8. DEEP LEARNING MODELS TO PREDICT ASD GENES USING GENE SEQUENCE ENCODING

The research is further progressed to the next stage with encoding schemes to transform the gene sequences to be given as raw input to deep models. Contemporary deep learning techniques differ from traditional machine learning in the manner representations are learned from the raw data. The representations learned through the deep architectures are self taught, data-driven and does not require any domain knowledge and feature engineering. These features are not designed by human engineers and not extracted manually as done in traditional machine learning. In deep learning methods feature learning and classification happens in the unified framework which helps to eliminate the tedious task of manual feature extraction. The key aspect of deep learning is that every layer in deep architecture generates a representation of the observed patterns based on the data it receives as inputs from the layers below, through optimization of a local unsupervised criterion. The deep architecture is thus capable of capturing unbiased, intellectual, assorted features which are vital for building the classifiers. Hence it is intended in this work to exploit the self-learning power of deep learning models by utilizing the gene sequences as raw input data and thereby avoid the time consuming task of feature engineering. A new mechanism of encoding the gene sequences, codon encoding and one hot encoding is proposed here to transform the sequences to provide direct input to the deep architectures for building the classifiers.

Chapter 6 discussed the development of deep neural network model which exploited user defined features for predicting ASD causing genes, their susceptibility to the disorder and triggering mutations. These tasks were implemented by utilizing the RNN and its variants LSTM, GRU architectures and have been elaborated in chapter 7. This chapter describes the method of two encoding schemes, codon encoding and one hot encoding and the problem modeling of deep learning models to predict ASD causing genes.

8.1 CODON ENCODING OF GENE SEQUENCES FOR DEEP LEARNING BASED PREDICTION OF ASD CAUSING GENES

Identifying genes causing the genetically transmitted Autism Spectrum Disorder (ASD) is still a challenging task. The rapid developments in the design of deep architecture models have

shown considerable success in sequential data processing tasks. As genomics data is dependent on domain specific experts for identifying relevant contributive features and as extracting hand-crafted attributes involves much time, an alternate effective solution is the need of the hour. Deep learning models examine the data to discover associations among the features and enable faster learning without being explicitly programmed to do so. Hence the primary goal of this work is to classify the ASD genes by employing deep learning based models without feature engineering. To prepare the input vectors, new encoding scheme have been defined and ASD gene sequences are transformed into numerical sequences.

Codon Encoding and Dataset Preparation

Codons are unlike in different gene families and are good discriminators for differentiating the genes and mutations. Codons are the nucleotide triplet that encodes an amino acid. The relation between the sequence of bases in DNA or its RNA transcripts and the sequence of amino acids in proteins is given by the genetic code. The features of the genetic code are given below.

- An amino acid is encoded by three nucleotides and proteins are built from a basic set of 20 amino acids.
- The code is nonoverlapping. Consider a base sequence ACCGTA. In a nonoverlapping code, ACC designates the first amino acid, GTA the second, and so forth. Genetics experiments again established the code to be nonoverlapping.
- The genetic code is degenerate. Some amino acids are encoded by more than one codon, and there are 64 possible base triplets and only 20 amino acids. Out of the 64 possible triplets only 61 specify particular amino acids and 3 triplets refer the stop codons that designate the termination of translation. Thus, there is more than one code word for most of the amino acids. For instance, there are six different ways to code the protein leucine. The code is highly degenerate and so only tryptophan and methionine are encoded by just one triplet each. Two or more triplets encode the other 18 amino acids. Indeed, leucine, arginine, and serine are specified by six codons each.

In this work the potential of codons in recognizing the gene sequences is explored. The simulated mutated sequences undergo the process of codon encoding and are converted into records having values ranging from 1 to 64 as there are 64 possible codons. A gene sequence S is composed of codons which are substrings s_i as defined by equation 8.1.

$S= \{s_1 s_2 \ldots . s_i \}$, i ranges from 1 to n where $n = |S| / 3$ (8.1)

The genetic code showing the relation between codons and amino acid is presented in Table XXXVII.

Table XXXVII Genetic Code Showing Relation Between Codons and Amino Acid

Codon	Amino Acid	Abbre viation	Codon	Amino Acid	Abbre viation	Codon	Amino Acid	Abbre viation
TTT	Phenylalanine	Phe	CCA	Proline	Pro	AAT	Asparagine	Asn
TTC	Phenylalanine	Phe	AGA	Arginine	Arg	AAC	Asparagine	Asn
TTA	Leucine	Leu	AGG	Arginine	Arg	AAA	Lysine	Lys
TTG	Leucine	Leu	GTT	Valine	Val	AAG	Lysine	Lys
TCT	Serine	Ser	GTC	Valine	Val	AGT	Serine	Ser
TCC	Serine	Ser	CCG	Proline	Pro	AGC	Serine	Ser
TCA	Serine	Ser	CAT	Histidine	His	GCA	Alanine	Ala
TCG	Serine	Ser	CAC	Histidine	His	GCG	Alanine	Ala
TAT	Tyrosine	Tyr	CAA	Glutamine	Gln	GAT	Aspartate	Asp
TAC	Tyrosine	Tyr	CAG	Glutamine	Gln	GAC	Aspartate	Asp
TAA	Termination	Ter	CGT	Arginine	Arg	GTA	Valine	Val
TAG	Termination	Ter	CGC	Arginine	Arg	GTG	Valine	Val
TGT	Cysteine	Cys	CGA	Arginine	Arg	GCT	Alanine	Ala
TGC	Cysteine	Cys	CGG	Arginine	Arg	GCC	Alanine	Ala
TGA	Termination	Ter	ATT	Isoleucine	Ile	GGA	Glycine	Gly
TGG	Tryptophan	Trp	ATC	Isoleucine	Ile	GAA	Glutamate	Glu
CTT	Leucine	Leu	ATA	Isoleucine	Ile	GAG	Glutamate	Glu
CTC	Leucine	Leu	ATG	Methionine	Met	GGT	Glycine	Gly
CTA	Leucine	Leu	ACT	Threonine	Thr	GGC	Glycine	Gly
CTG	Leucine	Leu	ACC	Threonine	Thr	GGG	Glycine	Gly
CCT	Proline	Pro	ACA	Threonine	Thr			
CCC	Proline	Pro	ACG	Threonine	Thr			

Ex: A gene sequence AGACTGGTTCCA… consists of codons AGA CTG GTT CCA…

Each codon of the DNA sequence codes for a single amino acid and each nucleotide unit consists of a phosphate, deoxyribose sugar and one of the 4 nitrogenous nucleotide bases, adenine, guanine, cytosine and thymine represented by A, G, C and T respectively. This codon mapping is mathematically formulated as given in equation 8.2.

si ε {xp{yq{zr}}}, where <xp / yq / zr> = A / C / G / T where p, q, r=1,2,3,4 (8.2)

For example the triplet <x1 y3 z1> represents the Codon AGA.

There are 64 possible combinations of codons composed of three nucleotide bases that specify the amino acids during protein assembling. Codons are mapped onto numbers ranging from 1 to 64. The entire gene sequence S is converted into a numerical vector V of codons CO as in (8.3).

$V_j = [\ CO_i\ ]_j$ where i = 1 to nr, n=4, r=3, j =total number of gene sequences (8.3)

For example AGA is encoded to 09 as per the following scheme.

AAA	**AAC**	**AAG**	**AAT**	**ACA**	**ACC**	**ACG**	**ACT**	**AGA**	**AGC**	**AGG**	**AGT**	**ATA**	**ATC**	**ATG**	**ATT**
01	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CAA	**CAC**	**CAG**	**CAT**	**CCA**	**CCC**	**CCG**	**CCT**	**CGA**	**CGC**	**CGG**	**CGT**	**CTA**	**CTC**	**CTG**	**CTT**
17	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**
GAA	**GAC**	**GAG**	**GAT**	**GCA**	**GCC**	**GCG**	**GCT**	**GGA**	**GGC**	**GGG**	**GGT**	**GTA**	**GTC**	**GTG**	**GTT**
33	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**
TAA	**TAC**	**TAG**	**TAT**	**TCA**	**TCC**	**TCG**	**TCT**	**TGA**	**TGC**	**TGG**	**TGT**	**TTA**	**TTC**	**TTG**	**TTT**
49	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**	**64**

The codon encoded SHANK3 gene sequence is given as follows

15 36 42 22 43 38 10 38 47 46 12 27
41 25 14 26 44 11 29 10 42 19 22 32
30 21 25 ……

In this manner codon encoding is done for all 1000 sequences available in the corpus. The length of the codon encoded sequences is variable and depends on the number of codons present in a gene sequence. All sequences are not of the same length, but in order to make it uniform, 0 padding is done to make them equal in length. The length of each record is taken as 2582 timesteps of a feature vector which is the maximum number codons in a gene sequence.

