IMPLEMENTING HUBBLE.2D6 - A TOOL FOR CYP2D6 HAPLOTYPE FUNCTION PREDICTION

TECHNICAL REPORT

Coleton Annett
Department of Computer Science
University of Victoria
Victoria, BC V8P 5C2
cannett@uvic.ca

August 11, 2021

ABSTRACT

CYP2D6 is a highly polymorphic pharmacogene responsible for the metabolism of up to 25% of all clinically prescribed drugs today. As such, the prediction of CYP2D6 phenotype in an individual is vital to personalised treatment, helping doctors determine correct dosages and evaluate risk of drug toxicity. However, due to the polymorphic nature of the gene, more and more haplotypes are being sequenced that require manual curation to determine their function. Hubble.2D6, developed by Gregory McInnes, et. al., is a Convolutional Neural Network (CNN) that predicts haplotype function with high accuracy, addressing the increasing need for automated CYP2D6 function prediction. This paper will discuss an attempt to implement Hubble.2D6 from scratch, basing methodologies, architecture, and evaluation outlined in the original paper. Despite data availability issues, the implementation achieves approximately 80% accuracy on the test set, and 60% agreement rate with the original Hubble.2D6 tool on both curated and uncurated CYP2D6 haplotype function predictions.

1 Introduction

1.1 CYP2D6 and Hubble.2d6

CYP2D6 is a gene whose produced proteins metabolize more than 20% of all clinically used drugs. However, it is highly susceptible to variations, and as a result is difficult for researchers to manually determine its function. In general, CYP2D6 haplotype function is classified as having either no function, decreased function, or normal function. This function is correlated to its metabolic activity and gives indication of an individual's ability to metabolize such drugs Gardiner and Begg [2006].

Hubble.2D6 is a CNN based tool that utilises transfer learning - a technique common in the image recognition domain - to predict haplotype function from annotated sequence data at a high degree of accuracy, by pre-training on both simulated data, as well as curated CYP2D6 microsomal data.

1.2 Problem statement and motivation

This report details an attempt to implement Hubble.2D6 from scratch, following techniques from the original paper *Transfer learning enables prediction of CYP2D6 haplotype function*, presented by Gregory McInnes, et. al.McInnes et al. [2019]. The purpose is to not only investigate the reproducibility of the tool on data provided, but to understand the benefit of transfer learning within the context of genetic analysis.

2 Methodology

2.1 Pre-processing

To follow the original tool as closely as possible, the implementation requires the reading and processing of data in Variant Call Format (VCF) into a format suitable for prediction and training within the model. Specifically, we require the transformation of the data in VCF into an $n \times 13$ matrix where the first 4 columns of each row correspond to the one-hot encoded vector of each nucleotide - A, C, T, and G, and the other 9 columns are one-hot encoded functional annotations. Robust functional annotations require the use of tools such as AnnoVar, VEP, and BCFtools, which was out of scope for this project, and therefore pre-computed annotation embeddings available from Hubble.2D6's repo were used. This leads to some encoded variants not having functional annotations which is a consideration made in the final evaluation.

Additionally, certain pre-training steps during the implementation of Hubble.2D6 require the encoding of diplotype sequences (two paired haplotype sequences). Since transfer learning requires similar input dimensions between training steps, the one-hot encoded matrix needed to encode two haplotype sequences within the same input matrix. The solution, which is also used in the original implementation, is to stack both encoded haplotype matrices on top of each other, separated by a 32 x 13 null matrix.

Custom scripts were written, largely based on similar scripts from the original author, to handle this required pre-processing of the data, with the scripts being extended to work on diplotypes encoded in VCF files for the first pre-training step. All scripts used for data pre-processing can be found in this project's repo.

2.2 Model architecture

Hubble.2D6 is a CNN model trained through the process of transfer learning, a method where weights from previously trained networks are transferred into a new network that trains on a different but related problem to the previous network. As a result, the implementation of Hubble.2D6 requires 3 very similar network architectures. However, as discussed later, one pre-training step had to be skipped for data availability reasons, and so only 2 network architectures will be presented.

Architectures for the 2 remaining networks follow the Basset network, as specified by the original paper. Figure 1 outlines the individual layers of both networks, as well as the activations, outputs, etc. and which layer weights were required to be transferred to the final network.

