RESEARCH

Mechanisms of Action Prediction - Kaggle Competition

Heitor Baldo[1*], Tayna da Silva Fiúza[2], Renata Lilian Dantas Cavalcante[2] and Hellen Paula Valerio[3]

*Correspondence:
hbaldo@ime.usp.br
[1] Institute of Mathematics and Statistics, University of São Paulo, São Paulo, Brazil
Full list of author information is available at the end of the article

Abstract

Background: Due to the availability of large amounts of biological data and well-established machine learning algorithms, the incorporation of computational pipelines in drug discovery has become increasingly common and useful. These pipelines may guide not only the development of new drugs but also provide new biological hypotheses and a better understanding of biochemical mechanisms underlying common diseases.

Result: Here we focused particularly on optimizing two machine learning algorithms to predict the mechanisms of action of drugs, using as input information about how these drugs remodel gene expression and affect cellular viability.

Conclusion: By optimizing the loss function of the models, we found that a deep neural network produces a minimum loss of 0.0182, outperforming a logistic regression model, which generates a minimum loss of 0.0268.

Keywords: Mechanisms of Action; Machine Learning; Neural Networks

Introduction

The drug discovery process aims at developing molecules that will alter a disease state by modulating the function of specific biomolecules. In this sense, the concept of mechanism of action (MoA) refers to the biochemical interaction between a ligand and a biological target, which will produce a pharmacological effect [1].

Pre-clinical drug discovery pipelines have relied mainly on screenings of cellular viability and gene expression remodeling after exposure of cells to potentially bioactive compounds. Cell viability assays are performed to determine the percentage of live cells after treatment with drugs [2], whereas gene expression is the process by which the information contained in the genetic material is decoded into a functional gene product. In general, the expression of a given gene is followed by the production of the corresponding protein, thus creating a functional protein [3]. This process takes place from a piece of cellular machinery that binds the DNA strand and reads the gene sequence in groups of three nucleotide bases (codon). Each codon corresponds to one of the 20 essential amino acids that are the raw material for the construction of new proteins. From the understanding of these concepts, we can then infer how the drugs are interacting with their targets in a given biological system [4].

Microarray and RNA-sequencing techniques have become important in revealing new biochemical mechanisms that play roles in health and disease, by allowing the

characterization of gene expression changes triggered by different types of perturbations [5]. In the last 5 years, research efforts have aimed at cataloging the effects of chemical and genetic perturbations in several cell types, providing data for the analysis of MoAs [6, 7]. These efforts have led to the development of more efficient, faster, and more systematic methods for the identification and prediction of MoAs.

In this context, Subramanian et al. [6] developed the Connectivity Map (CMap) platform, aiming to provide large-scale data for the identification of gene expression signatures triggered by genetic or chemical perturbations [8]. The database allows the comparison of new gene expression profiles with cataloged profiles, advancing the generation of new biological hypothesis and the investigation of MoAs of new drugs [6]. To this date, the database offers 1.3 million profiles that originated from 27.000 types of perturbations. Half of the database is composed of well-known and curated cellular signatures, whereas the other half is composed of perturbations triggered by unknown and potentially bioactive small molecules [6].

This paper will focus on the competition "Mechanisms of Action (MoA) Prediction: Can you improve the algorithm that classifies drugs based on their biological activity?", hosted by Kaggle (https://www.kaggle.com/c/lish-moa/). This challenge was proposed by the Connectivity Map initiative together with the Laboratory for Innovation Science at Harvard, aiming to advance drug development through the improvement of the algorithms used for the prediction of biological activity.

Materials and methods

Gene expression and cell viability data

Our study relies on a dataset containing gene expression and cellular viability measurements. The gene expression data encompasses the measurement of 772 transcripts, which have been shown to be representative of the whole transcriptome [6]. Drugs were tested at two different doses (D1 and D2) and measurements of transcript levels and cellular viability were taken at three different periods after exposure to the perturbation (24, 48 and 72 hours). Besides, information about a control group, treated only with the solvent used as a vehicle for the drugs, was included in the dataset. The control group was referred to in the dataset as *ctl_vehicle* in opposition to *cp_vehicle* and, per definition, does not have MoA annotations. Altogether, the features include the gene expression and cellular viability data, but also the type of treatment (*ctl_vehicle* or *cp_vehicle*), drug dosage and time after exposure to the perturbation. For the machine learning analysis, the type of treatment and drug dosage were coded as dummy variables.

