Developing better baselines for gene expression prediction

Kristy Mualim (260679030), Heinrick Lumini (260654386)

Abstract In this work, we take a new look at the gene expression work performed by Chen et al. and develop new baselines for the gene expression prediction task. We discuss the approaches taken in the original paper and provide new benchmarks for their neural network model, D-GEX.

***Index Terms*— Gene expression, feed forward neural network, multitask regression,**

I. INTRODUCTION

The fundamental goal of this paper seeks to discuss a suggested deep learning predictive model for gene expression and the baselines that were used by Yifei Chen et al. [1] to develop it, as well as the reproducibility of the work. The multi-task neural network system, annotated as D-GEX[1], aimed to improve the work of the LINCS program for gene expression data.

Gene expression profiling has been used as a tool to capture gene expression patterns in cellular responses to diseases, genetic perturbations and drug treatments. The Connectivity Map (CMAP) project was thus launched to create a large collection of expression patterns and has since discovered minute molecules that are functionally connected to receptors responsible for modulating expression. [2] However, to perform these tasks, CMap took to utilizing Affymetrix GeneChip microarrays, producing only ~564 genome wide gene expression profiles.

Given the intrinsic fact that most of the expression profiles relating to the ~22,000 genes present across the whole human genome are highly correlated, programs such as the LINCS program sought to construct regulatory networks of target and regulatory genes producing the LINCS consortium.

The LINCS consortium is a program that analyzed gene expression profiles from the Connectivity Map using principal component analysis. It was then discovered that carefully selecting ~1,000 genes could capture ~80% of the information presented in the cMAP data. Researchers are now able to access the expression signatures of these ~1,000 landmark genes to characterize cellular perturbations under various experimental conditions. In addition, the LINCS program also adopts a computational inference model utilizing linear regression to predict expression profiles of ~21,000 remaining genes, labelled as target genes, utilizing the expression profiles of landmark genes as input. D-GEX is a multi-layer feedforward neural network that sought to improve the methodology used by the LINCS consortium in inferring target gene expression. Whilst their D-GEX method has proven greater accuracy in predictions, their method was benchmarked solely against linear regression and kNN, which are hardly basic classifiers not representative of both what can be achieved with other state-of-the-art neural networks.

We thus propose to develop a more robust comparative baseline for this gene expression task by comparing their model proposed to other classifiers, as well as cutting edge deep learning models. This will allow us to assess the results obtained by Chen et al. based on sound baselines. This work will allow us to produce a brief survey of the efficiency of deep learning for gene expression modelling, and compare it to simple classifiers.

Improving predictions within the LINCS Consortium is an important problem as it serves to both accelerate accurate expression profiling over thousands of samples, eventually revealing meaningful patterns in regulatory DNA sequences and prioritizing different regulatory noncoding variants.

II. METHODOLOGY

A. *Datasets*

The *GEO expression dataset* curated by the Broad Institute contained 129,158 expression profiles from Affymetrix microarray platforms. The original GEO data was accessed from LINCS Cloud (http://www.lincscloud.org/), which has been quantile normalized into numerical ranges of 4 to 15. We removed duplicates from the dataset leaving 111,009 profiles in the end. The dataset was subsequently standardized by subtracting mean and dividing by standard deviation of each gene.

B. *Data Processing*

We worked on a manageable subset of the original data, that we were able to obtain from the raw data after running the scripts provided by the author of the original paper. We performed a 80/10/10 train/validation/test split on the retrieved data in order to train and select our models.

C. Metrics

The metric chosen for the development of was mean absolute error, (MAE), which was also the metric chosen in D-GEX. Here the standard deviation of our dataset is about 0.98, and the mean 0.02 therefore a MAE of .5 would mean our predicted values are on average about half a standard deviation away from the mean. However, we are mostly interested in how the models compares to each other rather than obtaining specific values for the task itself, so we will focus on the difference between the MAE recorded by the different models we analyze.

