Machine Learning Fall 2016

# Analysis on Deep Learning Methods for Predicting Patient Survival

Aaron Girard, Bonggun Shin, Ethan Zhou,
Henry Doupe, Henry(Yu-Hsin) Chen

December 22, 2016

## 1. Introduction

[Authors: Bonggun Shin and Henry Doupe]

In this work, we propose various methods and architectures of deep learning models with aims to improve performance in predicting patient survival with Glioma brain and breast cancer. Deep learning (DL) has shown success in various applications, particularly in learning non-linear and complex features. DL models tend to outperform statistical models when trained on large data sets such as image-net. However, the domain of this project, cancer data, suffers from the problem of limited data set size, since the complexity of DL models has high potential to over-fit the training data. Moreover, DL models can be seen as computationally inefficient due to the large number of features they try to learn.

Given this constraint, we propose two primary DL neural networks, feed-forward and convolutional, combined with regularization techniques, such as dimensionality reduction and residual network, to reduce the possibility of over-fitting. The Concordance-Index (C-Index) is used to measure the performance each of the proposed models and methods.

Among various deep learning methods, Convolutional Neural Networks (CNN) has shown success in various applications, such as computer vision [1, 2], speech recognition [3, 4], and sentiment analysis [5, 6] because CNN's provide an effective way of pooling adjacent data to be fed into the next layer by selecting the most salient information. It is also efficient because model weights are shared across different regions (or features). Therefore, CNN can be used as a light weight building block for deeper models. CNN is typically used on data sets with an obvious structure such as image data, but the survival data set only has a single high dimensional feature. We created image-like features in the survival data sets by stacking autoencoded and PCA features to create a 2D feature space. Although this artificial image allows us to technically apply CNN, it is not clear that CNN would perform well in predicting survival values since there is no explicit structure in the features. For this reason, we restrict the filter size to one so that the convolution can be done only along the same feature dimensions. The details of the CNN models are described below.

Our Giloma brain tumor data set is referred to as GBMLGG, and the breast cancer data set is referred to as BRCA. The GBMLGG data set has 1,137 patients and 17,568 features with 44% of the patients being censored. The BRCA data set has 1,098 patients and 17,584 features with 86% of the patients being censored.

# 2. Data Pre-processing

## 2.1 Selection and Imputation

[Author: Bonggun Shin]

Due to the incompleteness of the data sets, some greedy and predetermined rules for data selection and imputation are made in this work. The goal of such pre-processing is not only to remove useless data samples but also to prepare the feature values so that the data can be utilized for model building. We decided to conserve as much data as possible and focused on only the "nan" values in the data sets. Here are the three rules chosen for data selection and imputation:

1. All "nan" feature values are imputed with the mean values of the feature.
2. Samples with "nan" survival time are excluded from the data set.
3. The feature values of same features are normalized to between -1 and 1.

## 2.2 Feature Dimensionality Reduction

[Author: Henry(Yu-Hsin) Chen]

The two data sets have in average around 17,000 features for the samples, and it is impractical and computationally expensive to apply them all for building complex deep-learning models. Features used in this work are reconstructed through the method of Principle Component Analysis(PCA) [7] and Auto-encoder [8, 9]. The resultant features aim to reduce the complexity and to capture the linearity and non-linearity of the original feature set. PCA is conducted by taking advantage of existing library, Scikit Learn[1], and a simple feed-forward auto-encoder with one layer encoder and decoder is implemented and trained with mean squared error lost function in this work. In general, it takes longer for the training error to converge as the desired dimension of the reduced feature increases. The structure of the auto-encoder is shown in Figure 1.

# 3. Approach

[Author: Henry(Yu-Hsin) Chen]

In this work, several approaches are taken to gauge the performances of statistical models and deep-learning models on survival analysis of the GBMLGG and BRCA data sets. Elastic Net and feed-forward neural network are used as baseline approaches in comparison

[1] http://scikit-learn.org

to our more advanced deep-learning models, such as convolutional neural network(CNN) and its variations of implementations.

## 3.1 Model Selection and Validation

[Authors: Aaron Girard and Henry(Yu-Hsin) Chen]

For all our models, they are trained, validated, and tested with 70%, 15%, and 15% of the entire data set respectively, and C-index is selected as the evaluation metrics for the model performance in survival analysis. The models for testing are selected based on the lowest validation loss during model training. However, due to the difference in complexity of our models, the model validation processes are slightly different for the our baseline and advanced deep-learning approaches.

Our Elastic net and feed-forward neural network models are evaluated using 10-fold cross validation. For each fold, a model is trained and evaluated on validation data for number of times, the model is then evaluated on the test data, and model parameters recorded. Our CNN models are validated slightly differently due to their complexity. For each network, 10 models are trained, and the average of their training, validation, and testing results are reported. Data are shuffled and selected randomly for splitting for each trial to preventing model over-fitting to a certain portion of the data set.

