Pneumonia Detection: Comparison between ResNet50, EfficientNetB0 and DenseNet121

Nikola Bulat Alessandro Resta
{nikola.bulat, alessandro.resta}@studenti.unipd.it

Abstract

Pneumonia is a life-threatening disease that manifests in the form of either bacterial or viral infection in the lungs, causing the death of hundreds of thousands of children every year. X-rays in the chest area are mainly used for the diagnosis of this disease, which are then interpreted by radiologists. Since it is a challenging task even for human clinicians, there is a need for computer-aided diagnosis support systems. In this work, we used three pre-trained CNN architectures: ResNet50, DenseNet121, and EfficientNetB0 for transfer learning on a dataset containing 5,856 X-ray images of pediatric patients. Extensive experimentation of the models was then performed, taking into account changes such as different network-freezing settings, augmentation steps, optimizers, and learning rate schedulers. Then, an in-depth exploration of several evaluation metrics was conducted and used to compare the performance of the three models. Final results show that DenseNet121 was the top-performing model, reaching values of test accuracy, F1-score, and AUC of 93.28%, 94.71%, and 97.87%, respectively. The code is available at this link.

1. Introduction

Pneumonia is an infection caused by a virus, bacteria, or other germs. It results in inflammation in the lungs which can be life-threatening if not diagnosed in time. Moreover, pneumonia is one of the leading causes of death among children and old age people around the world. Chest X-ray is an important pneumonia diagnosis method worldwide, but expert knowledge is required to read the X-ray images carefully. Therefore, the process of pneumonia detection by reading X-ray images can be time-consuming and inaccurate [8].

For this reason, there is a pressing need for computer-aided diagnosis (CAD) systems, which can help radiologists in detecting different types of pneumonia from chest X-ray images immediately after the acquisition [13]. Among the deep learning techniques, convolutional neural networks (CNNs) have shown great promise in image classification and therefore widely adopted by the research community [10]. Some of the most recent methods take advantage of pre-trained networks and transfer learning [1], [5], [11], [13], [14], [17].

In our study, we use the Chest X-ray dataset provided by [9] for the binary classification task (pneumonia vs normal). We take advantage of transfer learning by comparing three different pre-trained architectures: ResNet50 [6], DenseNet121 [7], and EfficientNetB0 [16]. The choice of these pre-trained architectures is supported by the results from the literature (described in the following section) and our preliminary experimentation where we also tried other models such as Xception [2]. Moreover, we experiment with the following techniques in order to improve the performance: various transfer learning settings, different data augmentation techniques, changing the optimizer, learning rate schedulers, cost function adaptation, and an ensemble with the meta-model.

We assess these models on multiple levels: evaluation metrics, number of parameters, training time, and inference time. Our results suggest that unfreezing all the model parameters with strong data augmentation achieves the highest performance. In particular, DenseNet121 was able to reach the highest accuracy of 93.28% on the test set. However, our results also support that the model choice can depend on the application. For example, a small number of parameters and quick inference make the EfficientNetB0 suiting for resource-constrained scenarios.

2. Related Work

The application of deep learning models in medical image analysis has witnessed a significant rise in recent years. Extensive work has been done by researchers to exploit the capabilities of CNNs in predicting with high accuracy the presence or not of common diseases. In this section, we will explore some of the studies that have been conducted on X-ray image classification tasks, in particular for the pneumonia disease.

In [5] the authors trained and evaluated the CNN models ResNet18, Xception, InceptionV3, DenseNet121, and MobileNetV3 on an X-ray classification task for pneumonia detection. They achieved state-of-the-art results for all the tested models, and in particular with DenseNet121, which was able to attain the maximum testing accuracy (98%) and the minimum testing loss (0.064). They also built a weighted classifier to combine the predictions of the trained models, to further improve the test accuracy, and used CAMs [17] to demystify the activation map.

In the work done by [13] they trained the CNNs AlexNet, ResNet18, DenseNet201, and SqueezeNet to detect pneumonia cases in three combinations: normal vs. pneumonia, bacterial vs. viral, and normal vs. bacterial vs. viral. For all three experiments, the performance evaluation for the metrics test accuracy, precision, and recall reached their highest score for the DenseNet201 architecture.

In the analysis conducted by [1], the task of classifying pneumonia from normal cases given X-ray images was carried on using the CNN architectures: VGG16 and Xception. Results have shown that the highest test accuracy was reached by the VGG16 model (87%), while the true positive rate was higher in Xception (85%).