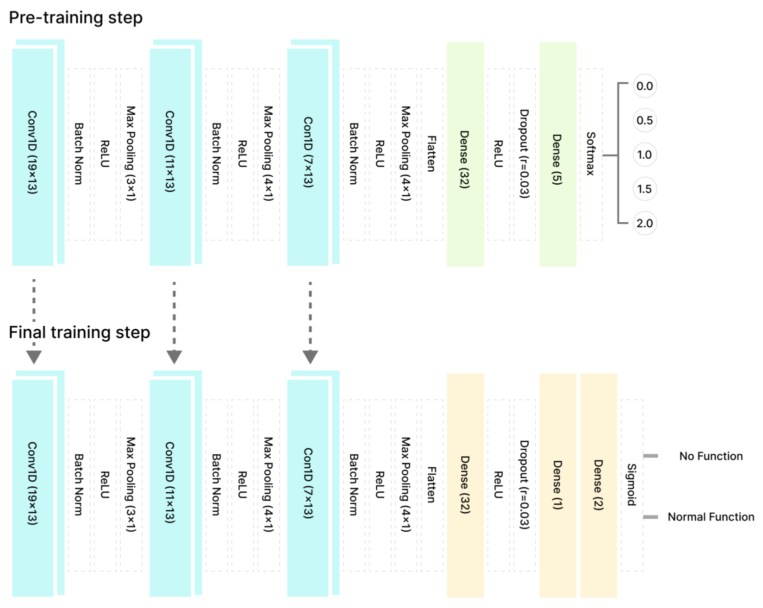


Figure 1: Overview of train model architectures

Model 1 is used for a multiclass classification task to predict CYP2D6 diplotype activity scores, whereas Model 2 is an ordinal regression task with 2 outputs, and thus the fully-connected section of both networks are separately configured and initialised. In the original paper, the model corresponding to the 2nd pre-training step also transfers one of it's fully-connected layers' weights to the final network.

The output of the final model is then compared against cut-off scores to encode into 1 of the 3 haplotype functions, which is used as the final prediction.

Each model takes a single $n \times 14868 \times 13$ tensor as its input, where n is the number of samples in the dataset.

2.3 Datasets and features

Hubble.2D6 requires the use of 3 independent datasets for each network trained in its pre-training steps. All data used within the implementation of Hubble.2d6 outlined in this report was taken from resources given within the original paper, or the official Hubble.2d6 GitHub repository.

Training data for the first training step involves a total of 60,000 simulated diplotypes. 50,000 were provisioned for training, with another 10,000 held out as the testing set. The script that produced this simulated data is provided in a repository by Gregory McInnes.

Data for the final training step, concerned with haplotype function prediction, contains all documented star alleles and their sub-alleles from PharmVar. Note that the dataset used in the implementation is the exact one provided by the paper (version 4.11.1, downloaded 10/25/2019), and therefore more recent curated functions may have been posted that were not analysed within this iteration. From this dataset, all star alleles with uncurated function were omitted from the training process and were instead used in final evaluation of the model, as well as a medium of prediction comparison with the Hubble.2d6 tool.

It is important to note that the dataset used to train the second pre-trained network contains curated liver microsome data that was provided to the original author by both St. Jude Children's Research Hospital and the University of Washington. As a result, this data is not freely available and I was unable to use it in the implementation, meaning that the entire second training step had to be skipped. This is addressed more in the Implementation section.

Lastly, labels corresponding to each star alleles' metabolic function for the final step's training data were computed with a custom script, pulling from an aggregated list made by the original authors that corresponds with the documented function of each CYP2D6 star allele from PharmVar. These labels, as well as all custom scripts can be found in the source repository.

3 Implementation and results

3.1 Training on simulated data

3.1.1 Process

The first step in the implementation was to train a CNN model to predict simulated CYP2D6 diplotype activity scores. These activity scores are classified as either 0.0, 0.5, 1.0, 1.5, or 2.0 and are computed as the sum of the activity scores of each haplotype that makes up the diplotype. Although roughly 250,000 simulated diplotypes were available from the original paper, a total of 60,000 were selected for training and evaluation.

