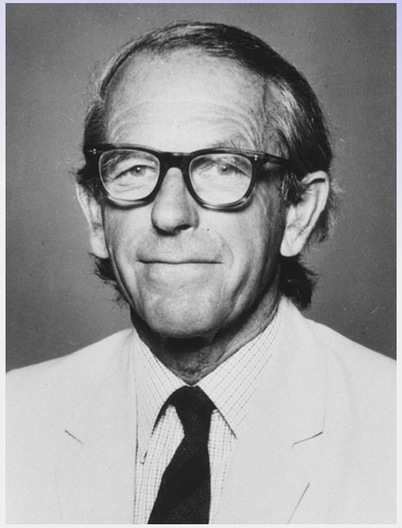
Novel Approaches For DNA Sequence Classification using Deep Learning Methods

Presented by:-
Shireen Shaikh (2021B1A43007G) &
Mridul Mishra (2021B5A42367G)

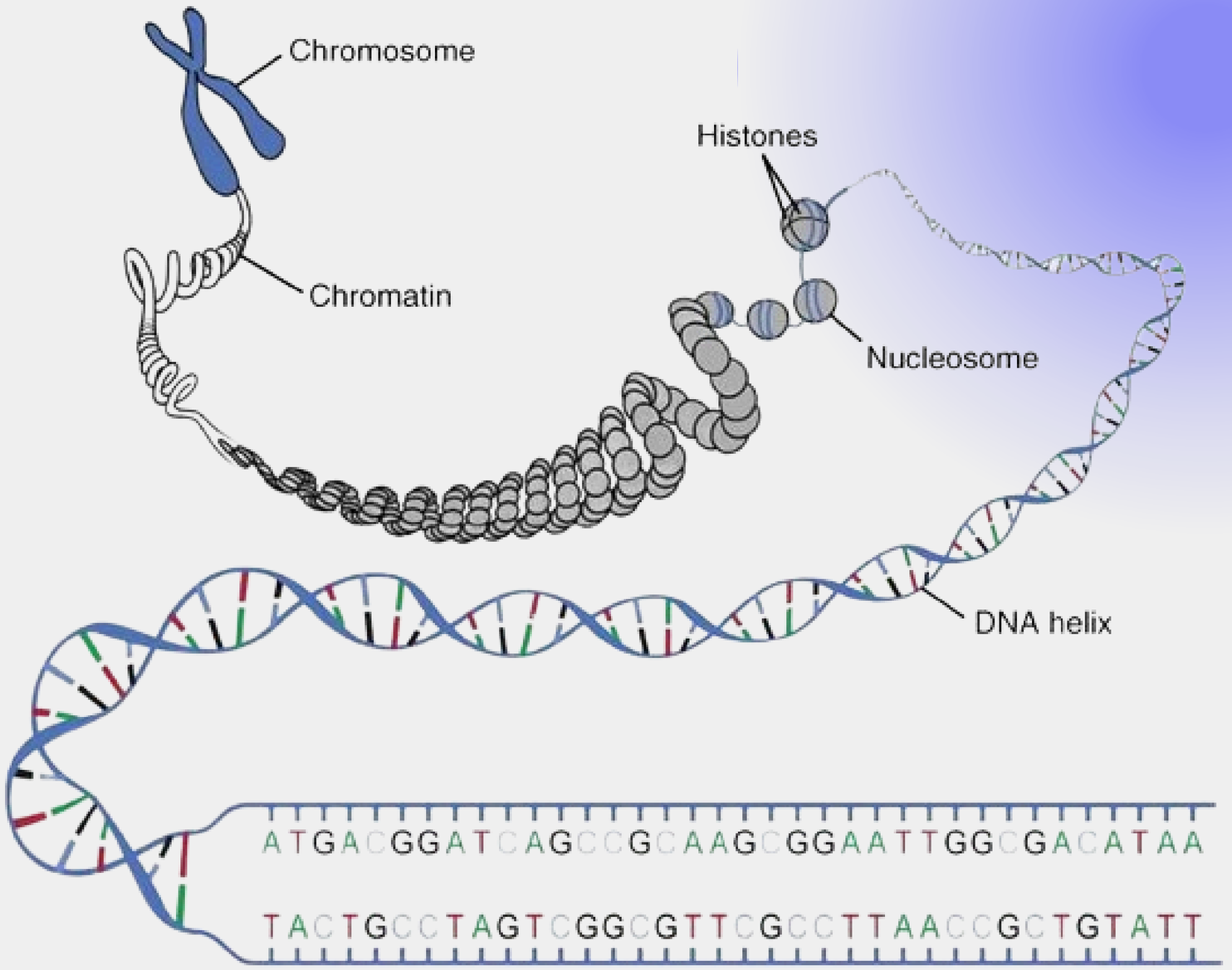
One of the only 4 people as of now to receive the Nobel Prize twice!