MoA annotations for the training set were codified as a binary matrix, where 1 specifies if a drug acts through a given MoA and 0 if not.

As usual, the dataset was split into training (85%), test (15%) and validation sets, where the validation set corresponds to 30% of the training set. The training and validation sets were used to train the learning algorithms and to estimate the loss, used here as the main metric of the model's performance, as required by the competition.

Principal components analysis

As we have a high dimensional database, with many features, in order to reduce its dimensionality and avoid some problems in the modeling process, we decided to perform a Principal Component Analysis (PCA).

PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated. In other words, it converts, via orthogonalization, a set of correlated variables in a set of mutually uncorrelated variables, called principal components, that have maximal variance [9].

Logistic regression

The generalization of binary logistic regression is the *multiple logistic regression*, in which we have more than two classes, and this generalization is explained in [10] and [11]. We will not emphasize this generalization for more than two classes because it is a multi-class model, and again, our problem is a *multi-label problem*.

A suitable approach to logistic regression in multi-label cases is to consider it as a neural network *with no hidden layers*. Thus, given a set of M classes, we have a network with M output neurons and the *sigmoid function* as activation function. This way of looking at the model is especially useful for cases where each class is very *unbalanced*, since the estimation of an independent model for each class could compromise the model.

Fitting the model as a neural network with no hidden layers, the parameters are estimated together. Thus, even if the classes are unbalanced, as a whole, the response matrix is not excessively sparse, so we avoid numerical problems in the estimation, which is usually done using stochastic gradient descent.

Furthermore, it is possible to think about the regularization parameters of the logistic regression aiming the treatment of a possible *overfitting* case. In this work, the logistic regression model via the neural network was tested using the L1 regularization, in which the term $\lambda||.||_1$ is added to the loss function, where λ is the penalty and $||.||_1$ is the $\mathcal{L}^1$ norm, therefore a large λ leads us to a greater number of coefficients with zero values.

Deep neural network

We optimized a deep neural network consisting of an input layer, 3 hidden layers and an output layer. The input layer receives as input 21.948 vectors of features, and the output layer is composed of 206 neurons, corresponding to the MoA classes. The three hidden layers were optimized to consist of 512, 256 and 128 neurons, respectively. Considering that neurons are connected by weights throughout the layers of the network, we aimed at optimizing the weights to minimize the loss in the training set.

To complement the general structure of the network, we used a batch normalization function to normalize the input distribution of each layer to a standard Gaussian distribution, reducing internal covariate shift as described in more detail by [12]. Also, to prevent overfitting, we used a regularization technique known as dropout, which consists of discarding randomly selected neurons during training [13]. Dropout was applied to the first hidden layer.

To optimize the best hyperparameters, we used a grid search algorithm [14]. The grid search scanned for batch size, number of epochs, type of optimizer, type of kernel initialization, type of activation function, number of neurons and dropout rate. The best model performance was achieved for a batch size of 2.300, 300 epochs, a dropout rate of 0.3 and 512 neurons per layer. The Adam optimizer, an algorithm for gradient-based optimization [15], was used for back-propagation of the predicted errors through the weights of the network since it generated the best model performance in the grid search.

A rectified activation function was used in the three hidden layers of the network, returning for each element of the input vectors:

$$f(x) = max(0, x) \tag{1}$$

This type of activation function has been shown to improve the performance of the learning algorithm. The output layer uses a sigmoid function, given that the input vectors should be transformed into a binary output (0 or 1), coding for the presence or absence of MoA annotations for each drug. Accordingly, during model compilation, minimization of the predicted errors was achieved by using binary cross entropy as a loss function for comparison of the binary true and predicted labels.

Computational implementation and model evaluation

The computational implementation was done using Python programming language. The models were implemented with methods from the Keras library (a high-level API capable of running on top of TensorFlow). The codes were executed on Jupyter notebooks on the Kaggle platform. Each notebook runs in a computational environment provided with a quad-core CPU and 16GB of RAM[1]. No parallel computing resources were used.