The Codon Encoded Dataset (CEDS) consisting of the encoded feature vectors is created with 1000 instances of dimension 2582 where each instance is assigned with one hot encoded class label ranging from 1 to 10.

Methodology

The mutated disease gene sequences are codon encoded and used in this multi-class pattern classification problem. Four different deep models based on DNN, BRNN, LSTM and GRU architectures are built by training with the codon encoded dataset. The methodology includes three elements namely datasets creation, model building, performance evaluation and the architecture of codon encoding based deep models for identifying ASD genes is depicted in Fig.8.1

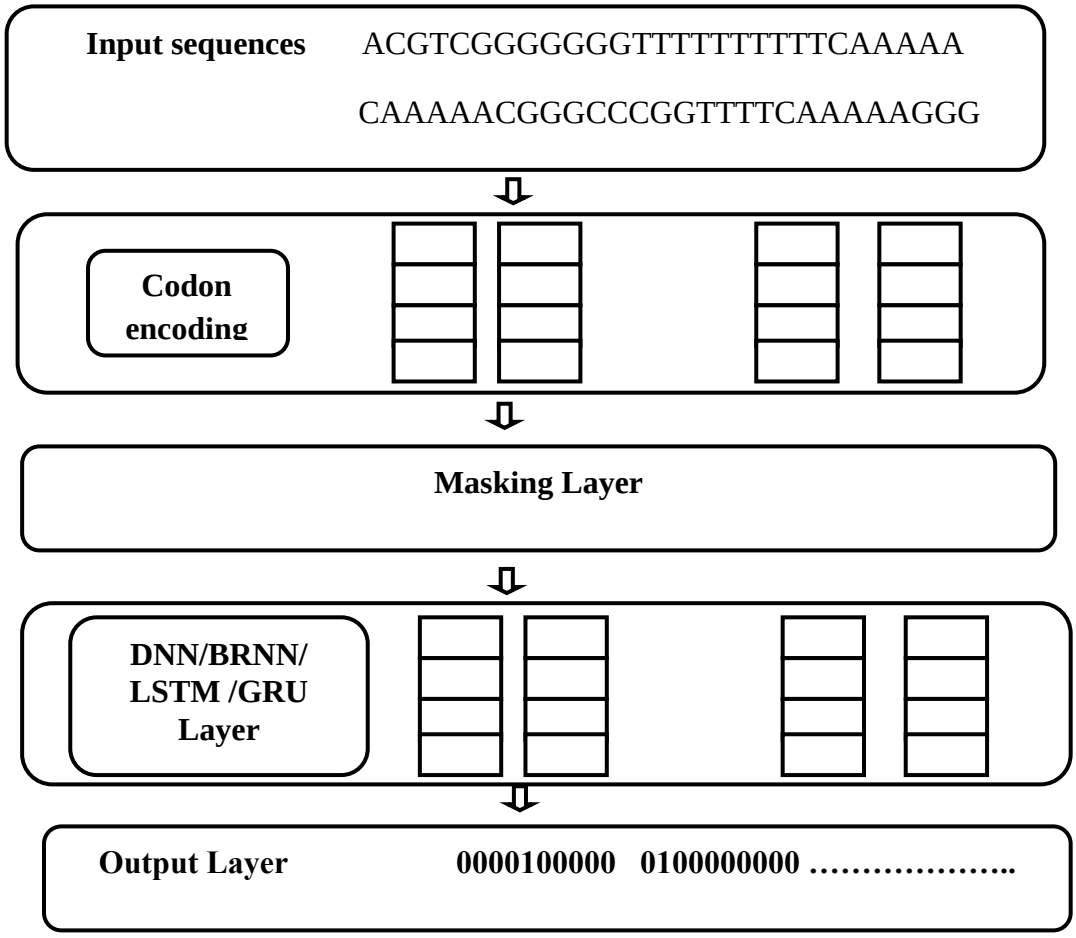

Fig. 8.1 Architecture of Codon Encoding Based Deep Models for Identifying ASD Genes

During the initial phase, the corpus built using 1000 mutated gene sequences accounting for ten types of ASD genes and four types of mutations as mentioned in Chapter 3 is used. Consequently codon encoding of these mutated gene sequences are done according to the method mentioned in the previous section. The input vectors are reshaped to have n time steps and k features, where n is the number of integers in the generated sequence and k is the set of possible integers at each time step. CEDS comprising of 1000 class labeled input vectors of dimension 2582 is used for building models.

The second phase consists of building deep neural networks that can automatically extract useful features from sequential patterns through high-level information associated with observed signals which in turn can be used for classification of pretentious genes that underlie ASD. Four different deep network architectures namely DNN, BRNN, LSTM, GRU are designed to build gene identification models. In order to ensure fair computation of the models the basic structure of the proposed deep models is designed uniformly with one input layer, 2 hidden layers with 8 memory units, a masking layer and an output layer. The output layer is a fully connected layer with 10 neurons for the 10 possible integers that may be output. The output layer provides a softmax activation function that allows the network to learn and output the distribution over the possible output values. In case of LSTM, the two stacked LSTM recurrent layers that are capable of automatically learning feature representations is used as done in section 7.2. In GRU, shared feature extraction is performed using two parallel submodels designed using Keras functional API as in section 7.3.

To improve the accuracy and efficiency of the models, a choice of hyperparameters such as batch size, epochs, dropout, learning rate and optimizer are taken into consideration. The parameters of the network are updated using mini - batch gradient descent. The epochs denote the number of times the network will work through the entire training dataset. The important regularization technique dropout enables to randomly ignore selected neurons during training. The significance of learning rate parameter is to decide the pace at which a deep model replaces the concepts it has already learned with the new ones. The optimization algorithm in a network enables to minimize the error function and Adam optimizer is the algorithm used in this implementation. The best configuration for the network is achieved by fine tuning the above mentioned hyperparameters. The four deep learning techniques such as DNN, BRNN, LSTM

and GRU have been trained with the CEDS dataset to build ASD gene type identification models.

In the concluding phase, the models are tested using 10-fold cross validation and evaluated for their predictive performance using various metrics such as precision, recall, F-measure, accuracy, log loss and specificity.

Experiment and Results

Implementation of DNN, BRNN, LSTM and GRU is done using Keras which is a high-level API for neural networks. The hyper parameters used in the previous experiments are used here also for the sake of uniformity. During training and testing, data are segmented on mini-batches of size 64 data segments. Varying dropouts from 0.2 to 0.5 are tested for this dataset and it is found that dropout of 0.3 is optimal. The learning rate of 0.01 is fixed and the network used the efficient Adam optimization algorithm. The log loss function is used while training which is suitable for multiclass classification problems. Varying epochs of 50, 100, 150, 200, 250 are experimented and the epoch size of 250 is fixed for the network. The four deep models are trained with the above parameter settings using CEDS dataset and the ASD causative gene identification models are built.

The performance of the four independent models was evaluated based on prediction accuracy, logarithmic loss, precision, recall and F-measure. The standard 10 - fold cross-validation technique was applied to split the data and to estimate their impact on the model's prediction performance for unknown samples. The epochwise accuracy of the models are tabulated in Table XXXVIII.

Table XXXVIII Epochwise Accuracy of Deep Models with Codon Encoding

Epochs	DNN	BRNN	LSTM	GRU
50	74.1%	77.4%	79.3%	79.5%
100	74.9%	79.2%	80.4%	80.6%
150	75.0%	80.3%	81.5%	81.4%
200	77.2%	80.9%	82.1%	82.8%
250	81.4%	81.5%	82.5%	83.9%

The experiment is carried out for different epochs and the results showed that GRU based ASD gene prediction model has achieved high accuracy of 83.9% at 250 epochs. The DNN and BRNN models demonstrated equal performances with accuracy of 81.4% and 81.5% for the gene identification model whereas LSTM attained 82.5% accuracy. It is found that the accuracy also increases to a considerable extent in all the four models as epochs are increased from 50 to 250. The epochwise log losses of the models are tabulated in Table XXXIX.

