Deep Learning Project

Mohanad Arafe (40042922) · Inés Gonzalez Pepe (40095696) · Rucha Shende (40205356)

April 26, 2022

1. Introduction

The project for the Deep Learning course involved featured two Kaggle-style problems and a competition. More specifically, we were tasked with two challenges. In the first, we were given a limited arbitrary sample from the PneumoniaMNIST dataset and told to create from scratch a model to produce the best results possible with no external help (i.e. no pre-trained models or external datasets beyond the data sample provided). Each team was to submit their results to a leaderboard, in order to determine who had the best model.
The second challenge was structured similarly to the first, but this time external resources were allowed and there was no leaderboard tracking each team's progress.
Our team attempted three different approaches for Challenge 1 and two different approaches for Challenge 2 in order to find the best methods, each method having varying results. For the first challenge, we attempted using data augmentation techniques, attention modules and a Siamese network. For the second challenge, we attempted transfer learning, a faithful reproduction of a ResNet50 model with focal loss and another Siamese network.
In the first challenge, we found that data augmentation had the best results while in the second challenge, transfer learning was the most effective approach. This being said, each approach was not without its advantages and drawbacks.

2. Literature Review

2.1. ResNet & Attention Modules

Ali et al. (2020) served as an inspiration for Challenge 1 as it proposed the concept of implementing a CNN model with attention modules in order to better predict pneumonia. As the paper explains, typically when predicting pneumonia from chest X-rays, a pipeline process is followed which involves lung segmentation, feature extraction and then classification. However, the first two steps are often manual and while deep learning methods can remedy this, they often can't replace an expert's analysis of the X-ray. Therefore, the goal of this paper is to create a model that supports the clinical decision and the authors managed to not only achieve this goal, but to build a model that outperformed among others, a standard ResNet50.
The beauty of this paper is that it takes existing concepts and applies them in new ways. The model is built off a popular existing CNN model, the ResNet50 first introduced in He et al. (2016). The ResNet model introduced the concept of residual layers, i.e. where former layers are reintroduced deeper into the model as input. Ali et al. (2020) then leverages Vaswani et al. (2017)'s attention modules which is a mechanism that assigns weights to parts of the input the model should emphasize and focus on.
In brief, the model proposed utilizes a ResNet50 architecture as the backbone, with parallel attention and positional encoding modules called a Dual Attention Layer before ending with a classifier layer. The main takeaway from this paper was that using attention modules could significantly help the model learn to identify pneumonia in lungs and remove the feature extraction portion of the pipeline process. Our implementation of such a model for Challenge 1, differs from Ali et al. (2020) as we are unable to train on a full dataset. Also, the paper does not detail how it constructed the ResNet50 or the Dual Attention Layer, therefore the exact reproduction of this model is difficult and much was left to our own discretion.

2.2. Siamese Network with BCELoss

Applying deep learning techniques can be difficult when training on a small dataset. This is especially true in the medical setting since patient identity protection limits the amount of data accessible in the scientific community. Some techniques such as one-shot learning or few shot learning have been proposed to address the problem. In Prayogo et al. (2020), the authors suggest a Siamese Convolutional Network (SCN) to classify chest X-ray (CXR) pneumonia images into 3 classes: normal patients, bacterial pneumonia and viral pneumonia. The goal of a SCN is to determine if two data points are the same or different. The input of a SCN are pairs of data points that either belong to the same class or are different. A SCN's architecture consists of twin convolutional network's that are joined by a connection function that measures the similarity between both networks. The output of the function is then forwarded into a fully connected layer. In the last layer, a sigmoid function

is used to a produce similarity result.
Prayogo et al. (2020) developed a SCN using 14 layers that contain 9 convolutional layers and 5 max-pooling layers. The connection function used is the cosine distance and the fully connected network contains 2 hidden layers with dropout layers in between. The dataset used contains 5863 CXR images from patients one to five years of age at Guangzhou Women and Children's Medical Center. In the preprocessing stage, images are resized to 224x224, randomly flipped on the horizontal axis and normalized based on mean and standard deviation. Additionally, since the SCN expects pairs of images, a method was developed to generate same class and different class pairs. Finally, the model built achieved an accuracy of 80.03%.
This publication is relevant to Challenge 1 since Siamese nets are popularly used in the context of small data challenges. While Prayogo et al. (2020) classifies pneumonia cases with a large dataset, we can still attempt to reproduce their work in the context of Challenge 1. An advantage of this paper is the fact that it describes in detail how the model is constructed and what hyperparameters are used which eases the process of reproducing the work.

