Predicting drug resistance in *M. tuberculosis* using a Long-term Recurrent Convolutional Networks architecture

Amir Hosein Safari[1] Nafiseh Sedaghat[1] Alpha Forna[1] Hooman Zabeti[1]
Leonid Chindelevitch[2] *

Maxwell Libbrecht[1] * †

[1] School of Computing Science, Simon Fraser University, Burnaby, BC, V5A 1S6, Canada
[2] MRC Centre for Global Infectious Disease Analysis; and the Abdul Latif Jameel Institute for Disease and Emergency Analytics (J-IDEA), School of Public Health, Imperial College, London, W2 1NY, UK

Abstract

Drug resistance in Mycobacterium tuberculosis (MTB) may soon be a leading worldwide cause of death. One way to mitigate the risk of drug resistance is through methods that predict drug resistance in MTB using whole-genome sequencing (WGS) data. Existing machine learning methods for this task featurize the WGS data from a given bacterial isolate by defining one input feature per SNP. Here, we introduce a gene-centric method for predicting drug resistance in TB. We define one feature per gene according to the number of mutations in that gene in a give isolate. This representation greatly decreases the number of model parameters. We further propose a model that considers both gene order through a Long-term Recurrent Convolutional Network (LRCN) architecture, which combines convolutional and recurrent layers. We find that using these strategies yields a substantial, statistically-significant improvement over the state-of-the-art, and that this improvement is driven by the order of genes in the genome and their organization into operons.

1 Introduction

Drug resistance is the phenomenon whereby an infectious organism (also known as a pathogen) develops resistance to the drugs that are commonly used in its treatment [1]. In this paper, our focus is on *Mycobacterium tuberculosis* (MTB), the etiological agent of tuberculosis (TB). It is the deadliest infectious disease today and is responsible for about 2 million deaths every year among 10 million new cases [2]. The challenge of drug-resistant TB is not only a concern for low and middle-income countries, but also high-income countries [3]. The importance of drug-resistant TB, and other drug-resistant pathogens, is due to the fact that without novel antimicrobial drugs, the total deaths due to drug resistance may exceed 10 million people a year by 2050, which is higher than the current annual mortality due to cancer [4].

One way to mitigate the risk of drug resistance is to carry out a sensitivity analysis by growing the bacterial isolate in the presence of different drugs and prescribing a regimen consisting of drugs

*These authors contributed equally
†Corresponding Author. Email: maxwell_libbrecht@sfu.ca

the isolate is susceptible to. However, this approach is time-consuming and labour-intensive, so treatment is typically started before its results become available, leading to poor outcomes.

The use of whole-genome sequence (WGS) data potentially makes it possible to identify drug resistance in hours rather than days. Prior methods for identifying drug resistance from WGS data can be divided into two categories. The first category consists of catalogue methods, which involve testing the WGS data of an isolate for the presence of known mutations associated with drug resistance. These mutations are primarily single nucleotide polymorphisms (SNPs), though they can also be insertions or deletions (indels) [5]. An isolate is then said to be resistant if it has one or more of such mutations [6, 7, 8, 9, 10]. However, this approach often has poor predictive accuracy [11], especially for situations involving novel drug resistance mechanisms or resistance to untested or rarely-used drugs [12].

The second category consists of machine learning methods, which aim to to predict drug resistance by using models trained directly on WGS and drug susceptibility test (DST) data [13, 14, 15, 16, 17, 18]. The current state of the art is wide-n-deep neural network (WnD) [19] which is suitable for generic large-scale classification problems with sparse inputs such as SNP data. While existing machine learning methods achieve better accuracy than catalogue methods, there remains much room for improvement, and such improvement in accuracy would greatly improve clinical outcomes.

Existing machine learning methods featurize the WGS data from a given isolate by defining one input feature per SNP. This representation ignores the central dogma of molecular biology, that most genomic function is driven by the sequence and expression of protein-coding genes.

Here, we introduce a gene-centric method for predicting drug resistance in TB. We define one feature per gene according to the number of mutations in that gene in a give isolate. This representation greatly decreases the number of features—and therefore model parameters—relative to defining one feature per SNP. To our knowledge, while gene-based statistical tests are sometimes used for microbial genome-wide association studies, this is the first use of such a gene-based features for drug resistance prediction using ML methods. We further propose a model that considers both gene order through a Long-term Recurrent Convolutional Network (LRCN) architecture, which combines convolutional and recurrent layers. LRCNs have recently seen great success in fields such as computer vision [20, 21] but, to our knowledge, this is the first use of an LRCN in computational biology. We also introduce a multi-task approach for this problem, in which we jointly train the model to predict drug of twelve drugs using a shared architecture [19, 22].

Remarkably, we find that using these strategies yields a substantial improvement over the state-of-the-art. This improvement is statistically significant and consistent across many drugs and settings. This improvement in accuracy requires both improvements—gene-based features and LRCN model—which may explain why it has not previously been reported. Through permutation experiments, we verified that the order of genes and their organization into operons drive the model's performance. Based on these results, we expect that this gene-centric method will prove useful across many genotype to phenotype prediction problems.

