Deep Learning in Drug Discovery

Model Test and Exploration

Qing Chen
Columbian College of Arts & Sciences
Washington, D.C.
qchen7@gwu.edu

***Abstract*—The goal of this paper is to test several deep learning models using Tensorflow for two interesting problems in drug discovery, target prediction and measurement value prediction. In Section I, we introduce the two problems in the context of given datasets, and provide a short summary of their background and current status. In Section II, we describe the architecture of the models we used, including single layer neural network, multi-layer neural network, convolutional neural network for target classification and include our best results for each model. We apply several simple regularization techniques to overcome over-fitting. The best model has an accuracy over 0.83. In Section III, we describe the architecture of models we used, which are similar to Section II but for regression, and we also include our best results for each model. Section IV provides a conclusion on our findings and a discussion on issues and possible future improvements.**

Keywords—Drug Discovery, Deep Learning, Convolutional Neural Network (CNN), Fingerprint

I. INTRODUCTION

According to [1], in recent years, it takes about $2.5 billion as the average price-tag for getting a new drug to market, with an estimated delivery date of 10-15 years. As Hinton'team won the Merck Kaggle challenge on chmical compound activity with deep networks in 2012, high potential of deep learning in drug discovery has been drawing more and more attention [2].

Therefore, it is likely to use deep learning driven approach to accelerate the drug discovery cycle and reduce the investment for getting a useful new drug.

A. Problem Statement

We have two datasets regarding Dopamin D2 receptor, which are available in ChEMBL database. Both datasets have the following features: "target," "smiles," "fingerprint," "occp," "measurement_value," and "measurement_type." "target" is basically what a compond can act functions on. "smiles" represents the formulas for each of the drugs. "fingerprint" is a 1024 bit binary identifier derived from the ECPC Extended-Connectivity Fingerprints (ECFPs), based on Morgan Algorithms [3]. "occp" stands for occupancy, which has 1024 dimensions, means the frequency of one substructure occurred in the drug compound. "measurement_value " is the value associated with "measurement_type", which is the metric that is used to take the measurement. The "measurement_type", for example, IC50, is a measure of the effectiveness of a substance in inhibiting a specific biological or biochemical function.

The first problem is classifcation. Given strucutre information from "fingerprint" and "occp," the goal is to predict whether the compound is active on a target class t. We use myFP_217_D2.csv dataset for target class prediction, since the other dataset myFP_CHEMBL217.csv only has one target. Therefore, myFP_CHEMBL217.csv is used for the regresssion task. The second one, a regression problem, is to predict the measurement_value given all other features. Since "smiles" is not what we undersand as a result of lack of domain knowledge, we abandoned it in both datasets. However, understand the internal structure of "smiles" and "fingerprint" migt help to build more intuitive convonlutional neural network.

B. Background

In 2017, Garrett B. Goh and his co-workers did a review of deep learning for target classification [4]. According to their review, the 2012 Merck challenge was the first foray of deep learning into quantitative structure activity relationship (QSAR), which aims at predictions of 15 drug targets. In 2014, Dahl and co-workers explored the effectiveness of multi-task (multiple output of different interest) neural networks for QSAR applications, based on the algorithms used in the Merck challenge. In the same year, Hochreiter and co-workers published a paper regarding application of multi-task DNN for quantitative structure activity relationship (QSAR) application using a very large dataset, which included 743,336 compounds, approximately 13 million chemical features, and 5069 drug targets [5]. They use ECFP4 fingerprints as input data. [5] also provides a comparison of performance accuracy (in terms of AUC metrics) of deep neural network against several traditional machine learning algorithms, using a curated database obtained from ChEMBL, as shown in Fig. 1. While most traditional machine learning algorithms range from 0.7-0.8 AUC, DNN achieved an AUC of 0.83 [4-5]. [6] provides a consistent performance accuracy comparison across 3 different datasets (PCBA, MUV, Tox21) when using multi-task deep neural network (MT-DNN) as compared to logistic regression (LR), random forest(RF), and single-task neural network (ST-NN), as demonstrated in Fig. 2. In [4, table1], Goh and his co-workers also summarized a wide range of reasonable comparison between DNN Models and state-of-the-art non-DNN models,

and the table shows that DNN outperforms other non-DNN models in general.