2.3. Data Augmentation

Data augmentation is a strategy used to increase the diversity of available data by applying realistic transformations on available data to generate new data. Mi et al. (2021) has shown that one can obtain 3% to 7% increase in classification accuracy after employing data augmentation approaches. Satoto et al. (2020) was a primary inspiration for the first challenge as it employed data augmentation techniques on a similar dataset (pneumonia COVID19 chest X-rays) with great success.
Satoto et al. (2020) implements a CNN model, but adds the novelty of implementing auto contrast to the pre-processing step and custom layers to the CNN. The auto contrast is meant to clarify the pneumonia in the images while the custom layer aims to reduce the computation time of CNNs, which are known to take awhile. The model proposed resizes and increases the intensity of the differentiation between the pneumonia and the background through auto contrast. Augmentation is then done by resizing, rotating and reflecting the original images before passing in the enlarged dataset to the CNN which predicts four possible classes; normal, COVID19, pneumonia acteria or pneumonia irus.
Hyperparameter tuning as well as experimenting with the number of custom layers was also applied in order to not only find the best possible performance, but the optimal computational time. This resulted in a CNN with 34 custom layers and an Adam optimizer

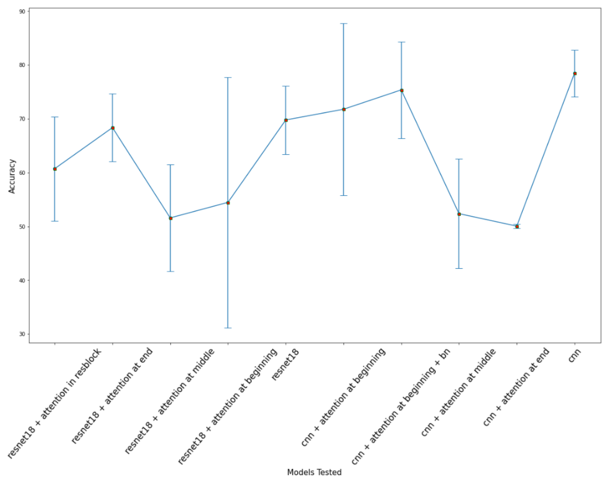


Figure 1: Challenge 1 CNN + Attention Integration Approach

that outperformed other benchmarks in the field and achieved 98.7% accuracy.
Nonetheless, while this paper is promising, the data augmentation techniques as well as the pre-processing applied can find itself especially dependent on the type of data and model used. Therefore, while Satoto et al. (2020) remains a primary inspiration, other avenues were explored. In fact, due to the similarity between MRI scans and X-rays, pre-processing techniques used in brain tumor segmentation can also be explored. In Cirillo et al. (2021), brightness augmentation and elastic deformation are used very successfully and is worth keeping in mind when completing the first challenge.

3. Methodology

3.1. Challenge 1

3.1.1 Data Augmentation

In Challenge 1, we wanted to assess if combinations of augmentation techniques previously implemented in relevant works such as 2.3 can increase a network's performance. Before jumping into the each augmentation reproduced, we wrote a custom CNN model that includes 4 convolutional layers and 4 max-pooling layers. The number of channels in each layer are 16, 32 and 64. Each convolutional filter uses a kernel size of 3, stride of 1 and padding of 1. Additionally, batch normalization is performed between each convolutional layer as well as ReLU activations. Finally, dropout is used with p=0.15 before feeding the output to a fully connected network of size 1x1x64. Figure 2 depicts the architecture used for Challenge 1.

