Analysis of Multi-Class Classification Methods In MolData Drug Discovery Dataset

Jainish Mehta [1]

Abstract

Drug discovery has current challenges in using existing pharmaceutical understanding while developing new drugs and in the integration of data since it is highly diverse. This work thereby aims to support personalizing pre-clinical filtering of drugs. This paper aims to analyze classification methods on a relatively new dataset, MolData, by applying novel feature selection and clustering models on molecular compounds and their descriptions. Various supervised learning algorithms, such as k-NN and XGBoost, were performed before feeding clusters into neural networks, such as Radial Basis Function Neural Networks, to develop and predict optimal drug selection and predict specific drug needs, such as target proteins. Extreme learning machines produced the highest performance accuracy of **36.8%**, which is slightly greater than an altered MLP architecture and SVM.

1. Introduction and Motivation

Drug discovery aims to find new compounds to treat diseases and medications by identifying substances with a desirable therapeutic effect. This often is a lengthy process that can take 12-15 years and costs in excess of US 1 billion dollars with only one in 1000 drugs entering the pre-clinical trials stage that is able to be tested in humans [1]. A significant field of application is in cancer treatment as, in 2021 alone, 1.9 million new cancer cases were diagnosed and 608,570 cancer deaths were reported in the United States [2]. Yet, there exist challenges in understanding how to leverage existing knowledge of compounds to better predict how current drugs may react to particular diseases. A paper by Aittokallio describes how there is a lack of prediction of pairwise drug-dose combination effects, especially for more than two drugs [3]. This paper will attempt to address this using a multi-class classification approach. MolData is the molecular benchmark for disease and target-based machine learning and provides unique categories of diseases that the drug targets, such as if the drug relate to pulmonary, cancer, or obesity problems, as well as proteins that the drug targets, such as ion channels, transferase, and oxidoreductase [4]. The datasets consist of binary representations of the disease and protein targets of interest for certain bioassays. There are 14 protein targets that act as classes and 16 diseases that will act as features. While a more advanced version is used for other papers, this simplified dataset has not been referenced greatly, hence working with new multi-class data.

While advanced neural networks such as graph neural networks are used for molecular compounds, current pattern recognition methods, such as K-nearest neighbors and decision tree classifiers can be potentially higher performing. These will be utilized and analyzed to find highly accurate methods for classification for determining future drug discovery and molecular pattern understanding. With the advent of deep learning and neural networks (NN), they can outperform classical machine learning techniques and will be heavily investigated. These mitigate potential problems in classical supervised machine learning techniques such as feature engineering, feature selection, as well as scale for larger datasets. This is particularly useful for overcoming challenges in discovering complex molecular patterns and diseases to target classification.

2. Methodology

This paper will compare classification accuracy from traditional machine learning methods to more novel neural networks.

2.1. MolData

Upon preprocessing the dataset, it was determined that the first three classes, membrane, enzyme, and nuclear receptor, had the largest proportion of targets associated with them. Class balancing was done using oversampling, specifically SMOTE (Synthetic Minority Oversampling Technique), which selects examples that are close to the feature space to draw new samples from the minority class [5]. This is especially useful while feeding the dataset into a neural network as previously the loss of the test set was increasing over epochs. The data set is divided 80% training and 20% testing.

2.2. MLP Architecture

A four-layer MLP is used initially with the architecture of

```
 - Linear (16, 512)- ReLU
- Linear (512, 1024) - BatchNorm (1024)
- Linear (1024, 1024) - ReLU
- Linear (1024, 14) - ReLU - Dropout (0.1)
```

The cross-entropy loss is initially used as the loss function and SGD as the optimizer.

2.3. ELM Algorithm

Various neural networks will be explored, one of which is the Extreme Learning algorithm, which is as follows:

Algorithm 1 ELM

```
Input: number of hidden neurons and number of variables
Output: predict label, the label of the point
repeat
  Initialize random input weights and biases.
  for $i = 1$ to $n$ training points do
    $H\beta = T$ as Hessian matrix.
    $\hat{\beta} = hT$ where h is Moore Penrose inverse
  end for
until $epochs$ is $>= 20$
```

In the above algorithm,

$$H = \begin{bmatrix} g(w_1x_1 + b_1) & . & g_(w_nx_1 + b_1) \\ g(w_1x_N + b_1) & . & g(w_nx_N + b_N) \end{bmatrix}$$

$$\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ . \\ \beta_n \end{bmatrix}$$

and T is a output vector similar to matrix β. Outputs in ELM are calculated as: $f_L(x) = \sum_{i=1}^{L} \beta_i g(w_i x_j + b_i)$ where L is the hidden units, β is a special matrix with a weight between the hidden layer and output, g is the leaky ReLU activation function, b is the bias vector, and x is the input vector [6].