## 3.2 Baseline Models

### 3.2.1 Elastic Net

[Authors: Henry Doupe]

Elastic Net (EN) with cox regression is used as the statistical baseline method. The package, glmnet[2], and its Python wrapper, glmnet_python[3], are used to implement this model [10]. EN was chosen for its own merit and the insightful contrast it provides with the primary model which uses Deep Learning. Since EN uses the Cox Proportional Hazards Model [11], it models the survival nature of the data well. It also addresses the high dimensionality of the data with the regularization penalty as shown below.

$$P_\alpha(\lambda,\ \beta)\ =\ \lambda(\alpha\sum_{i=1}^{p}|\beta_i| + \frac{1}{2}(1-\alpha)\sum_{i=1}^{p}{\beta_i}^2)$$

The parameter, $\lambda$, provides the level of the regularization, and $\alpha$ moves the model towards LASSO regression when it approaches one or to Ridge regression when it approaches zero. This yields the the benefit of discrete feature selection from LASSO and the ability to handle correlated features from Ridge regression.

The hyper parameters, $\lambda$ and $\alpha$, are chosen by maximizing the C-Index on the validation dataset. A grid search is used over values of $\alpha$ from 0 to 1 increasing by increments of 0.1. Then, for each $\alpha$, the package, glmnet, estimates a model for 30 values of $\lambda$. The value, 30, was chosen because it appeared to give sufficient level of resolution but was still computationally feasible.

[2]https://cran.r-project.org/web/packages/glmnet

[3]https://github.com/bbalasub1/glmnet_python

### 3.2.2 Feed-Forward Neural Network

[Authors: Aaron Girard]

A Feed Forward Neural Network (FFNN) is used as our second baseline method. The model was chosen as a secondary approach because of its simpler implementation compared to other more advanced neural network models proposed as the in this work. Thus, FFNN is an ideal candidate that provides performance insights of deep-learning methods on our task. The model implemented, as illustrated in Figure 2, has a single hidden layer with one hyper-parameter, the number of hidden nodes. It uses mean squared error loss function for back-propagation. The models are trained over different feature sets of the BRCA and GBMLBGG data. These sets are the original raw features, PCA reduced features, and auto-encoded features.

## 3.3 Primary Deep-learning Models

### 3.3.1 Basic Convolutional Neural Network

[Authors: Bonggun Shin and Henry(Yu-Hsin) Chen]

Despite numerous options of neural networks, a convolutional neural network (CNN) is chosen to be the primary approach for our task. Unlike conventional neural network models that consider all information at once, CNN selects the most salient information through pooling operations. This allows different combinations and subsets of features to be considered in predicting patient survival. Furthermore, the model is efficient since its feature weights are shared across regions. These properties of CNN motivate us to use this network over the others as the building blocks for more advanced deep-learning models for survival analysis.

Our basic model for CNN, as shown in Figure 3, has one convolutional layer and one maximum pooling layer to capture meaningful features for predicting patient survival. A dropout layer is then added to randomly set a portion of the feature weights to zero in order to preventing the network from over-fitting to the training data. For each patient, given two feature sets, PCA reduced and auto-encoded features, of size $p$ each, an artificial image of dimension of $2 \times p$ is created as input to the network. With filter size of 1 and $nf$, number of filters, as a hyper-parameter, the convolutional layer convolutes over the input image and constructs a representation of dimension $2 \times nf$. The representation is then maximum pooled over its columns to generate a feature weight vector of length $nf$, which is used for prediction the survival time of the given patient.

### 3.3.2 Modified Basic Convolutional Neural Network

[Authors: Henry(Yu-Hsin) Chen]

Using our basic CNN model as a sub-network, a modified version is proposed to provide additional insights into the given features with the intention of improving predictive power of the original model. The modified CNN, as presented in Figure 4, takes the pooled output representation of size $nf$ concatenated with high-variance features of size $p'$ as the input to a feed-forward neural network for prediction. High-variance features are extracted from the raw feature set and excluded from the PCA reduced and auto-encoded feature sets.

The top $p'$ features with the highest variance are selected. By taking consideration of these features separately, helpful "noise" is introduced to the network. Due to their high variance, we expect them to provide additional predictive power and, thus, improve our model's performance.

### 3.3.3 Multi-layer Convolutional Neural Network

[Authors: Bonggun Shin]

Building upon our CNN structure, we add more CNN blocks and regularization in order to discourage over-fitting. The structure of the basic CNN building block is shown in Figure 5. The CNN block contains sub-modules, such as batch normalization(BN) [12], ReLu [13] activation, and convolutional operations. Different combinations of these sub-modules demonstrate different behavior due to how each sub-module controls information propagation.

The combination of sub-modules is chosen based on the previous research of computer vision proposed by He at el. [14]. It consists of batch normalization sub-modules that precede convolutional operations. As the layer goes deeper, the probability of signal distortion gets higher. Thus, normalization is performed before applying the non-linear transformation in order to help regularize and keep the weights from diverging. With this design of a CNN block, we then construct a deep and multi-layer CNN model by stacking the blocks on top of each other, as shown in Figure 5.

### 3.3.4 Residual Neural Network

[Authors: Bonggun Shin]

Since we constructed a deeper and, thus, more complex multi-layer CNN model, we expect the performance and the difficulty of training to increase. Hence, we use the reformulation method, residual networks (ResNet) as introduced by He et al. [15].