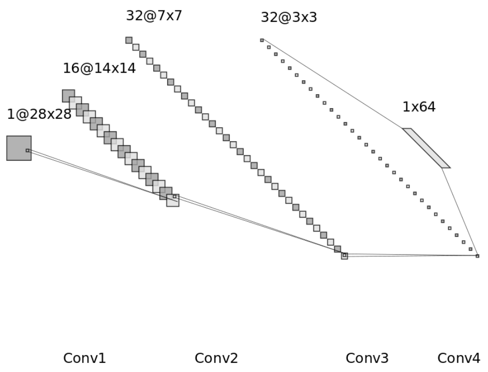


Figure 2: Model architecture

The first paper we reproduced is Satoto et al. (2020)'s work where augmentations include auto contrast, random rotations of 90°, arbitrary scaling of 0.5, random X shearing from -45° to 45° and random Y shearing of -60° to 60°. Before running our model, we wanted to measure the performance of the baseline model (initial test bed) to have a base to draw comparisons. The base model runs 50 seeds in 24.3 seconds in the CPU averaging 79.83% +- 5.72 accuracy. Using an Adam optimizer with $\gamma = 3x10^{-4}$, the model took 74 seconds to run 50 seeds and averaged 65.27% +- 6.40 accuracy. Cirillo et al. (2021) demonstrated that brightness augmentation and elastic deformation are the best augmentation techniques for brain tumor segmentation. Since MRI and CXR scans are not entirely different, it is not completely random to attempt the same transformations in the context of pneumonia detection. Therefore, we ran our models with the new set of augmentation techniques and observed positive results. The model ran in 48.3 seconds over 50 seeds scoring 82.75% +- 3.63 in accuracy.
While a 3% increase is not significant, it remains a successful result to use since it demonstrates the power of data augmentation techniques. We did however have issues when uploading this result to codalab. In the codalab platform, we got an accuracy of 79.89% which is lower than what we expected. After performing analysis, we noticed that the data in codalab is normalized. This definitely caused us problems since the results we expected were based on raw imaging data. Nonetheless, data augmentation was a successful method to use for Challenge 1.

3.1.2 Attention Mechanism

As discussed previously in 2.1, Ali et al. (2020) was a major inspiration on how to tackle the first challenge. The principal idea was to build a CNN model and introduce attention modules into it. While Ali et al. (2020) used a ResNet50, we attempted integrating attention modules as well as positional encodings into a residual network (we implemented a ResNet18 from scratch instead of ResNet50 to reduce computational cost) as well as a more classic CNN as provided in the initial test bed. Further variance from the paper involved introducing numerous attention modules and shifting their location throughout the model in order to determine which variation produced the best results.
As we can see in Figure 1, the results were fairly unimpressive. The ResNet18 model on its own performed reasonably well but whenever attention modules were added to it, performance dropped. On the other hand, not only did the standard CNN outperform the standard ResNet18, but the attention integrated CNNs also outperformed the attention integrated ResNet18s. It was a CNN with an attention layer integrated right after the model's first layer that performed the best out of all the different models in this approach, though it wasn't the most successful model out of all approaches tried for Challenge 1. Adding batch normalization and pooling to the model was explored but proved unsuccessful in conjunction with the attention modules. Furthermore, increasing the number of attention modules, hyperparameter tuning across all models as well as varying model depth was explored. It was found that more attention did not result in better performance. However, it was noted across all models, the integration of attention modules and positional encodings rendered the performance of the models highly volatile. As can be seen in Figure 1, the standard deviation is quite large which speaks to the drastic change in performance a change in test sample can produce.
In brief, while the attention integrated ResNet18 model did not perform as well as anticipated, the attention integrated CNN performed admirably when the attention was integrated at the beginning of the model, though this approach was not the most successful out of all the ones attempted for Challenge 1.