2.4. Hyperparameter Tuning

Upon determining the optimal neural network, hyperparameter tuning will be performed, specifically for the loss function, optimizer, and learning rate. Various loss functions can be used for multi-class classification, including the Kullback-Leibler (KL) divergence loss and cross-entropy loss. In supervised learning, forward KL is used where for two probability distributions P and Q, KL divergence measures how a probability distribution differs from another one: $D_{KL} = E_{x\sim P}[log\frac{P(x)}{Q(x)}]$ [7]. In a model, the loss function minimizes the risk of it and is denoted by $L(f(x), y)$. KL divergence loss will optimize over the distribution of models, f_θ such that $argmin_\theta E_{(x,y)\sim D}[L(f_\theta(x), y)]$. For cross-entropy loss, a separate loss is calculated for each class label, $-\sum_{c=1}^{M} y_{o,c} log(p_{o,c})$ where p is the predicted probability of observation, o, and class, c and y is the binary indicator if the class label is a correct classification [8]. Optimizers include non-adaptive ones such as stochastic gradient descent and Nesterov accelerated gradient, as well adaptive ones such as AdaDelta and Adam. AdaDelta uses exponentially weighted averages over previous gradients to address issues of diminishing learning rates [9]. It is expressed as: $\eta_t^{'} = \frac{\eta}{\sqrt{S_{dw_t} + \epsilon}}$ where η is the learning rate, ϵ is a small positive number, and S_{dw_t} is $\beta S_{dw_{t-1}} + (1-\beta)(\frac{\delta L}{\delta w_{t-1}})^2$ where w is the weight parameter and β is usually 0.9 to 0.95; 0.95 will be used. Adam optimizer adapts the learning rate parameter using the average of the second moments of the gradients, the uncentered variance [10]. Lastly, the activation function is altered with softmax and tanh. Softmax is often used as the last layer to normalize the output to a probability distribution [11]. It is represented by $\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$ where K is the number of classes and e^{z_i} and e^{z_j} are the standard exponential function for the input and output vector, respectively.

3. Results

3.1. KNN Classification

The most evident approach to classification is K-nearest neighbours algorithm, which is a non-parametric supervised learning method. The distance metric is Euclidean without the need for normalizing as there are categorical labels. One hot encoding is already performed on the dataset. Multiple K-values are utilized and evaluated with evidently low accuracy scores, not exceeding more than 48.3 percent before SMOTE oversampling and 27.8% after, as shown in figure 1.

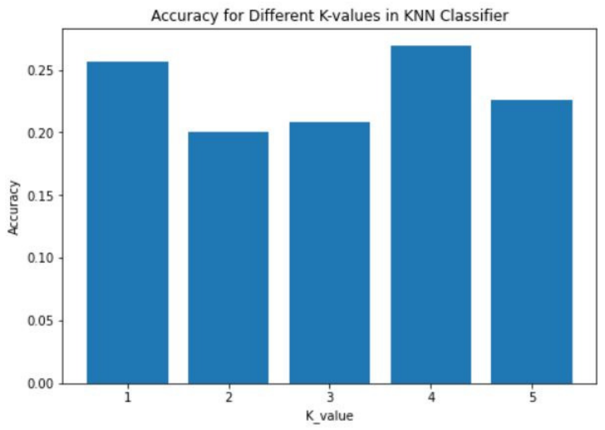

Figure 1. k-NN classification over various K values

There are few previous arts to base the results from, however, the KNN is evidently producing meaningful results with little accuracy. This can arise from the fact that there are 14 classes. There is much overfitting. More training data points and less regularization can help with this issue. The curse of dimensionality might also be a factor where increasing the number of variables will make it difficult to predict the output of a new data point. Reducing the feature space may help using PCA or the Integrated Gradients (IG) method [12]. Other approximate nearest neighbour algorithms can also be used such as locality-sensitive hashing (LSH) or an advanced algorithm proposed by Hou et al., KNN-KD-tree algorithm, combining KNN and KD-tree [13].