ResNet creates a bypassing channel for feature information to flow directly from the input to the output of a model block, as illustrated in Figure 6. The channel allows proceeding layers to utilize "fresh" feature information rather than diminished feature information. A residual block is defined as $y = x + F(x, W)$ where $F$ is a residual sub-network and $x$, $y$, $W$ are the input, output, and sub-network weight, respectively. In this work, two convolutional layers are used to estimate the change in $x$ as suggested by He et al. [15].

## 3.4 Bayesian Optimization

[Authors: Ethan Zhou]

In order to gauge the performance of our prototype models, we follow a line-search-like procedure where initial hyper-parameters are randomly initialized and individually optimized for performance. The procedure, however, is limited by its inability to change multiple hyper-parameters simultaneously, so we used Bayesian Optimization as a smarter approach for hyper-parameter tuning. Rather than looking at one parameter at a time, the approach takes a more extensive view at the search space and estimates the changes for the parameters based on probability.

Due to time constraints, we only applied Bayesian Optimization on one model, the basic convolutional neural network model. The framework is set to optimize over two hyper-parameters, number of filters and dropout keep rate, since they are directly related to the network. The search is conducted over a range of values, predetermined based on results of several trials of manual tuning: 30-70 for numbers of filters and 0.6-1.0 for dropout keep rate.

# 4. Result and Discussion

## 4.1 Baseline Models

### 4.1.1 Elastic Net

[Authors: Henry Doupe]

Our Elastic Net model achieved testing C-Index of 70.22% and 49.31% on GBMLGG and BRCA, respectively, using raw features. The performance improves to 72.29% and 58.00% maximum testing C-index when we applied PCA reduced feature of dimension 200 and 800 from GBMLGG and BRCA respectively. Results are shown in Table 1.

Elastic Net performed better on the GBMLBGG data set than the BRCA data set. Furthermore, less features are required to reach a stable C-Index level for the former data set. This agrees with Figure 7 which shows that less principal components are required to explain the same level of variance in the GBMLBGG data set than in the BRCA data set. On average, the standard deviation of C-Index over CV folds for the BRCA data set (8.0%) was twice as great as that of the GBMLBGG data set (4.0%). These results most likely arise from the BRCA data set being more complex and, perhaps, containing more noise.

### 4.1.2 Feed-forward Neural Network

[Authors: Aaron Girard]

Overall, the Feed-forward Neural network performed better on the GBMLBGG data set. Figure 17 shows the spread of the testing C-Index accuracy values on the FFNN with 400 hidden units, which was the best performing model. Past 400 hidden units, the models gradually began to over-fit. On average, the model achieved 74.37% and 71.36% accuracy using data preprocessed using PCA and autoenconding, respectively. On the BRCA data, the FFNN achieved 66.60% and 63.82% accuracy using PCA and autoenconding respectively. The reduced accuracy on the BRCA data set is consistent with other results presented in this paper. In sum, a simple FFNN with little tuning performed better than chance creating the groundwork for exploration of this data with more complex NN frameworks.

## 4.2 Primary Deep-learning Models

### 4.2.1 Basic Convolutional Neural Network

[Authors: Henry(Yu-Hsin) Chen]

Our basic CNN models yield 77.89% and 76.14% average validation and testing C-index on the GBMLGG data set and 67.00% and 65.18% on the BRCA data set, shown in Table 1.

To better understand the result of the model, we analyze the its changes in performance with respect to individual hyper-parameters. From Figure 8, we observe that as the input feature dimension increases, the training and validating C-index increase. However, the test results only show initial improvements and decline afterward. The trend is primarily attributed to model over-fitting due to the increase in feature dimensions. From Figure 9, the same trend can be found as the number of filters increases. A larger number of filters means more information is convoluted and generated from input images, and thus the model suffers the same issue of over-fitting. From Figure 10, we see that the models perform better with a dropout keep rate around 0.8, which means only 80% of the feature weights are used for prediction and back propagation. Though the dropout layer's purpose is to prevent model over-fitting, a lower dropout keep rate, or a more regularized model, does not necessarily yield a better result in testing C-index due to the decreased feature information provided. On the other hand, a non-regularized model where the dropout keep rate is 1.0 will highly over-fit to the training data.

### 4.2.2 Modified Basic Convolutional Neural Network

[Authors: Henry(Yu-Hsin) Chen]

With the introduction of high-variance features as "helpful" noises, our modified CNN model provides slight improvement in performance compared to the basic CNN. The model yields 77.98% and 76.65% average validation and testing C-index on the GBMLGG data set and 81.62% and 70.72% on the BRCA data set, as shown in Table 1.

From the result, we see that the modified model's performance improves more on the BRCA data set than on the GBMLGG data set when compared to the basic model. This arises primarily out of the difficulty of the data sets and the additional features required to describe them well. Hence, when we take a closer look at the added hyper-parameter, number of hidden nodes, of the modified model, we can find the same over-fitting trend as the number of hidden nodes increases. The trend is illustrated in Figure 11.