Due to the large number of samples, as well as the large matrices built once each sample was transformed into it's one-hot encoded form, it was infeasible to directly store the encoded data, and as such all data was passed into the model via a Python generator. This generator encoded each batch file one at a time and yielded samples that would then be passed into the constructed network for training.

3.1.2 Results

The first model predicts the activity score of 10,000 samples within the testing set with 85% accuracy. In comparison, the original paper specifies that their first pre-training step resulted in a 100% test set accuracy. This difference could be attributed to the missing functional annotations for some of the variants during encoding, or better tuning of hyperparameters.

3.2 Training on liver microsome data

3.2.1 Process

Based on the original paper, the weights from the convolution layers of the previous model are then transferred to a new network, which is trained to predict the metabolic activity of curated human liver microsome samples. This is a regression problem, and so a new architecture for the full-connected layers is necessary.

3.2.2 Complications

Data used by the original implementation was provided by the facilities that curated the microsome samples (St. Jude Children's Research Hospital and University of Washington. This made it inaccessible for me without contacting them, or the original author. As a result, this pre-training step was skipped, and weights from the first step were instead transferred directly to the final step's network. The weights from one of the fully-connected layers of the second model were supposed to be transferred into the final model after training meaning both layers in the final model must be randomly initialised.

3.3 Training on curated CYP2D6 function data

3.3.1 Process

The final step in implementing Hubble.2d6 is training a network that will predict the ultimate target label, in this case two outputs: probability of no function, and probability of normal function. As highlighted earlier, the actual tool predicts haplotype function, which has 3 separate classifications, and so a scoring system is used to encode the labels into a format suitable for the network to predict these two probabilities.

The scoring system presented in the original paper contains two binary labels. The first label corresponds to no function alleles and is set to 0 if an allele has no function, and 1 otherwise. The second score, the normal score, is then set to 1 if an allele has normal function, and 0 otherwise. According to the paper, this system is more beneficial compared to a default classification problem, as it allows the network to learn the closer similarity between no function and decreased function alleles compared to normal function alleles. Cutoff values pulled from the original Hubble.2d6 implementation are then used as thresholds that decode scores into one of the three function classes.

Similarly with the first pre-training step, samples from the dataset used in the final training are encoded into a one-hot-encoded matrix with functional annotations pulled from pre-computed embedded annotations. As a result, they occur the same penalty as before where information is inherently being lost as a result of incomplete functional annotations.

Within the full dataset, 71 star alleles have uncurated, or unknown, function and are omitted from the dataset to be used for evaluation and comparison metrics with Hubble.2d6 later on. Once uncurated haplotypes are removed, 31 star alleles and their respective suballeles are selected for training, and 24 for the test set. Stratified 5-fold cross validation is used over the training set to select hyperparameters, with 10% of the training set being taken as the validation set. Note, some hyperparameters such as convolution kernel size, or stride number, were obtained from the saved model architectures within the Hubble.2d6 repo, since this network being trained should be the final model and thus as close to the original as possible. Once hyperparameters are selected, 7 separate networks are trained on the full training set and averaged together to form an ensemble model.

As stated previously, the output of the model consists of two scores. This was done to frame the prediction task as an ordinal regression problem rather than a classification problem. Mean Squared Error (MSE) is regarded as one of the better loss functions to use for this problem Cheng [2007] and therefore is what was employed in the final implementation.

AUROC was used as the metric during training to identify the effectiveness of each network during training based on the output scores, whereas the final ensemble model evaluation metric involves an accuracy measure on the predicted functional class. For accuracy to be informative, the output scores must be decoded into their respective functional classes. The cutoff values used were 0.42 for the no function score, and 0.76 for the normal score probabilities, as specified by the original paper.

Note, during training, the inherited weights of each network are frozen for the majority of the training, and then unfrozen and "fine-tuned" with a very low learning rate to better align them to the ordinal regression task. This was done since the original paper specifically mentions fine-tuning the transferred weights at each iteration rather than each network completely inheriting the weights without tweaking.