Based on the MoA annotations, the accuracy of the solutions will be evaluated through the average value of the cost function applied to each *drug-MoA* annotation pair. For each `sig_id`, we will predict the probability that the sample had a positive response for each MoA target. For N lines `sig_id` and M MoA targets, we will make $N \times M$ predictions. Our objective, therefore, is to build multi-label classification models that minimize the following cost function:

$$L(y, \hat{y}) = -\frac{1}{M}\sum_{m=1}^{M}\frac{1}{N}\sum_{i=1}^{N}\left[y_{i,m}\log(\hat{y}_{i,m}) + (1 - y_{i,m})\log(1 - \hat{y}_{i,m})\right], \tag{2}$$

where N is the number of `sig_id` observations in the test data, M is the number of scored MoA targets, $\hat{y}_{i,m}$ is the predicted probability of a positive MoA response for `sig_id` ($i = 1, ..., N$ and $m = 1, .., M$), and $y_{i,m}$ is the ground truth, 1 for a positive response, 0 otherwise.

The competition's public (and private) scores are measured through this loss function.

[1] https://www.kaggle.com/docs/notebooks

Results and discussion

Exploratory data analysis

Data for 21.984 and 3.982 drugs are available in the training and test sets, respectively. Each observation has 876 features, including expression levels for 772 genes, cellular viability percentages across 100 cell lines, and three categorical variables: the period of time after exposure to the treatment (24, 48 or 72 hours), type of treatment (control or not) and drug dosage (high or low). The dataset does not contain any missing values and the pre-processing consisted of turning the categorical variables into numerical values. The data was also standardized, so that each feature in the dataset had a mean of 0 and a standard deviation of 1.

Most of the observations in the dataset concern samples that were exposed to the drugs ("cp_vehicle"), compared to only 8% of controls ("ctl_vehicle"). The other categorical features are evenly balanced in the dataset, since similar numbers of drugs were measured 24, 48 and 72 hours post exposure to the treatment and also at high and low doses (see Figure).

The dataset contains a total of 206 MoA classes. The most frequent term in the MoA classifications is "inhibitor" (54%), followed by "antagonist" (15%) and "agonist" (13%). Drugs can act through more than one MoA and this is also reflected in the dataset. So, 39% of the samples of the training set belong to the control group and do not have a MoA class, while 52% act through one MoA and 8% act through multiple MoAs. Besides, while the most frequent MoA classes are activated by hundreds of drugs, the least frequent are activated by dozens, suggesting a considerable class imbalance that should be taken into account for model optimization (Figure 2).

Dimensionality reduction via PCA

We performed a Canonical Correlation Analysis (CCA) in the gene expression and the cell viability datasets, and we got a correlation coefficient approximately equal to 0.9906, suggesting a linear relationship between them. Therefore, we used principal component analysis to reduce the feature space of the dataset.

By calculating the percentage of explained variance per principal component, we see that the first component explains about 25% of the variance of the whole dataset, while the other components explain less than 5% each, suggesting that there is not much redundancy in the data. Moreover, by distinguishing the categorical variables by colors, we see some clusterization effects (Figure 3). This indicates that these features have a significant impact on gene expression and cellular viability, specially text *cp_type*. Importantly, the number of MoAs per drug also seems to affect gene expression and cellular viability, as drugs with 2 MoAs tend to cluster separately from drugs with 0 or 1 MoA (Figure 3).

Reduced data sets

The application of the classification models was not only performed on the original data made available in the competition but also on three "polished" data sets in an attempt to reduce the observations and features that could eventually "generate noise" for the classification task. Namely, these data sets are:

- Reduced data: the original dataset was filtered so that the control group, without MoA classifications, was not considered for the machine learning analysis.

