Determining the merging capability of Wide and Deep Network and Long-term Recurrent Convolutional Network

Akash Sindhu
Simon Fraser University
akash_sindhu@sfu.ca

Abstract

The goal of this project is to understand Wide and Deep Neural Network WDNN [1] and Long-short Recurrent Convolutional Network [2]. Based on these two models, a new model LRCN_WDNN is created by merging and tested on SNP-based and gene burden-based data. Since the LRCN_WDNN model is complex, Bayesian optimization is performed to approximate the best set of parameters. The results show that LRCN_WDNN improved model is able to achieve training accuracy of 96.52% and validation accuracy of 85.63% which is higher than LRCN. Though the accuracy is higher, the model is overfitted. To overcome this, dropout rates, kernel size, filters are increased and the number of units in dense layers are decreased in descending order. The final training accuracy was 84% and validation accuracy was 82% a significant decrease in accuracy. This project shows, LRCN_WDNN model is able to learn equivalent to LRCN. All the code can be found on Github.

1 Introduction

The goal of this project is to understand the behavior of WDNN and LRCN models when combined. Studies have shown that WDNN performed better than Random Forest whereas LRCN performed better than WDNN. Though they have their own set of advantages, we can combine them to take advantage of their advantages. LRCN_WDNN model is the name given to this new model. As the LRCN_WDNN model is complex after merging, we know that it will overfit. However, we will check it by performing analysis and see if we can remove the overfitting by introducing Bayesian optimization on LRCN_WDNN.

2 Motivation

The motivation for this project came from the genotype to phenotype slides introduced in the class lectures. Later, for the course project, M.tuberculosis paper [2] was given as a reference to study and further develop on top of it. Besides that, M.tuberculosis is a major public health threat worldwide. South Africa is one of the eight countries that accounted for two-thirds of the world's TB cases in 2019. The high HIV and Covid-19 prevalence fuel the TB epidemic in South Africa, and the country is a hot spot for drug resistance TB, with one of the highest burdens of multi-drug resistance TB in the world. This disease kills 2 million people every year among 10 million new cases. Pathogens if treated by antibiotics develop drug resistance over time. This leads to the ever-increasing number of deaths due to drug resistance against antibiotics. The number of deaths may increase by more than 10 million per year by 2050 which is higher than cancer [4].

3 Related work

In the past, many studies have been performed in this field. The two of the studies are as follows:

WDNN [1] as the name suggests has two portions in the model. These two portions learn different information from the model. The first is the 'wide' portion that acts like logistic regression and learns the genotype to phenotype relationship whereas the 'deep' portion learns the non-linear relationship in the train data and labels. Three hidden layers each with 256 rectified linear units (ReLU) [5], dropout [6], batch normalization [7], L2 regularization [9] are used in the network. Dropout and L2 regularization is used to prevent overfitting of the models to the training data. The last layer is a sigmoid layer for the multi-drug model. Adam optimizer is used and the model was trained on 100 epochs starting with random weights determined by Xavier uniform initialization.

LRCN [2] model is based on long-short term memory (LSTM) [8] and convolutional neural networks [10]. The combination of LSTM and CNN enables the model to take into account the linear order and local structure of genes in the genome. The genes were fed in the same order in which they appear in the M.tuberculosis reference genome [2].

4 Approach

4.1 Data

The data used in this project was provided in [3]. For WDNN input, the Single Nucleotide Polymorphism (SNP) data was used whereas LRCN input was gene burden-based data. SNP data is represented as a binary matrix of 7,845 isolates (in rows) by 742,620 variants (in columns) as shown in Figure 1. [2] introduced a new gene-burden based method "... [that] define one numerical feature per gene corresponding to the number of mutations in that gene in a given isolate.". This method significantly reduces the number of model parameters and gathers the knowledge of all the mutations found in the gene as shown in Figure 2. Gene burden-based features are represented as an integer matrix of 7,485 isolates (in row) by 3,960 variants (in column). Each entry indicates the number of variants in a given gene within a given isolate.

SRR1166318	0	0	0	0	0	0
ERR176810	0	0	0	0	0	0
ERR181956	0	0	0	0	0	0
SRR2100379	0	0	0	0	0	0
SRR924706	0	0	0	0	0	0
SRR1169163	0	0	0	0	0	0
ERR550791	0	0	0	0	0	0
SRR1172131	0	0	0	0	0	0

Figure 1: SNP data snapshot.