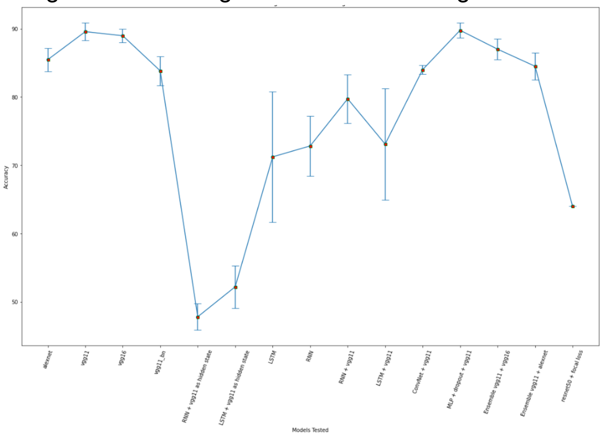
3.1.3 Siamese Network

After attempting to perform variations in the data and using an attention mechanism, we wanted to research solutions more specific to small dataset classification problems. We came across Prayogo et al. (2020)'s work which uses a Siamese network to classify pneumonia patients using CXR images. The main difference between a typical deep learning model and a Siamese network is the fact the former learns to classify an image while the latter learns a similarity function. Therefore, we attempted to reproduce this paper and see if we can get good results.
The first step was to define the convolutional network. Since Prayogo et al. (2020) uses images of size 224x224 which is 8 times larger than our images (28x28), we had to limit the size of the model we built compared to Prayogo et al. (2020)'s. Therefore, our Siamese network contained two CNN's with 5 convolutional layers and 2 max pooling layers. The output

of both networks were placed in a cosine similarity function. Finally, the output embeddings were placed in a fully connected layer with a sigmoid activation at the end representing the class probability. We implemented a data generator that created a dataset containing pairs of images with the label representing whether they are of the same class or not. The probabilities above 0.5 represent pairs of the same class while probabilities under 0.5 represent pairs of different class. As for data transformations, random horizontal flipping and normalization was performed.
The Siamese net ran 50 seeds in 46 seconds averaging 49.89% +- 1.49 accuracy which was an underwhelming result for us. The result could be due to many factors. Firstly, we used a binary cross entropy loss function and an Adam optimizer with $\gamma = 1e^{-6}$. The loss in each epoch was not decreasing nor increasing, it seemed to stay constant. Therefore, we tried to vary the model slightly by using a contrastive loss function and measuring the euclidean distance between both CNN's. This helped slightly decrease the loss per epoch but produced very similar results which were discarded. All in all, our attempt to use a Siamese network to classify pneumonia images in a few-shot learning setting failed.

3.2. Challenge 2

Figure 3: Challenge 2 Transfer Learning Results



3.2.1 Transfer Learning

Since the second challenge removes the ban on external datasets and pre-trained models, it felt logical to take advantage of this and attempt a transfer learning approach using pre-trained models available on PyTorch. We tried to leverage other MedMNIST datasets as well as a chest X-ray Kaggle dataset, but given that the initial results showed many signs of negative transfer, the idea of integrating other datasets was abandoned.
The transfer learning approach was multi-tiered as for each pre-trained model tried, three goals were kept in mind; 1) how well the pre-trained model performed as is on the PneumoniaMNIST samples, 2) how well an additional model could be integrated on top of the pre-trained model, 3) was it possible to use the pre-trained model as a feature extractor of sorts. Many pre-trained models tested failed the first goal and we were ultimately left with AlexNet and VGG, performing well, but VGG11 performing best.
After the first goal was cleared, the second and third goals were developed in parallel. For the second goal, CNNs, RNNs, LSTMs and MLP models were built on top of the VGG model. The third goal was only successfully accomplished in regards to the implementation of RNNs and LSTMs. However, as seen in Figure 3, the performance of the models was greatly varied. In general, RNNs and LSTMs performed badly, especially when using the input passed through the pre-trained VGG as a hidden state. It is interesting to note that RNNs had a tendency to perform better than LSTMs, most likely due to the fact that the models weren't particularly deep, therefore forgetting information from a couple layers ago isn't a serious concern. Nonetheless, it can be concluded that attempts to use the pre-trained VGG model as a feature extractor were unsuccessful and produced the worst models of the transfer learning approach. Meanwhile, building on top of the pre-trained VGG model had more success. Once again, RNNs and LSTMs performed worst of all. Surprisingly, a simple 2-layer MLP with dropout built on top of the VGG achieved the best results, though it only surpassed the standard VGG by 0.3%.
A second approach was made to leverage the pre-trained models tested by creating an ensemble of the best performers, though the ensemble was never able to top the individual models' performance and took a particularly long time to run. While the other models tested in this approach could take hours to run on CPU, the ensemble models tested were unable to fall within the 2 hour CPU run time constraint and since the results weren't impressive, this method was discarded.
It can be concluded that when using a pre-trained model on a limited data sample, a simple MLP or even a single linear layer can suffice in order to achieve over 85% performance in terms of accuracy with the right pre-trained model.