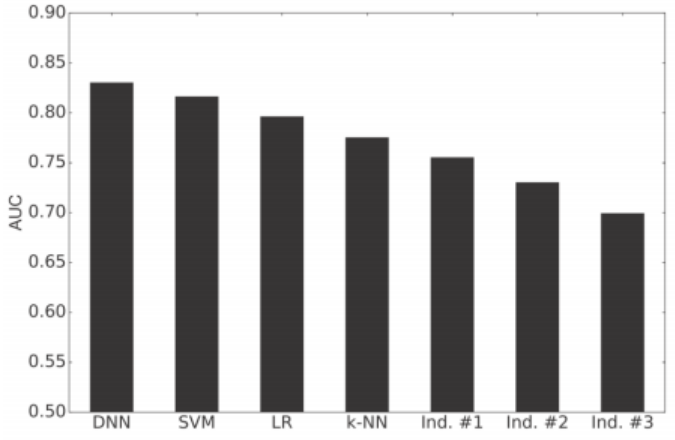

Fig. 1. AUC performance comparsion (DNN vs. traditional ML algorithms)

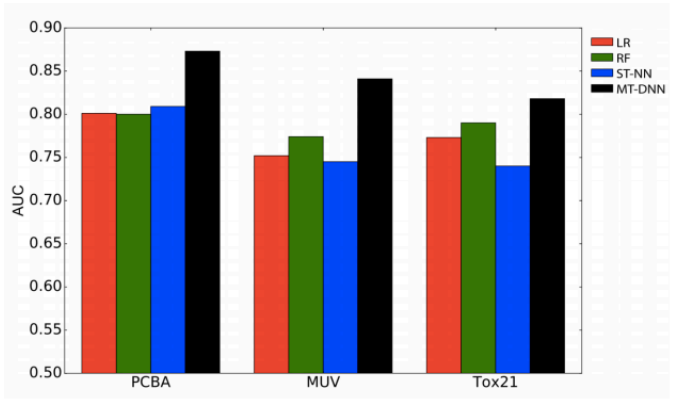

Fig. 2. AUC performance comparsion across 3 different datasets.

II. Classification

The orignial dataset is randomly split into 3 parts, including a training set (70% of orignial data), a development test set or validation set (20%), and a test set (10%).The training set is for model training, while the development test set is used to test the generalization error of the model trained. One can tune model parameters to improve model performance on the development set. However, the test should be considered blind during the training and model tuning process.Otherwise, more or less, we tend to overfit the test set.

The classfication of "target" is a multi-class (7 classes) classification problem. Fig.1 shows a frequency distribution of target from the dataset. Since the dataset is not extremely unbalanced or totally dominant by one particular class, using accuracy is reasonable as the evaluation metric for this problem. Predicting all output as the dominant class would only yield to an accuracy of approximately 0.3. AUC and F1 score are more complicated to implement in multi-class classification, and it is unnecessary to use them for this particular dataset.

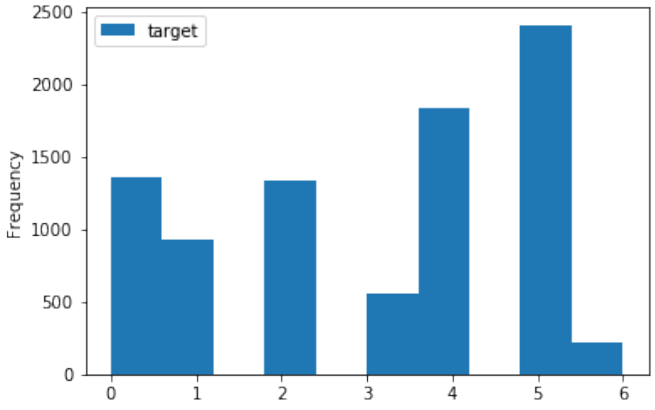

Fig. 3. Frequency distribution of target

Three models are used for this classification problem, which are softmax regression, general feedforward neural network, and convolutional neural network. All models are tested under dozens of parameter settings (including hyperparameters and model parameters) to improve accuracy of the development test set prediction, and they all use cross entropy as the cost function to be optmized. The best model settings and results are presented in this paper. The architecture of each model are introduced in II.A-C. We also use one-hot coding for target classifcation, which binarize the original one.

A. Softmax Regression

The softmax regression is one of the simplest models for multi-class classification. It can be viewed as a single-layer neural network with softmax function as its activation function at the output layer. The input layer has 2048 neurons and 7 outputs.