3.2. Decision Tree Classification

These are supervised learning method that takes on a discrete set of values called classification trees where the leaves are the class labels and branches are features for the class labels. A maximum depth of 4 is used for the tree and the accuracy output is 40% before SMOTE and 21.7% after. Compared to the KNN classification, the decision tree has lower accuracy. Some reasons may include that as the tree grows in size, it may be prone to overfitting and pruning may be needed. The decision tree can be visualized in the figure below.

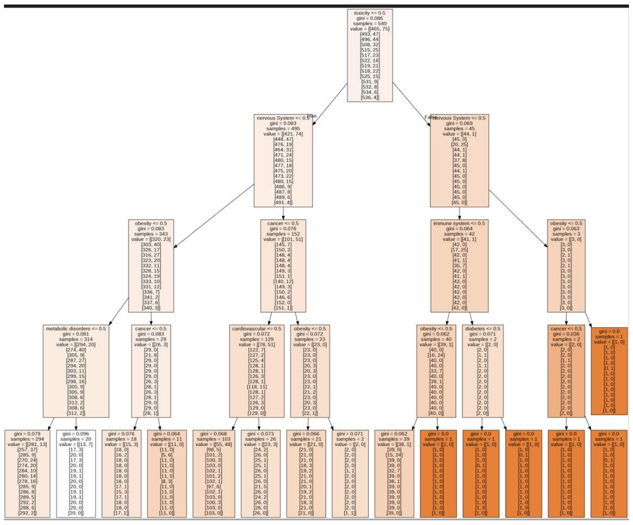

Figure 2. Decision Tree Leaves and Branches

The figure illustrates how the leaf and branches are chosen with the Gini index calculated as: $G = \sum_{i=1}^{C} p_i \cdot (1 - p_i)$ where p_i is the probability of label i being chosen [14]. The low Gini values for almost all the features indicate the purity of classification, which is relatively optimal as these features are preferred.

The random forest classifier constructs a multitude of different decision trees at training time and represents the output selected by most trees. It produces an accuracy of 41.6% before SMOTE and 32.2% after, which is slightly higher than that of the decision tree classification. This is logical as it avoids and prevents overfitting by using multiple tree instances and performing majority voting from them. Each tree does not consider all the features, hence the feature space is reduced.

3.3. Ensemble Model Classification

Ensemble modeling is useful as they use a finite set of alternative models and allows for a flexible structure to obtain higher accuracy predictive performance than from any one algorithm. XGBoost is used to build each model to emphasize any training instances that previous models misclassified. Multilabel classification is needed for this dataset, so instead of using a one-vs-the-rest classifier for fitting one classifier per class, a multi-output classifier is used where it is fitting for the classifier per target. This produced much more meaningful results for binary data with the accuracy being roughly 45% before oversampling and 33.0% after. This is better than the KNN classification technique and indicates overfitting. The algorithm minimizes overfitting by uniquely penalizing the model through LASSO (L1) and Ridge (L2) regularization, which are beneficial to prevent unnecessary overfitting [15].

3.4. Support Vector Machines

Support vector machines perform classification by finding a hyper-plane that differentiates any two classes. These are effective in high-dimensional space, as is the case for MolData [15]. Different kernel functions, which take the data and transform them into a required form in the output, are experimented it. The linear kernel function is defined by $K(x, y) = (x^T y + c)$ and is a simple kernel that would not project data into higher dimensions. The polynomial kernel function will be of more interest as it takes the inner product from higher dimensional spaces [16]. It is expressed as $K(x, y) = (ax^T, y)^d$ where d is the number of dimensions. The radial basis (RBF) kernel function will also produce meaningful results as it looks into the inner product of two data points in an infinite number of dimensions as $K(x, y) = e^{-\gamma \|x-y\|^2}$ where γ scales the amount of influence 2 points have on one another [17]. Lastly, the sigmoid kernel function is $K(x, y) = tanh(\alpha * x^T y + c)$ where α is a kernel parameter and is useful for binary classification. Gammas are the radius of the area of influence of the support vector with large gammas leading to overfitting, while too small values cannot capture the complexity of the data. The C-value, which tells the SVM model how much to avoid misclassifying each training example, is already tuned for optimally to act as a relatively accurate regularization parameter. Experimental results of hyperparameter tuning with corresponding model accuracy are illustrated in figure 3.