In the task of Covid-19 detection from X-ray images, the authors of [14] and [11] exploited the CNNs ResNet50 and EfficientNet, achieving state-of-the-art results. The output of [11] is very interesting cause it shows how lightweight networks such as EfficientNet perform comparably good with heavier architectures like ResNet50 for this task.

All the above studies made use of pre-trained models on the ImageNet dataset [3], exploiting transfer learning and fine-tuning techniques. Our work differs from the cited ones, by focusing on a specific pneumonia dataset and exploring how different freezing options impact the performance of three specific architectures, namely ResNet50, DenseNet121, and EfficientNetB0. Moreover, we tried to improve the models by exploring different data augmentation steps, learning rate scheduling, per-class-weights in the cost function, and an ensemble of the top-performing models.

3. Dataset

A total of 5,856 images were used for the dataset provided by [9]. The dataset had two divisions which are the training set and the test set. Pneumonia-infected cases were recorded as both bacterial and viral pneumonia. Importantly, the viral and bacterial co-infection cases were not part of the dataset. Chest X-ray images (anterior-posterior) were selected from retrospective cohorts of pediatric patients of one to five years old from Guangzhou Women and Children's Medical Center, Guangzhou. By then, two specialized doctors evaluated and ranked according to their understanding of the picture before they could be used to train the AI framework.

3.1. Splitting

The initial dataset lacks a designated validation set, prompting us to generate one by sampling 10% of the training images at random. To prevent any inadvertent data leakage during the sampling procedure, we initially extracted all patient IDs from the training data. Subsequently, we randomly selected 10% of these IDs and populated our validation set exclusively with images associated with those chosen IDs. As mentioned earlier, the test set was provided by [9], and contains about the 12% of all the images. A partition of our dataset is shown in table 1.

Category	Training set	Val set	Test set
Normal	1215	135	234
Pneumonia	3494	388	390
Total	4709	523	624

Table 1. Dataset training, validation and test split.

3.2. Data Augmentation

Effective neural network training relies on having a substantial amount of data. In cases of limited data, such as ours, the model's parameters may be compromised, resulting in poor generalization of learned networks. Addressing this challenge involves employing data augmentation techniques, optimizing the use of available data. By expanding the training dataset, data augmentation prevents the model from overfitting to the existing data. The specific data augmentation techniques utilized in this study are detailed in Table 2. Two augmentation types, *normal* and *strong*, were designated. Normal augmentation, involving rotation, horizontal flip, and color jitter, was employed solely for training the classifier (feature extraction). On the other hand, strong augmentation, incorporating all techniques, was applied when more than just the classifier underwent training (fine-tuning). This distinction was crucial due to the substantial increase in the parameters to be learned when unfreezing more network's layers, necessitating more data.

3.3. Handling Class Imbalance

The distribution of data for the two classes in the training set (but also in the validation and test sets) exhibited a significant imbalance, as one can see from Table 1. Such an imbalance can potentially result in training a biased model that may not effectively generalize to the under-represented class. To address this concern, we opted to leverage the WeightedRandomSampler feature offered by PyTorch [12].

Technique	Setting
Rotation	20
Horizontal Flip	0.5
Color Jitter	1
Shift Scale Rotate	0.5
Perspective Change	0.5

Table 2. Data augmentation techniques applied during training. For the "Rotation", the Setting indicates the maximum degree while it denotes the probability of occurrence for the other techniques.

This method (partially) solves the issue by employing a blend of oversampling and undersampling techniques, ensuring an equal number of samples for each class. In particular, the over-represented class, pneumonia in this case, undergoes a downsampling, while the under-represented one, i.e. normal, an upsampling.

4. Method

4.1. Models

In this section, we will introduce and discuss the architectures ResNet50, DenseNet121, and EfficientNetB0 that we adopted in this project to create the models for the pneumonia detection task.

4.1.1 ResNet50

ResNet50 is a variant of the ResNet architecture that was introduced in [6] by Microsoft Research in 2015. The residual learning framework eases the training of networks substantially deeper than those used previously by mitigating the problems of degradation and vanishing gradient.

ResNet50 contains 25.5M parameters, and it is organized into four primary sections: the convolutional layers extract features from the input image, the identity block and convolutional block process and transform these features, and the final classification is achieved through the fully connected layers. By incorporating identity mappings, certain layers within the block are skipped, empowering the network to learn the residual functions that describe the relationship between the input image and its output.