B. ANN (2048-512-128)

The ANN we use has 3 hidden layers, with 2048-512-128 neurons at the corresponding h1-h2-h3 layers. Activation function for all the hidden layers are rectified linear units (RELU) to avoid gradient vanishing effects. We also test ANN with 1 and 2 layers, which would be improved by adding layer to 3. We do not test ANN with over 4 layers, since going deeper rarely helps more, which is not the case in convolutional neural network [7].

C. CNN (2048-512(5)-128(8))

The convolutional neural network contains two convolutional layers and two fully connected layers, with 1d convolution (to be more precisely, cross-correlation) and without max-pooling. We use 5 filters for the first convolution layer, 8 filters at the second convolution layer, and set stripe size to 4. The dropout rate we set is 0.2. All activation functions are RELUs.

D. Results

(1) Softmax regression:

Training accuracy: 0.992712796

Development test accuracy: 0.725376606

Test accuracy: 0.709154129

(2) ANN (2048-512-128):

Training accuracy: 0.994700253

Development test accuracy: 0.814020872

Test accuracy: 0.772885263

(3) CNN (2048-512(5)-128(8)):

Training accuracy: 0.994700253

Development test accuracy: 0.809965253

Test accuracy: 0.780996501

From the above results, we can observe that overfitting occurs. Fig. 4. shows the training cost versus (validation) development test cost along the number of epochs. There is a "U" shape for the generalization error. Therefore, adding regularizers should be a good way to help us increase development test accuracy as well as test accurcay possibly. [8] introduce many common regularization techniques, due to the time limit, we only implement early stopping and L2 regularization (weight decay).

Early stopping is very easy to implement. The basic idea is to stopping iterations of training before the "U" shape of generalization error grows too far. In my code, two kinds of early stopping are implemented, one is based on validation cross entropy, the other is based on validation accuracy.

On the other hand, the L2 regularization strategy drives the weights closer to the origin by adding a quadratic regularization term of weights, driving the weights closer to the origin [8].

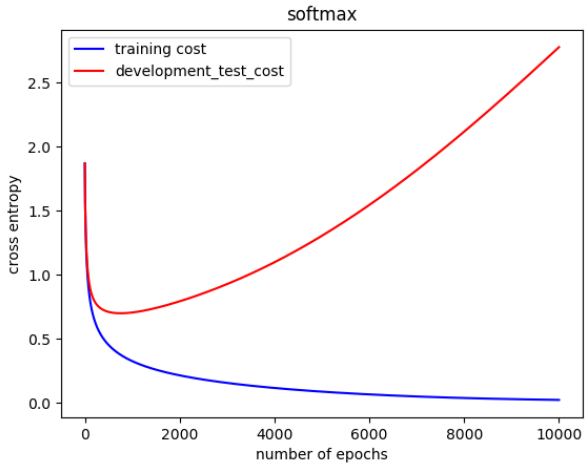

Fig. 4. training error vs generalization error of softmax model

E. Results using Early Stopping

(1) Softmax regression:

Training accuracy: 0.901457429

Development test accuracy: 0.781575918

Test accuracy: 0.767091513

(2) ANN (2048-512-128):

Training accuracy: 0.994700253

Development test accuracy: 0.838354588

Test accuracy: 0.807647765

(3) CNN (2048-512(5)-128(8)):

Training accuracy: 0.994700253

Development test accuracy: 0.817497075

Test accuracy: 0.783314049

F. Results using L2 Regularization

(1) Softmax regression:

Training accuracy: 0.868168294

Development test accuracy: 0.774044037

Test accuracy: 0.765932798

(2) ANN (2048-512-128):

Training accuracy: 0.994700253

Development test accuracy: 0.824449599

Test accuracy: 0.806488991

(3) CNN (2048-512(5)-128(8)):

Training accuracy: 0.994700253

Development test accuracy: 0.799536526

Test accuracy: 0.783314049

From D-F, we can see that currently ANN with 3 hidden layers has the best performance. We can also observe that both early stopping and L2 regularization increase test accuracy a little bit (For softmax regression, there are relatively significant improvements). Due to the time and computational power limitation, we are going to test more regularization strategy in future work. After we get sufficient reliable models, we are going to use model ensemble to see whether we can further improve our accuracy.