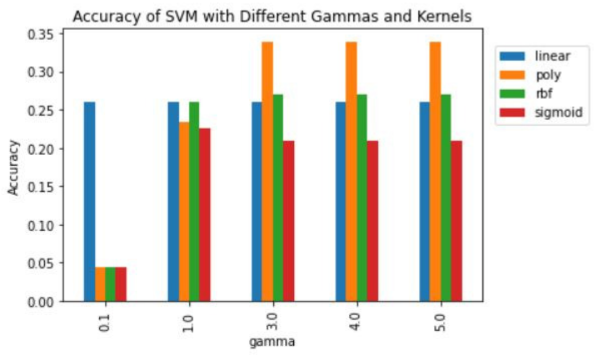

Figure 3. SVM Performed with Various Gammas and Kernels

Evidently, the polynomial kernel function had the highest accuracy and a 4.0 gamma value produced the best results. This may be attributed to the fact that the data is non-linear, high-dimensional, and effectively normalized. The RBF kernel also performed well since it projects higher-dimensional data and searches linear separations within it.

3.5. Summary of Traditional ML Techniques

Technique	Accuracy	Precision	Recall
KNN (K=4)	27.8	37.6	27.0
Decision Tree	21.7	13.7	20.7
Random Forest	32.2	36.5	31.4
XGBoost	33.0	43.4	32.2
SVM (γ=4, poly)	33.9	43.3	33.0

Technique	F1 Score	Class 2 ROC-AUC OvR
KNN (K=4)	24.9	59.0
Decision Tree	14.9	82.6
Random Forest	27.0	82.6
XGBoost	28.0	82.1
SVM (γ=4, poly)	30.0	78.9

Table 1 Performance of Traditional ML Techniques

Note that all values in the table above are in percent. SVM and XGBoost produced the highest accuracy, precision, recall, F1 score, and close to the highest ROC-AUC one vs rest score for class 2. This correlates with findings by Wu et al. and their analysis of the MoleculeNet dataset, which consists of properties of 700000 compounds [18]. Among the conventional methods for machine learning models, XGBoost and SVM methods had high testing accuracy. SVM works well with the Moldata dataset as it is more effective in high-dimensional spaces and when there is a clear margin of separation between classes.

3.6. Neural Networks: Radial Function Neural Network

Neural networks are beneficial as it detects significant features efficiently and easily, are computationally efficient, and are beneficial for multi-class classification such as the case for MolData. Radial Function Neural Networks are artificial neural networks (ANN) where the radial basis function is the activation function that consists of a real-valued function φ that depends on the distance between the input and a fixed point, c i.e. $\varphi(x) = \widehat{\varphi} = \|\mathbf{x} - \mathbf{c}\|$. The results on MolData are shown in the following figures.

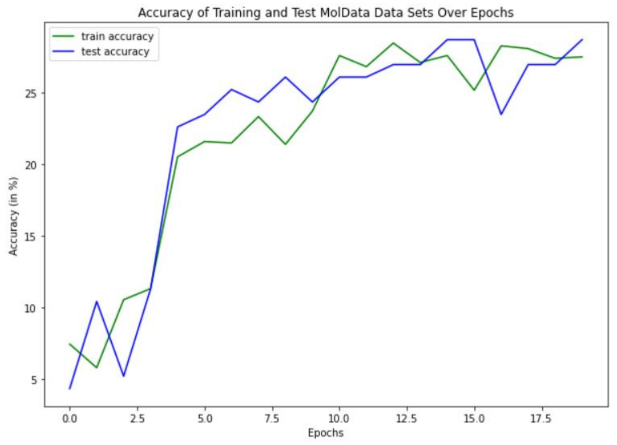

Figure 4. Accuracy of MolData on RFNN Over 20 Epochs

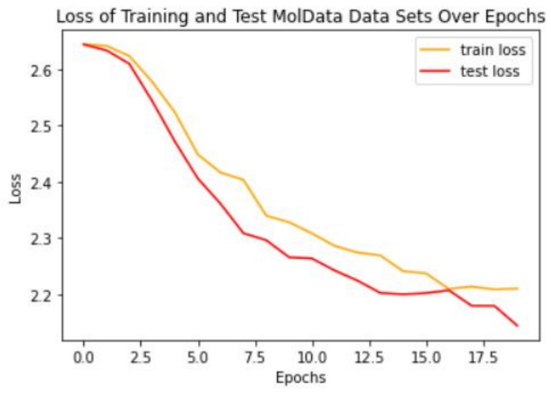

Figure 5. Loss of MolData on RFNN Over 20 Epochs

As shown from figures 4 and 5, the training and testing accuracy is quite low at roughly 30 percent at its peak and a training and testing loss at roughly 2.2.