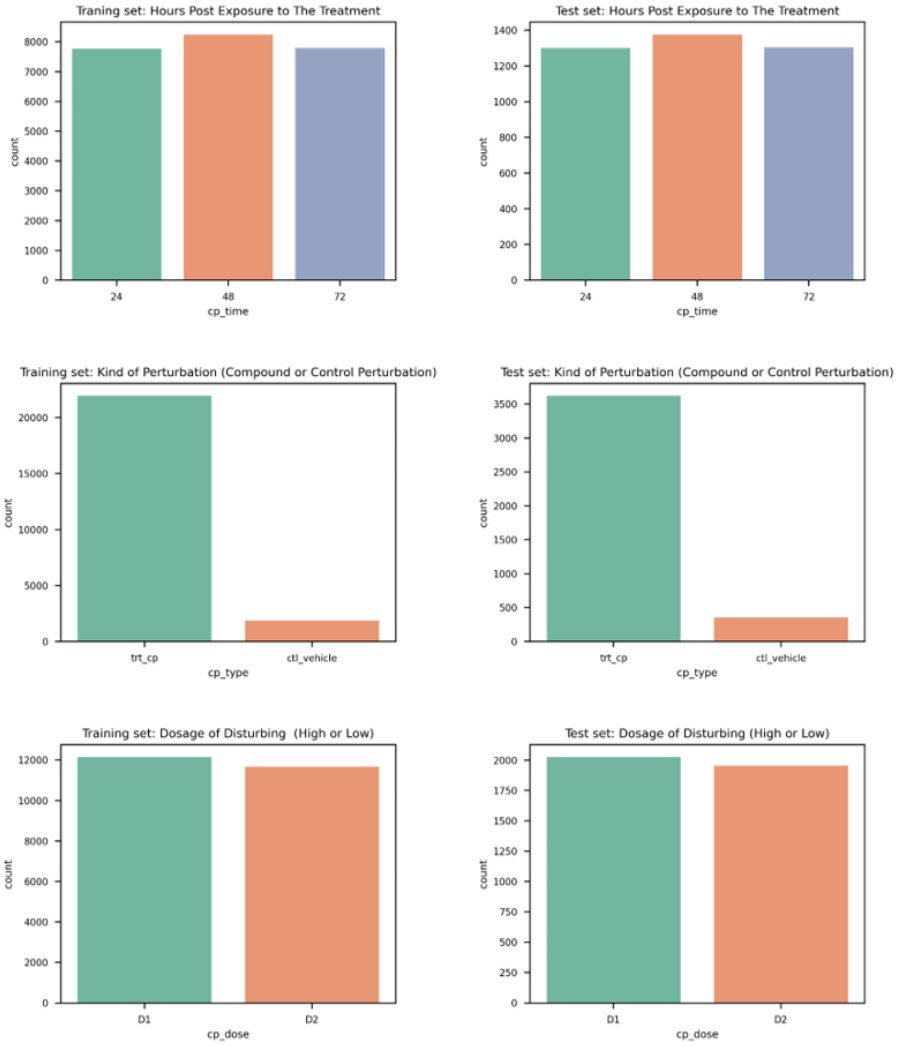

Figure 1 Distribution of data in training (left) and test (right) set, taking into consideration the features: cp_type, cp_time and cp_dose, indicates samples treated with a compound (cp_vehicle) or with a control perturbation (ctrl_vehicle); treatment duration (24, 48, 72 hours) and dose (high or low) respectively.

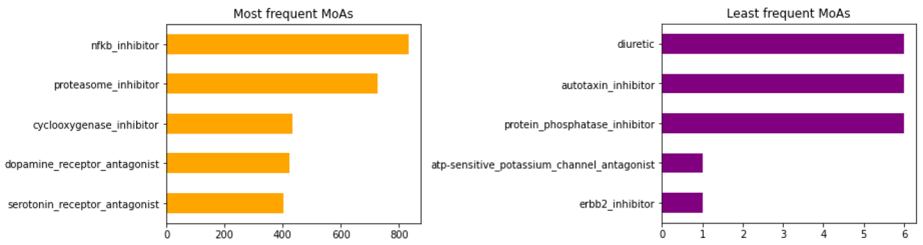

Figure 2 Most and least frequent MoAs represented in the training set.

- PCA-transformed data: After PCA transformation of the data, only the first principal components that accounted for 90% of the data variance were retained for posterior analysis.

Table 1 Explained variance by the respective number of PCs.

Explained variance	Number of PCs
94.861%	540
94.605%	530
94.342%	520
94.071%	510
93.794%	500
93.596%	493
93.423%	487