Frederick Sanger
(Father of Genomics)

A brief into DNA and histone :-

a type of protein



But why do we care if the histone wraps around DNA at all?

BIG PROBLEMS WITH IMPROPER PACKAGING !

Improper DNA packaging

Gene expression regulation failure

Congenital disorders, Diabetes, Immune system dysfunction and many more...

Exposed and unprotected DNA

Susceptible to damage from UV rad, chemicals.

Increased Apoptosis

Programmed cell death

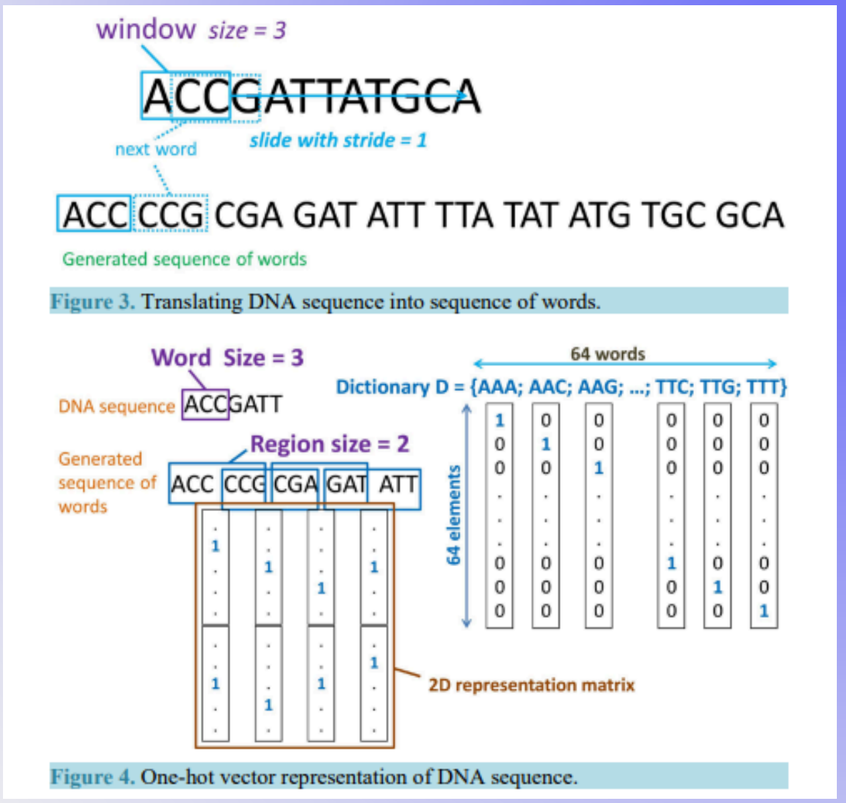
DNA Sequence Classification by Convolutional Neural Network by Nguyen G. N. et al.

- Recognised as a pioneer paper in the intersectional field of Biology and Machine Learning.
- Worked on 12 datasets of sequenced DNA which classified DNA as ‘1’ if it wraps around histone and ‘0’ if it doesn’t.
- We selected the data set H3K4me1 and performed an initial data set exploratory analysis.