Table XXXIX Epochwise Log Loss of Deep Models with Codon Encoding

Epochs	DNN	BRNN	LSTM	GRU
50	1.6742	0.9439	0.8875	0.9157
100	1.0557	0.9120	0.8696	0.8433
150	0.9726	0.8351	0.8281	0.7952
200	0.9521	0.8678	0.8174	0.7640
250	0.8248	0.7902	0.8025	0.7014

The log loss function punishes the classifiers for the inaccuracy of predictions. The experimental results indicate that in the early epochs the log loss is high and drastically comes down as epochs escalate. In 50 epochs DNN has the highest log loss whereas LSTM has the least. The GRU method has the least log loss of 0.7014 at 250 epochs for ASD gene identification model whereas it is 0.7902, 0.8025 and 0.8248 for BRNN, LSTM and DNN respectively. The performance comparisons of the models are summarized in Table XL.

Table XL Performance Results of Deep Models with Codon Encoding

Metrics	DNN	BRNN	LSTM	GRU
Precision	0.81	0.83	0.83	0.85
Recall	0.82	0.8	0.82	0.83
F- Measure	0.81	0.81	0.82	0.84
Accuracy	81.40%	81.50%	82.50%	83.90%
Correctly classified instances	407	407	412	419
Incorrectly classified instances	93	93	88	81
Specificity	73.6%	73.9%	74.5%	78.9%

The result analysis indicates that GRU model is effective in predicting the ASD genes with a precision of 0.85, recall of 0.83 and F-measure of 0.84. The LSTM classifier stands

second with a precision of 0.83 and recall of 0.82 whereas DNN and BRNN show equal performance with F-measure of 0.81 and accuracy of 81.4%, 81.5%. GRU based gene prediction model has correctly identified 419 instances. When evaluating the specificity, GRU gives a prominent score value of 78.9% for identifying the genes whereas it is 74.5% for LSTM based ASD gene prediction model. The following charts from Fig.8.2 to Fig.8.8 depict the experimental results.