III. Regression

Compare with the classification problem, regression on measurement_value seems to be a much more challenging problem. For the dataset, we did not use a unseen test set. We only split the dataset to training set (70%) and (development) test set (30%), since the performance was not even good for the training set. Unlike the overfitting issues in Section II, we first

encountered serious underfitting issues here. The evaluation metric and the cost function we use here are both Mean Squared Error (MSE). In addition, we also make a plot to compare all predictions versus the actual value of measurement value, and the best prediction should generate most points on or very close the the indentity line y = x. At first, after tuning parameters for hunderd of times, we still could not get a model that finely fit the training set, no matter we did normalization or not. It turned out ANN and CNN frequently fitted a "constant" with very small fluctuations. One strange thing we observed that is ANN tend to predict the mean of training set as its best prediction. But finally, we find that one of our CNN setting is able to fit the training set.

Furthermore, since high measurement_value is not what we desire in drug discovery, suggested by Dr. Chen Zeng, we decide to remove records of high meansurement_value based on a boxplot and IQR analysis. Fig. 5. shows these huge outliers.

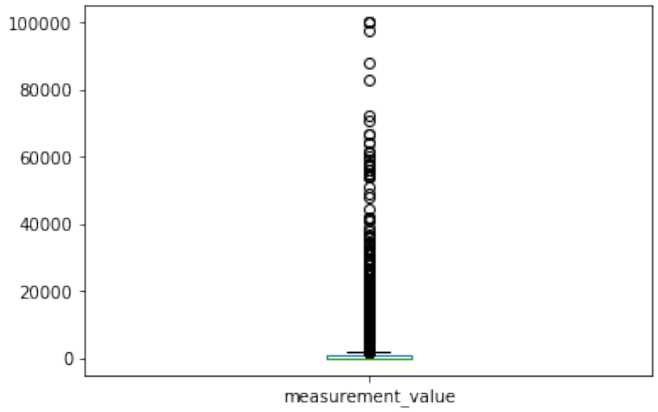

Fig. 5. Boxplot of measurement value

After removing these outliers, we still have more than 85% of the original data. Although under-fitting issues still occurs in linear model and ANN, all 3 models perform better.

A. Linear Regression

The linear regression can be viewed as a single-layer neural network with no activation function or linear identity function as its activation function at the output layer. The input layer has 2050 neurons (weights) and 1 output.

B. ANN (2048-1024-2048)

The ANN we use has 3 hidden layers, with 2048-512-128 neurons at the corresponding h1-h2-h3 layers. Activation functions for all the hidden layers are sigmoid functions. The large number of neurons in h3 layer is due to the squashing effect of the sigmoid functions (avoid weight being too large).

C. CNN (2050-410(5)-82(8))

The convolutional neural network contains two convolutional layers and two fully connected layers, without 1d convolution (to be more precisely, cross-correlation) and max-pooling. We use 5 filters for the first convolution layer, 8 filters at the second convolution layer, and set stripe size to 5. The dropout rate we set is 0.2.

D. Results

(1) Linear regression:

Training MSE: 88358.648437500

Test MSE: 119437.796875000

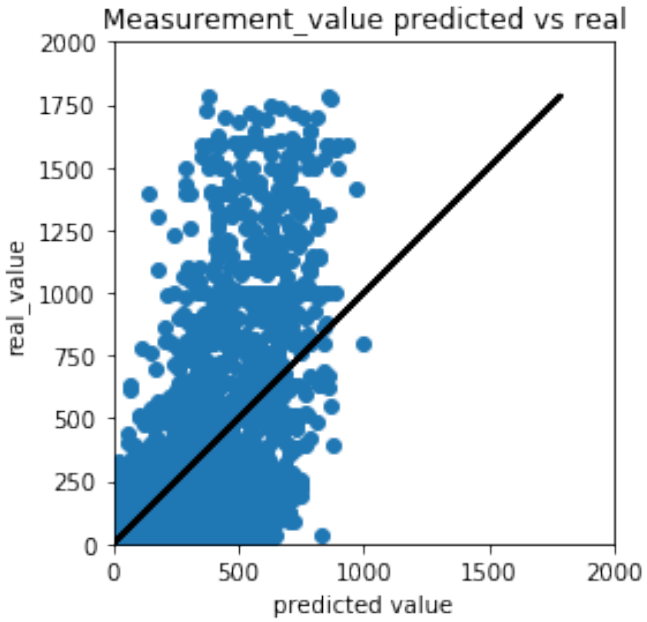

Fig. 6. Linear model training fitting

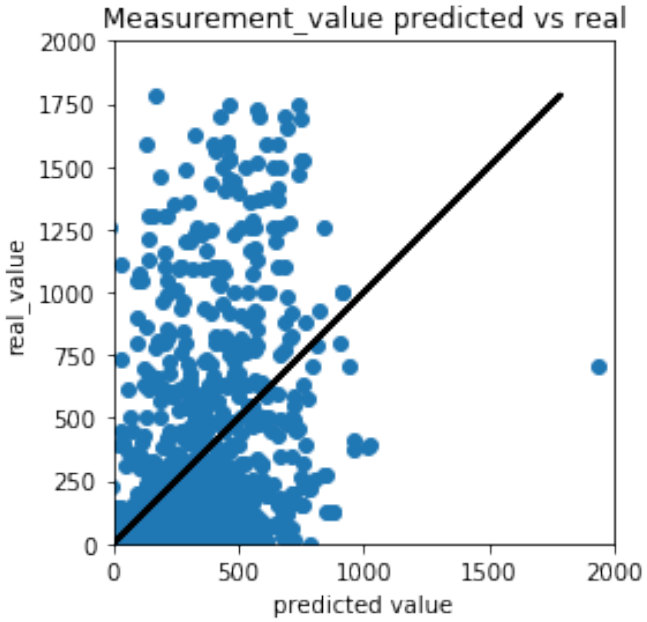

Fig. 7. Linear model test fitting

(2) ANN (2048-1024-2048):

Training MSE: 87789.164062500

Test MSE: 110842.968750000

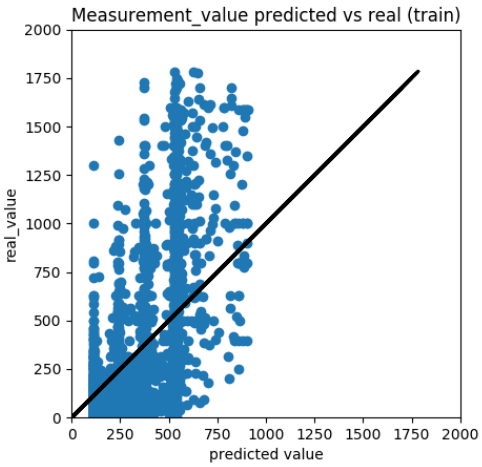

Fig. 8. ANN model training fitting

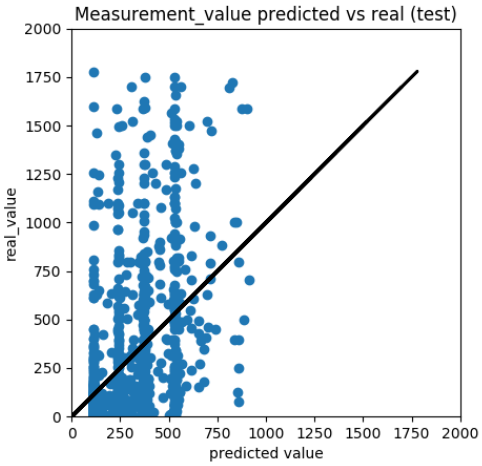

Fig. 9. ANN model test fitting

(3) CNN (2050-410(5)-82(8)):