SRR1166318	0	0	1	0	0	4
ERR176810	0	0	1	0	2	3
ERR181956	0	0	1	0	1	3
SRR2100379	0	0	1	0	0	4
SRR924706	0	0	1	0	2	3
SRR1169163	0	0	1	0	0	1
ERR550791	1	0	1	0	0	1
SRR1172131	0	2	1	0	2	7

Figure 2: Gene data snapshot.

5 Architectures

In terms of architecture, project mainly focused on LRCN model and LRCN_WDNN model types.

5.1 LRCN

LRCN model is first introduced in [2] and used a new technique called the gene burden-based method to pre-process the data before sending it to the model. This technique helped the model to avoid overfitting to training data because the number of parameters decreased a lot from SNP data.

The LRCN architecture used in this project has this architecture. First, the input layer takes data with shape (None, 200, 20), connected with convolutional 1d layer that has 8 filters, kernel size as 3 and RELU activation, and same padding. Maxpooling is performed with pool size as 3 and padding as same. LSTM is added with 518 units, return sequence as False, recurrent dropout as 0.3. A dropout of 0.1 is added to avoid overfitting. A dense layer with 64 units is added. Again dropout of 0.1 is added followed by the last dense layer of 12 units (same as the number of drugs). The sigmoid function is used to make the model the multi-drug classification model. Adam as an optimizer, custom loss function, and custom accuracy function is used that ignores the missing labels, and therefore, these missing labels do not affect the network weights [2]. The highest training accuracy achieved is 84% and validation accuracy is 81.58%. Figure 4 shows that validation loss is decreasing as training loss goes down. This shows no signs of overfitting.

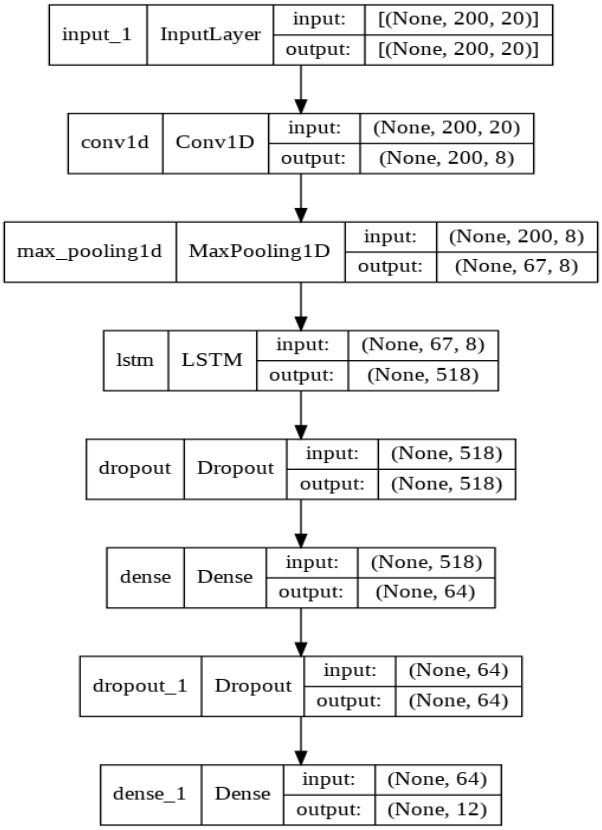
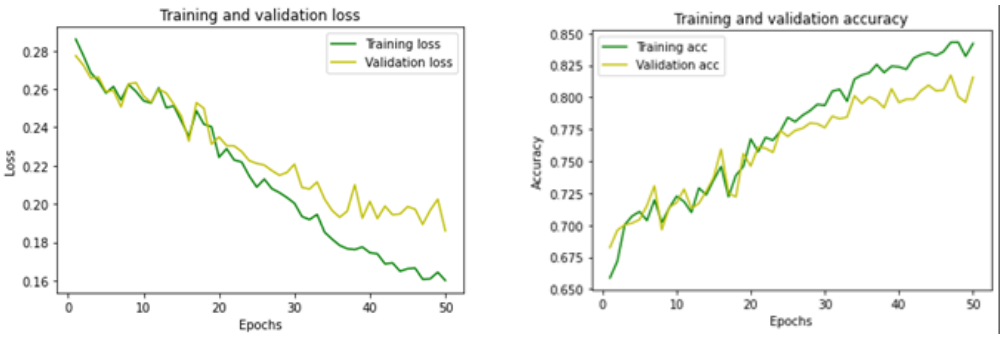


Figure 3: LRCN model architecture.