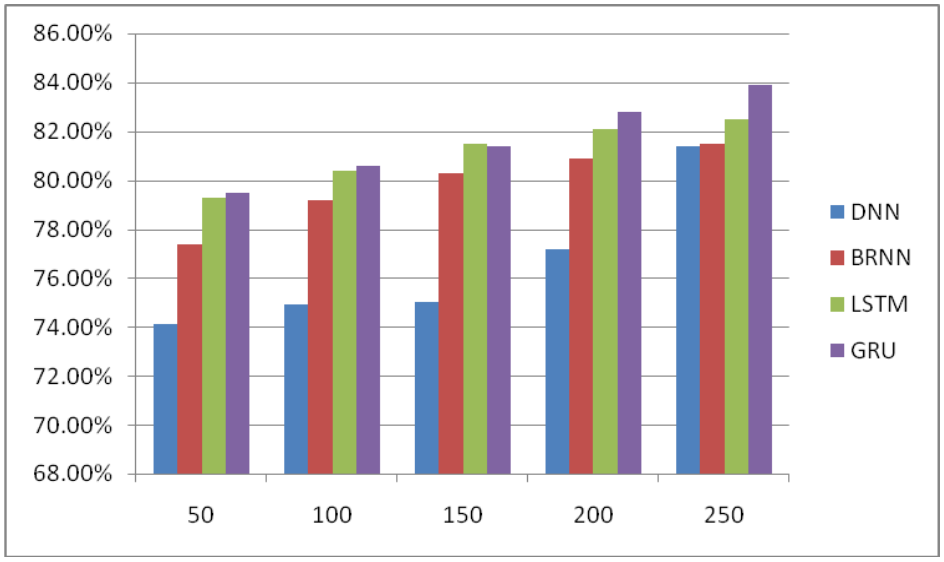

Fig. 8.2 Epochwise Accuracy of Deep Models with Codon Encoding

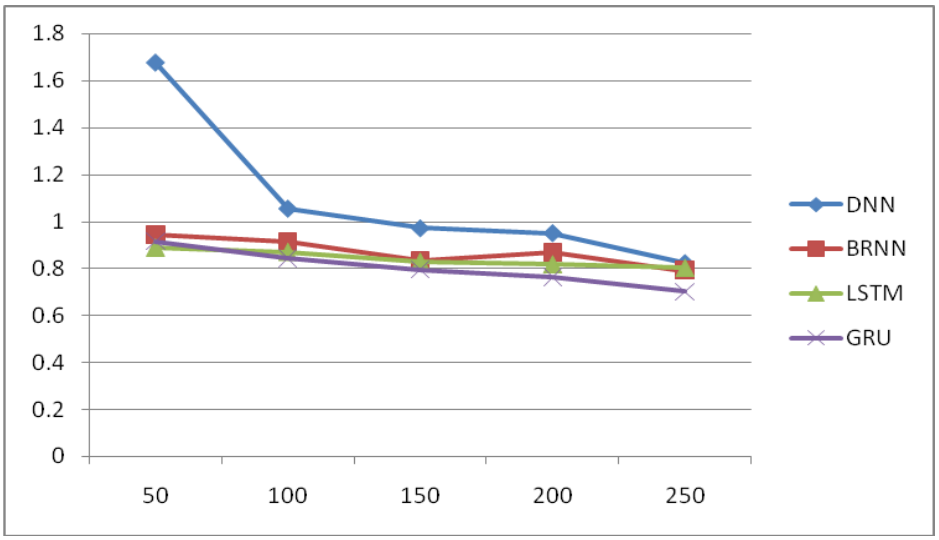

Fig. 8.3 Epochwise Log Loss of Deep Models with Codon Encoding

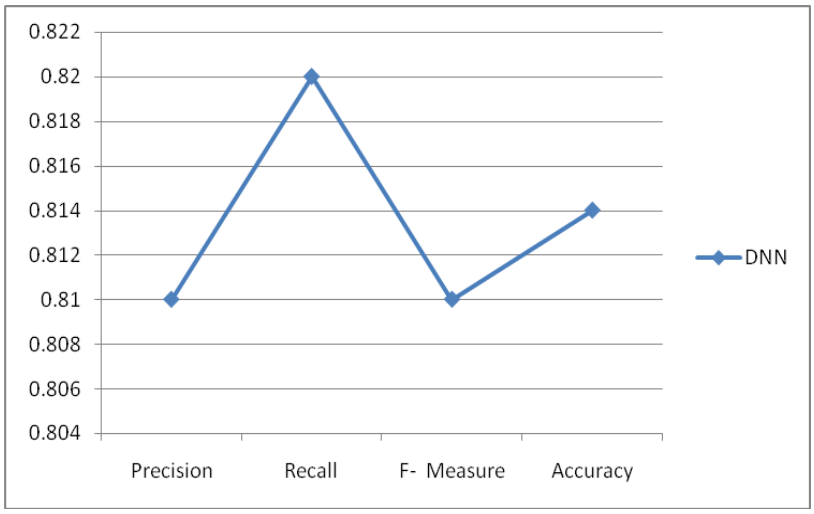

Fig. 8.4 Performance of DNN Model with Codon Encoding

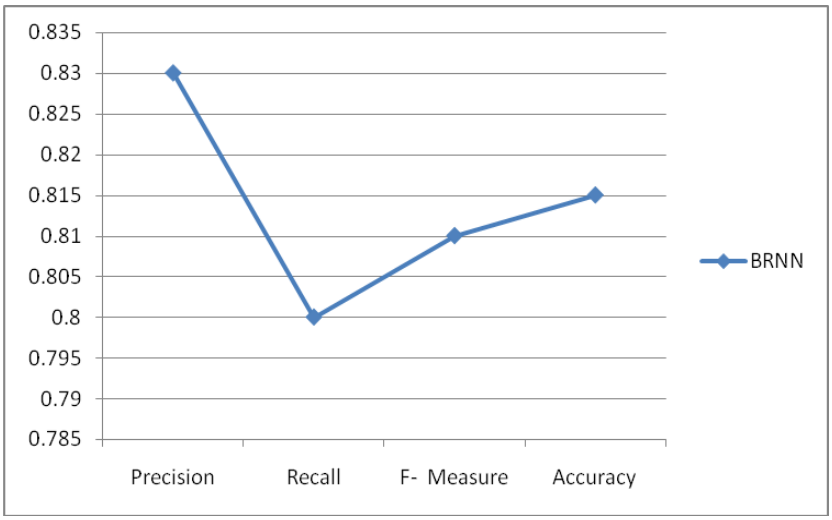

Fig. 8.5 Performance of BRNN Model with Codon Encoding

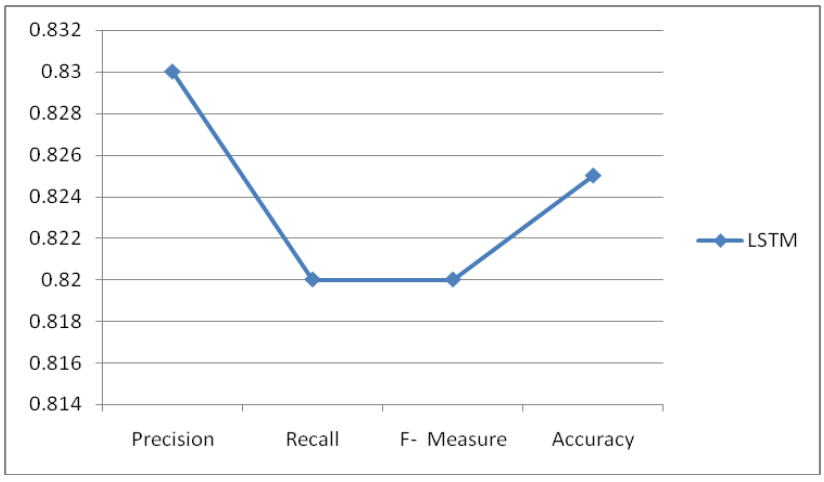

Fig. 8.6 Performance of LSTM Model with Codon Encoding

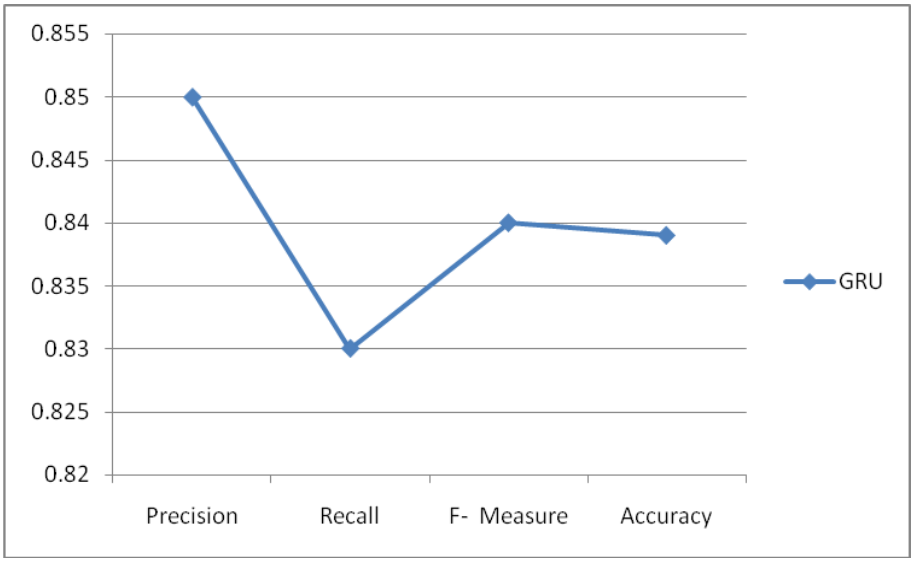

Fig. 8.7 Performance of GRU Model with Codon Encoding

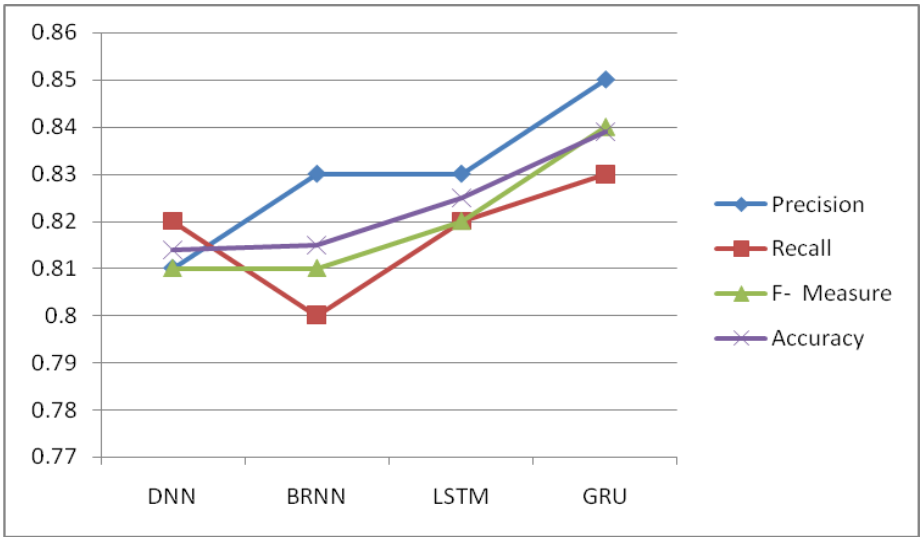

Fig. 8.8 Performance Comparison of Deep Models with Codon Encoding

Fig.8.2 shows that the epochwise accuracy values are higher for GRU based gene recognition model. The logloss which is initially high for all models reduces with increased epochs and is observed to be minimum for the GRU gene prediction model which is illustrated in Fig.8.3. The performance of the four models is illustrated in Fig.8.4 to 8.7 and it is noticed that the precision, recall and accuracy of gene prediction model is high compared to other models.

Comparison of Deep Learning Models with Self Learned Vs User Defined Features

The effectiveness of the deep models developed with codon encoded dataset (CEDS) in predicting ASD gene type is compared with the models trained using hand crafted features(CMDS) in the previous experiments described in chapter 6 and chapter 7. The performance measures like precision, recall, accuracy, F-measure and log loss are used to compare the models and the comparative performance is reported in Table XLI - Table XLII.

Table XLI Comparative Performance of Deep Models - Self Learned Features (CEDS) Vs User Defined Features (CMDS)

Metrics	Self Learned features - Codon encoding (CEDS dataset)				User Defined Features (CMDS dataset)			
	DNN	BRNN	LSTM	GRU	DNN	BRNN	LSTM	GRU
Precision	0.81	0.83	0.83	0.85	0.79	0.80	0.81	0.83
Recall	0.82	0.8	0.82	0.83	0.81	0.77	0.78	0.81
F- Measure	0.81	0.81	0.82	0.84	0.80	0.78	0.79	0.82
Accuracy	81.40%	81.50%	82.50%	83.90%	80.8%	81.3%	81.9%	82.5%

It is observed that deep learning models have attained improved accuracy when trained with the codon encoded dataset. The GRU model outperforms with 83.9% accuracy in identifying ASD genes using self learned features and 82.5% with user defined features. LSTM shows 0.6% enhanced accuracy working with the encoded gene sequences. BRNN shows almost equal accuracy 81.5% and 81.3% in both the methods and increased F- measure of 0.81. DNN achieves 0.6% improved accuracy and its precision has also improved. The comparison of log loss of four models on CEDS and CMDS is presented in Table XLII.

Table XLII Log Loss of Deep Models - Self Learned Features (CEDS) Vs User Defined Features (CMDS)

Architecture	Self Learned features – Codon encoding (CEDS dataset)	User Defined Features (CMDS dataset)
DNN	0.8248	0.8641
BRNN	0.7902	0.8010
LSTM	0.8025	0.8284
GRU	0.7014	0.7162

With regard to logarithmic loss GRU has comparatively less misclassifications for both self learned and user defined features. It has evidenced reduced values of 0.7162 and 0.7014 for CMDS and CEDS respectively. The comparative performance of the deep learning models using CEDS and CMDS is depicted in Fig.8.9 – Fig.8.10.