D. *Principle Component Analysis (PCA)*

Given that there are 943 features available in the prediction of an individual gene's expression, we wanted to utilize PCA to reduce the number of similar features and reduce dimensionality. Principle Component Analysis (PCA) is a dimension-reduction tool that is used to reduce a large set of variables to a small set that contained most of the information in the large set. The first principal component accounts for as much of variability in the data as possible, and each succeeding component accounts for as much of remaining variability as possible.

III. MODELS

A. *Support Vector Machine*

Support Vector Machines (SVMs) are supervised learning models and constitutes a linear classifier. A support vector machine formally constructs a hyperplane or set of hyperplanes in a high-dimensional space. A good separation is achieved by the hyperplane with the largest distance to the nearest training-data point of any class. The input x is first mapped onto a m-dimensional feature space using some fixed (nonlinear) mapping and then a linear model is constructed in this feature space. We used the validation set in order to tune the SVM penalty parameter of error, C.

B. *k-Nearest Neighbours*

k-Nearest Neighbours (k-NN) is a non-parametric method used where the input consists of k closest training examples in the feature space. The output, in this case, is the predicted gene expression for the particular target gene. The value is the average of the values of its k nearest neighbours. k-NN is a type of instance-based learning where the function is approximated locally and thus is particularly sensitive to the local structure of the data. Standard kNN regression may be biased when duplicated samples frequently exist in the data, such as the GEO microarray data. Thus, in gene expression inference, a commonly adopted alternative would be to query k nearest genes rather than k nearest samples. We trained this model by calculating euclidean distances to all landmark genes of target gene, illustrating the k nearest landmark genes of target gene as the k nearest landmark genes with least euclidean distances. kNN does not impose any prior assumptions on learning machine. Thus, is flexible to model nonlinearities contained within the dataset. However, performing prediction involved building and querying data structures, kNN could suffer from poor scalability to growing data size. Optimal k was obtained based on validation sets.

C. *Feed-forward neural network*

Feed Forward Neural Networks are created by utilizing multi-layered networks of sigmoid units. This means each layer is a collection of non-linear activation functions, so the overall network can model non-linear functions. The overall architecture consists of an input layer, an output layer and k hidden layers,

such that the neurons present at the i^{th} layer form the inputs to the neurons at $(i+1)^{th}$ layer. The output layer then gives the predicted outputs.

We utilized backpropagation as a means for weight updates, where the error obtained upon our output prediction values are fed back through the network to allow the algorithm to adjust weights of each connection in order to reduce the value of the error function by a small amount.

This is performed multiple times either until the network converges or the maximum epoch number is reached.

We experimented with several neural network architectures. Starting with one hidden layer and working our way up to three. We also worked with different layer widths and loss function. The final feed forward neural net we selected is composed of a single hidden layer containing 225 units, using a rectified linear unit as the activation function (ReLU). ReLU units, developed in 2000 by Hahnloser et al. [4], and have been shown extremely efficient for neural networks applied to regression tasks. The network is optimized using the MAE loss described previously.

D. Convolutional neural network

Convolutional Neural Networks (CNNs) form a class of deep, feed-forward artificial neural network designed to work identify higher level patterned in structured data [3].

That property motivated us to attempt to apply CNNs to our problem, under the assumption that the expression that there exists broad patterns in the expression of landmark genes[1] that can be brought to a higher level representation and exploited by a CNN, since these have shown good results being applied to regression tasks as well [7].

CNNs, similar to feed-forward neural nets, consist of an input and an output layer, with multiple hidden layers. Each individual hidden layer can be one of a convolutional, pooling, fully connected or normalization layers .

Convolutional layers apply a convolution operation to the input and passes the output as input to the next layer. Each convolutional neuron processes data only for its receptive field. The convolution operation applied helps to reduce the number of free parameters, allowing the network to be deeper in the second layer. After each convolutional layer, it is conventional to apply a nonlinear layer in an attempt to introduce nonlinearity to a system that was previously computing linear operations.