5.2 LRCN_WDNN baseline

The LRCN_WDNN model was implemented by concatenating the outputs of WDNN and LRCN. The dense layer is added after the concatenation layer and the final dense layer with 12 units (same as no. of drugs) with softmax activation function is used to make the model a multi-drug classification model. Three LRCN_WDNN model types are tested as described below:

The LRCN_WDNN baseline model was able to achieve the training accuracy of 91.15 and validation accuracy of 83.45. Figure 6 (a) shows training and validation loss and Figure 6 (b) shows training and



(a) LRCN model training and validation loss.

(b) LRCN model training and validation accuracy.

Figure 4: LRCN model.

validation accuracy. Validation loss started to increase after 15 epochs whereas training loss keeps on increasing till 50 epochs. This behavior shows that model is able to learn too closely to the training data. After looking at Figure Y, we are sure that training accuracy was increasing and validation accuracy was not increasing after 25 epochs, further proving our analysis.

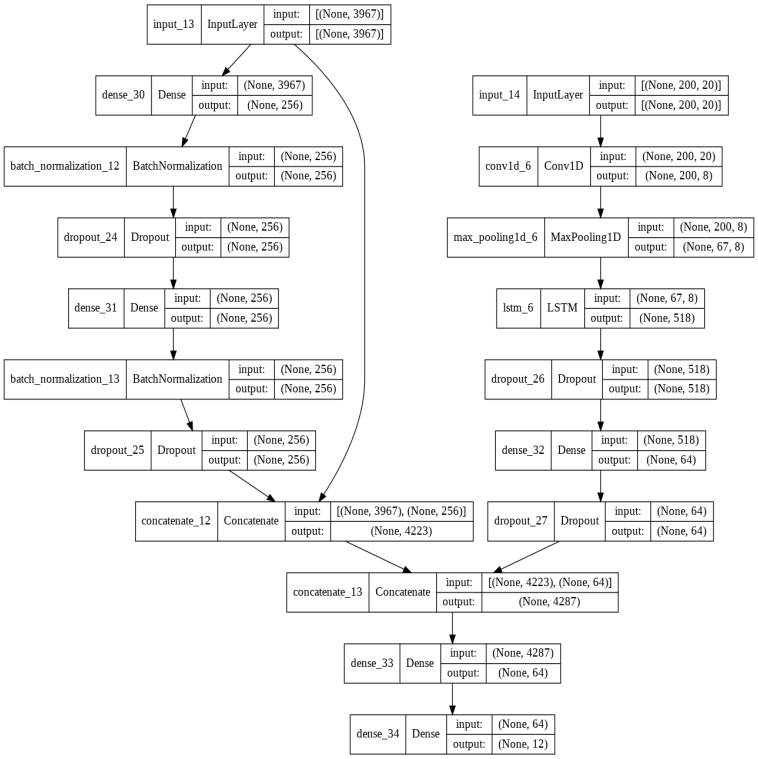
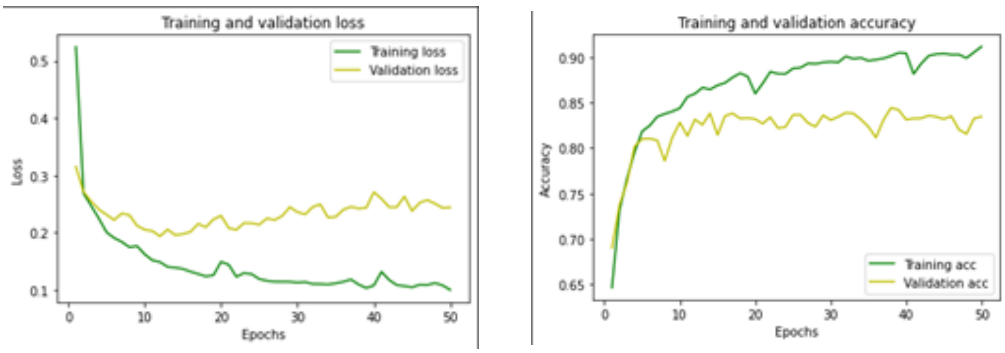


Figure 5: LRCN_WDNN baseline model architecture.