The implementation of our feature preprocessing pipeline[1] and our LRCN model[2] are publicly available, as is our complete dataset[3].

[1] https://github.com/AmirHoseinSafari/Genotype-collector-and-SNP-dataset-creator
[2] https://github.com/AmirHoseinSafari/LRCN-drug-resistance
[3] https://github.com/AmirHoseinSafari/M.tuberculosis-dataset-for-drug-resistant

2 Materials and Methods

2.1 Data

To train and evaluate our method, we created a dataset by using the Pathosystems Resource Integration Center (PATRIC) [23] and the Relational Sequencing TB Data Platform (ReSeqTB) [24] datasets to collect just about 8000 isolates, together with their resistance/susceptible status (labels) for twelve drugs: rifampicin, isoniazid, pyrazinamide, ethambutol, streptomycin, amikacin, capreomycin, kanamycin, ethionamide, ciprofloxacin, moxifloxacin, and ofloxacin (Table 1) [25, 26]. The short reads containing the whole-genome sequences of these 7845 isolates are downloaded from the European Nucleotide Archive [27] and the Sequence Read Archive [28]. (Supplementary Section 1 contains a list of accession numbers of all data sets used.)

To obtain the SNP information from the raw reads, we used a standard protocol [26, 29]. We first map the raw sequence data to the reference genome using `bwa-mem` [30], then call the variants (SNPs, insertions, and deletions) for each isolate using two different established pipelines, SAMtools [31] and GATK [32]. To make the calls more robust, we used the intersection of variant calls between the two tools; less than 4% of SNPs or insertions, and deletions (indels) were removed in this step. This SNP-based dataset can be represented as a matrix of 7845 isolates by 742,620 SNPs and indels.

We created a gene-based dataset by identifying, for a given isolate, the number of mutations in each gene. We extracted the information on the boundaries corresponding to each known TB gene from MycoBrowser [33]. Of the 4,187 known TB genes, we found that 3,960 have a mutation in at least one isolate in our dataset. We represent this gene-based dataset as a feature matrix in which each row represents an isolate, each column represents a gene, and where the value in a given cell indicates the number of mutations in that gene for that isolate.

Drug	Number of labelled isolates	Number of resistant isolates
Streptomycin	5,125	2,104 (41%)
Rifampicin	7,715	2,968 (38%)
Pyrazinamide	3,858	754 (20%)
Ofloxacin	2,911	800 (27%)
Moxifloxacin	961	129 (13%)
Kanamycin	2,436	697 (27%)
Isoniazid	7,734	3,445 (45%)
Ethionamide	1,516	498 (33%)
Ethambutol	6,096	1,407 (23%)
Ciprofloxacin	443	37 (8%)
Capreomycin	1,991	552 (28%)
Amikacin	2,033	573 (28%)

Table 1: Summary of the number of isolates and the label distribution in our data.

2.2 LRCN model

Our method is based on a combination of long short-term memory (LSTM) [34] and convolutional neural network (CNN) [35] architectures, known as a Long-term Recurrent Convolutional Network (LRCN) [20]. An LSTM is an artificial recurrent neural network that can learn long-distance

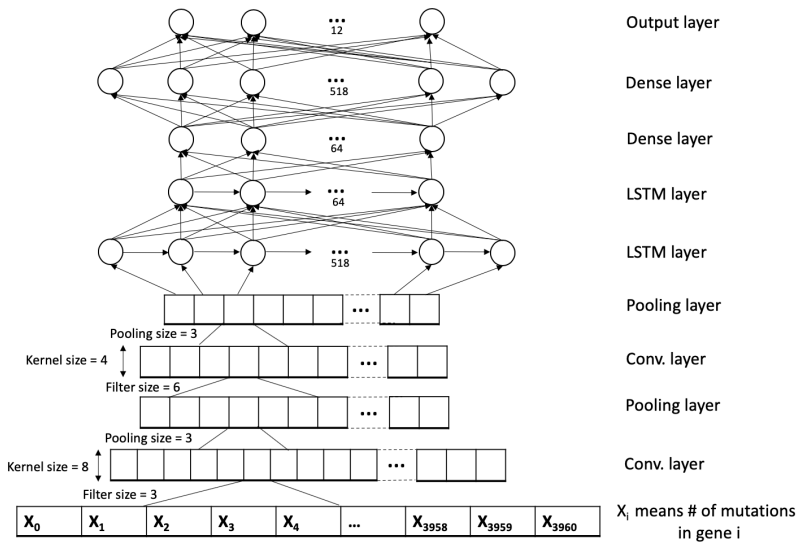


Figure 1: The architecture of our LRCN network.

dependencies by using feedback connections, which is appropriate for our situation given the multifactorial nature of the drug resistance phenotype [36]. A CNN is a feed-forward neural network designed for processing structured arrays of data, especially data where the order is important, such as for our gene data (we elaborate on this in Section 3.2). By combining a CNN with an LSTM, the model can take into account the order and locality structure of genes.