How do you input the sequences into the Model?

```
>YPR201W_ARR3_940968
TAGAGGTTATCAATTTATCCACGAAATTGGTTCTGCAATATTGTGCTTTGTCCCATTGGTGCTTTACTTCTTTATTGCATGGTTTTTGACCTTCGCATTAATGAGGTACTTATCAATATCTAGGAGTGATACACAAAGAGAATGTAGCTGTGACCAAGAACTACTTTTAAAGAGGGTCTGGGGAAGAA
AGTCTTGTGAAGCTAGCTTTTCTATTACGATGACGCAATGTTTCACTATGGCTTCAAATAATTTTGAACTATCCCTGGCAATTGCTATTTCCTTATATGGTAACAATAGCAAGCAAGCAATAGCTGCAACATTTGGGCCGTTGCTAGAAGTTCCAATTTTATTGATTTTGGCAATAGTCGCGAGGATC
CTTAAACCATATTATATATGGAACAATAGAAATTAATTGTTGACTCACCAAAAAATTACTTGGGCACCAATGAATAATACCACTTATAACTCTATATTAATCCAAGTTCACCCTAAAAACAGAT
1
>iYPR201W_941173
TTTTCTATTACGATGACGCAATGTTTCACTATGGCTTCAAATAATTTTGAACTATCCCTGGCAATTGCTATTTCCTTATATGGTAACAATAGCAAGCAAGCAATAGCTGCAACATTTGGGCCGTTGCTAGAAGTTCCAATTTTATTGATTTTGGCAATAGTCGCGAGGATCCTTAAACCATATTATAT
ATGGAACAATAGAAATTAATTGTTGACTCACCAAAAAATTACTTGGGCACCAATGAATAATACCACTTATAACTCTATATTAATCCAAGTTCACCCTAAAAACAGATAAAATTGACAATTTTTTTTGAAGGACAACTGATTGATCTTAGTGATGGTTAGGATGAAAAAATGGGTTTGTTCACGCCATC
CCGCATCATTCAAAAACTCCGCACAGTTTTGGAGTTCAGTAAGAAGGGGACGTAATCTTACTATTGCATTTTTGAAAGTCTAATGAACTCAAGTGGCAAACTGTGTTACGTACATCACTTCTAA
1
>iYPR201W_941752
ATTAGTAATGCTTTAAACCCAAGTATTGTAAGCCCGATGGAATAGCACATGGTGATGGTACAACAATGGCTTCATAGATCATTCGCATAAAAATGGAATTCGTGCATCTAGTGTCTAATTAAAATTAACGTAACGAAATAGGACAAGATAAGCTTATGTCGTAGTACCCGCATAAGTACGTGTAGCTG
CCATTTTATAATGCATGAAGTTGAAGTCATTAGGATGAAGAGGTGACATATAATGTTTCAAAATGGTGCCAGTATAAGTTTGGGATATCATGGAGCGGCATTCGTTTTAGTAAATGACTGTATTGTATAGTACTGTCTGTATGATTATCAGTTGAGTTGGTGAGATGAAGTAAAGATATTATAGCATC
TGCGTTAATGTGTATTGCTATTATAGTATAGCATAGTGGTGTTATATAGTGGCACCGGGAATGAGTATAGGGAGTAAGAAGATCTTGTAGTTGCGTTTGAATAGCGACATGAGGACACGTCTTA
1
```

- The paper follows a **3-mer representation** for sequence representation.
- This was followed by sequence embedding using **one-hot encoding method**, forming a 2D matrix representation.



Model architecture of the base model


```
# Define the model architecture
model = Sequential()
model.add(Conv1D(filters=32, kernel_size=4, activation='relu', input_shape=(X_train.shape[1], 1)))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=32, kernel_size=4, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(100, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(y_train.shape[1], activation='softmax'))
```

1. 1D convolutional layer:-

- each filter learns a different feature
- Kernel:- size of convolution window, number of input data that the filter will slide over.

2. MaxPooling:-

- Performs downsampling, by taking a maximum value over a window of size defined.

3. Flatten:-

- Transforms the convoluted 2d output into 1d before pushing it to fully connected **Dense** layers.

4. Dropout:-

- Sets a fraction of input units to 0 at each update during training time, which helps prevent the model from becoming too dependent on specific neurons (overfitting). Here, 50 percent of input units.