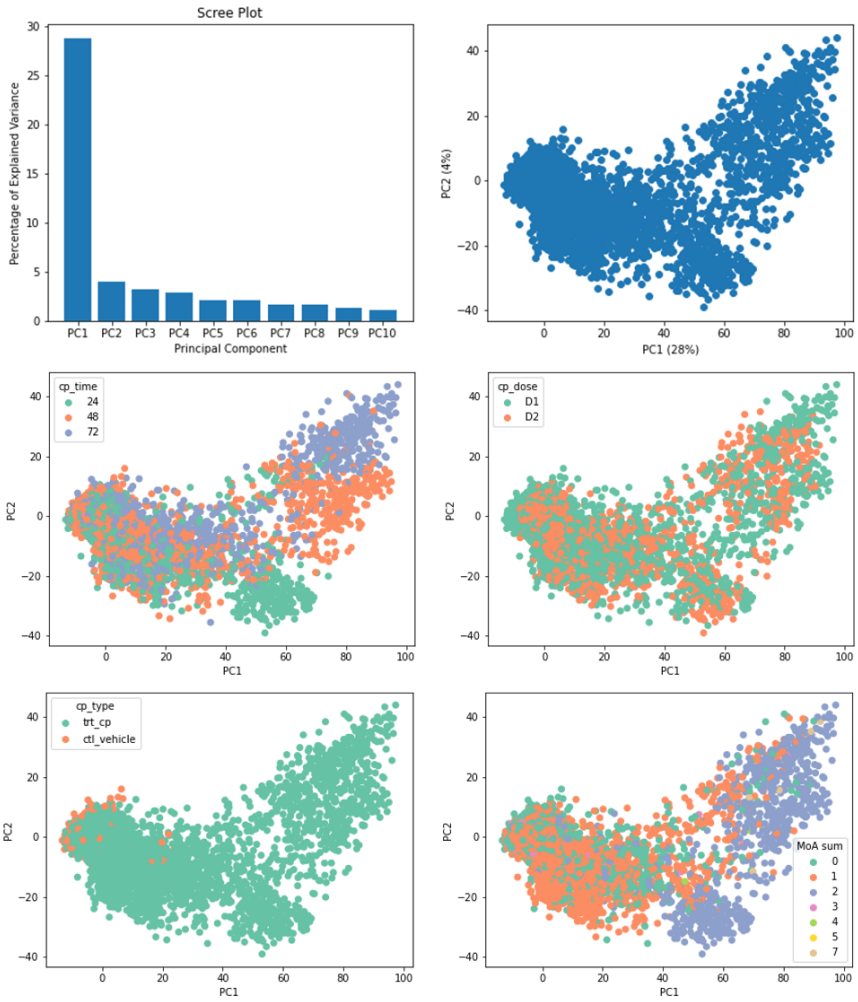

Figure 3 The scree plot and the two-dimensional plot of the first principal components obtained in the Principal Component Analysis. The categorical variables (cp_type, cp_time and cp_dose) were distinguished by colors for visualization of clusterization effects. Colors were also used to distinguish drugs with different numbers of MoAs.

- PCA-reduced data: The reduced dataset, without the control group, was PCA-transformed and PCs that accounted for 90% of the data variance were retained for modeling.

Results for the logistic model

The logistic regression model via the neural network was adjusted for the four respective scenarios: complete data, without control data, PCA data, PCA of without control data).

We tried to optimize the regularization parameter λ with cross-validation using validation/test data sets via the grid search method. In all cases, the optimal λ was equal to 0.

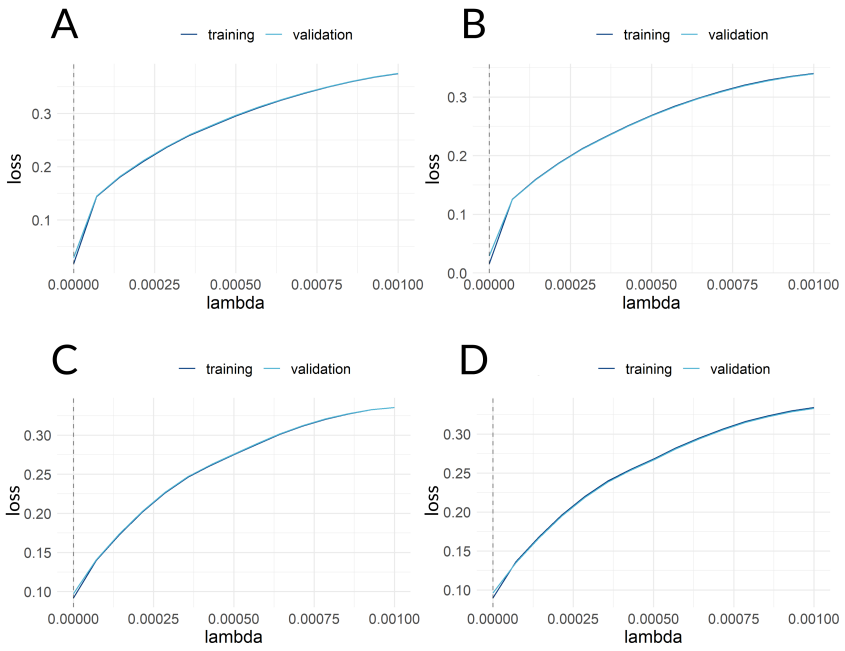

Figure 4 Training and validation curves. **A** Original dataset. **B** Reduced dataset. **C** PCA dataset. **D** PCA-reduced dataset.