### 4.2.3 Multi-layer and Residual Convolutional Neural Network

[Authors: Bonggun Shin]

With deeper CNN models, we are able to achieve 73.54% and 75.24% average validation and testing C-index on the GBMLGG data set and 73.79% and 69.73% on the BRCA data set with the multi-layer CNN model. With our residual network model, we reach 73.87% and 78.02% average validation and testing C-index on the GBMLGG data set and 68.07% and 73.14% on the BRCA data set.

We vary the depth of the two deep models in order to see to see the effect of residual network's bypassing mechanism. For the GBMLGG data set shown in Figure 12, as the depth of the models increases, the performance of ResNet increases slightly, while that of multi-layer CNN model remains the same until the depth 8. From the depth 16 and on, performance of the multi-layer CNN model plummets to below 50%. The efficacy of the ResNet based model is more noticeable in the results for the BRCA data set 13.

The median of the C-indices for the ResNet model gradually increases with some slight oscillation, while that of the multi-layer CNN model constantly goes down as the models grow deeper. Despite the fact that the multi-layer CNN model sometimes outperforms ResNet model, the general trends of performance are well in alignment with the hypothesis that the bypassing mechanism is effective in tackling the over-fitting issue that arises from the deeper and more complex models. Thee results indicate that ResNet is a better deep model than multi-layer CNN for our task.

By comparing our deep models with our single CNN models as mentioned before, we discover that deeper models do not necessarily improve performance. Since our data sets suffer from high dimensionality, one explanation is that the complexity of our deep CNN models might be too high. Hence, a single, basic CNN block might be more appropriate for our task. Another possible explanation is that the input features might contain redundant or noisy information since our dimensionality-reduced features are constructed based on all given features without any filtering or selection. Therefore, noise reduction should be considered to increase the performance of the deep models.

## 4.3 Bayesian Optimization

[Authors: Ethan Zhou]

The resulting outputs from Bayesian Optimization do not provide a clear improvement over the original scores. Number of filters tended towards the upper boundary for BRCA (63.43) while GBMLGG (56.71) remained in the middle ground. Dropout/keep rates remained consistently around 1.0 (1.0 for BRCA and 0.97 for GBMLGG).

Our Bayesian Optimization is conducted using both five and ten iterations. With five iterations, the overall trend-line generally follows that of the original results, but had greater variance. To decrease variance, we performed testing on the BRCA data set with ten iterations per step as shown in Figure 16 and only up to 60 features, due to time constraints. The result is indecisive and more testing is required to provide a complete perspective of the benefits from applying Bayesian Optimization.

# 5. Conclusion

[Authors: Henry(Yu-Hsin) Chen]

Our work focuses on analyzing the performances of deep-learning models for predicting patient survival time. Using a statistical model, Elastic Net, and the simplest type of neural network, feed-forward neural network, as baselines, we propose more advanced convolutional neural network and attempt to improve the prediction accuracy in C-index from the baseline approaches. Advantages and disadvantages of the models are also discovered and explained along the process. As result, our best performing model is able to achieve 78.02% and 73.14% testing C-index for the GBMLGG and BRCA data sets.