3.3.2 Results

Results are given on two sets of alleles, one containing each star allele and their respective suballeles, and one that includes only star allele samples. This is for expressiveness sake as the original paper only reports accuracy on star allele function predictions, and not any inaccuracies in sub allele prediction. Since they did not include how they deal with suballele prediction scores, it was decided within this paper to report prediction scores for both sets. Table 1 displays the relative accuracies, and reports a 79% evaluation accuracy on the test set containing only star allele predictions. This is compared to the reported 88% test accuracy of Hubble.2d6's final training.

Table 1: Final training evaluation

	Star alleles only	Star alleles + suballeles
Training set	83.87%	79.87%
Testing set	79.16%	70.17%

Additionally, it is important to note that suballele prediction accuracy is affected by some star alleles having more suballeles than others, therefore weighting some functional classes over others.

In addition to raw accuracy metrics, Figure 2 displays heatmaps of predicted functions against true functions on the star allele set, showing the implementation's tendency to predict no function alleles as having decreased function instead.

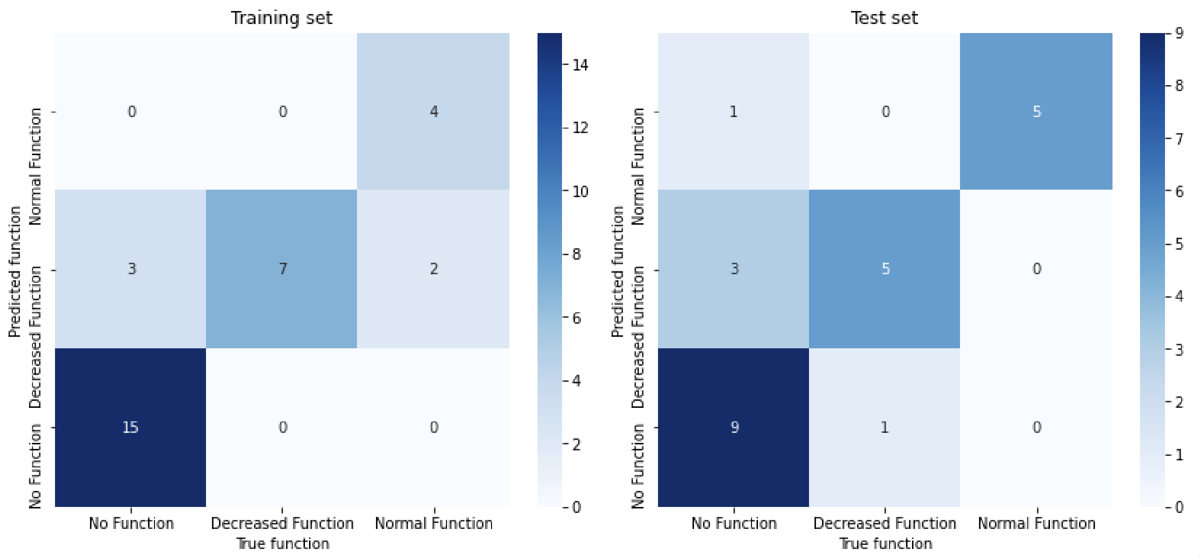
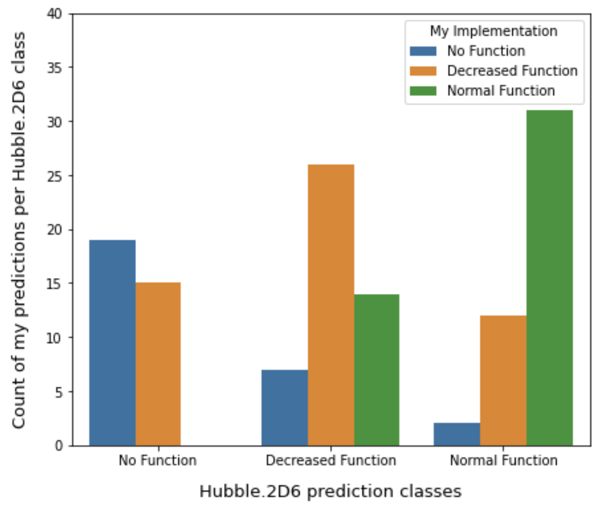