Pooling layers help to combine the outputs of neuron clusters at one layer into a single neuron in the next layer.

CNNs are usually deemed more efficient than feed-forward neural networks in relation to image classification given its use of local receptive fields, parameter sharing and pooling [3]. Local receptive fields help to focus on important features isolated to a small region of interest in the image.

IV. RESULTS

A. Principle Component Analysis

We tested number of components ranging from 1 to 943 and eventually selected for 551, maintaining an overall sample variance of at least 85% of the original sample data. The use of PCA enabled us to work extensively on designing our other algorithm with much more manageable runtime than with the initial 943 features, while still obtaining sensibly similar results.

B. Support Vector Machine

Utilizing support vector machines, we tested C values from 0.1 to 0.8, inclusive for 8 increments.

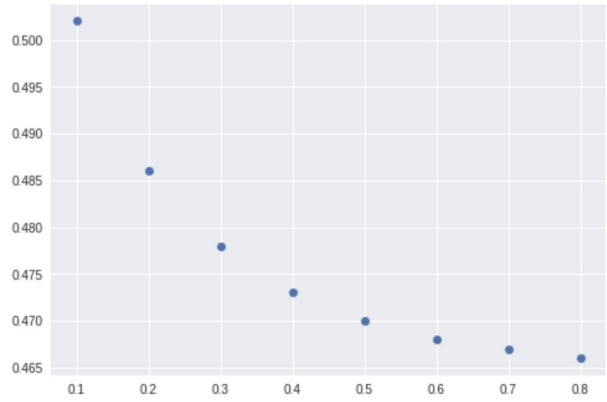

Figure 1. Validation MAE v.s Value of C

Utilizing the information gathered from Figure 1, we used the C value of 0.8 as the hyperparameter for our GEO test data, achieving a mean absolute error of 0.491.

Testing SVM models on GTEx data and 1000G data respectively, given the same hyperparameter, mean absolute error was at 0.583 and 0.756 respectively. This indicated that the GTEx data and 1000G data may not be as linearly separable as GEO, generating higher error values.

C. k-Nearest Neighbours

We did hyperparameter tuning for k-Nearest

neighbours using a range of values from 0-30.

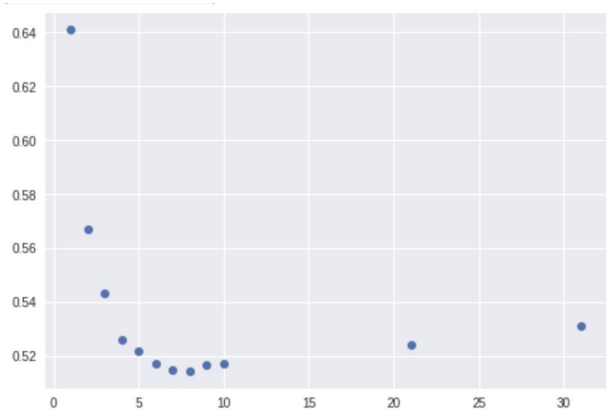

Figure 2: Validation MAE v.s Number of nearest neighbours

Utilizing the information gathered from Figure 2, we used number of neighbours = 8 as the hyperparameter for our GEO test data, achieving a mean absolute error of 0.535.

Testing kNN on GTEx data and 1000G data respectively, given the same hyperparameter, mean absolute error was at 0.458 and 0.756 respectively.

D. Feed-forward neural network

We developed and selected the feed forward neural network described through several attempts at finding the right architecture. Using more than one layer would lead to severe overfitting, obtaining an MAE as low as .08 on the training set and an MAE of over 2.3 on the validation set. The explanation for this is that our model needed more bias to prevent the overfitting. We resorted to using a single layer feed forward neural network, with a width of 225 units, and an input size of 551, the number of features that were selected using PCA. The results yielded MAE values that were only similar to the previous, simpler classifiers, as even the simplest of our feed forward neural networks seem to heavily overfit the training set, unable to compete with simpler baselines.