4.1.2 DenseNet121

DenseNet121 (8M parameters) is a form of the DenseNet architecture introduced in [7]. DenseNets have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

DenseNet, short for "Densely Connected Convolutional Networks," earns its name by linking each layer to every other layer in a feedforward fashion. What sets DenseNet apart from other CNN architectures are two main characteristics. Firstly, it employs a dense block structure, ensuring that each layer is connected to every other layer in a feedforward manner. Secondly, it incorporates bottleneck layers, effectively cutting down on parameters while maintaining the number of features learned by the network.

4.1.3 EfficientNetB0

The authors of [16] propose a novel approach to model scaling, where they carefully balance the network's depth, width, and resolution to achieve optimal performance. Moreover, they designed a new family of models called EfficientNets to achieve better accuracy and efficiency than previous convolutional networks.

Drawing inspiration from [15], the authors developed their baseline network by leveraging a multi-objective neural architecture search that optimizes both accuracy and FLOPS. This search provides an efficient network named EfficientNetB0 which contains 5.3M parameters.

4.2. Transfer learning and fine-tuning

The traditional approach to training Machine Learning models was to train each model from scratch for a specific task. However, two popular techniques often used to leverage pre-trained models are fine-tuning and transfer learning. In this study we made use of these techniques to overcome the small dataset size and the computational burden of training large networks, by leveraging models pre-trained on the ImageNet dataset [3].

In particular, we have four different settings for leveraging the pre-trained models: *class*, *feat1*, *feat2*, and *all*. "Class" refers to using the model as a feature extractor where we freeze all the parameters, but the fully-connected classification layer. "All" denotes the setting in which we unfreeze and train the whole model. "Feat1" and "feat2" are two versions between the described extremes, where we unfreeze some of the last layers of the network. We kept the ratio of trainable parameters constant across different pre-trained architectures in order to have a fair comparison between them. In particular, "feat1" trains around 13% of the total parameters, while "feat2" trains around 20%.

5. Experiments

5.1. Experimental Setup

We structured our experiments with the goal of assessing different pre-trained architectures, transfer learning settings, data processing techniques, and hyperparameter choices on the pneumonia detection task. For our baselines,

Model	Acc.	Prec.	Rec.	F1	AUC
EfficientNetB0-class (baseline)	86.03%	83.08%	96.92%	89.47%	95.35%
EfficientNetB0-feat1	89.53%	86.13%	98.72%	92.00%	96.49%
EfficientNetB0-feat2	89.88%	86.55%	98.91%	92.32%	96.60%
EfficientNetB0-all	91.98%	89.51%	98.46%	93.77%	97.49%
DenseNet121-class (baseline)	83.88%	80.08%	97.95%	88.12%	94.1%
DenseNet121-feat1	86.25%	82.13%	98.97%	89.77%	96.43%
DenseNet121-feat2	88.13%	84.13%	**99.23%**	91.06%	96.96%
DenseNet121-all	**93.28%**	**91.02%**	98.72%	**94.71%**	97.87%
ResNet50-class (baseline)	85.04%	83.3%	94.62%	88.6%	93.53%
ResNet50-feat1	88.37%	86.24%	96.41%	91.04%	96.01%
ResNet50-feat2	90.13%	86.91%	98.72%	92.44%	96.79%
ResNet50-all	92.25%	90.21%	**99.23%**	94.51%	**98.52%**

Table 3. Accuracy, F1-Score, AUC, Precision and Recall of all the models computed on the Test set. The post-fix in the model's name indicates up to which layer the network was unfrozen, starting from the last.

we exploited the three pre-trained architectures using "class" transfer learning (training just the classification layer). Moreover, for the baselines, we used Adam optimizer with a learning rate of 0.001, and the "normal" data augmentation (described in Section 3.2). The baselines and all the other models were trained for 20 epochs.

We tried to improve our models by experimenting with:

- different transfer learning settings ("feat1", "feat2", and "all", described in Section 4.2);
- stronger data augmentation, as more trainable parameters require more data;
- Stochastic Gradient Descent (SGD) optimizer;
- learning rate schedulers CosineAnnealingLR10 and ReduceLROnPlateau5 (available in PyTorch [12]);
- increasing the weight of the underrepresented class (normal) in the cost function;
- ensemble with a simple meta-model trained on the best model's predictions.