The architecture of our model is as follows. We start with CNN layers with specific filter, kernel, and pool sizes (see Section 2.5). These layers use the rectified linear unit (ReLU) activation function and the "same" padding. Next are LSTM layers, with a determined number of nodes. They use the hyperbolic tangent activation function and a recurrent dropout. At the end come dense layers. Between every pair of layers, we use a dropout with a rate of 0.1. A schematic representation of this architecture is shown in Figure 1.

We use a multi-task model, which means we predict all 12 drugs in one network. The advantage of this model is that if two drugs have highly correlated resistance status, then the patterns learned from one drug can compensate for the lack of training data on the other drug; the pairwise correlation of the status vectors for each pair of drugs is shown in the Supplementary Section 2.

The challenge of using the multi-task model in our dataset is that many isolates lack labels for some of the drugs. For this reason we replace the usual loss function with a masked loss function. The masked loss function ignores the missing labels in calculating the loss, and therefore, these missing labels do not affect the network weights. We use the binary cross-entropy as the loss function. Specifically, we use the loss function

$$\text{Loss } = \sum_{i=1}^{I} \sum_{d=1}^{D} 1(X_{i,d} \text{ is available}) H(X_{i,d}, Y_{i,d}) \tag{1}$$

where I and D are the numbers of isolates and drugs respectively, $X_{i,d}$ and $Y_{i,d}$ are the true and predicted resistance values, H is the binary cross-entropy function, and 1 is the indicator function.

2.3 Train-Validation-Test split

To evaluate our method, we split our data into training, validation, and test sets. We use two approaches for this purpose.

To obtain the results shown in Figure 3, we used a nested cross-validation approach. We split the data into 10 equal parts, and trained the model 10 times, using 8 folds for training, 1 fold for validation, and 1 for testing each time. In each run, we used Bayesian Optimization to maximize the Area Under The Receiver Operating Characteristic Curve (AUC-ROC) on the validation set, and after tuning the hyperparameters, we tested the best model on the test set. At the end of this process, we ended up with 10 different models for each method, from which we determined the mean and standard error of the AUC-ROC.

Because approach above is computationally expensive, we used un-nested cross-validation for figures other than Figure 3 We used 10% of the data as the testing set. We chose hyperparameters via 10-fold cross-validation on the remaining dataset by selecting the model that achieved the highest mean AUC-ROC. After hyperparameter tuning, we evaluated our model by testing it on the test set.

2.4 Evaluation

In order to evaluate the accuracy of our predictions, we used the AUC-ROC values. This metric plots the true positive rate (TPR) against the false positive rate (FPR) at different classification thresholds, and computes the area under the resulting curve.

Since many clinical applications require a predictor with at least a 95% specificity, we also evaluate our method using the sensitivity at 95% specificity. This metric tells us what fraction of resistant isolates can be identified as such when that their threshold for prediction is set so as to give at least 95% specificity. Several methods have no threshold that achieves 95% specificity, meaning that they cannot be used when 95% specificity is required.

2.5 Hyperparameter optimization

For tuning the hyperparameters of the model and determining the ideal number of layers of each type of network (CNN, LSTM, dense) in our model, we used Bayesian optimization. Bayesian optimization is a method suited to optimizing high-cost functions, such as hyperparameter search for deep learning model, by using a combination of a Gaussian process and Bayesian inference [37]. We used the Python implementation of Bayesian Optimization [38], with 15 iterations to tune the parameters of all the models used in this paper, with a uniform optimization approach to ensure a fair comparison.

The parameters of the optimized model (for the second approach in section 2.3) are as follows. For the CNN layer, we have 2 layers, whose filter, kernel, and pool sizes are (8,3,3) and (4,6,4), respectively. Both layers use the rectified linear unit (ReLU) activation function and the "same" padding. Regarding the LSTM layer, we have two layers with 518 and 64 nodes, respectively. Their recurrent dropout value is 0.3. For the dense layers, we have two layers with 64 and 518 nodes, respectively (Figure 1). Between every pair of layers, we use a dropout with a rate of 0.1. Also, We use Adam optimizer with a learning rate of 0.01 to train the model. The parameters chosen by the nested cross-validation approach (Section 2.3) are listed in Supplementary Section 3.

2.6 State-of-the-art methods

In order to compare our method to state-of-the-art methods, we use an LSTM, a wide-n-deep neural network (WnD) [19], a support vector machine (SVM), logistic regression (LR), and random forests (RF) [19, 13, 14, 15]. We did not compare to k-mer based approaches such as KOVER [39] as they are out of the scope of the SNP-based paradigm.