That overfitting problem is visible on the following Figure 4, showing the evolution of the loss when training our model.

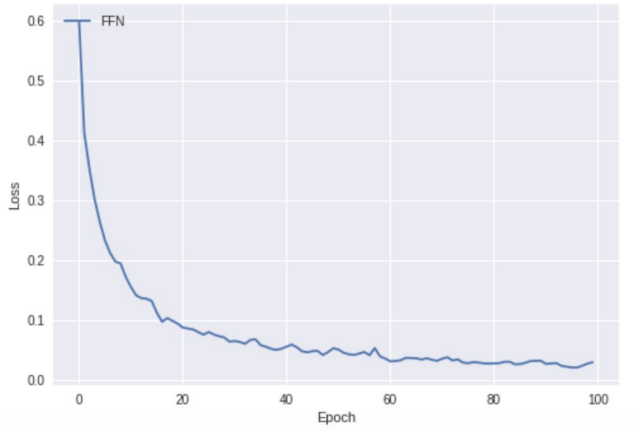

Figure 3: Training MAE v.s Number of Epoch

After only about 80 epochs, our model is already close to perfectly fitting the training set, while the results on validation set for such model are mediocre. The following curves show our attempt at fixing this problem through early stoppage at several different epochs.

Figure 4: MAE v.s Number of Epochs

We reached an accuracy of .53 on the test set using this model.

DISCUSSION

A. Large Datasets

The paper used large datasets that were supposedly sustainable on a CPU. However, during data preprocessing, we faced significant issues in having our computers handle such massive amounts of data due to the computational inefficiency as well as lack of available RAM and storage. Thus, we took to utilizing a google cloud instance to help us preprocess data. Additionally, working with datasets that consisted of 129,158 gene expression profiles, each

comprising of 22,268 probes, corresponding to 978 landmark genes and 21,290 target genes were beyond the computational means that we had access to. However, we wanted to preserve the integrity of the datasets and thus conserved the 978 landmark gene. We therefore made the decision to work on the prediction of one of the target genes, using all the landmark genes comparable to D-GEX - a multitask system, equivalent to having a different classifier for each gene. This allowed us to maintain the given task of gene expression inference at a more manageable level. Any means to extend the results obtained in this project and D-GEX would be via training models with the entire dataset.

B. Principle Component Analysis

In the context of analyzing gene expression, utilizing just 551 features dramatically reduced the training time of the models, therefore enabling us to train and experiment more easily with different models.

C. Training Neural Networks

We observed as illustrated in the D-GEX paper [1], D-GEX is presented as a single model capable of predicting the expression of any of the 9250 genes presented originally. However, it turns out D-GEX actually several multitask models, configured and trained each on a partition of the target genes. The value reported in the original paper did not represent the real performance of a single D-GEX model, but the aggregated performance of several models, presented as one.

Our approach to handling the enormous amounts of data we were presented with was to reduce the dimensionality of the data, both through PCA, and through predicting the expression of a single gene. This way of handling the issue could have been the source of the reduced performance and overfitting of our neural networks.

Moreover, on the full dataset, in the original paper[1], D-GEX reported an MAE of 0.44 while linear regression obtained 0.47. Our neural network had comparable results to that of D-GEX where 9000 hidden units and 3 hidden layers were used. Our results being obtained from simplified data, this indicates the data might not carry enough complexity for complex neural networks to yield convincing performances on it. It would be good to further explore the use of other models for gene expression tasks rather than neural networks that might not be biased enough, as our work shows.

