes predictions based on the similarity scores between these image parts and the learned prototypes. This approach allows the model to reason transparently, providing a clear understanding of how it arrives at its predictions. By comparing image parts to learned prototypes, ProtoPNet offers a level of interpretability that is absent in other deep models.

➤ **What the explanation prototypes look like**

In ProtoPNet, the explanation prototypes are learned parts of an image that the network uses to compare with parts of a new image. These prototypes are used to reason about the new image and make predictions based on the similarity scores between the parts of the image and the learned prototypes. The network dissects the image and identifies several parts where it thinks that a particular part of the image looks similar to a prototypical part of a certain class. This reasoning process is integral to the way the network makes its predictions, and it provides a transparent reasoning process for interpretability.

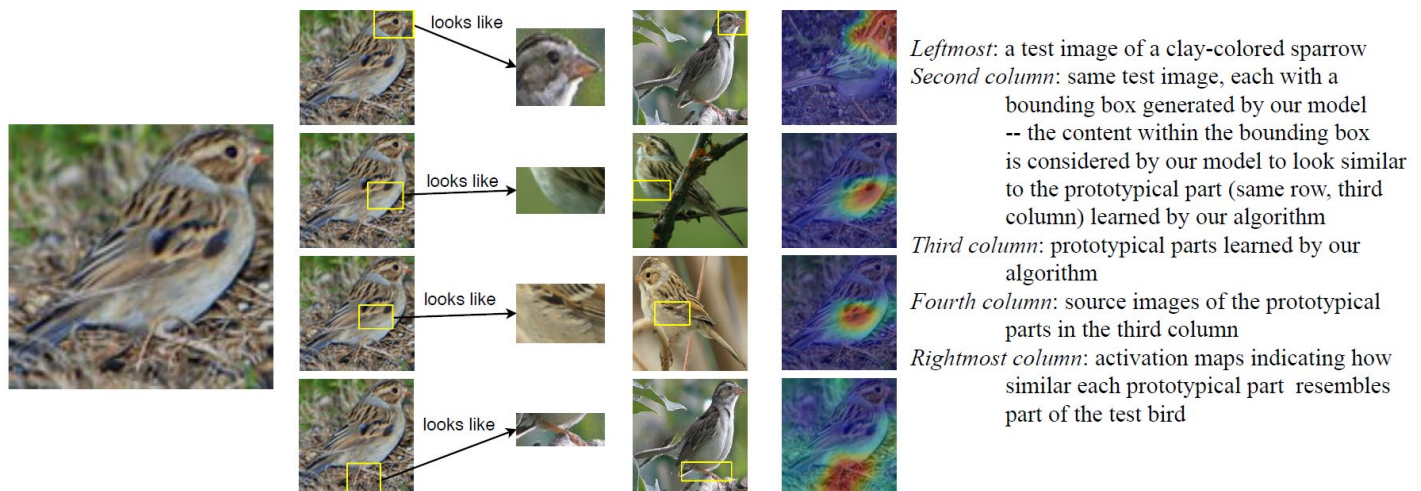


Figure 1: Image of a clay colored sparrow and how parts of it look like some learned prototypical [1]

[1] Chen, C., Li, O., Tao, C., Barnett, A. J., Su, J., & Rudin, C. (2018). This Looks Like That: Deep Learning for Interpretable Image Recognition. *ArXiv*. /abs/1806.10574