We implemented our experiments using the PyTorch Lightning [4] and the Google Colab platform. The link to the code is available in the abstract. The following sections are dedicated to describing how exactly we compared the models, reporting the results, and providing the interpretation.

5.2. Evaluation Metrics

Given that the dataset is unbalanced and that true positives (discovering pneumonia cases) are crucial for our application, the choice of evaluation metrics presents a key decision in our experiments. Moreover, having a consistent evaluation with the literature allows us to properly assess our results. For this reason, we decided to include the same metrics as the authors of [5], [13], and [1], i.e. accuracy, precision, recall, F1, and AUC. The combination of these metrics allows us to compare different models on multiple levels.

5.3. Results and Discussion

In this section, we report, interpret, and discuss the results we obtained. First, we give the big picture of all the experiments. Then, we provide a detailed assessment of our overall top-performing model, DenseNet121-all. Finally, we compare our models regarding multiple criteria and show how the choice of the model can depend on the particular application.

5.3.1 Results Overview

Our results are reported in Table 3. The model names are composed of the pre-trained model name followed by the transfer learning setting. "Normal" data augmentation (see Section 3.2) was used for the baselines (EfficientNetB0-class, DenseNet121-class, and ResNet50-class) while the "strong" data augmentation was used for all the other models because they require more data. The "strong" data augmentation was also tested on the baselines for fair comparison, but it didn't show improvements.

From the table, we can see that, in our experiments, training more layers constantly leads to improvements in terms of metrics. However, it's important to note that training more parameters without "strong" data augmentation didn't lead to improvements. We tested this with several models, and we reached the same or even worse performance compared to the baselines. A crucial observation is that the recall is very high for all the models, even for the ones that do not have such high accuracy. This is a consequence of the unbalanced dataset, and keeping the precision high while having almost perfect recall presents the main challenge in this domain.

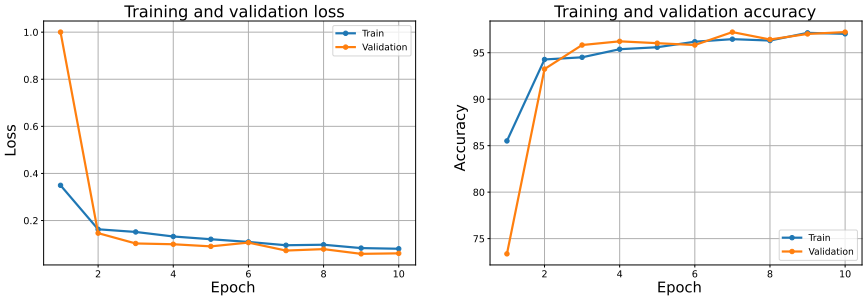

Figure 1. Learning curves over the epochs for both training and validation loss and accuracy.

Another interesting insight coming from the table values is that the proportion to which each metric changes, as less freezing is used, significantly varies among the architectures. For instance, the test accuracy difference between ResNet50-feat2 and ResNet50-class is 5.09%, for EfficientNetB0-feat2 and EfficientNetB0-class is 3.85% while for DenseNet121-feat2 and DenseNet121-class is 4.25%. This is despite the fact that feat2 is chosen such that about 20% of the network parameters are trained. The same reasoning applies to the other metrics.

Figure 2 depicts the accuracy gap between the baselines and the best models. The ResNet50-all model had the best results using the Adam optimizer and 128 batch size, while DenseNet121-all and EfficientNetB0-all excelled using SGD with CosineAnnealingLR10 and ReduceLROnPlateau5 learning rate schedulers, respectively, and 64 batch size. The most drastic improvement that we obtained is the DenseNet121 architecture where our changes lead to almost 10% improvement in accuracy.

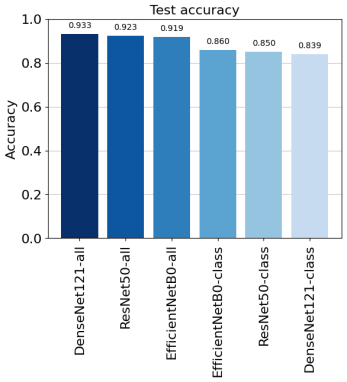

Figure 2. Accuracy comparison between the best models and the baselines

The final two changes that we experimented with didn't lead to improvements. Increasing the weight of the normal class in the cost function didn't succeed in reducing the gap between precision and accuracy. In fact, it reduced the accuracy by 1% and 5% with the factor of 3.0 and 5.0, respectively. Moreover, the ensemble in form of a simple SVM meta-model, trained on 200 hold-out instances from the training set using predictions of the three best models, also reduced the accuracy to 92.47%.