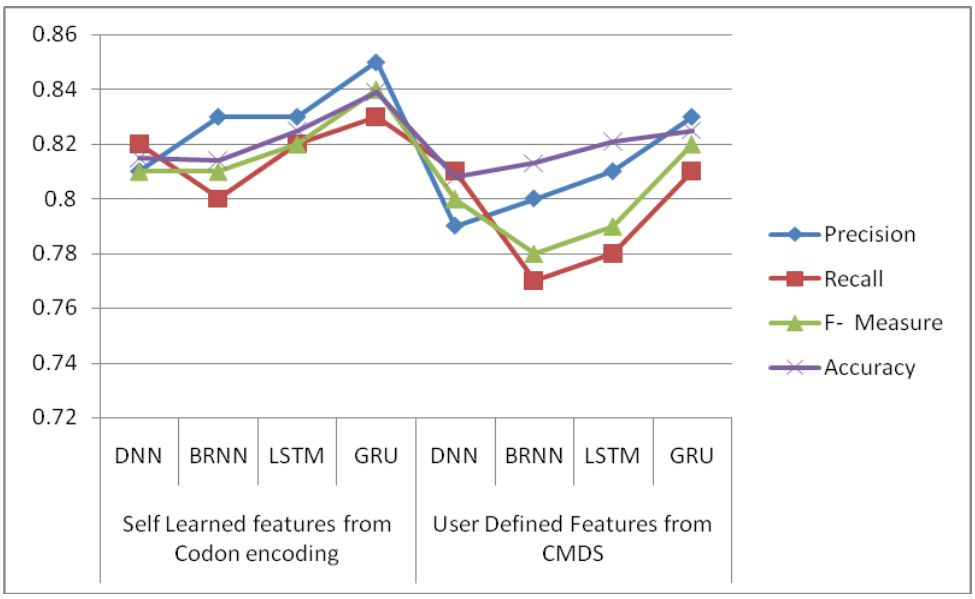

Fig. 8.9 Comparative Performance of Deep Models - Self Learned Features (CEDS) Vs User Defined Features (CMDS)

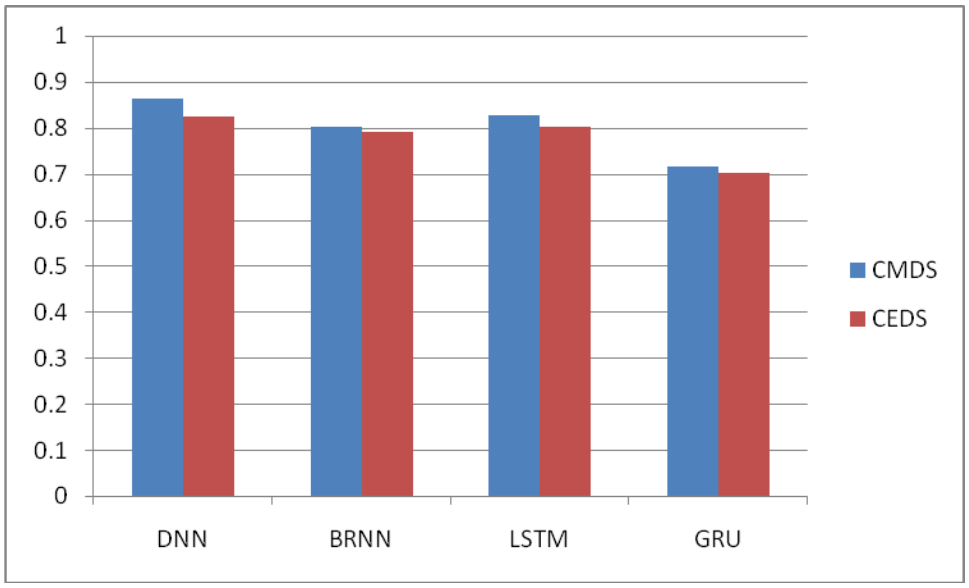

Fig. 8.10 Log Loss Comparison of Deep Models - Self Learned Features (CEDS) Vs User Defined Features (CMDS)

It is evident that the deep models show superior performance with codon encoded dataset rather than with user defined CMDS dataset for identifying genes. GRU models have high accuracy in predicting the genes and outperforms with high precision and recall than its near equivalent LSTM as shown in Fig.8.9. The recall values of BRNN, LSTM and GRU appear the same whereas accuracy is constantly soaring for GRU than the other models. The performance of the models with regard to log loss are given in Fig.8.10 and it shows that the all the models show slightly higher log loss for gene identification with user defined features than with CEDS dataset. Among the four models, GRU based model has a minimum log loss for ASD gene identification.

Findings

From this work it is found that the deep learning models have extracted gene characteristics automatically from codon encoding of gene sequences through self learning and has exhibited superior performance. These deep models surpass their respective performance for gene prediction using hand crafted features by capturing the influential, intricate and inconspicuous features directly from the encoded gene sequences which may not be noticed by humans. It is evident that deep models have eliminated the need for domain expertise and feature engineering thus reducing the time taken. Empirical experiments on encoded datasets confirm that the GRU model outperforms other deep learning models as the long-term temporal

dependencies of gene sequence observations are jointly learned by the integrated sub models. The reliability of the gene identification model is proved by self taught features learnt through deep learning. These experiments also prove that the proposed method can be used as a general deep learning framework for classification of gene sequences and disease prediction as direct input is given without the need for careful extraction of features.

8.2 ONE HOT ENCODING OF GENE SEQUENCES FOR DEEP LEARNING BASED PREDICTION OF ASD CAUSING GENES

The previous experiment utilized the codon encoding scheme of the gene sequences to train the deep models which are able to classify the ASD gene sequences. The codon patterns are read by the networks and it self-learns features from the varied sequences. But positional information of each nucleotide in the sequence is not explicitly specified to the deep networks. Since gene sequences are denoted as a sequence of successive letters without space, it is proposed to use one-hot encoding to give binary representation for the input sequences without losing positional information of each nucleotide. In this work, the utility of using one hot encoding method to encode a gene sequence as vector of numerical values for building the deep models to predict the type of ASD causative genes is explored.

One Hot Encoding and Dataset Preparation

The simulated mutated sequences undergo the process of one hot encoding where each input sequence of length l is transformed into a $4 \times l$ representation. The nucleotide bases adenine (A), cytosine (C), guanine (G), thymine (T) match the components from top to bottom respectively. If one of the nucleotide appears, the corresponding component is set to one and the others are set to 0. All sequences are not of the same length, but in order to make it uniform, 0 padding is done to make them equal in length. The maximum length of the available CDNA sequence is 7746 which corresponds to CHD8 gene. Hence each input sequence is transformed into a 7746 x 4 one-hot encoded vector. A one hot encoded sample of size 10-base pair for the gene sequence ACGTGTCCAG is shown below.

1	0	0	0	0	0	0	0	1	0
0	1	0	0	0	0	1	1	0	0
0	0	1	0	1	0	0	0	0	1
0	0	0	1	0	1	0	0	0	0

In this manner one hot encoding is done for all 1000 sequences available in the corpus. The One Hot Encoded Dataset (OHEDS) consisting of encoded feature vectors is created with 1000 instances of dimension 7746 x 4 where each instance is assigned with one hot encoded class labels ranging from 1 to 10 for the ten possible genes.

Methodology

Four different deep models based on DNN, BRNN, LSTM and GRU architectures are built by training the above OHEDS. The process includes three elements namely dataset creation, model building, performance evaluation and the architecture of one hot encoding based deep models for identifying ASD genes is depicted in Fig.8.11.