5.3 LRCN-WND improved

The LRCN_WDNN improved model with the best set of hyper-parameters that are achieved after 200 epochs shows that it is able to learn the training data even more closely than the LRCN_WDNN baseline model. Figure 8 (b) shows that the training accuracy is higher and validation accuracy has not increased. This behavior of the model shows that though the parameters are best, they helped the model to learn too much about the training data and the model started to memorize the training data instead of generalizing.



(a) LRCN_WDNN baseline model training and validation loss.

(b) LRCN_WDNN baseline model training and validation accuracy.

Figure 6: LRCN_WDNN baseline

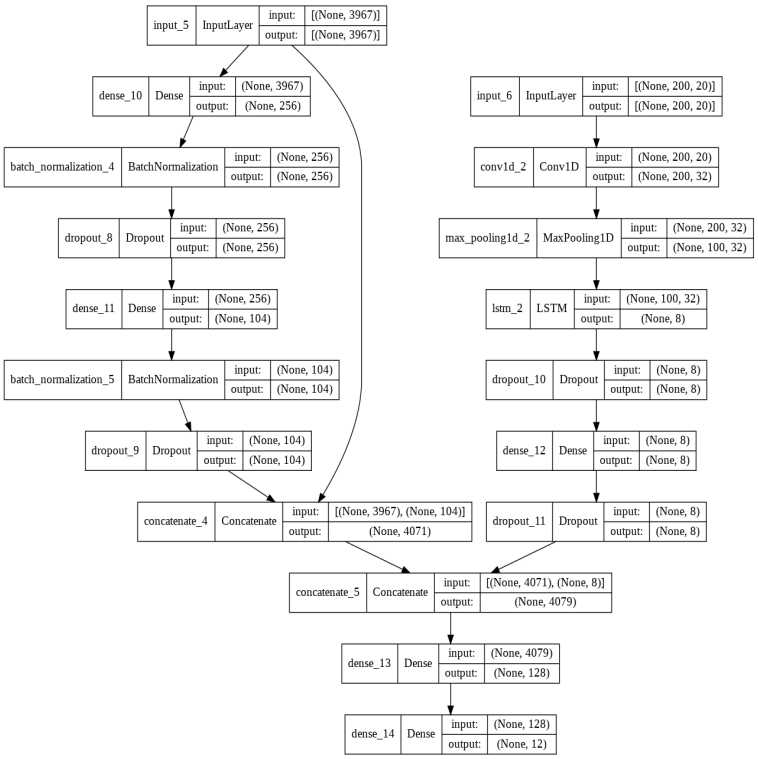


Figure 7: LRCN_WDNN improved model architecture.

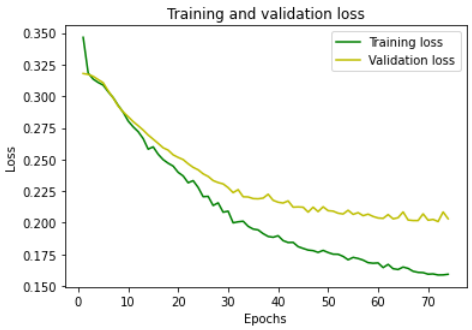
5.4 LRCN_WDNN extended

To avoid the overfitting from the LRCN_WDNN improved model, LRCN_WDNN extended model is implemented. This model has higher dropout rates in all the dropout layers. Earlier dropout rates were 0.2 for the WDNN part of the network and 0.1 for the LRCN part of the network whereas the new dropout rate increases from 0.5 at starting layer and goes to 0.8 at the last layer. This helped the model to have both the best set of parameters that help the model to learn better along with better dropout layers. The results in Figure 10 (a) and (b) of the LRCN_WDNN extended model show that the model is able to learn and generalize better.

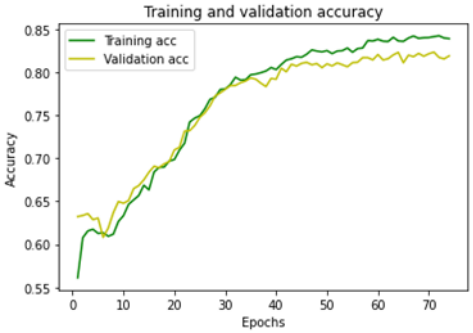
(a) LRCN_WDNN improved model training and validation loss.

(b) LRCN_WDNN improved model training and validation accuracy.