3.7. MLP

Multi-layer perceptions (MLP) are a fully connected feedforward artificial neural network (ANN) that use backpropagation for training. A paper by Chen et al. references how deep learning is used extensively in drug discovery and QSAR modeling [18]. As such, an MLP will be performed on the MolData dataset.

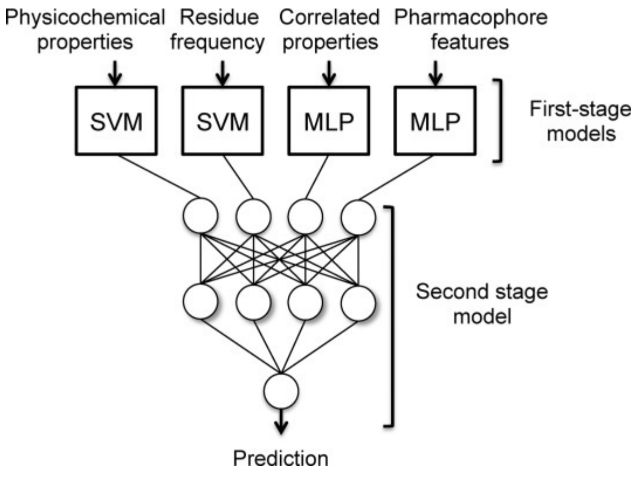

Figure 6. MLP Used in Drug Discovery Pipeline

The following graphic illustrates how MLP is used to determine pharmacophore features for later prediction and is hence an integral part of the deep learning pipeline [19].

This produced the following plot for the accuracy and loss of the training and testing dataset.

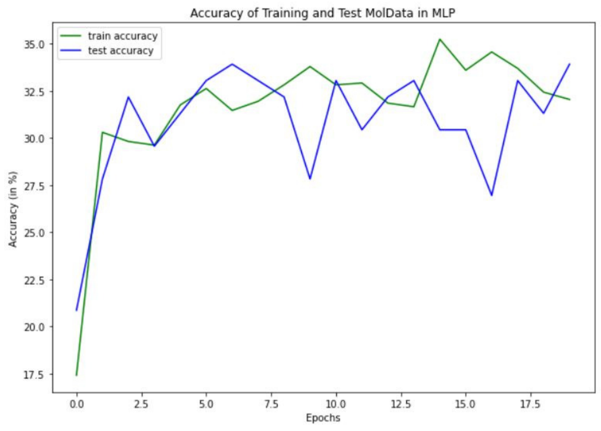

Figure 7. Accuracy of MolData on MLP Over 20 Epochs

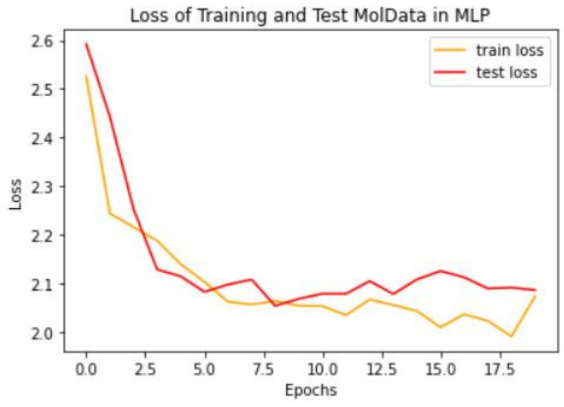

Figure 8. Loss of MolData on MLP Over 20 Epochs

This produces interesting results as the train loss increases slightly near the end of the epochs. The learning rate is lowered from 0.01 to 0.005, however that did not make an impact. A major possibility is that there is potential overfitting of the data with the accuracy still low. Since there are only 600 original data points and 918 with SMOTE, the training data is too small to accurately represent all input data values. There is further evidence with the accuracy from the MLP being roughly 36 percent and a training and testing loss at roughly 2.1. This is better than RBFNN. Namely, RBFNN uses Gaussian activation functions, $h(x) = e^{(-\frac{(x-c)^2}{r^2})}$, which makes neurons more locally sensitive, whereas MLP uses sigmoidal activation functions and has dropout regularization. This makes MLP better for extrapolation and testing on new datasets [20].

3.8. Extreme Learning Machines

Extreme learning machines do not require backpropagation techniques that MLP does, but instead, require a Moore-Penrose generalized matrix inverse for weights [21]. The following plots show the accuracy and loss of the training and testing dataset.