# 6. Appendix

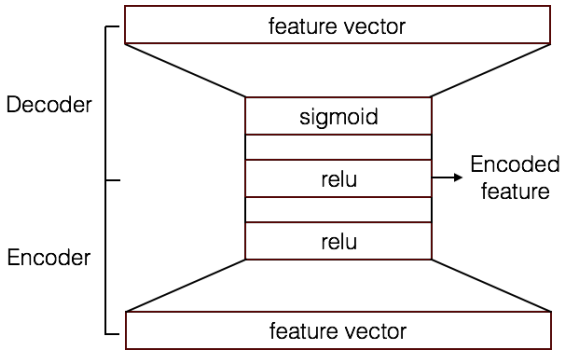


Figure 1: Architecture of our auto-encoder implementation

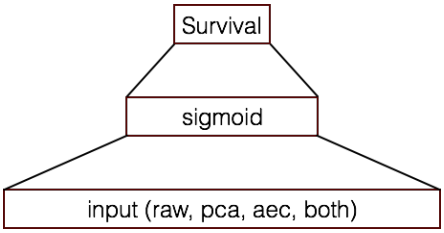


Figure 2: Architecture of our feed-forward neural network implementation

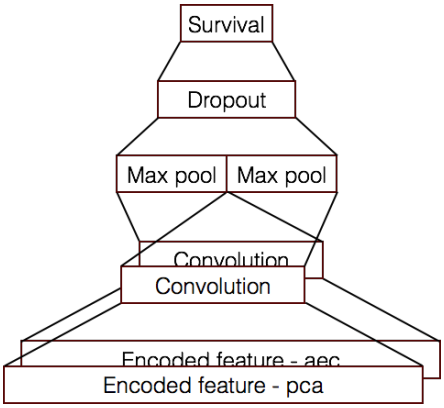


Figure 3: Architecture of our basic CNN implementation

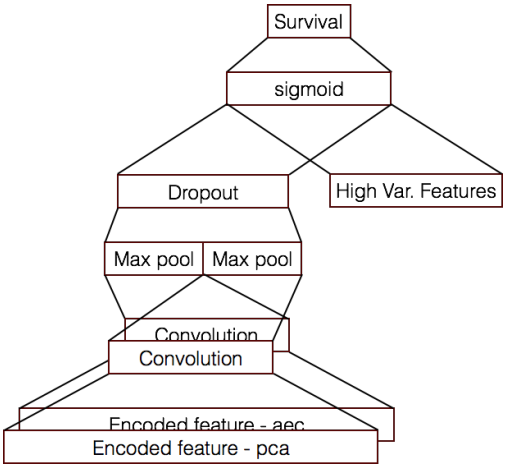


Figure 4: Architecture of our modified basic CNN implementation

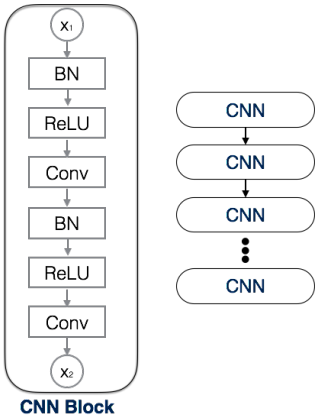


Figure 5: Architecture of our multi-layer CNN implementation

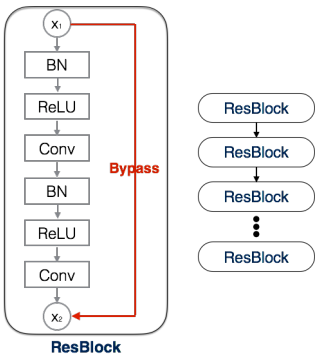