For all the comparison models we used the exact same data and procedure (e.g. Bayesian Optimization to choose the parameters) that we used for the LRCN. The chosen parameters for the SVM model are: a linear kernel and $C = 0.1$. For LR, they are $C = 0.1$ and the ℓ_2 penalty. For RF, they are: 140 estimators, a minimum sample split of 4, no bootstrapping, and a maximum depth of 50. For the LSTM network, they are 3 LSTM layers, with 355, 455, and 343 nodes, respectively, followed by 4 dense layers with 359, 219, 230, and 147 nodes, respectively. For the wide-n-deep network, they are 5 layers with 518, 518, 64, 518, and 64 nodes, respectively, and a kernel regularizer value of 0.1. The parameters chosen by the nested cross-validation approach (Section 2.3) are listed in Supplementary Section 3.

2.7 Implementation

We used the Python programming language to implement all the methods in this paper. We used the Keras [40] library for the deep neural networks, and the Scikit-learn [41] library for the standard machine learning models.

3 Results

3.1 LRCN outperforms state-of-the-art methods on a large dataset

We evaluate our LRCN method by comparing it to other state-of-the-art methods, namely, LSTM, wide-n-deep, SVM, LR, and RF (Section 2.6). We found that the LRCN method achieves a better performance, as measured by the AUC-ROC, than the state-of-the-art methods for all twelve drugs. Furthermore, The LRCN approach achieves the best area under the precision-recall curve (AUC-PR) for 11 of the 12 drugs, while on isoniazid, it performs equally well as RF (Figure 3b).

For a prediction of resistance to be usable in diagnosis, it must be made with a high specificity [42]. Therefore, we also evaluated methods according to the maximum sensitivity they achieve with a minimum of 95% specificity (Section 2.4). That is, we find the smallest threshold at which the model has 95% specificity, then we calculate the sensitivity at this threshold. If the model is unable to reach the 95% specificity for any threshold, it cannot be used for diagnosis, and therefore scores zero in this metric. The LRCN method has the best sensitivity among all the methods on 10 of the 12 drugs, although, for the two remaining drugs, amikacin and ethionamide, the wide-n-deep method achieves a slightly higher specificity. Furthermore, many standard ML approaches perform poorly on this metric, which may limit their usefulness in clinical applications (Figure 3c).

To evaluate whether the differences in model performance are robust and not due to random chance, we calculated confidence intervals for each AUC-ROC, AUC-PR, and sensitivity at 95% specificity. Since we used 10-fold cross-validation, we came up with 10 different models and performances for each method (Section 2.3). We calculated a confidence interval according to 1.96 times the standard error across these 10 performance values. Also, For all drugs other than pyrazinamide, and ciprofloxacin, the 95% confidence intervals for the LRCN do not overlap with those

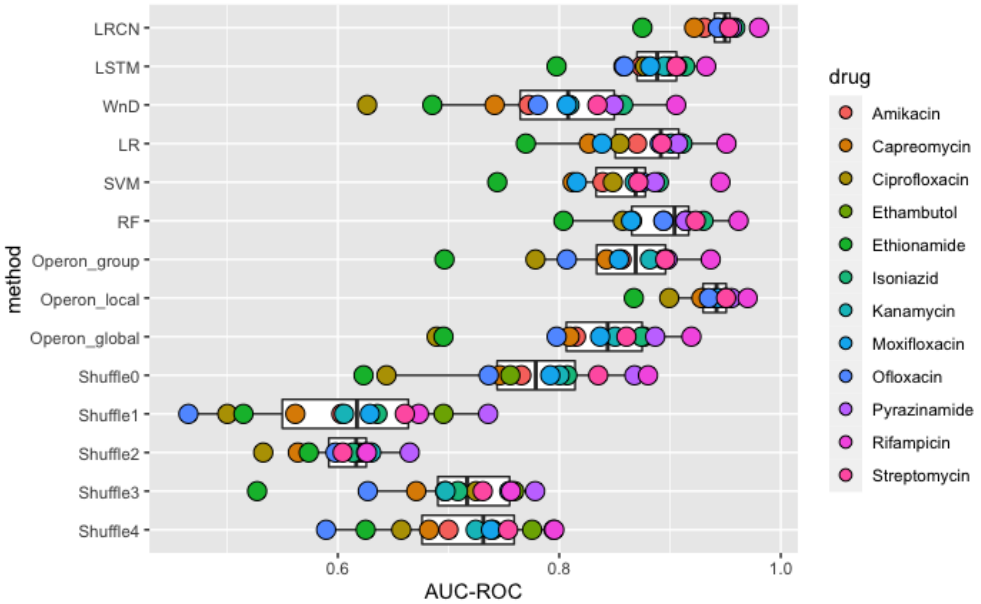


Figure 2: Summary of performance of all methods tested in this manuscript. The vertical axis represents the different methods that we used, each dot represents a drug, and the horizontal axis showing the AUC-ROC performance. The white rectangles represent the mean and standard deviation of each method. "Operon" and "Shuffle" methods refer to the results descrbied in section 3.2 and 3.4 respectively.

of the other methods, meaning that the difference in performance is statistically significant (Figure 3a). Furthermore, the approach that we used to split the data (Section 2.3) examined the LRCN model on multiple different random test sets. This gives us further confidence in the robustness of LRCN's performance.