Considering $\lambda = 0$, the model loss for the original and reduced data sets, fitting with 200 epochs and batch-size of 2000, was pretty similar. The performance on the test partitions was practically identical, but in the validation partitions, the model did slightly better for the original dataset.

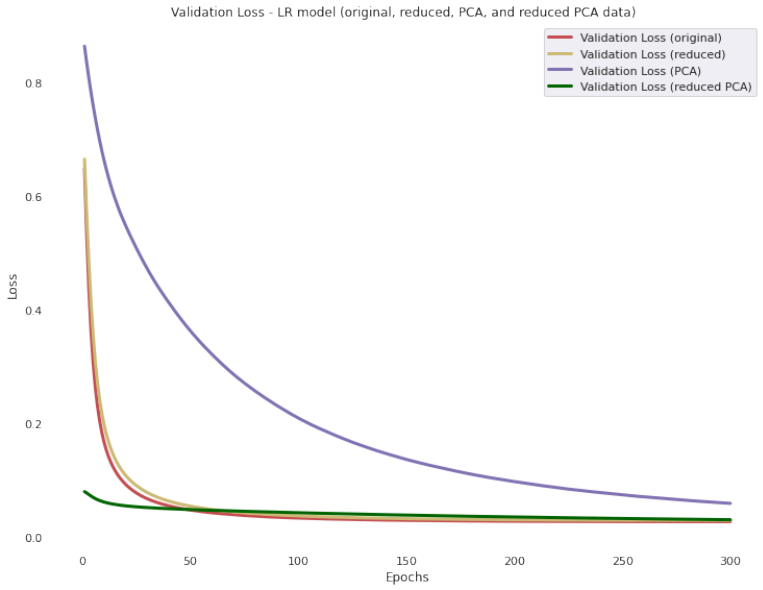

Figure 5 Loss curves by epoch for the logistic regression model in the validation partitions of the original, reduced, PCA, and reduced PCA datasets.

Results for the neural network model

Following a logical path, we tried to improve the results obtained by the logistic regression model by building a more complex model: the deep neural network model.

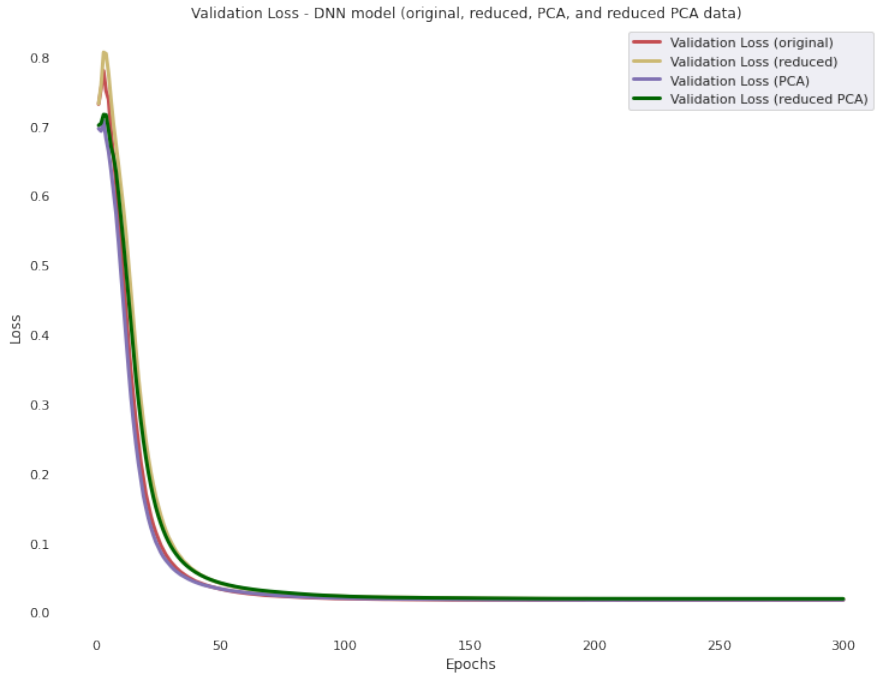

Figure 6 Loss curves by epoch for the DNN model in the validation partitions of the original, reduced, PCA, and reduced PCA datasets.

We observed that the DNN model, for the test data sets, obtained better performance (smaller loss value) in the PCA data. However, for the validation data sets, the model performed better in the original data. Furthermore, we realize that the absence of observations treated with placebo (`ctl_vehicle`) yielded a negative effect on the classification power of the model.