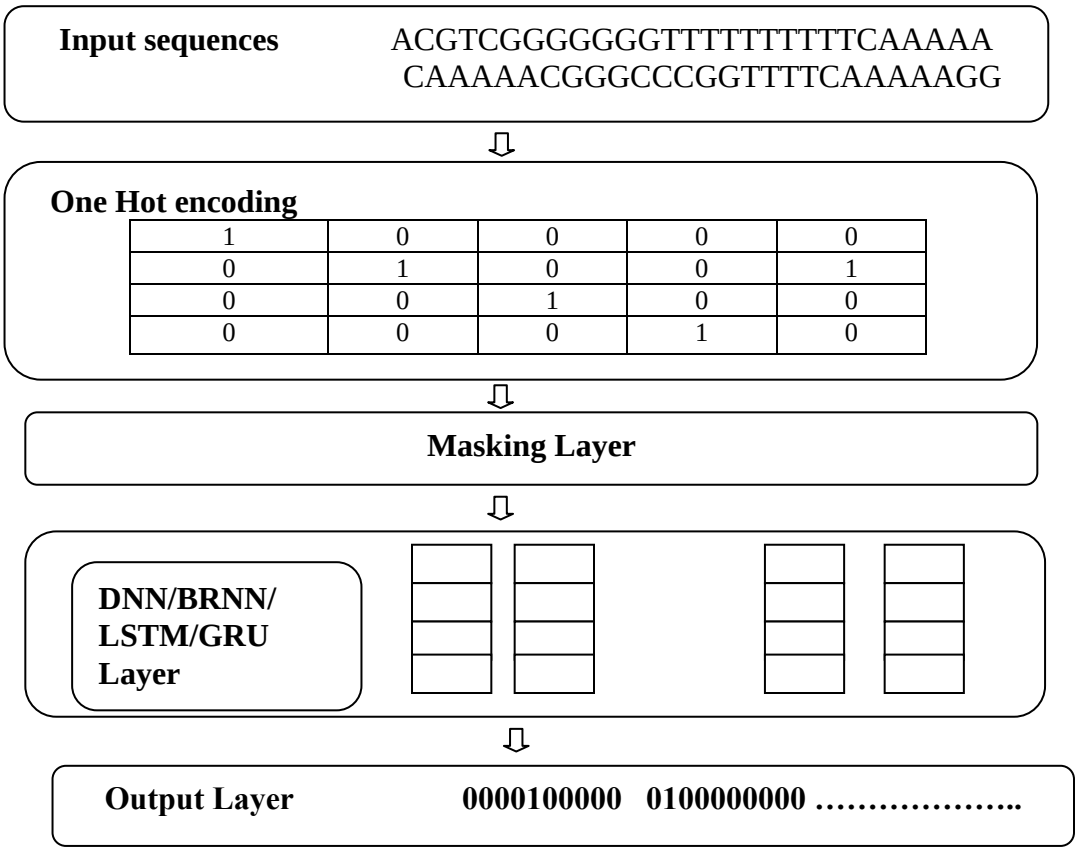

Fig. 8.11 Architecture of One Hot Encoding Based Deep Models for Identifying ASD Genes

During the initial phase, the corpus built using 1000 mutated gene sequences accounting for ten types of ASD genes and four types of mutations as mentioned in Chapter 3 is used. Consequently one hot encoding of these mutated gene sequences are done according to the method mentioned in the previous section. The input vectors are reshaped to have n time steps

and k features, where n is the number of integers in the generated sequence and k is the set of possible integers at each time step. The dataset OHEDS comprises of 1000 class labeled input vectors of dimension 7746 X4 with 10 class labels and is used for building models.

In the second phase four different deep network architectures namely DNN, BRNN, LSTM, GRU are designed to build gene identification models. To ensure unbiased investigations, the basic structure of the deep models configured in the previous work is employed here. That is the network with one input layer, 2 hidden layers with 8 memory units, a masking layer and an output layer which is a fully connected layer with 10 neurons for the 10 possible gene classes is designed. The softmax activation function which is placed in the output layer allows the network to learn and output the distribution over the possible output values. In case of LSTM, the two stacked LSTM recurrent layers is used as done in section 7.2 and with GRU, shared feature extraction is performed using two parallel submodels as in section 7.3.

A choice of hyperparameters such as batch size, epochs, dropout, learning rate and optimizer are similar to that used in the earlier work. Mini - batch gradient descent is used to update the parameters of the network. The epochs denote the number of times the network will work through the entire training dataset. The important regularization technique dropout is used to reduce the effect of overfitting. The predictive model is constructed with sparse categorical cross entropy loss function for training and suitable for prediction problems. The speed at which a deep model replaces the concepts it has already learned with the new ones is decided by the learning rate. The error function is minimized by the optimization algorithm and Adam optimizer is the algorithm used in this implementation. The above mentioned hyperparameters are fine tuned to achieve the best configuration for the deep networks. ASD gene type identification models are built by training the four deep learning techniques using one hot encoded dataset.

In the final phase, 10 - fold cross validation is used to test the models and their predictive performance is evaluated using various metrics such as precision, recall, F- measure, accuracy, log loss and specificity.

Experiment and Results

Experiments have been carried out by implementing DNN, BRNN, LSTM and GRU algorithms using Keras which is a high-level API for neural networks. The hyper parameters

used in the previous experiments are used here for the sake of uniformity. During training and testing, data are segmented on mini-batches of size 64 data segments. The network used a dropout rate of 0.3 and a learning rate of 0.01. The experiment was conducted for varying epochs of 50, 100, 150, 200, 250 and the results are tabulated. The network used the sparse categorical cross entropy loss function while training, suitable for multiclass classification problems and the efficient Adam optimization algorithm. The four deep classifiers are trained with the above parameter settings using OHEDS dataset and the ASD causative gene identification models are built.

The standard 10 - fold cross-validation technique was applied to estimate the predictive performance of the models. The performance of the four independent models was evaluated based on prediction accuracy, logarithmic loss, precision, recall and F-measure. The epochwise accuracy of the models is tabulated in Table XLIII.

Table XLIII Epochwise Accuracy of Deep Models with One Hot Encoding

Epochs	DNN	BRNN	LSTM	GRU
50	74.8%	77.7%	79.6%	80.3%
100	75.2%	80.1%	80.6%	80.8%
150	76.6%	80.5%	81.7%	81.6%
200	78.2%	81.4%	82.4%	83.1%
250	81.6%	81.8%	82.9%	84.3%

The tabulated results show that GRU based ASD gene prediction model has achieved an accuracy of 84.3% at 250 epochs which is higher than that of other three models. At 50 epochs the GRU prediction model achieved an accuracy of 80.3%, gradually increased to 81.6% at 150 epochs and reached 84.3% at 250 epochs. There is an increase of about 4% accuracy for the GRU prediction model from 50 to 250 epochs. The experiments show that accuracy of all three models increases as epochs are increased The accuracy of DNN, BRNN, LSTM models at 250 epochs is 81.6%, 81.8% and 82.9% respectively. The epochwise log loss of these three models is shown in Table XLIV.

Table XLIV Epochwise Log Loss of Deep Models with One Hot Encoding

Epochs	DNN	BRNN	LSTM	GRU
50	1.0124	0.8539	0.8474	0.8356
100	0.9951	0.8312	0.8151	0.8130
150	0.9277	0.8151	0.7878	0.7623
200	0.8528	0.8054	0.7302	0.7345
250	0.8047	0.7810	0.7150	0.7004

The results illustrate that the log loss reduces as epochs increase and the models have achieved better performance by reducing its misclassifications of genes. Initially at 50 epochs the DNN, BRNN, LSTM, GRU models had log loss of 1.0124, 0.8539, 0.8474 and 0.8356 respectively for predicting genes. This gets reduced at the end of 250 epochs for all the four models. Eventually the log loss associated with GRU model in classifying the ASD causative genes is 0.7004 which is comparatively less when compared to that of 0.8248 of DNN model, 0.7802 of BRNN and 0.7150 of LSTM model. The overall performance results of these four models are shown in Table XLV.

Table XLV Performance Results of Deep Models with One Hot Encoding

Metrics	DNN	BRNN	LSTM	GRU
Precision	0.82	0.83	0.84	0.86
Recall	0.81	0.82	0.82	0.84
F- Measure	0.82	0.82	0.83	0.85
Accuracy	81.6%	81.8%	82.9%	84.3%
Correctly classified instances	408	409	414	421
Incorrectly classified instances	92	91	86	79
Specificity	82.6%	82.9%	83.7%	84.9%

The result analysis indicates that GRU model shows promising performance for the ASD gene prediction. It is effective in predicting the ASD genes with a precision of 0.86, recall of 0.84 and F-measure of 0.85. The DNN and BRNN classifiers achieve almost equal F-measure of 0.82 whereas the LSTM classifier attains a slight upper edge with that of 0.83. The GRU based

gene prediction model has correctly identified 421 instances and has achieved specificity of 84.9%. Among the other three models LSTM has correctly identified 414 instances whereas DNN and BRNN has recognized 408 and 409 respectively. The experimental results of GRU based models are illustrated in Fig.8.12 to Fig.8.17.