3.2.2 ResNet50 with Focal Loss

While this approach technically falls under transfer learning, it was inspired by Hanif Hossain et al. (2021), a particularly successful case of transfer learning applied to pneumonia classification. Hanif Hossain et al. (2021) implements a pre-trained ResNet50 model along with custom dense layers which we tried to reproduce but resulted in poor accuracy possibly because of the use of a different dataset. Improving performance through hyperparameter tuning was attempted and when we replaced the pre-trained ResNet50 with a pre-trained AlexNet model and an Adam optimizer, we obtained the better results.

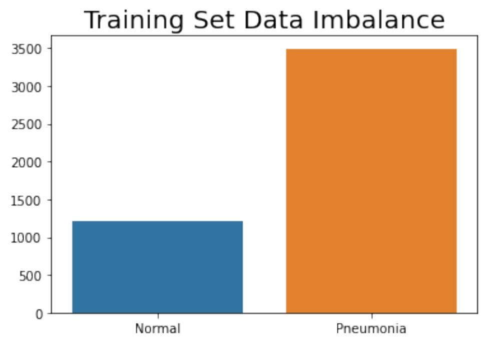


Figure 4: Data Imbalance

An important aspect of Hanif Hossain et al. (2021)'s work is using focal loss to prevent bias in an imbalanced dataset. As can be seen from the bar plot in Figure 4, the given pneumonia dataset is highly imbalanced with 1294 samples belonging to the normal category and 3495 belonging to the pneumonia category. However, since the samples fed into the network themselves are perfectly balanced, our implementation of the focal loss did not result in a performance jump. We theorize that implementing focal loss would most likely have benefited us had we had full access to the dataset. Nonetheless, its inclusion showed a rise in the accuracy by 1-2% compared to the initial test bed results of 85%. Since the focal loss is an extension of Cross-Entropy loss, we theorize that this slight improvement in performance can be due to an improved loss function. The focal loss is defined as:

$$\mathrm{FL}\left(p_t\right) = -\left(1 - p_t\right)^{\gamma} \log\left(p_t\right) \tag{1}$$

where γ is the modulating factor which smoothly adjusts its rate and reduces the loss contribution from easy examples by down-weighting them and increases the importance of correcting mis-classified examples. When $\gamma = 0$, it is simple Cross Entropy Loss, therefore through experimenting with γ we concluded that $\gamma = 2$ works best in our implementation.
To conclude, while our implementation of ResNet50 with focal loss did not live up to Hanif Hossain et al. (2021)'s results, by modifying the implementation we were able to achieve decent performance as demonstrated in Table 2. We expect that if this model is trained over an imbalanced data set, we could expect a drastic improvement in the performance.