D. Convolutional Neural Networks

We sought to exploiting convolutional neural networks for the task and ran into several conceptual issues. In order to exploit a CNN, we would need to make use of some sequentiality in the data, whether it is temporal or spatial. While commonly utilized on time series, speech (temporal sequentiality) or image data (spatial sequentiality), we wanted to explore gene expression inference utilizing a CNN regression framework with suitable representation given its successful usage in other problems that required the use of initially unspecified features [5, 6].

Our current data utilizes correlation between the expression of different landmark genes, however with no obvious notion of sequential events associated with it. In developing a CNN model, we sought to alter data representation. Training CNN models to convolve across a dataset that lacked intrinsic sequentiality proved challenging.

An alternative was to split the data over groups of features, allowing us to feed our data through the CNN sequentially. However, artificially introducing a form of sequentiality in the data that did not exist, might in turn alter conclusions to gene expression inference.

However, we propose the idea of multiplying a binary mask to inputs of discriminator network, utilizing a mask constructed using random Bernoulli distribution with probability p(mask) having 0 and 1 elements. Training would start with using the mask with high probability (having more zero elements) to only show a small portion of genes initially and then progressively decreasing the probability to finally show all the genes at the end of the training process.

V. CONCLUSION

This work has allowed us to to show simple baselines can compare to complex models, but also that the use of deep learning is not always easy to justify while working with gene expression data.

Despite only utilizing a partial section of the dataset, it could be observed that general baseline

parametric and non-parametric methods, such as SVM and kNN both obtained performance comparable to the neural networks on gene expression inference. The results mentioned in the paper also coincide with our observations and error values.

The use of neural networks applied to our dimensionally reduced data generated insightful observations. The feed forward neural network allowed us to look into how the complexity of the data was affected when its dimensionality was reduced in several ways . The fact that the neural network obtained performance comparable to D-GEX while heavily overfitted the training set, implies that neural networks might be too complex to work on our relaxed version of the gene expression problem. In addition, while the deep learning algorithms such as those implemented in D-GEX managed to reduce prediction error to 32% from 38% as illustrated in LINCS L1000 [1]. A prediction error registered at 32% is still too high for applications in gene inference hence further work needs to be done at generating an appropriate model in gene expression inference.

Despite the successful use of CNNs in regression problems in the litterature [7], our attempt at utilizing CNN allowed us to realize that it may impose drawbacks in this particular task. These results, however, have propelled further thoughts into fully exploiting different data representations. A way to apply CNNs to this task in the future could be found through insightful manipulation of the data.

Overall, our exploration of possible baselines for the gene expression task performed in the D-GEX paper allowed us to see that the simple baselines compete with complex model on simple data, and that a simplifying the data could be a way to develop baselines before developing more advanced technique on a multitask dataset.

References

[1] Yifei Chen, Yi Li, Rajiv Narayan, Aravind Subramanian, Xiaohui Xie, *Gene expression inference with deep learning*, bioRxiv 034421; doi: https://doi.org/10.1101/034421

[2] Justin Lamb, Emily D. Crawford, David Peck, Joshua W. Modell, Irene C. Blat, Matthew J. Wrobel, Jim Lerner, Jean-Philippe Brunet, Aravind Subramanian, Ken- neth N. Ross, Michael Reich, Haley Hieronymus, Guo Wei, Scott A. Armstrong, Stephen J. Haggarty, Paul A. Clemons, Ru Wei, Steven A. Carr, Eric S. Lander, and Todd R. Golub. 2006. *The connectivity map: Using gene-expression signatures to connect small molecules, genes, and disease*. Science, 313(5795):1929–1935.

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. *Gradient-based learning applied to document recognition*. Proceedings of the IEEE, 86(11):2278–2324, Nov.

[4] Hahnloser, R.; Sarpeshkar, R.; Mahowald, M. A.; Douglas, R. J.; Seung, H. S. (2000). *Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit*. Nature. 405: 947–951. doi:10.1038/35016072

[5] Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. NIPS (2014)

[6] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)

[7] Babu, G.S., Zhao, P., & Li, X. (2016). Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Useful Life. DASFAA.