Figure 6: Architecture of our residual network implementation

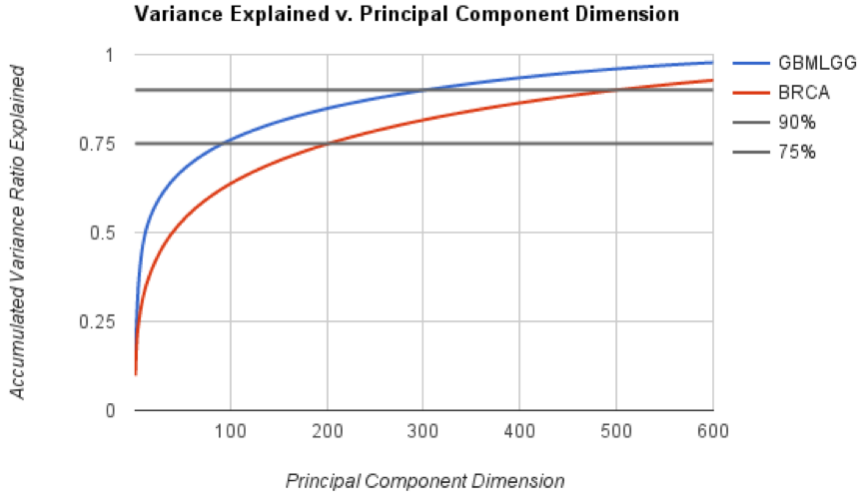


Figure 7: Analysis of PCA variance explained

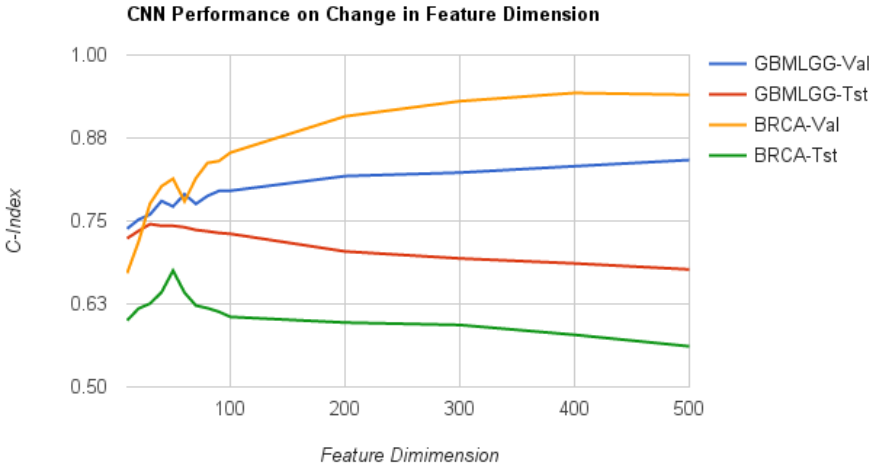


Figure 8: Performance of basic CNN model with changes in feature dimensions

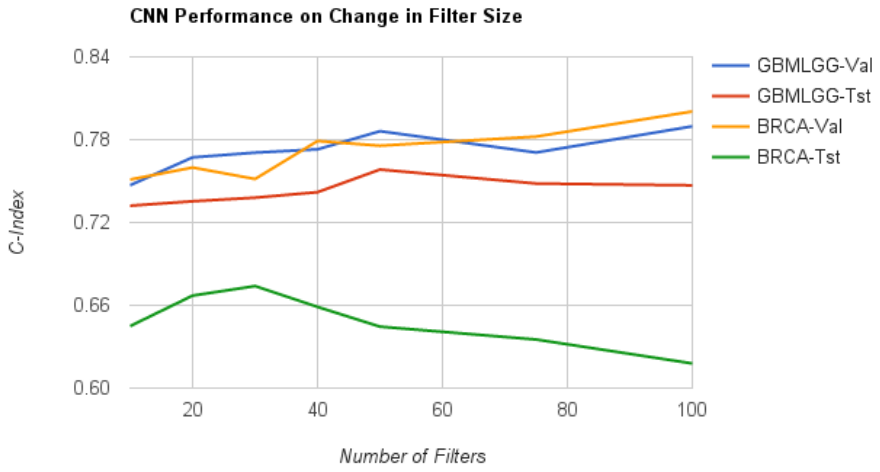


Figure 9: Performance of basic CNN model with changes in numbers of filters

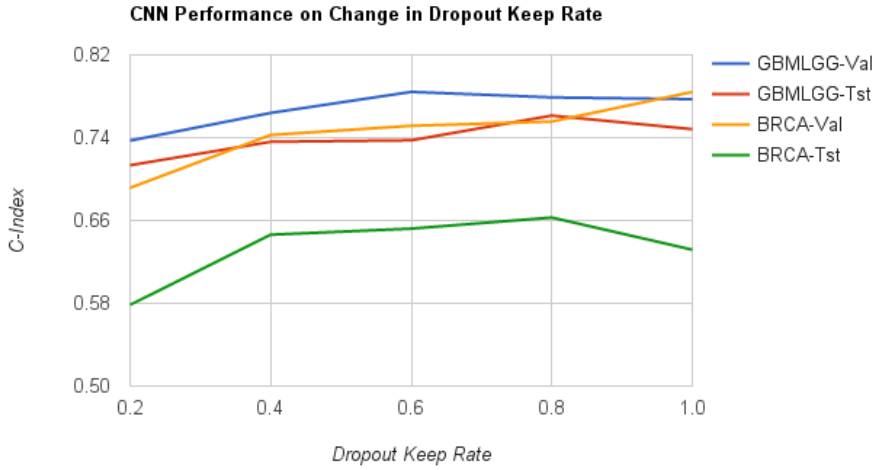


Figure 10: Performance of basic CNN model with changes in dropout keep rates

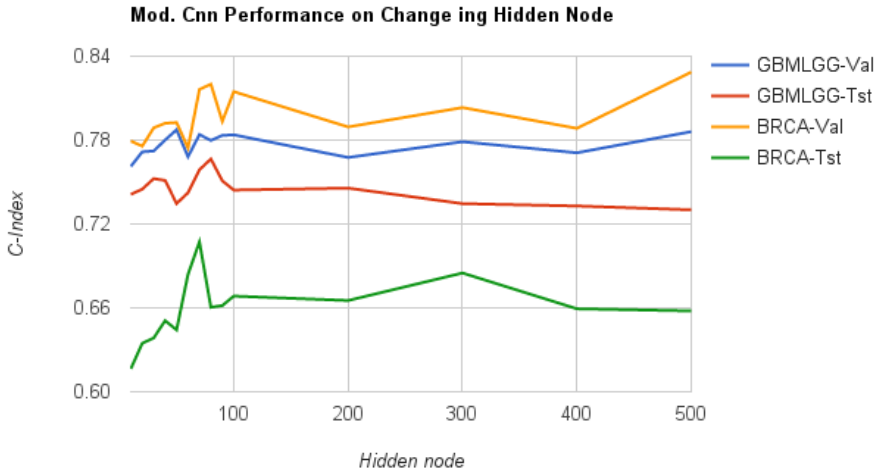


Figure 11: Performance of modified CNN model with changes in numbers of hidden nodes