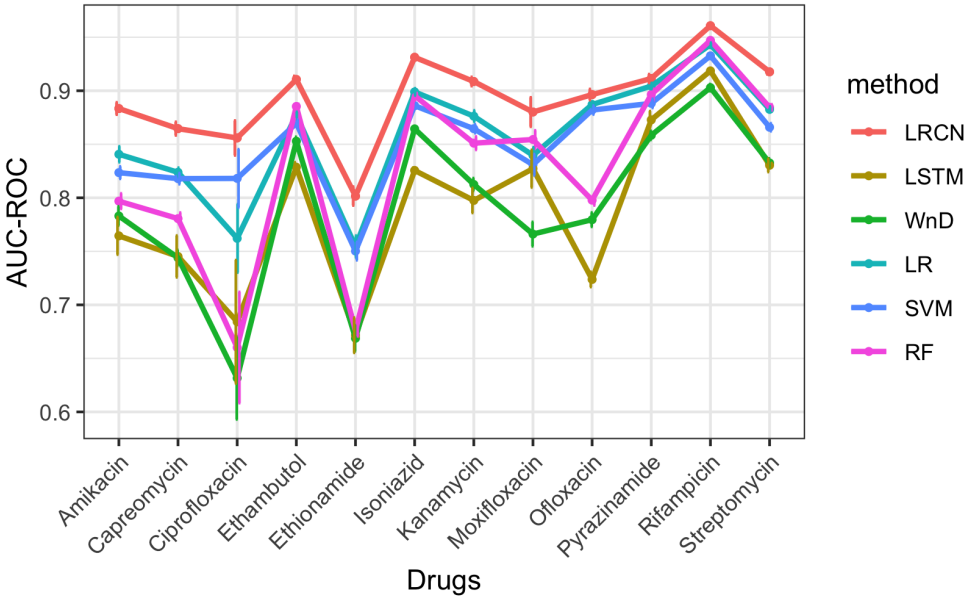
3.2 LRCN exploits gene locality

We hypothesized that the improvement in performance in LRCN is due to the importance of gene order. Indeed, the main difference between the LRCN and the other methods is the combination of CNN and LSTM layers. These layers may help the LRCN to recognize the genes' spatial relationships in the genome.

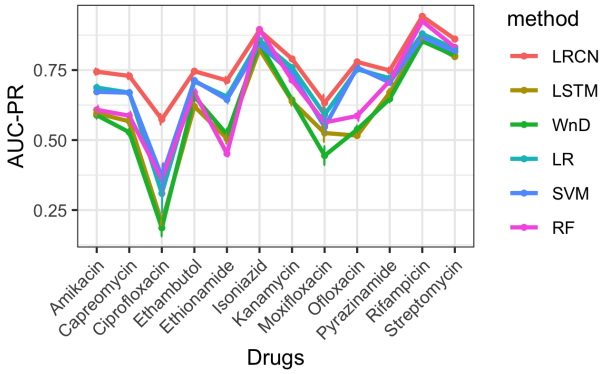
To test this hypothesis, we randomly permuted the genes to put them in a random order. We then trained and tested our method with the new data. This permutation is analogous to shuffling the pixels of an image. We shuffled the genes' order five times and for each new permutation, we carried out the entire process including Bayesian Optimization hyperparameter tuning.

As expected, we found a significant decrease in the LRCN's performance on the shuffled data, making it comparable to that of a random model. This implies the importance of genes' order in our model (Figure 4).

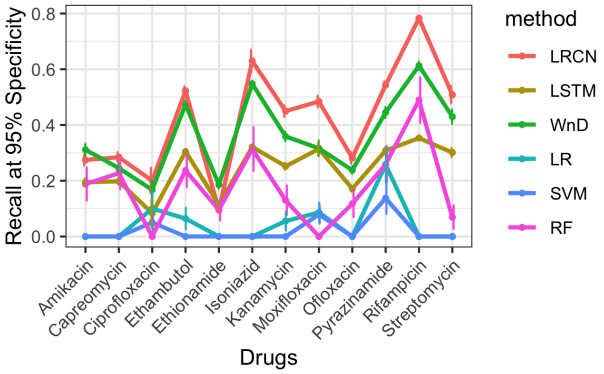
3.3 Gene-based features enable a high accuracy on our dataset



(a)