3.2.3 Siamese Network with Triplet Loss

For our third approach for Challenge 2, we decided to once again use Siamese networks, however, this time taking advantage of the access to external datasets, unlike in Challenge 1. Further differentiating this Siamese network from the previous one is that we did not implement it using BCELoss. We followed an implementation used in Ren & Xue (2020) which uses triplet loss by extracting anchor samples and positive samples from the same category and negative samples from a different category. Ren & Xue (2020) then uses the feature embedding model to map the triplets to D-dimension Euclidean space, and the distances are measured in the resulting latent space.
We followed a similar approach when doing our own implementation, although we leveraged the TripletMarginLoss class from PyTorch and used a random sampling strategy to generate the triplets. Before training on the 10 given training samples of the pneumonia MNIST dataset, we attempted to train a joint embedding from the pneumonia MNIST dataset and the BreastMNIST dataset to make sure our model implementation was headed in the correct direction. Both the datasets were loaded with a batch size of 16 and the embedding model was trained using the pre-trained Alexnet.
However, while training our model over 20 epochs, it was observed that the losses oscillated instead of decreasing and the model never arrived at an accuracy higher than 6%. We experimented with the training time and margin values of the loss but were unable to achieve better results. We theorize that the volatility of our model and the poor performance is most likely due to some flaw in the network implementation or triplet generation that we were unable to discover. Another possible reason is that the PneumoniaMNIST and BreastMNIST datasets were too dissimilar and/or had too little data to properly ascertain similarity. Nonetheless, we believe the first source of error is the most likely one and concluded that Siamese networks were unsuccessful approaches for both Challenge 1 and Challenge 2.

4. Discussion

Table 1: Challenge 1 Model Average Performance Across 50 Seeds

Approach	Accuracy
Satoto et al. (2020) Augmentation	65.27% +- 6.40
Cirillo et al. (2021) Augmentation	82.75% +- 3.63
CNN with attention	71.78% +- 16.01
Siamese Network	49.89% +- 1.49

For Challenge 1, the main approaches attempted was 1) integrating attention modules into a CNN model, 2) augmenting and preprocessing data and 3) implementing a Siamese network.
As we can see in Table 1, the Cirillo et al. (2021) data augmentation produced the best results, though this

Table 2: Challenge 2 Model Average Performance Across 10 Seeds

Approach	Accuracy
VGG11	89.57% +- 1.30
MLP + VGG11	89.77% +- 1.10
VGG Ensemble	87.00% +- 1.50
Resnet50 with Focal Loss	64% +- 0.00
Alexnet with Focal Loss	86.27% +- 3.45
Siamese Netowrk	6.73% +- 0.35

implementation was unable to surpass the best model on the leaderboard. Theoretically, it is logical that a model with more data to train on should perform more successfully, however, it was disheartening to see the lackluster performance of the attention integrated CNN. As mentioned in 3.1.2, its performance was particularly volatile and was unable to surpass a regular CNN's performance. In hindsight, some obstacles in this model's performance most likely lie in the small dataset and experimental difficulties with integrating attention into a CNN when most sources on the subject simply integrate attention into RNNs or MLPs. To this end, the practical difficulty of integrating transformer-like networks with CNNs particularly when dealing with images (as discussed in Lecture 12 of the class) is also to be considered a source of error. Knowing what we do now, future work with attention in computer vision tasks, would pivot towards vision transformers and perhaps the Masked Autoencoders discussed in class.

Moreover, attempts were made to combine approaches by using the augmented data with an attention-integrated CNN, but the results were worse than each method separately, so efforts were then diverted to improving each method separately. This being said, the Siamese network was never combined with the other approaches due to its unique structure and its implementation late in the project's development which left us short on time.

After attempting data augmentation and attention models, we turned our focus to Prayogo et al. (2020)'s work. Our goal was to implement a solution that is more suitable to small data learning problems. The Siamese network we developed did not realize the hope we had had for it. We expected the model to be our best performer but it turned out to be the worst. As mentioned in 3.1.3, the loss of the network did not decrease which meant that the network was never learning. We tried to improve the model by eliminating the sigmoid function and instead measuring the contrastive loss between the output of both networks. This helped decrease the loss slightly but still gave similarly poor performance results.