Comparing the models

Finally, we compare the performances of the two models in the data sets in which both obtained the best results: in the original data and in the PCA data (figure 7).

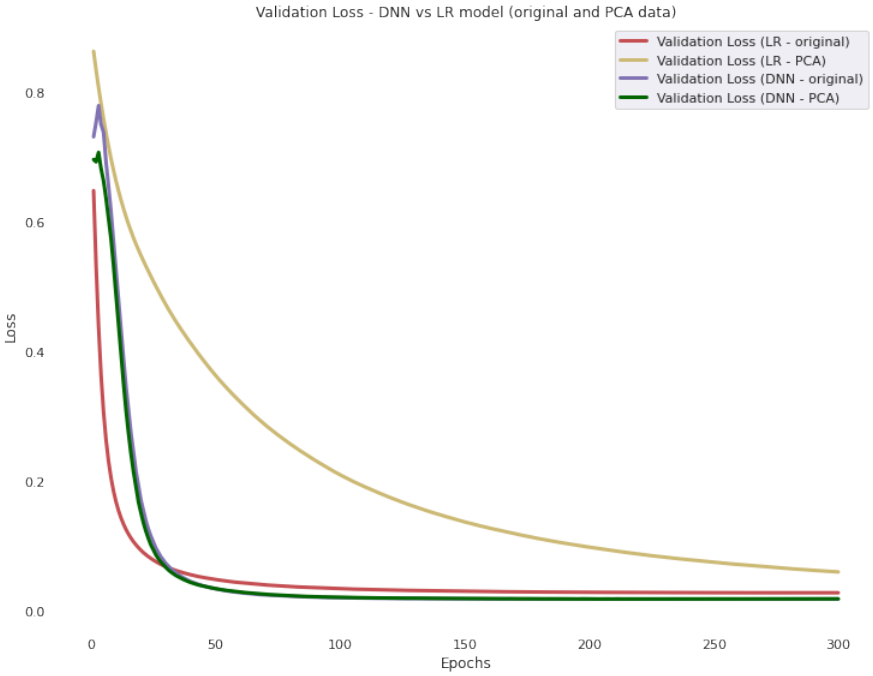

Figure 7 Loss curves by epoch for the DNN model in the validation partitions of the original, reduced, PCA, and reduced PCA datasets.

Table 2 Scores obtained by the DNN and RL models in all data sets considered.

Data set	DNN (validation)	LR (validation)	DNN (test)	LR (test)
Original	0.0178	0.0274	0.0182	0.0268
Reduced	0.0191	0.0297	0.0192	0.0303
PCA	0.0180	0.0596	0.0184	0.0597
Reduced PCA	0.0195	0.0308	0.0196	0.9030
Test (Competition)	-	-	0.0219	0.0669

We can see clearly that the DNN model performed better (smaller loss value) than the logistic regression model in all the data sets.

The results obtained in the competition's test dataset represent the *private scores* for each method, which were taken into account in the competition ranking. The best private score obtained (0.01929) reached the position 3,382 of 4,373 participants[2]. The participant who occupies the first place in the competition obtained a private score of 0.01599.

[2]See the availability of data and materials section.

The accuracy of both models fluctuated within the range of 10% to 20%. However, due to the high imbalance of the MoA classes, this is not a suitable evaluation measure, therefore we did not consider this measure.

Discussion

Despite the high imbalance of the MoA classes data, the deep neural network model employed demonstrated a good generalization ability in the classification task, achieving values below 0.022 for the loss function (2). The logistic regression model, on the other hand, as it is an overly simplified model, did not reach values smaller than 0.026. The results of both models in the reduced data sets were the worst obtained, indicating that observations without MoA (`ctl_vehicle`) can be useful in the classification task. Moreover, distinguishing between treatments that elicit or not specific MoAs may be biologically relevant in drug discovery pipelines.

Conclusions

In relation to the logistic regression model, built via neural networks, the best model was the one using the original dataset, achieving a loss of approximately 0.027 in the test dataset. In contrast, the deep neural network model obtained the best scores for the loss function, mainly with respect to the validation sets, for all data sets analyzed. We could certainly improve this model by improving its hyperparameters, e.g., by increasing considerably the number of epochs and applying cross-validation in training, however, such improvements would imply the use of more powerful computational resources, such as parallel computing.