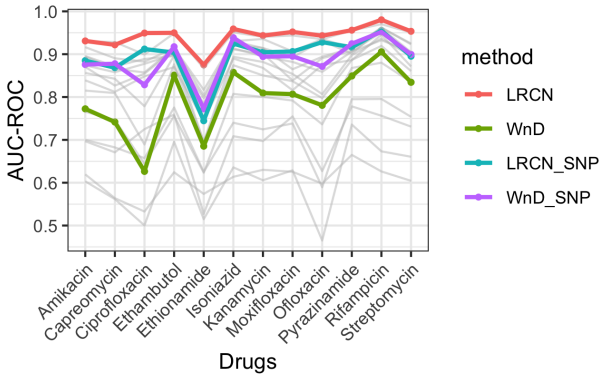


(b)

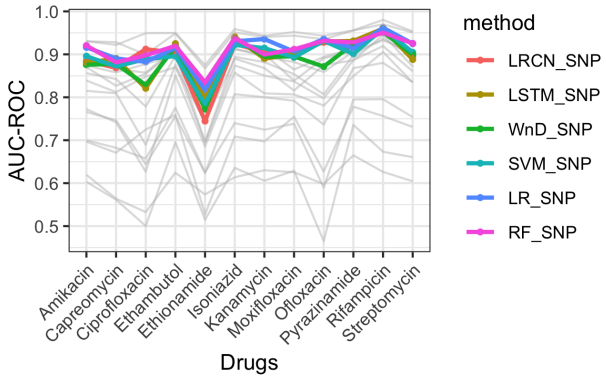


(c)

Figure 3: Comparison of state-of-the-art model performance. The error bars represent the confidence intervals of performance, calculated according using to a nested cross-validation approach (Section 2.3). Vertical axis indicates performance: (a) AUC-ROC, (b) AUC-PR and (c) sensitivity at 95% specificity.



(a) Comparison of LRCN and WnD between the gene-based and SNP-based datasets.



(b) Performance on SNP dataset

Figure 5

To evaluate the efficacy of the gene-centric approach, we applied the same methods to the SNP-based dataset (Section 2.6). Using SNP data, we trained and tested the state-of-the-art methods [13, 14, 15, 19]. We found that the LRCN approach on the SNP data performed similarly to other methods (Figure 5b). But there is a significant difference between the LRCN on SNP and LRCN on gene data (Figure 5a). This means using either just gene data or just LRCN does not achieve the best performance; both techniques together are required for improved performance.

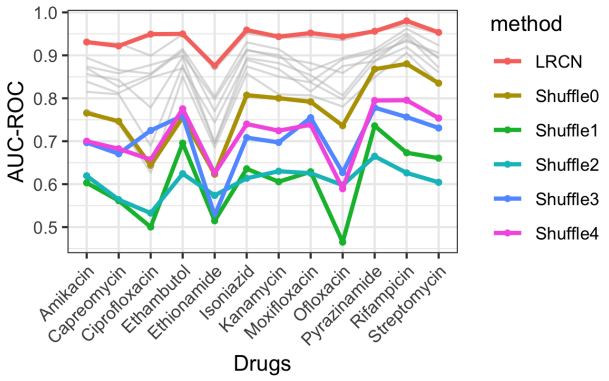


Figure 4: Comparison between the test AUC-ROC of our LRCN model using shuffled gene order. For comparison, the gray lines represent the result of other models discussed in other sections.

3.4 The organization of genes into operons drives performance

We hypothesized that the LRCN leverages the fact that neighboring genes in operons have related functions. Operons are clusters of neighbors or co-regulated genes with related functions [43]. Among the 3960 genes with at least one mutation in our data, 879 genes fall within an operon. For this experiment, we shuffled the operon genes in three different ways in order to disrupt their order, To test three hypotheses related to operons, we tested three ways of shuffling their order, described below. For each, we trained and tested our method on the resulting data.

First, we hypothesized that the order of the operon genes has a greater effect on the LRCN's results comparing to the order of non-operon genes. To test this, we randomly permuted all the genes found within operons without touching the other genes. We found a significant decrease in the accuracy of the LRCN, similar to that seen when we randomly permute all the genes (Figure 6). This suggests the importance of operon genes' order in comparison to that of non-operon genes.

Second, we assumed that as the operons' length is usually short (from 2 to 14), the order of genes within each operon should not affect the model's performance, and we should not expect a considerable difference with the original result. To test this, we individually permuted the genes within each operon, while leaving all the other genes untouched. We observed that, as hypothesized, with this approach the LRCN's performance is comparable to its performance on the original data (Figure 6).

Our third hypothesis was that if we keep the gene order fixed within each operon, but permute the locations of the operons, this should negatively affect the accuracy of the LRCN. This is similar to what would happen if we shuffled the frames of a video, which would render the videos meaningless. We have indeed found a considerable drop in the LRCN's performance in this case (Figure 6).