Figure 2: Heatmaps of predictions against true CYP2D6 star allele functions

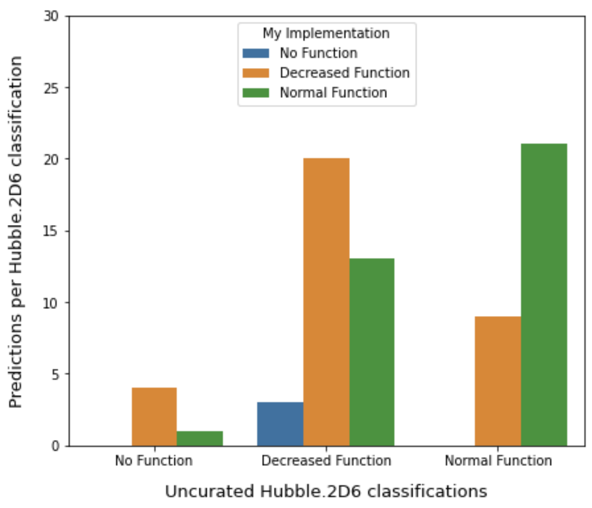
3.4 Comparison to Hubble.2D6

Here, we compare the final model predictions on CYP2D6 star alleles against the original Hubble.2d6 tool. Two comparison set are considered: Hubble.2d6's official predictions released with the paper on the documented star alleles and suballeles from PharmVar, and generated Hubble.2d6 predictions on just the uncurated function alleles within that dataset. Through comparison, we can gain a better realisation of the effect the omission of a second training step and our choice of certain hyperparameters had on final predictions vs Hubble.2d6. On comparing predictions from the final implementation against these sets, we see that our model has approximately 60% agreement in predictions with Hubble.2d6 on both sets.

From the comparisons against the full allele dataset (Figure 3a), we see that my implementation classifies a majority of normal functional haplotypes similarly to Hubble.2d6. However, with no function and decreased function star alleles, the implementation disagrees almost half the time on both with Hubble.2d6's predictions. Additionally, we see the trend seen in test set evaluation, where lower function alleles are over-classified as being of a higher function class than Hubble.2d6 predicts.



(a) Curated prediction comparison



(b) Uncurated prediction comparison

Figure 3: Heatmaps of predictions against true CYP2D6 star allele functions

On uncurated function alleles (Figure 3b), the implementation attains a 57% agreement rate with Hubble.2d6 predictions. With this set, we see the same over-classification trend noticeable in other evaluation observations.

4 Discussion and next steps

From the results and comparison against Hubble.2d6 predictions, it is clear that the missing pre-training step on liver microsome samples as well as other considerations impacted the model's ability to predict on the final dataset. For example, both fully-connected layers in every network of the final ensemble model are randomly initialised, as opposed to the deeper layer having inherited weights from the second pre-training step, as the original paper intended. In particular, fine-tuning the convolution layers on a pure regression task may have aided its ability in the ordinal regression task within the final training. An email has been sent to the original author of Hubble.2d6, asking about the availability of the data for the second step.

5 Conclusion

Due to data availability and other concerns, a true implementation of Hubble.2d6 cound not be achieved. However, the attempted implementation was able to generate predictions that agree with the official tool at a rate of 60%, and predict to 80% accuracy the star allele functions contained in the evaluation set.

6 Data availability

Data used in this implementation of Hubble.2d6 can be found both from the original paper McInnes et al. [2019], and the GitHub repository paired with this report: `https://github.com/Locrian24/seng474-term-project`

References

Sharon J. Gardiner and Evan J. Begg. Pharmacogenetics, drug-metabolizing enzymes, and clinical practice. *Pharmacological Reviews*, 58(3):521–590, 2006. ISSN 0031-6997. doi:10.1124/pr.58.3.6. URL `https://pharmrev.aspetjournals.org/content/58/3/521`.

Gregory McInnes, Rachel Dalton, Katrin Sangkuhl, Michelle Whirl-Carrillo, Seung-been Lee, Russ B. Altman, and Erica L. Woodahl. Hubble2d6: A deep learning approach for predicting drug metabolic activity. *bioRxiv*, 2019. doi:10.1101/684357. URL `https://www.biorxiv.org/content/early/2019/06/27/684357`.

Jianlin Cheng. A neural network approach to ordinal regression. *CoRR*, abs/0704.1028, 2007. URL `http://arxiv.org/abs/0704.1028`.