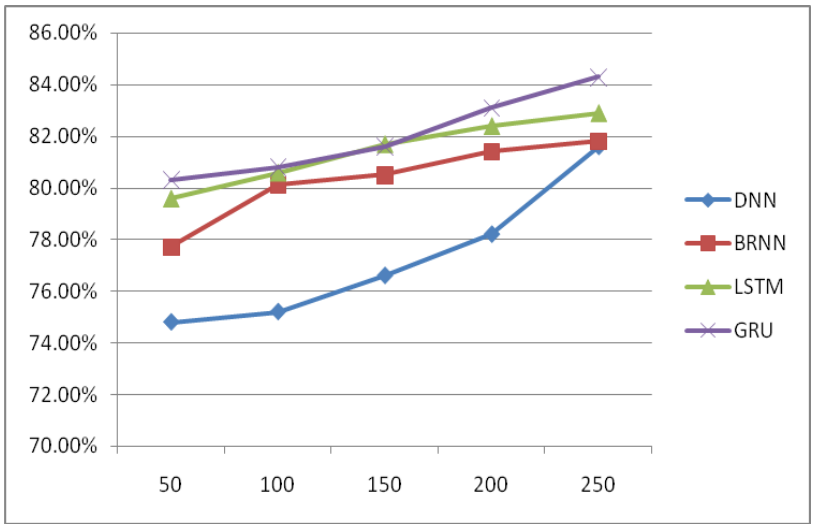

Fig. 8.12 Epochwise Accuracy of Deep Models with One Hot Encoding

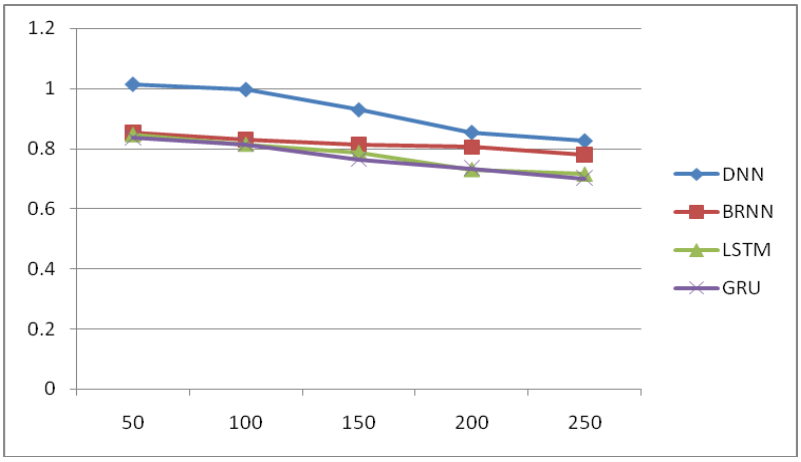

Fig. 8.13 Epochwise Logloss of Deep Models with One Hot Encoding

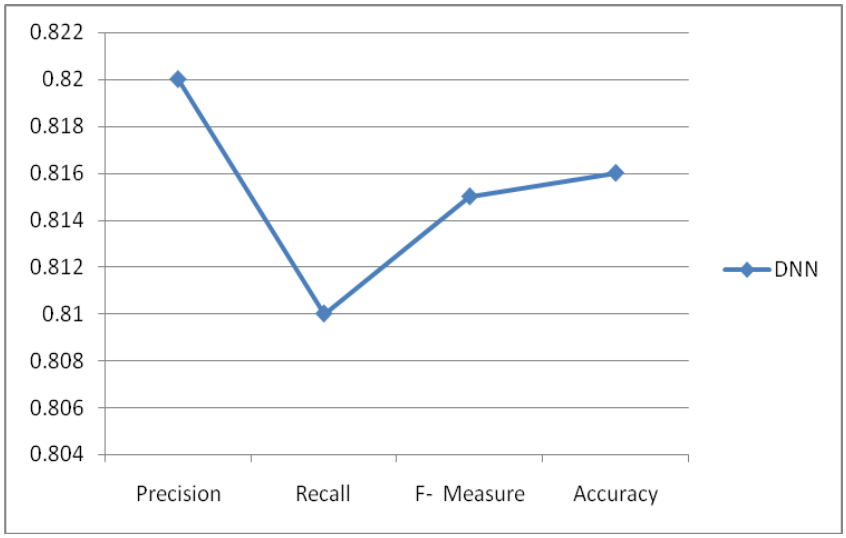

Fig. 8.14 Performance of DNN Model with One Hot Encoding

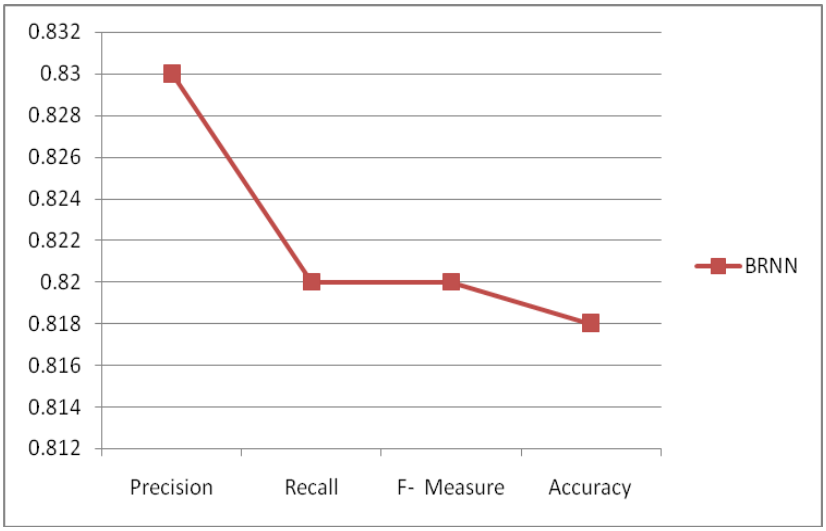

Fig. 8.15 Performance of BRNN Model with One Hot Encoding

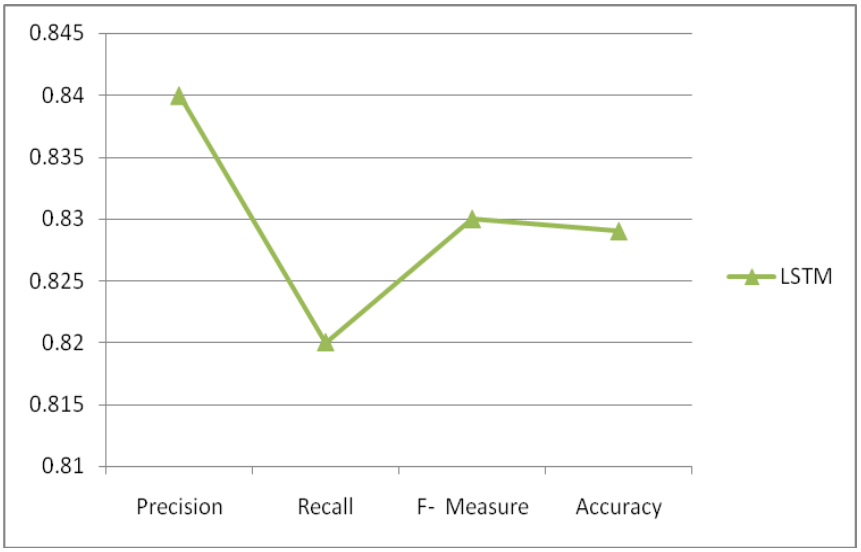

Fig. 8.16 Performance of LSTM Model with One Hot Encoding

Fig. 8.17 Performance of GRU Model with One Hot Encoding

Fig.8.12 shows that the epochwise accuracy values are steadily increasing for GRU based gene recognition model. The logloss in identifying genes reduces with increased epochs and is observed to be minimum for the GRU model which is illustrated in Fig.8.13. The performance of DNN, BRNN, LSTM and GRU models is depicted from Fig.8.14 to Fig.8.17. It is noticed that while the precision of gene prediction model is high for GRU the recall has a dip of 0.02. The F-measure and accuracy of LSTM model is almost equal and BRNN model shows constant recall and F-measure.