Moving on to Challenge 2, the approaches attempted were 1) transfer learning and 2)a Resnet50 with focal loss model and 3) a siamese network.

For the transfer learning approach, we built complex models such as CNNs, RNNs or LSTMs on top of pre-trained models to attempt to improve upon the pre-trained model's performance, but these combined models did not perform as expected. While they succeeded on average in achieving results within the 70-90% range, which is respectable, they were unable to pass the performance of an unmodified VGG11 model, which made these classifiers built on top of the VGG11 to be redundant. What we learnt was that simpler is better when it comes to Challenge 2, as can be seen in Table 2. The best results that we obtained across all approaches for Challenge 2 came from a pre-trained VGG11 model with a simple MLP built on top of it with 0.7 dropout, but even a simple linear layer achieves similar results within the 89% accuracy range. Even our attempt to faithfully reproduce the Resnet50 with focal loss from Hanif Hossain et al. (2021) let us down and required significant modification to be comparable to other methods attempted. Attempting to build complex classifiers, using pre-trained models as featured extractors or using ensemble models, all failed in comparison to a single pre-trained model.

Of course, the use of pre-trained models with other models built on top of them took significantly more time to run than the models implemented in Challenge 1. For example, while the best model from Challenge 1 took approximately 50 seconds over 50 seeds on CPU, the best model and one of the simplest from Challenge 2 would take approximately 7.7 hours to run over 50 seeds on CPU, which is why for the second challenge the seeds were decreased to 10 in order to finish running in approximately 1.6 hours, thus keeping within the time constraints given to us. Furthermore, like in the previous challenge, the Siamese network was still our worst performer, but this time it was unable to even clear 10% accuracy on the test set even with the use of triplet loss.

5. Conclusion

Our project has allowed us to explore numerous approaches to dealing with a small dataset in deep learning. In developing our project, we sought to choose methods that are backed up by solid theoretical reasoning and published papers on the subject. We believe we succeeded in this regard, though our own implementation sometimes let us down. Nonetheless, this project allowed us to reproduce models and ideas we read about as well as encounter first hand the issues the authors might have faced themselves, i.e. perhaps integrating an attention layer into a CNN or finetuning a Siamese network.

All in all, we can conclude that sometimes a simpler approach is optimal, considering that our best method for Challenge 1 involved data augmentation and a CNN while for Challenge 2 it was a pre-trained VGG11 model with an MLP classifier.

References

Ali, G., Shahin, A., Elhadidi, M., & Elattar, M. (2020). Convolutional neural network with attention modules for pneumonia detection. In *2020 international conference on innovation and intelligence for informatics, computing and technologies (3ict)* (pp. 1–6).

Cirillo, M. D., Abramian, D., & Eklund, A. (2021). What is the best data augmentation for 3d brain tumor segmentation? *2021 IEEE International Conference on Image Processing (ICIP)*. doi: 10.1109/icip42928.2021.9506328

Hanif Hossain, S. M., Raju, S., & Ritahani Ismail, A. (2021). Predicting pneumonia and region detection from x-ray images using deep neural network. *arXiv e-prints*, arXiv–2101.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 770–778).

Mi, Q., Xiao, Y., Cai, Z., & Jia, X. (2021). The effectiveness of data augmentation in code readability classification. *Information and Software Technology*, *129*, 106378. doi: 10.1016/j.infsof.2020.106378

Prayogo, K. A., Suryadibrata, A., & Young, J. C. (2020). Classification of pneumonia from x-ray images using siamese convolutional network. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, *18*(3), 1302. doi: 10.12928/telkomnika.v18i3.14751

Ren, F., & Xue, S. (2020). Intention detection based on siamese neural network with triplet loss. *IEEE Access*, *8*, 82242–82254.

Satoto, B. D., Utoyo, M. I., Rulaningtyas, R., & Koendhori, E. B. (2020). Custom convolutional neural network with data augmentation to predict pneumonia covid19. In *2020 3rd international conference on biomedical engineering (ibiomed)* (pp. 71–76).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.