Training MSE: 5032.369628906

Test MSE: 167180.968750000

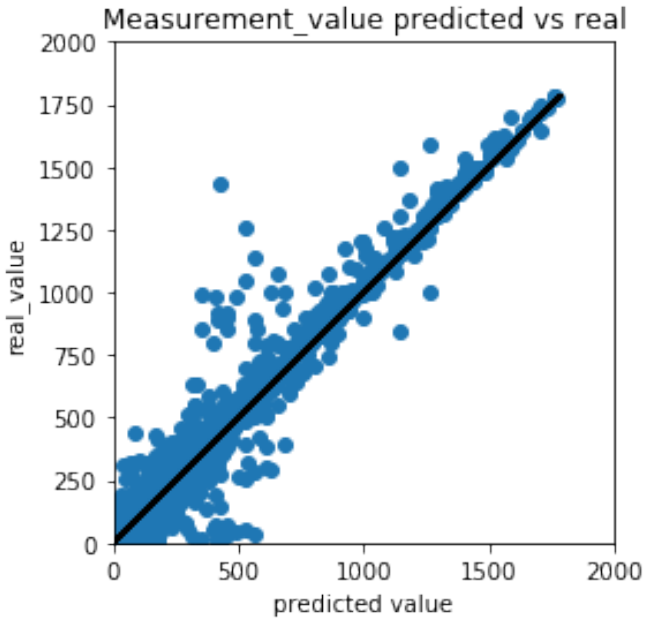

Fig. 10. CNN model training fitting

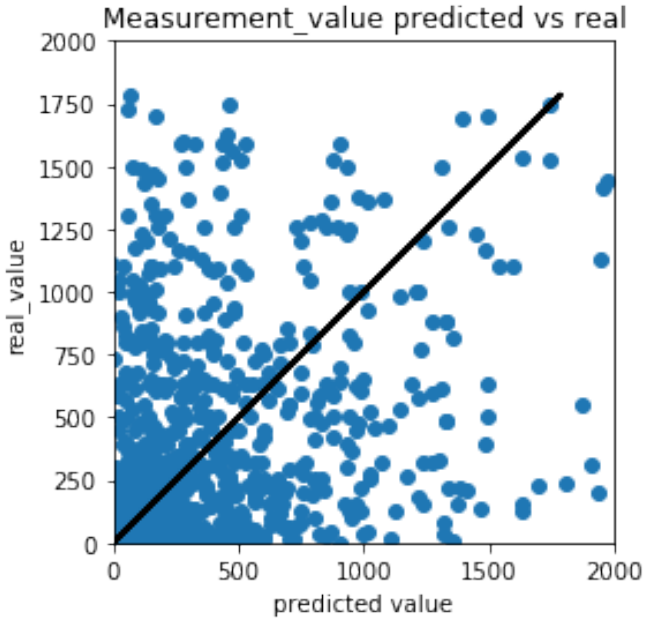

Fig. 11. CNN model test fitting

It can be easily observed that the CNN model overfit the training set (also with much lower training MSE, compare with the other two models). However, compare with the other two models, even it has larger test MSE, the plot of model test fitting shows that it is still more robust than the other two models. In addtion, it also turns our underftting issues into overfitting issues, which are easier to handle using different existing techniques such as L1 and L2 regularization. Due to limited time frame and computational power, regulariztion would be done in future work.

.

IV. Conclusion

For this project, we test several basic deep learning models on both classificaiton and regression tasks in drug discovery. We use early stopping and L2 regularizers to improve model performance on the test set. The accuracy of target classifcation using deep learning is reasonble and can be potentially further improved by adding more appropriate regularizers. For regression task on measurement value prediction, we eventually get a CNN that fits the training set well, after numerous attempts of different hyperparameters and model parameters choices. Regularization strategy can also be used since the situation is changing from under-fitting to over-fitting.

However, we should do more studies on the convergent behavior of deep learning model. If we can mathematically understand problems such as why and when ANN would converge to the mean of data, we can design better and more robust models.

For future work, more regularizers can be analyzed and added to my models. In addtion, more evalution metrics, such as AUC and F1 score for multi-class classification, should be used to further validate whether my models are robust. In addtion, the hyperparameter of L2 regularizers was not optimized due to limited computational power. We might be able to do some analysis to choose lambda/beta (decay rate) of weight decay more carefully to get better model performance. Ensemble methods would also be used in the future for performance improvement.

After we gain a basic understanding of how to set up parameters and regularizers to make deep neural network work reasonable well on both classification and regression tasks for this small dataset, we can study and apply multi-task DNN (as mentioned in [4] and [5]) to this dataset (or larger dataset), to see whether the model can learn acorss different tasks to boost the overall performance.

References

[1] "Introducing AtomNet-Drug design with convolutional neural networks," Dec. 2, 2015 [Online], Available: http://www.atomwise.com/introducing-atomnet

[2] J. Markoff, "Scientists see promise in deep-learning programs," Nov. 23, 2012 [Online], Available: http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html?_r=2&

[3] H. L. Morgan, "The generation of a unique machine description of chemical structures-A technique developed at chemical abstracts service," *J. Chem*, pp. vol. 5, 107-113, 1965.

[4] G. B. Goth, N. O. Hoda, and A. Vishnu, "Deep learning for computational chemistry," *Journal of Computational Chemistry*, 529, 2017.

[5] T. Unterthiner *et al*., *Conference Neural Information Processing Systems Foundation*, 2014.

[6] B. Ramsundar *et al*., arXiv:1502.02072 2015.

[7] A. Karpathy, "CS231n Convolutional Neural Networks for Visual Recognition," Stanford University, 2017.

[8] I. Goodfellow, Y. Bengio, and Aaron Courville, *Deep Learning*, MA: MIT Press, 2016.