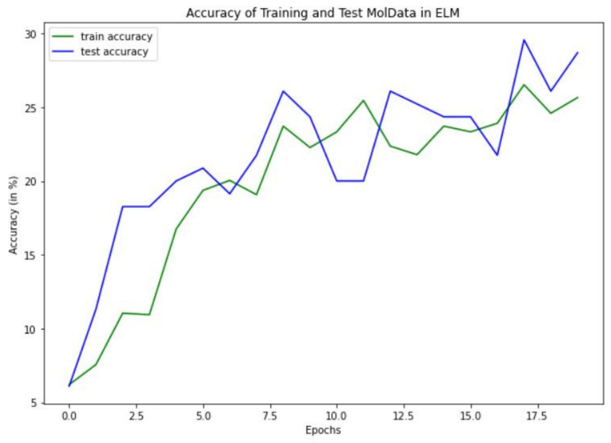

Figure 9. Accuracy of MolData on ELM Over 20 Epochs

The accuracy and loss are slightly worse than in MLP, however better than radial basis function neural networks. This is only a little consistent with some papers, such as by Markowska-Kaczmar et al., that claim ELM have a prediction accuracy comparable to MLPs with far less computing effort [22]. This may be credited to a need in ELM for developing smart algorithms for the inverse matrix calculation to determine weights.

3.9. Hyperparameter Tuning

Various loss functions and optimizers are experimented with. The following table is changes made to the MLP neural network [23]. Using the AdaDelta loss function, the MLP

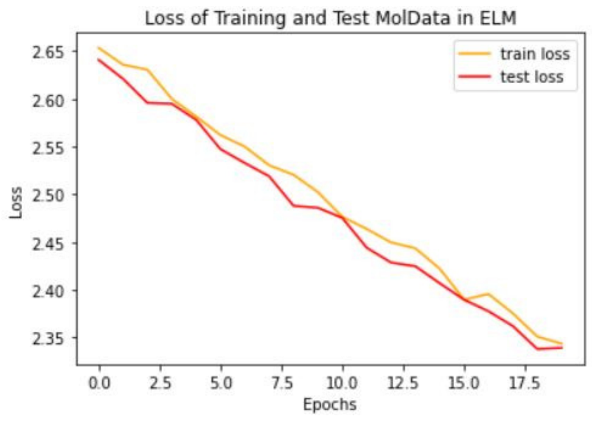

Figure 10. Loss of MolData on ELM Over 20 Epochs

architecture is altered to improve accuracy. Some techniques include using the ReLU activation function to prevent the vanishing gradient problem, doing batch normalization near the output layer, and adding dropout to prevent overfitting. Consequently, the batch normalization in the second layer was removed and batch normalization to the third layer with a dropout of 0.1 is added. This increased the accuracy to 33.4%, but increased the MSE to 23.7. A paper by Maier et al. mentions the optimum number of hidden neurons was determined by varying the number of neurons from I to $2I$+1,where I is the number of input variables [24].

Technique	Accuracy (in %)	MSE
NAdam (Nesterov)	6.8	57.2
AdaDelta	27.3	13.7
Adam	17.5	51.8

Table 2 Performance of MLP with Optimizers

The following table is changes made to the ELM neural network.

Technique	Accuracy (in %)	MSE
NAdam (Nesterov)	34.9	3.7
AdaDelta	17.6	22.9
Adam	36.8	24.2

Table 3 Performance of ELM with Optimizers

Note that the training accuracy and average training MSE after 20 epochs are recorded. The lower the MSE value, the more ideal the model [25]. After hyperparameter tuning of the models, ELM performed significantly better than the MLP neural network. Loss functions have also been experimented with using Kullback-Leibler (KL) divergence and the cross-entropy function. These are the most common for multi-classification problems. The ELM neural network had an accuracy of 7.6 percent on the training set, while MLP had an accuracy of 6.6 percent for the KL loss function. These are relatively low values compared to cross-entropy, which is counter-intuitive as it is functionally equivalent to cross-entropy loss since that is what the KL divergence minimizes towards. Nonetheless, the cross-entropy loss function will be used. Lastly, the softmax and tanh non-linear activation functions were tuned in the MLP architecture. The softmax produced an accuracy of 7.43%, while tanh produced 25.1% accuracy on the training set. This is interesting as softmax is recommended for classification problems since it resembles the derivative of the cost function C with respect to input z, $\frac{\partial C}{\partial z}$ in the last layer. Therefore, the ELM neural network produced the highest accuracy with the Adam optimizer, 0.001 learning rate, and leaky ReLU activation function. Adding additional layers to the MLP architecture with AdaDelta optimizer and cross-entropy loss function made it close to performance with ELM.