Figure 8: LRCN_WDNN improved



(a) LRCN_WDNN extended model training and validation loss.



(b) LRCN_WDNN extended model training and validation accuracy.

Figure 9: LRCN_WDNN extended

6 Optimization

Hyper-parameter optimization is performed on the LRCN_WDNN baseline model that is achieved by concatenation of LRCN and WDNN. Bayesian optimization from the Keras tuner library is used to achieve this. The parameters set for optimization are the number of units in layers, dropout rates, learning rate, kernel size, pool size, filters, and early stopping.

The trend seen in the best hyperparameters obtained from 200 trails was the number of units decreased and the dropout rate increased as we go deeper into the network. So, the final model LRCN_WDNN extended was again tested on these parameters and adjusted a little bit more by increasing the dropout rate for all the dropout layers.

7 Discussion

LRCN_WDNN baseline model was already too complex and it can be guessed that this model will overfit. Perhaps, in the experiments, we saw this model learned too closely to the training data as expected. But this, does not turn away the argument that can this model perform well? If we compare the LRCN accuracy with LRCN_WDNN extended, we can see that they are very close. It is difficult to say which one performs better considering the stochastic nature of neural networks.

However, we saw that the LRCN model can be challenged by LRCN_WDNN extended model. Considering the complexity of LRCN_WDNN, the true capabilities of the model can only be studied when more data is available for training and validation.

8 Conclusion

This project studied WDNN and LRCN models and researched their merging capabilities. The results show that we were successfully able to merge these two models. The merged model was overfitted to the training data that was further studied using hyper-parameter optimization. We found out that increasing the dropout rates for each dropout layer, filters, and kernel size and decreasing the number of units in dense layers helped reduce the complexity of the model. This reduces the overfitting and generalization of the model. This project can be further studied by changing the structure of the model.

References

[1] Chen, M. L., Doddi, A., Royer, J., Freschi, L., Schito, M., Ezewudo, M., Kohane, I. S., Beam, A., amp; Farhat, M. (2019). Beyond multidrug resistance: Leveraging rare variants with machine and statistical learning models in mycobacterium tuberculosis resistance prediction. EBioMedicine, 43, 356–369. https://doi.org/10.1016/j.ebiom.2019.04.016

[2] Safari, A. H., Sedaghat, N., Zabeti, H., Forna, A., Chindelevitch, L., amp; Libbrecht, M. (2021). Predicting drug resistance in M. tuberculosis using a long-term recurrent convolutional network. Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics. https://doi.org/10.1145/3459930.3469534

[3] Safari, A. H. (n.d.). Amirhoseinsafari/m.tuberculosis-dataset-for-drug-resistant: The SNP and gene datasets of M. tuberculosis for drug resistance prediction. GitHub. Retrieved December 18, 2021, from https://github.com/AmirHoseinSafari/M.tuberculosis-dataset-for-drug-resistant

[4] World Health Organization. (n.d.). Global tuberculosis report 2019. World Health Organization. Retrieved November 6, 2021, from https://www.who.int/publications/i/item/9789241565714.

[5] Agarap, A. F. (2019, February 7). Deep learning using rectified linear units (ReLU). arXiv.org. Retrieved December 18, 2021, from https://arxiv.org/abs/1803.08375

[6] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., amp; Salakhutdinov, R. (n.d.). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Retrieved December 18, 2021, from https://www.cs.toronto.edu/ rsalakhu/papers/srivastava14a.pdf

[7] Ioffe, S., amp; Szegedy, C. (2015, March 2). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv.org. Retrieved December 18, 2021, from https://arxiv.org/abs/1502.03167

[8] Staudemeyer, R. C., amp; Morris, E. R. (2019, September 12). Understanding LSTM – a tutorial into long short-term memory recurrent neural networks. arXiv.org. Retrieved December 18, 2021, from https://arxiv.org/abs/1909.09586

[9] Cortes, C., Mohri, M., amp; Rostamizadeh, A. (2012, May 9). L2 regularization for learning kernels. arXiv.org. Retrieved December 18, 2021, from https://arxiv.org/abs/1205.2653

[10] Yamashita, R., Nishio, M., Do, R. K. G., amp; Togashi, K. (2018, June 22). Convolutional Neural Networks: An overview and application in radiology - insights into imaging. SpringerOpen. Retrieved December 18, 2021, from https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9