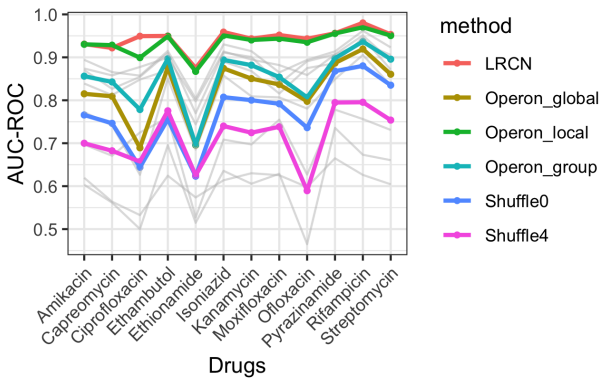


Figure 6: Comparison between of LRCN model on different types of data. The "operon_global" represents the first approach of shuffling the operons, the "operon_local" represents the second approach and the "operon_group" is for the third approach. The "Shuffle0" and the "Shuffle4" are the highest accuracies we got from section 3.2.

4 Conclusion

In this paper, we introduce a new method for predicting drug resistance in *M. tuberculosis*. Our method is a combination of CNN and LSTM layers and thus has the advantage of considering both gene sequences and their locality. We found that our method outperforms other state-of-the-art methods in a variety of settings, using three different evaluation metrics, including one suitable for clinical applications.

We observed that the LRCN achieves its highest performance if we use the number of mutations in each gene, rather than the SNP data. This number allows us to take into account the importance of individual genes while the model architecture allows us to consider their order in the genome.

We found that using CNN layers before the LSTM layers in our architecture is most beneficial for drug resistance prediction, which illustrates the importance of gene locality. In order to evaluate the importance of gene order, we used two different experiments (Section 3.2 and 3.4). Importantly, these experiments suggest that the good performance of the LRCN model on TB isolates is robust and not due to chance.

In conclusion, we introduced a novel state-of-the-art method for predicting drug resistance by using gene locality information alongside sequence information. Our results suggest that gene-centric prediction is effective in the context of an LRCN. This paper focuses on *M. tuberculosis*, but we plan to apply the LRCN approach to predict the drug resistance of other pathogens such as *Escherichia coli* in future work. Furthermore, as the lack of interpretability is a considerable disadvantage of the LRCN model as well as of most other neural network-based approaches, adding information to the LRCN method to make it more interpretable is an important direction for future research.

5 Acknowledgments

Leonid Chindelevitch acknowledges funding from the MRC Centre for Global Infectious Disease Analysis (reference MR/R015600/1), jointly funded by the UK Medical Research Council (MRC) and the UK Foreign, Commonwealth & Development Office (FCDO), under the MRC/FCDO Concordat agreement and is also part of the EDCTP2 programme supported by the European Union; and acknowledges funding by Community Jameel.

References

[1] World-Health-Organization. Antimicrobial resistance: global report on surveillance. Tech. Rep., WHO (2014).

[2] World-Health-Organization. Global tuberculosis report 2019. Tech. Rep., WHO (2019).

[3] Raviglione, M. C. & Smith, I. M. XDR tuberculosis—implications for global public health. *New England Journal of Medicine* **356**, 656–659 (2007).

[4] O'Neill, J. Antimicrobial resistance: Tackling a crisis for the health and wealth of nations. Tech. Rep., Review on Antimicrobial Resistance (2014).

[5] Carter, J. J. *et al.* Prediction of pyrazinamide resistance in mycobacterium tuberculosis using structure-based machine learning approaches. *bioRxiv* (2019). URL `https://www.biorxiv.org/content/early/2019/12/18/518142`. `https://www.biorxiv.org/content/early/2019/12/18/518142.full.pdf`.

[6] Steiner, A., Stucki, D., Coscolla, M., Borrell, S. & Gagneux, S. KvarQ: targeted and direct variant calling from fastq reads of bacterial genomes. *BMC Genomics* **15** (2014).

[7] Coll, F., McNerney, R., Preston, M. *et al.* Rapid determination of anti-tuberculosis drug resistance from whole-genome sequences. *Genome Med.* **7**, 51 (2015).

[8] Bradley, P., Gordon, N., Walker, T. *et al.* Rapid antibiotic-resistance predictions from genome sequence data for Staphylococcus aureus and Mycobacterium tuberculosis. *Nature Communications* **6** (2015).

[9] Iwai, H., Kato-Miyazawa, M., Kirikae, T. & Miyoshi-Akiyama, T. CASTB (the comprehensive analysis server for the Mycobacterium tuberculosis complex): A publicly accessible web server for epidemiological analyses, drug-resistance prediction and phylogenetic comparison of clinical isolates. *Tuberculosis* 843–844 (2015).

[10] Feuerriegel, S. *et al.* PhyResSE: a web tool delineating Mycobacterium tuberculosis antibiotic resistance and lineage from whole-genome sequencing data. *Journal of Clinical Microbiology* **53**, 1908–1914 (2015).

[11] Schleusener, V., Köser, C., Beckert, P. *et al.* Mycobacterium tuberculosis resistance prediction and lineage classification from genome sequencing: comparison of automated analysis tools. *Scientific Reports* **7** (2017).