4. Discussion

SVM performed the best from the traditional machine learning methodologies explored with an accuracy of 33.9%, while the ELM neural network had the highest accuracy of 36.8% with the Adam optimizer, 0.001 learning rate, and leaky ReLU activation function. While ELM is more computationally expensive and more time-consuming, it produced more accurate results. This may be attributed to non-linearities in the dataset and more hidden layers for complexity that can learn the structure and parameters of the data. Nonetheless, the accuracies were similar potentially due to the fact there is not a large volume of training data. Before SMOTE was implemented, KNN with a K-value of 4 had an accuracy of 48.3%, which is greater than ELM. Overall, the dataset had a very low overall accuracy both before SMOTE and after, hence more feature selection or more training points would be ideal to make better predictions and prevent potential overfitting. There is also a possibility there is little correlation between the features and classes, however, this is unlikely because a paper by Arshadi et al. has performed correlation analysis to investigate drug repurposing and found sets of bioassays highly correlated in both active and inactive molecules [26]. For the ELM and MLP neural networks, the testing accuracy was slightly greater than the training accuracy and with both of the values being quite low, there is underfitting at play as well.

5. Future Work

A more advanced dataset consisting of 1.4 million with 600 label columns can be used to compare the novel ELM algorithm with the previous multitask learning state-of-the-art model for molecular artificial intelligence proposed by Arshadi et al. This would likely increase model performance, especially for the neural networks. Research by Jayaweera et al. suggests ELM with a RBF kernel with different input weights is an effective classification model and likewise, ELM with a polynomial kernel, as found to be optimal for SVM, can be experimented with [6]. Finally, leaving some or all training with empty targets can allow for semi-supervised or unsupervised learning and experimentation with other neural networks, such Restricted Boltzmann machine (RBMs) with deep belief networks formed by stacking RBMs with backpropagation can potentially beat state-of-the-art models for classifying disease-target drugs. Future challenges in drug-dose combination with more than two drugs ca be investigated with ELM neural network proposed in this paper against the state-of-the-art classifier chains as explored by Nam et al [27].

6. References

[1]J. P. Hughes, S. Rees, S. B. Kalindjian, and K. L. Philpott, "Principles of early drug discovery," British journal of pharmacology, Mar-2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3058157/. [Accessed: 08-Dec-2022].

[2]"Cancer Facts Figures 2021," American Cancer Society. [Online]. Available: https://www.cancer.org/research/cancer-facts-statistics/all-cancer-facts-figures/cancer-facts-figures-2021.html. [Accessed: 08-Dec-2022].

[3] T. Aittokallio, "Full article: What are the current challenges for machine learning in drug discovery and repurposing?," Taylor Francis Online. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/17460441.2022 .2050694. [Accessed: 08-Dec-2022].

[4] "Transilico/Moldata: A molecular benchmark for disease and target based machine learning," GitHub. [Online]. Available: https://github.com/Transilico/MolData. [Accessed: 08-Dec-2022].

[5] J. Brownlee, "Smote for Imbalanced Classification with Python," MachineLearningMastery.com, 16-Mar-2021. [Online]. Available: https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/. [Accessed: 08-Dec-2022].

[6] C. D. Jayaweera and N. Aziz, "Development and comparison of extreme learning machine e and multi-layer perceptron neural network models," Journal of Physics: Conference Series. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1123/1/012032. [Accessed: 08-Dec-2022].

[7] D. Ghosh, "KL divergence for Machine Learning," The RL Probabilist, 07-Aug-2018. [Online]. Available: https://dibyaghosh.com/blog/probability/kldivergence.html. [Accessed: 08-Dec-2022].

[8] "Loss Functions," Loss Functions - ML Glossary documentation. [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/lossfunctions.html [Accessed: 08-Dec-2022].

[9] G. Mayanglambam, "Deep learning optimizers," Medium, 18-Nov-2020. [Online]. Available: https://towardsdatascience.com/deep-learning-optimizers-436171c9e23f. [Accessed: 08-Dec-2022].

[10] J. Brownlee, "Gentle introduction to the adam optimization algorithm for deep learning," MachineLearningMastery.com, 12-Jan-2021. [Online]. Available: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. [Accessed: 08-Dec-2022].