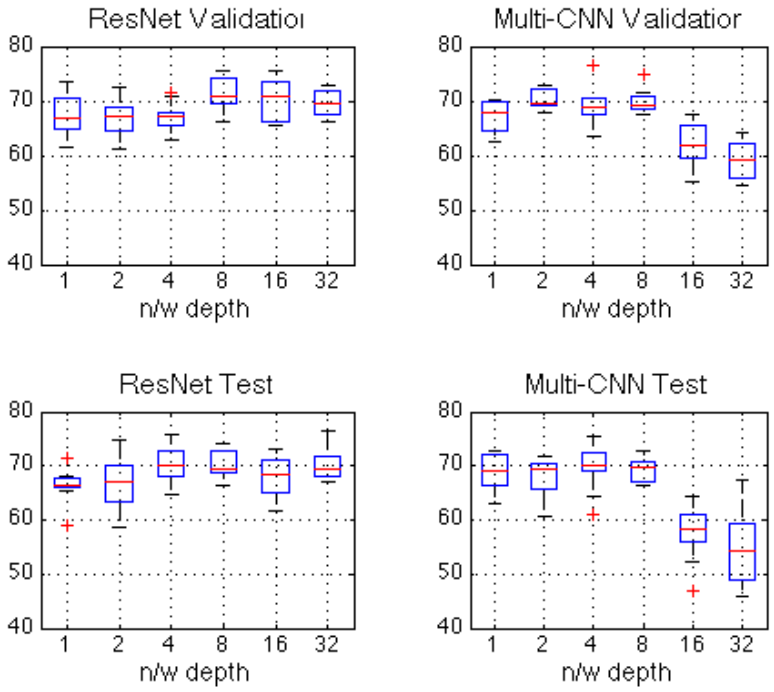


Figure 12: Performance of multi-layer and residual CNN models on GBMLGG data set

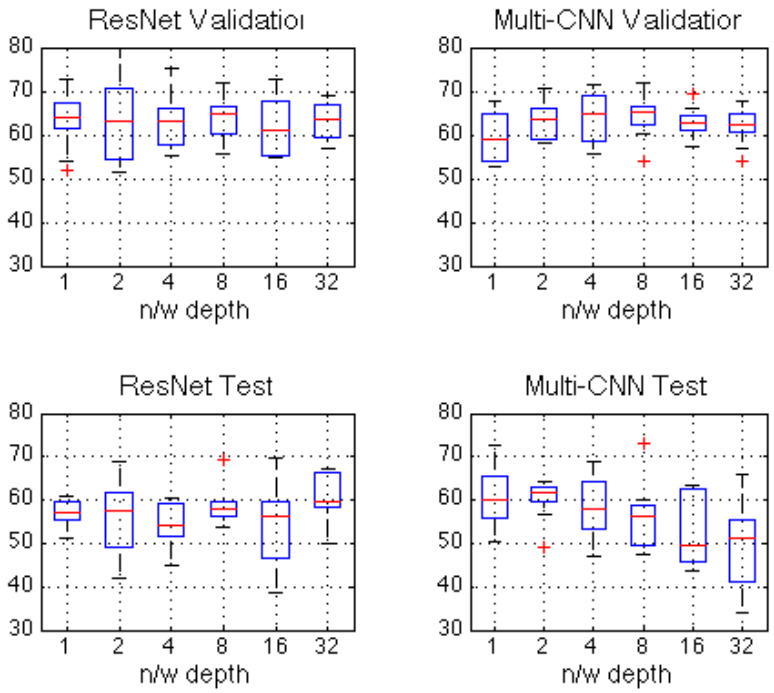


Figure 13: Performance of multi-layer and residual CNN models on BRCA data set

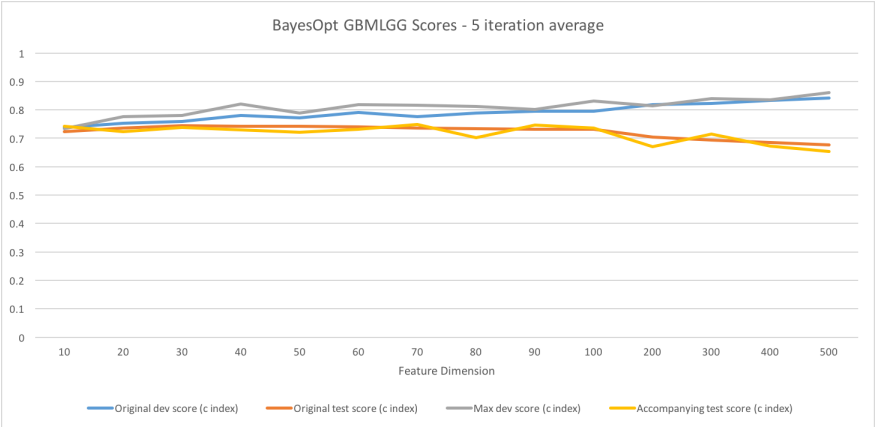


Figure 14: Basic CNN model with Bayes Optimization on GBMLGG data set

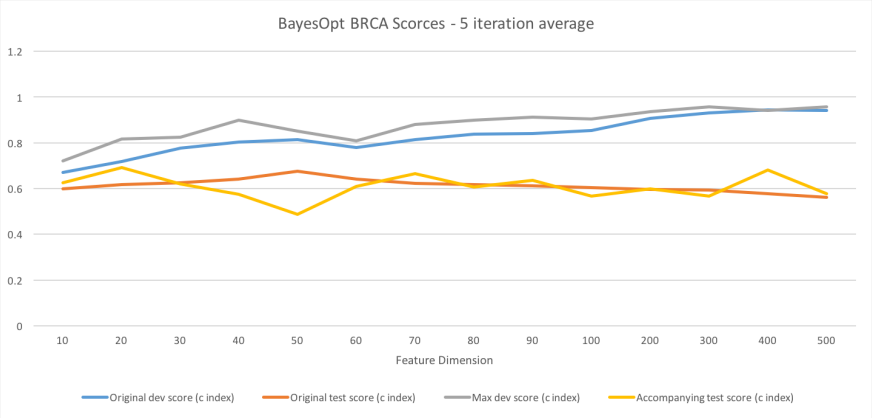


Figure 15: Basic CNN model with Bayes Optimization on BRCA data set (5 iterations)