5.3.2 Best model: DenseNet121-all

The table illustrated in 3 shows that, given the metrics described in Section 5.2, our top-performing model is DenseNet121-all. In fact, the highest value was reached in three out of five metrics, namely test accuracy, precision, and F1-score. Figure 1 shows its learning curves in terms of loss and accuracy for both training and validation data. The model was trained for only 10 epochs because both training and validation loss/accuracy have reached a steady state.

To better understand the performance of DenseNet121-all, we inspect its confusion matrix shown Figure 3. This plot also shows comforting results, as the secondary diagonal contains low values compared to the main one. As the high recall suggested, the false positives (38) are significantly higher than the false negatives (5), meaning that our model tends to predict pneumonia cases more often. It is important to notice how having a higher recall in the medical domain is preferable to high precision, as this reduces the cases where patients don't undergo further visits because of a false negative, possibly risking their lives.

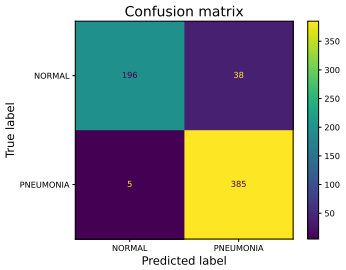

Figure 3. Confusion matrix of DenseNet121-all.

Another important metric to consider is the ROC curve, shown in figure 4. This curve correlates the false positive rate with the true positive rate for the DenseNet121-all model. As we can see from the plot, the curve is near its optimal shape, meaning that the model's performance in distinguishing between the positive and negative classes is very high. This is not surprising, as table 3 already anticipated a high AUC value.

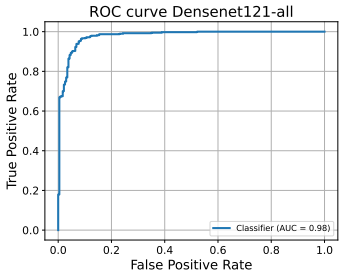

Figure 4. ROC curve of DenseNet121-all.

5.3.3 Multi-level Comparison

In the previous section we talked in details about DenseNet121-all. However, it should be noted that different models we designed might be preferable in different applications. Now, we discuss the advantages of the other two pre-trained architectures' best models, and in which scenarios they could be the best choice.

Pneumonia detection can be used in real-time applications, where there is the need for low inference time or, in other words, a fast classification given an X-ray image. To this extent, another important metric that could be used for comparison is the inference running time. Table 4 reports this value for the three best models along with training running times for completeness. Evidence shows that EfficienNetB0 reached the lowest inference time (0.0165 seconds), while ResNet50 had the lowest training time (2009 seconds). The first result is due to the fact that EfficientNetB0 is a lighter architecture compared to the others, in fact, Section 4.1.3 reported 5.3 M parameters, which is lower than 8M and 25.5 M of Densenet121 and ResNet50, respectively. Even though all three models may seem to have a very low inference time, we must take into account that we computed them using the powerful GPU T4 (provided by Google Colab). In real scenarios, edge devices with limited computational capacity are likely to be used, therefore running time gaps may be even more prominent. Thus, EfficientNetB0 might constitute a good compromise between accuracy and inference time. The second result is due to the usage of a larger batch size in the training (128 vs. 64), that allowed to speed-up the process at the expense of utilizing more GPU memory.

Model	Training time (s)	Inference time (s)
ResNet50	**2009***	0.0205
Densenet121	2919	0.0199
EfficientNetB0	2648	**0.0165**

Table 4. Training time and inference time in seconds of the top-performing models. Inference time is the time taken by the model to make a prediction on a single test sample (X-ray image).
(*) Note that here we used batch size = 128 (instead of 64).

Another interesting observation is that ResNet50-all achieved higher recall and AUC values than DenseNet121-all while still being close in other metrics. Considering how close they are in terms of performance and the lower training time of ResNet50, further experiments could be conducted in future work for a more in-depth comparison.