[11] K. E. Koech, "Softmax activation function," Medium, 18-Nov-2021. Available: https://towardsdatascience.com/softmax-activation-function-how-it-actually-works-d292d335bd78. [Online].

[12] R. Khandelwal, "Understanding deep learning models with integrated gradients," Medium, 04-Nov-2020. [Online]. Available: https://towardsdatascience.com/understanding-deep-learning-models-with-integrated-gradients-24ddce643dbf. [Accessed: 08-Dec-2022].

[13] W. Hou, D. Li, and C. Xu, "An advanced k nearest neighbor classification algorithm ... - IEEE xplore," IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/8690508. [Accessed: 09-Dec-2022].

[14] N. Tyagi, "Understanding the gini index and information gain in decision trees," Medium, 30-Sep-2020. [Online]. Available: https://medium.com/analytics-steps/understanding-the-gini-index-and-information-gain-in-decision-trees-ab4720518ba8 [Accessed: 08-Dec-2022].

[15] A. Nagpal, "L1 and L2 regularization methods," Medium, 14-Oct-2017. [Online]. Available: https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c. [Accessed: 08-Dec-2022].

[16] S. Ray, "SVM: Support Vector Machine Algorithm in machine learning," Analytics Vidhya, 29-Nov-2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Accessed: 08-Dec-2022].

[17] S. Dye, "An intro to kernels," Medium, 05-Mar-2020. [Online]. Available: https://towardsdatascience.com/an-intro-to-kernels-9ff6c6a6a8dc. [Accessed: 08-Dec-2022].

[18] Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande, "MoleculeNet: A benchmark for Molecular Machine Learning," Chemical Science, 31-Oct-2017. [Online]. Available: https://pubs.rsc.org/en/content/articlelanding/2018/SC/C7SC02664A.

[Accessed: 08-Dec-2022].

[19] H. Chen, O. Engkvist, Y. Wang, and M. Olivecrona, “The rise of Deep Learning in Drug Discovery,” Drug Discovery Today, 31-Jan-2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1359644617303598?via [Accessed: 08-Dec-2022].

[20] E. Gawehn, J. Hiss, and G. Schneider, “Deep learning in drug discovery - gawehn - wiley online library,” Wiley Online Library, 30-Dec-2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/minf.201501008. [Accessed: 08-Dec-2022].

[21] “Extreme Learning Machines,” Extreme Learning Machines: Random Neurons, Random Features, Kernels. [Online]. Available: http://www.extreme-learning-machines.org/. [Accessed: 08-Dec-2022].

[22] U. Markowska-Kaczmar and M. Kosturek, “Extreme learning machine versus classical feedforward network - neural computing and applications,” SpringerLink, 30-Aug-2021. [Online]. Available: https://link.springer.com/article/10.1007/s00521-021-06402-y. [Accessed: 08-Dec-2022].

[23] A. P. Singh, “Steps you should follow to successfully train MLP,” Medium, 08-Aug-2020. [Online]. Available: https://medium.com/analytics-vidhya/steps-you-should-follow-to-successfully-train-mlp-40a98c3b5bb3. [Accessed: 08-Dec-2022].

[24] H. R. Maier, “Use of artificial neural networks for predicting optimal alum doses and ...,” Research Gate, May-2004. [Online]. Available: https://www.researchgate.net/publication/220275281 [Accessed: 08-Dec-2022].

[25] W. Rowe, “Mean square error R2 score clearly explained,” BMC Blogs, 05-Jul-2018. [Online]. Available: https://www.bmc.com/blogs/mean-squared-error-r2-and-variance-in-regression-analysis/: :text=There%20is%20no%20correct%20value,to%20what%20R2%20should%20be. [Accessed: 08-Dec-2022].

[26] A. Keshavarzi Arshadi, M. Salem, A. Firouzbakht, and J. S. Yuan, “Moldata, a molecular benchmark for disease and Target Based Machine Learning - Journal of Cheminformatics,” BioMed Central, 07-Mar-2022. [Online]. Available: https://jcheminf.biomedcentral.com/articles/10.1186/s13321-022-00590-y. [Accessed: 08-Dec-2022].

[27] J. Nam, E. Loza Mencía, H. J. Kim, and J. Fürnkranz, “Maximizing subset accuracy with recurrent neural networks in multi-label classification,” Advances in Neural Information Processing Systems, 01-Jan-1970. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/2eb5657d37f474e4c4cf01e4882b8962-Abstract.html. [Accessed: 08-Dec-2022].