We conclude, in retrospect of the analysis, that even faced a large and high imbalanced database, the neural network models demonstrated a good generalization capacity on the classification task.

Acknowledgements

To Prof. Dr. Marcelo da Silva Reis from University of São Paulo - USP for contributing in the discussions of this project. To the University of São Paulo - USP and Federal University of Rio Grande do Norte - UFRN for structural support and resources have been ceded.

Funding

This work was supported by CAPES (Brazilian Federal Agency for the Support and Evaluation of Graduate Education - processes 88887.464712/2019-00 and 88887.488330/2020-00), FAPESP (São Paulo Research Foundation - process 2016/11430-4) and CNPq (National Council for Scientific and Technological Development - process 141945/2020-6).

Abbreviations

MoA: mechanism of action; PCA: principal components analysis; LR: logistic regression; DNN: deep neural networks; CCA: canonical correlation analysis.

Availability of data and materials

Project home page: https://github.com/msreis/IBI5031-2S-2020/tree/main/projects/group-6. The data sets, along with the codes used in this analysis are available at https://www.kaggle.com/heitorbaldo/moa-prediction-python (version 1). The best private score of 0.01929 was obtained in the version 7 of the R notebook https://www.kaggle.com/heitorbaldo/moa-prediction-pca-dnn-r, and the private score of 0.02189 was obtained in the version 2 of the Python notebook https://www.kaggle.com/heitorbaldo/moa-prediction-pca-dnn-python. Additional analyzes can be found at https://www.kaggle.com/fiuzatayna/moa-prediction-python and https://www.kaggle.com/hellenpaulavalerio/moa-prediction-python.

Competing interests

The authors declare that they have no competing interests.

Author details
[1] Institute of Mathematics and Statistics, University of São Paulo, São Paulo, Brazil. [2] Digital Metropolis Institute, Federal University of Rio Grande do Norte, Natal, Brazil. [3] Institute of Chemistry, University of São Paulo, São Paulo, Brazil.

References

1. Spratto, G., Woods, A.: Delmar Nurse's Drug Handbook 2011: Special 20 Year Anniversary. Nelson Education, Toronto, ON (2010)
2. Stoddart, M.J.: Cell viability assays: introduction. In: Mammalian Cell Viability. Methods in Molecular Biology, pp. 1–6. Humana Press, New York, NY (2011)
3. SNUSTAD, D.P., SIMMONS, M.J.: Fundamentos da Genética. Sétima Edição. Guanabara Koogan (2017)
4. Schenone, M., Dančík, V., Wagner, B.K., Clemons, P.A.: Target identification and mechanism of action in chemical biology and drug discovery. Nature chemical biology **9**(4), 232 (2013)
5. Catlett, N.L., Bargnesi, A.J., Ungerer, S., Seagaran, T., Ladd, W., Elliston, K.O., Pratt, D.: Reverse causal reasoning: applying qualitative causal knowledge to the interpretation of high-throughput data. BMC Bioinformatics **14**(1) (2013). doi:10.1186/1471-2105-14-340
6. Subramanian, A., Narayan, R., Corsello, S.M., Peck, D.D., Natoli, T.E., Lu, X., Gould, J., Davis, J.F., Tubelli, A.A., Asiedu, J.K., *et al.*: A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. Cell **171**(6), 1437–1452 (2017)
7. Choobdar, S., Ahsen, M.E., Crawford, J., Tomasoni, M., Fang, T., Lamparter, D., Lin, J., Hescott, B., Hu, X., Mercer, J., *et al.*: Assessment of network module identification across complex diseases. Nature methods **16**(9), 843–852 (2019)
8. Musa, A., Ghoraie, L.S., Zhang, S.-D., Galzko, G., Yli-Harja, O., Dehmer, M., Haibe-Kains, B., Emmert-Streib, F.: A review of connectivity map and computational approaches in pharmacogenomics. Briefings in Bioinformatics, 112 (2017). doi:10.1093/bib/bbw112
9. Johnson, R.A., Wichern, D.W.: Applied Multivariate Statistical Analysis. Pearson Prentice Hall, New Jersey (2007)
10. James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning: with Applications in R. Springer, New York, NY (2013)
11. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York, NY (2006)
12. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
13. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
14. Liashchynskyi, P., Liashchynskyi, P.: Grid search, random search, genetic algorithm: A big comparison for NAS. CoRR **abs/1912.06059** (2019). 1912.06059
15. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization (2014). http://arxiv.org/abs/1412.6980