[12] Kadura, S. *et al.* Systematic review of mutations associated with resistance to the new and repurposed Mycobacterium tuberculosis drugs bedaquiline, clofazimine, linezolid, delamanid and pretomanid. *Journal of Antimicrobial Chemotherapy* (2020). Dkaa136.

[13] Zabeti, H. *et al.* An interpretable classification method for predicting drug resistance in m. tuberculosis. *bioRxiv* (2020).

[14] Yang, Y. *et al.* Machine learning for classifying tuberculosis drug-resistance from DNA sequencing data. *Bioinformatics* **34**, 1666–1671 (2017).

[15] Kouchaki, S. *et al.* Application of machine learning techniques to tuberculosis drug resistance analysis. *Bioinformatics* **35**, 2276–2282 (2018).

[16] Yang, Y. *et al.* Machine learning for classifying tuberculosis drug-resistance from DNA sequencing data. *Bioinformatics* **34**, 1666–1671 (2018).

[17] Drăghici, S. & Potter, R. B. Predicting HIV drug resistance with neural networks. *Bioinformatics* **19**, 98–107 (2003).

[18] Arango-Argoty, G. *et al.* DeepARG: a deep learning approach for predicting antibiotic resistance genes from metagenomic data. *Microbiome* **6**, 1–15 (2018).

[19] Chen, M. L. *et al.* Deep learning predicts tuberculosis drug resistance status from whole-genome sequencing data. *bioRxiv* (2018).

[20] Donahue, J. *et al.* Long-term recurrent convolutional networks for visual recognition and description (2015).

[21] McLaughlin, N., del Rincon, J. M. & Miller, P. Recurrent convolutional network for video-based person re-identification (2016).

[22] Yuan, H., Paskov, I., Paskov, H., González, A. J. & Leslie, C. S. Multitask learning improves prediction of cancer drug sensitivity. *Scientific Reports* **6**, 31619 (2016).

[23] Wattam, A. R. *et al.* PATRIC, the bacterial bioinformatics database and analysis resource. *Nucleic acids research* **42**, D581–D591 (2014).

[24] Starks, A. M. *et al.* Collaborative effort for a centralized worldwide tuberculosis relational sequencing data platform. *Clinical Infectious Diseases* **61**, S141–S146 (2015).

[25] Ngo, T.-M. & Teo, Y.-Y. Genomic prediction of tuberculosis drug-resistance: benchmarking existing databases and prediction algorithms. *BMC bioinformatics* **20**, 68 (2019).

[26] Deelder, W. *et al.* Machine learning predicts accurately Mycobacterium tuberculosis drug resistance from whole genome sequencing data. *Frontiers in Genetics* **10**, 922 (2019).

[27] Leinonen, R. *et al.* The European Nucleotide Archive. *Nucleic Acids Research* **39**, D28–31 (2011).

[28] Leinonen, R., Sugawara, H. & Shumway, M. The Sequence Read Archive **39**, D19–21 (2011).

[29] Coll, F. *et al.* A robust snp barcode for typing mycobacterium tuberculosis complex strains. *Nature Communications* (2014).

[30] Li, H. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *arXiv* (2013).

[31] Li, H. *et al.* The sequence alignment/map format and samtools. *Bioinformatics* **25**, 2078–2079 (2009).

[32] Poplin, R. *et al.* Scaling accurate genetic variant discovery to tens of thousands of samples. *bioRxiv* (2017).

[33] Kapopoulou, A., Lew, J. & Cole, S. The mycobrowser portal: a comprehensive and manually annotated resource for mycobacterial genomes. tuberculosis .

[34] Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780 (1997).

[35] Schmidhuber, J. Deep learning in neural networks: An overview. *CoRR* **abs/1404.7828** (2014). `1404.7828`.

[36] Trauner, A., Borrell, S., Reither, K. & Gagneux, S. Evolution of drug resistance in tuberculosis: Recent progress and implications for diagnosis and therapy. *Drugs* (2014).

[37] Snoek, J., Larochelle, H. & Adams, R. P. Practical bayesian optimization of machine learning algorithms 2951–2959 (2012).

[38] Nogueira, F. Bayesian Optimization: Open source constrained global optimization tool for Python (2014–). URL `https://github.com/fmfn/BayesianOptimization`.

[39] Drouin, A. *et al.* Interpretable genotype-to-phenotype classifiers with performance guarantees. *Scientific Reports* **9**, 4071 (2019).

[40] Chollet, F. *et al.* Keras. `https://github.com/fchollet/keras` (2015).

[41] Pedregosa, F. *et al.* Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011).

[42] Su, M., Satola, S. W. & Read, T. D. Genome-based prediction of bacterial antibiotic resistance. *Journal of clinical microbiology* **57**, e01405–18 (2019). 30381421[pmid].

[43] Osbourn, A. E. & Field, B. Operons .