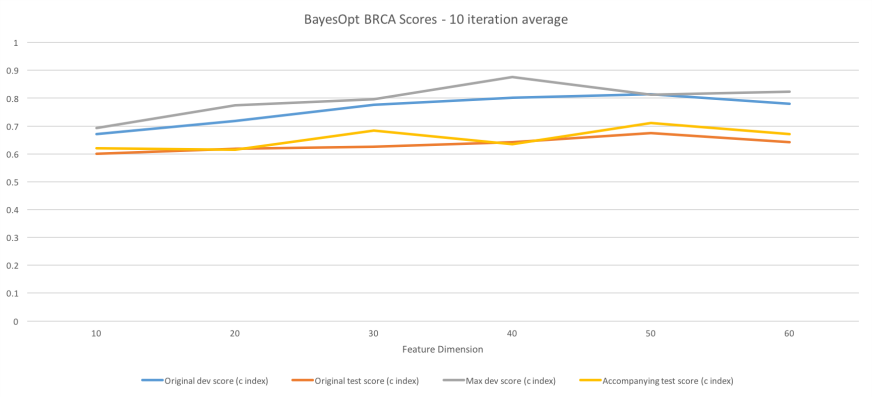


Figure 16: Basic CNN model with Bayes Optimization on BRCA data set (10 iterations)

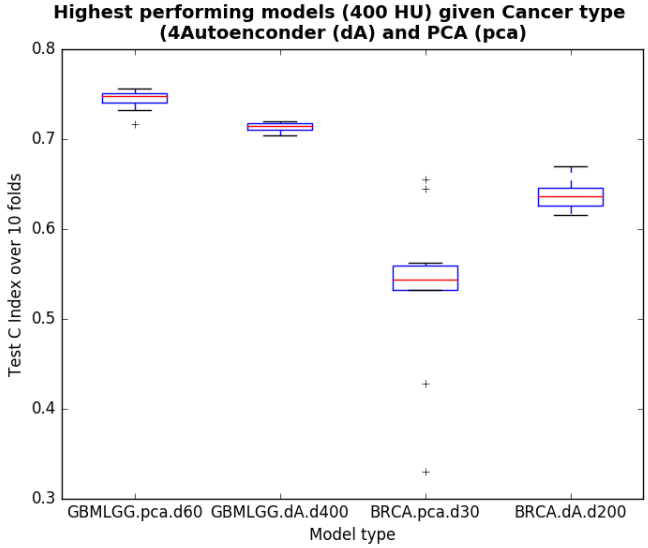


Figure 17: Performance of FFNN on GBMLGG and BRCA data set with autoencoder and pca

| Model & Features | GBMLGG Valid | GBMLGG Test | BRCA Valid | BRCA Test |
|---|---|---|---|---|
| Elastic Net (Raw) | – | 70.22% | – | 49.31% |
| Elastic Net (PCA) | – | 72.29% | – | 58.00% |
| Feed-forward NN (Raw) | 75.00% | 71.23% | 67.00% | 65.18% |
| Feed-forward NN (AEC) | 73.88% | 71.36% | 56.36% | 63.82% |
| Feed-forward NN (PCA) | 73.72% | **74.37**% | 69.28% | **66.60**% |
| Basic CNN | 77.89% | 76.14% | 75.56% | 66.29% |
| Modified CNN | **77.98**% | 76.65% | **81.62**% | 70.72% |
| Multi-layer CNN | 73.54% | 75.24% | 73.79% | 69.73% |
| Residual Network | 73.87% | **78.02**% | 68.07% | **73.14**% |

Table 1: Validation and test results in C-index of all models

# References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun, "Pedestrian detection with unsupervised multi-stage feature learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3626–3633.

[3] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[4] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[5] B. Shin, T. Lee, and J. D. Choi, "Lexicon Integrated CNN Models with Attention for Sentiment Analysis," ArXiv, Tech. Rep. 1610.06272, 2016. [Online]. Available: https://arxiv.org/abs/1610.06272

[6] S. Poria, E. Cambria, and A. Gelbukh, "Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis," in *Proceedings of EMNLP*, 2015, pp. 2539–2544.

[7] K. Pearson, "Principal components analysis," *The London, Edinburgh and Dublin Philosophical Magazine and Journal*, vol. 6, no. 2, p. 566, 1901.

[8] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representations by error propagation, parallel distributed processing, explorations in the microstructure of cognition, ed. de rumelhart and j. mcclelland. vol. 1. 1986," *Biometrika*, vol. 71, no. 3, pp. 599–607, 1986.

[9] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

[10] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for cox proportional hazards model via coordinate descent," *Journal of Statistical Software*, 2011.

[11] D. Cox, "Regression models and life-tables," *Journal of the Royal Statistical Society*, 1972.

[12] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015.

[13] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.

[14] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” *ECCV*, 2016.

[15] ——, “Deep residual learning for image recognition,” *CVPR*, 2016.