Comparison of Deep Learning Models with Self Learned Features of OHEDS and User Defined Features of CMDS

The performance of the deep models developed with one hot encoded dataset (OHEDS) is compared with the corresponding models trained with user defined codon measures dataset (CMDS) for predicting ASD gene type in the previous experiments described in section 8.1. The comparative analysis is performed with respect to various performance measures like precision, recall, accuracy, F-measure and log loss and the comparative results is reported in Table XLVI - Table XLVII.

Table XLVI Comparative Performance of Deep Models - Self Learned Features (OHEDS) Vs User Defined Features (CMDS)

Metrics	Self Learned Features One Hot Encoded Dataset (OHEDS)				User Defined Features (CMDS dataset)			
	DNN	**BRNN**	**LSTM**	**GRU**	**DNN**	**BRNN**	**LSTM**	**GRU**
Precision	0.82	0.83	0.84	0.86	0.79	0.80	0.81	0.83
Recall	0.81	0.82	0.82	0.84	0.81	0.77	0.78	0.81
F- Measure	0.82	0.82	0.83	0.85	0.80	0.78	0.79	0.82
Accuracy	81.6%	81.8%	82.9%	84.3%	80.8%	81.3%	81.9%	82.5%

From the comparative results it is observed that deep learning models have attained improved accuracy when trained with the one hot encoded dataset. The GRU model outperforms with 84.3% accuracy in identifying ASD genes when one hot encoding is used whereas the accuracy of 82.5% is reported with CMDS dataset. The precision, recall and F-measure of GRU has shown improvements than its own performance when using CMDS. BRNN and LSTM shows 0.5% and 1% enhanced accuracy working with the one hot encoded gene sequences than CMDS. DNN achieves 0.8% improved accuracy and its precision has also improved. The comparison of log loss of four models on OHEDS and CMDS is presented in Table XLVII.

Table XLVII Log Loss of Deep Models - Self Learned Features (OHEDS) Vs User Defined Features (CMDS)

Architecture	Self Learned Features One Hot Encoded Dataset (OHEDS)	User Defined Features (CMDS dataset)
DNN	0.8047	0.8641
BRNN	0.7810	0.8010
LSTM	0.7150	0.8284
GRU	0.7004	0.7162

With regard to logarithmic loss all the deep models have comparatively less misclassifications for one hot encoded datasets. GRU model has evidenced reduced log loss values of 0.7162 and 0.7004 for CMDS and OHEDS respectively. LSTM model is the next best model with 0.7150 log loss. The comparative performance of the deep learning models using self learned features from one hot encoding and user defined features from CMDS dataset is depicted in Fig.8.18 – Fig.8.19.

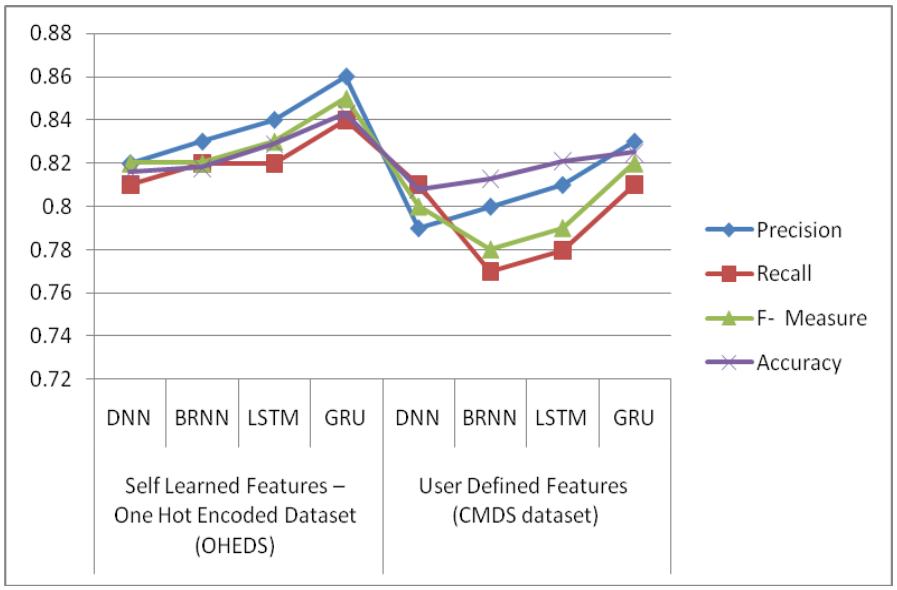

Fig. 8.18 Comparative Performance of Deep Models - Self Learned Features (OHEDS) Vs User Defined Features (CMDS)

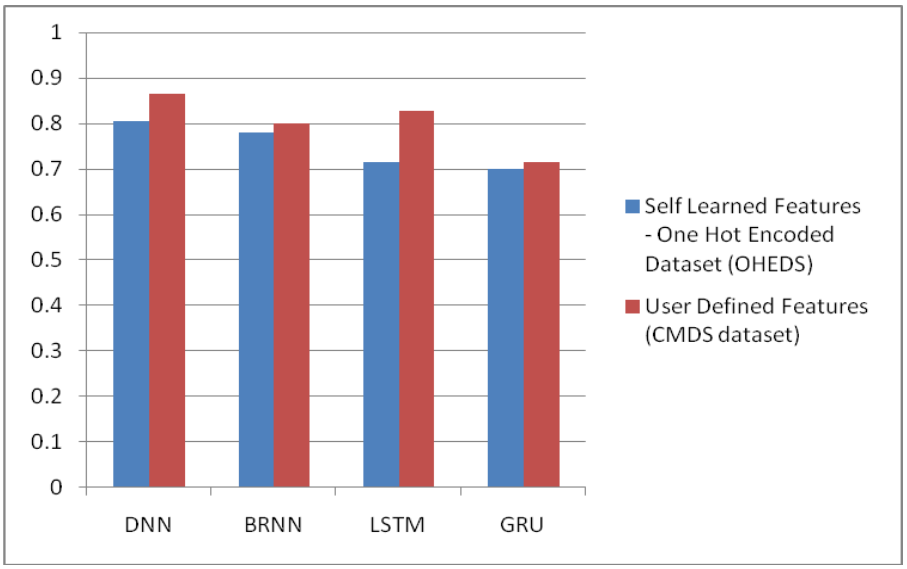

Fig. 8.19 Log Loss Comparison of Deep Models - OHEDS Vs CMDS

It is apparent that the deep models built with one hot encoding show enhanced performance than the models learnt with user defined featured from CMDS dataset for identifying genes. GRU models have high accuracy in predicting the genes and outperform its near equivalent LSTM with high precision and recall as shown in Fig.8.18. The recall values of DNN, BRNN, LSTM appear the same whereas accuracy is constantly soaring for GRU than the other models. The performance of the models with regard to log loss is given in Fig.8.19 and it shows that the all the models show slightly reduced log loss for gene identification with OHEDS dataset than CMDS dataset. GRU based model shows a major difference in the log loss for ASD gene identification when compared to other models.

Comparison of Models with Two Encoding Schemes

On comparing the two encoding schemes it is found that the one hot encoding scheme performs better than the codon encoding scheme. The deep models DNN, BRNN, LSTM and GRU with one hot encoding scheme outperformed their respective models using codon encoding for ASD gene type identification.

Findings

The results confirm that the deep models are able to capture the prominent, intricate and hidden features that humans can miss by preserving specific position information of each individual nucleotide in sequences. The deep models have eliminated the complex task of feature

engineering and have also reduced the time involved. From the comparative results it is found that the deep learning models have improved in self learning and exhibited better performance with respect to various performance measures when using one hot encoding of gene sequences. Empirical experiments confirm that by using one-hot vectors to develop GRU network model, significant performance improvements has been achieved in predicting the ASD causative genes. These experiments also prove that the proposed method can be used as a generalized effective solution to characterize the variable length gene sequence data with masking and one hot encoding.

SUMMARY

Deep learning models using DNN, BRNN, LSTM and GRU developed to classify ASD gene sequences using two different encoding methods have been elaborated in this chapter. Two encoding mechanisms such as codon encoding and one hot encoding proposed to represent the gene sequences were illustrated and the preparation of two corresponding datasets have been described. The models were trained with these two different datasets and the effectiveness of these models was evaluated using different performance metrics to explore the reliability of the method. The implementation results were also demonstrated in this chapter with tables and charts.