6. Conclusion

In this work we presented an extensive evaluation of three deep-CNN architectures, namely ResNet50, DenseNet121, and EfficientNetB0 for the pneumonia detection task. We employed transfer learning and fine-tuning techniques in order to leverage the pre-trained models on the ImageNet [3] dataset, reducing training times and improving models' overall performance. We experimented with different freezing options, data augmentation steps, optimizers, learning rate schedulers, class weights in the cost function, and ensemble using a meta-model. Then, we performed a multi-level comparison of the best-performing models discovered in our experiments, involving the metrics: test accuracy, precision, recall, F1-score, AUC, training times, and inference time. Within our setting, results show how making all the parameters learnable always improves the first five considered metrics, provided enough data using, for instance, strong augmentation. Our best model, DenseNet121-all, reached the highest values in 3 of the 5 evaluation metrics considered. In particular, it reached accuracy, F1-score, and AUC values of 93.28%, 94.71%, and 97.87% on the test set, respectively.

However, our multi-level comparison suggests that the choice of the best model might depend on the particular application. The most important lines of future work include: using a bigger dataset to further improve and confirm our results, using k-fold cross-validation for the robustness of dataset splits, incorporating a larger variety of pre-trained architectures, and designing custom architectures for specific applications. Our results show potential in supporting specialists to discover pneumonia cases and they align with the existing literature, but drawing definite conclusions requires further experimentation.

References

[1] Enes Ayan and Halil Murat Ünver. Diagnosis of pneumonia from chest x-ray images using deep learning. In *2019 Scientific Meeting on Electrical-Electronics Biomedical Engineering and Computer Science (EBBT)*, pages 1–5, 2019.

[2] François Chollet. Xception: Deep learning with depthwise separable convolutions, 2017.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[4] William Falcon and The PyTorch Lightning team. PyTorch Lightning, Mar. 2019.

[5] Mohammad Farukh Hashmi, Satyarth Katiyar, Avinash G Keskar, Neeraj Dhanraj Bokde, and Zong Woo Geem. Efficient pneumonia detection in chest xray images using deep transfer learning. *Diagnostics (Basel)*, 10(6):417, June 2020.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[7] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

[8] Amit Kumar Jaiswal, Prayag Tiwari, Sachin Kumar, Deepak Gupta, Ashish Khanna, and Joel J.P.C. Rodrigues. Identifying pneumonia in chest x-rays: A deep learning approach. *Measurement*, 145:511–518, 2019.

[9] Daniel S. Kermany, Michael Goldbaum, Wenjia Cai, Carolina C.S. Valentim, Huiying Liang, Sally L. Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, Justin Dong, Made K. Prasadha, Jacqueline Pei, Magdalene Y.L. Ting, Jie Zhu, Christina Li, Sierra Hewett, Jason Dong, Ian Ziyar, Alexander Shi, Runze Zhang, Lianghong Zheng, Rui Hou, William Shi, Xin Fu, Yaou Duan, Viet A.N. Huu, Cindy Wen, Edward D. Zhang, Charlotte L. Zhang, Oulan Li, Xiaobo Wang, Michael A. Singer, Xiaodong Sun, Jie Xu, Ali Tafreshi, M. Anthony Lewis, Huimin Xia, and Kang Zhang. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122–1131.e9, Feb. 2018.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[11] Eduardo Luz, Pedro Silva, Rodrigo Silva, Ludmila Silva, João Guimarães, Gustavo Miozzo, Gladston Moreira, and David Menotti. Towards an effective and efficient deep learning model for covid-19 patterns detection in x-ray images. *Research on Biomedical Engineering*, 38(1):149–162, Apr. 2021.

[12] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703, 2019.

[13] Tawsifur Rahman, Muhammad E. H. Chowdhury, Amith Khandakar, Khandaker R. Islam, Khandaker F. Islam, Zaid B. Mahbub, Muhammad A. Kadir, and Saad Kashem. Transfer learning with deep convolutional neural network (cnn) for pneumonia detection using chest x-ray. *Applied Sciences*, 10(9), 2020.

[14] V Sreejith and Thomas George. Detection of covid-19 from chest x-rays using resnet-50. *Journal of Physics: Conference Series*, 1937(1):012002, jun 2021.

[15] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. Mnasnet: Platform-aware neural architecture search for mobile, 2019.

[16] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019.

[17] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. *CoRR*, abs/1512.04150, 2015.