Model Optimizer, loss and Accuracy metrics

```
# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

- **ADAM:-** short for Adaptive Moment Estimation ,it computes adaptive learning rates for each parameter.
- uses the concept of momentum by maintaining an exponentially decaying average of past gradients and squared gradients, helping to accelerate convergence.
- Combines the advantages of both **ADAGRAD** and **RMSPROP.**

$$L(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i)$$

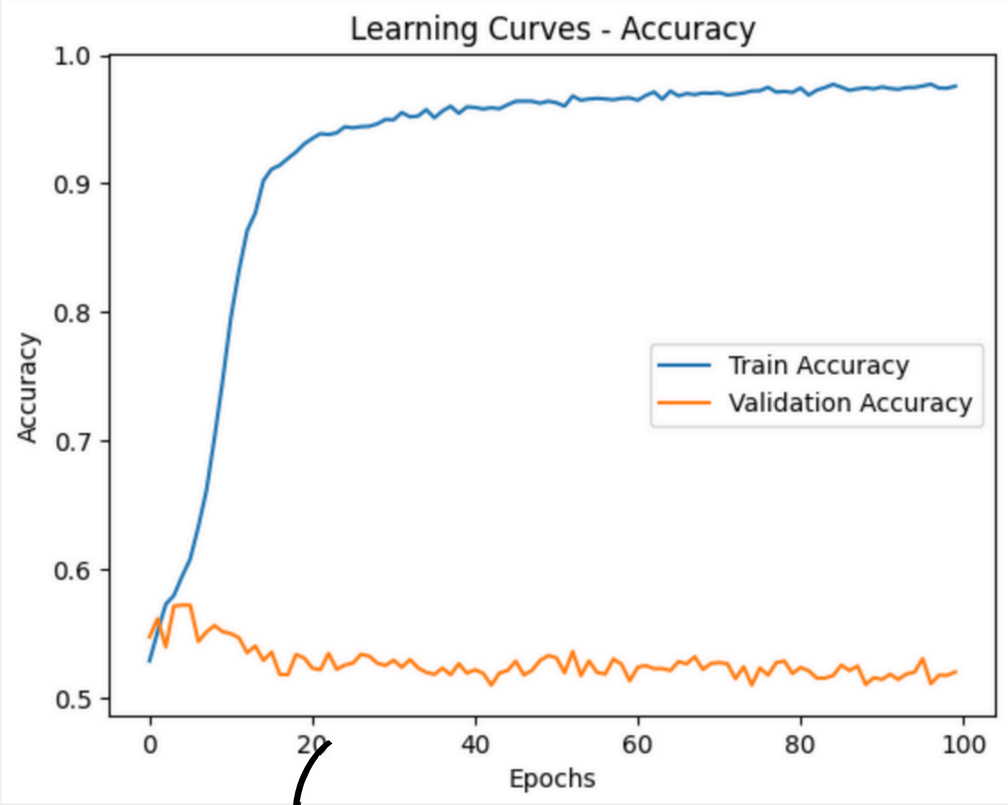
- measures the dissimilarity between true labels (y) and the predictive label model output (y cap).
- Used specifically for multiclass classification problems where the target is categorical.

- To evaluate the performance of the model.
- Several metrics like recall, precision and f1 score calculated.
- But for performance comparison we have used Accuracy.
- Accuracy is the ratio of #true predictions and #total predictions.

Hyperparameters.

Finally the model was trained for 100 epochs with a batch size of 32.

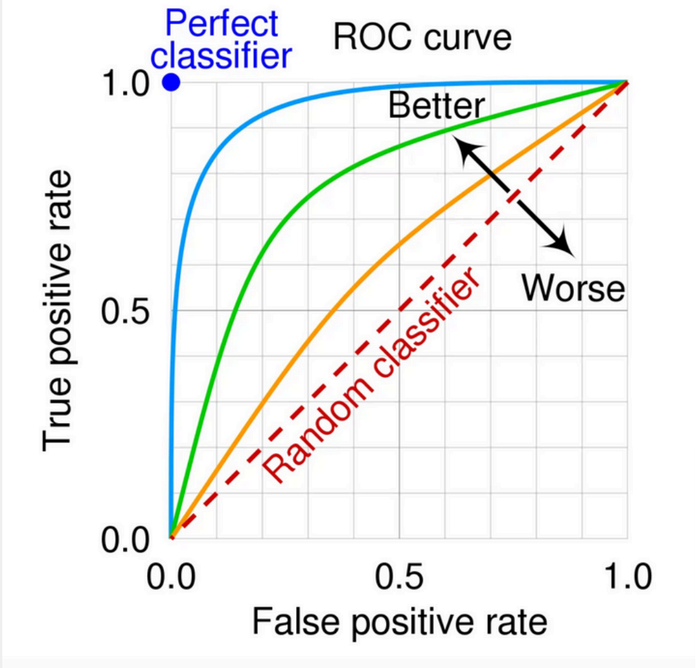
Results From base model implementation

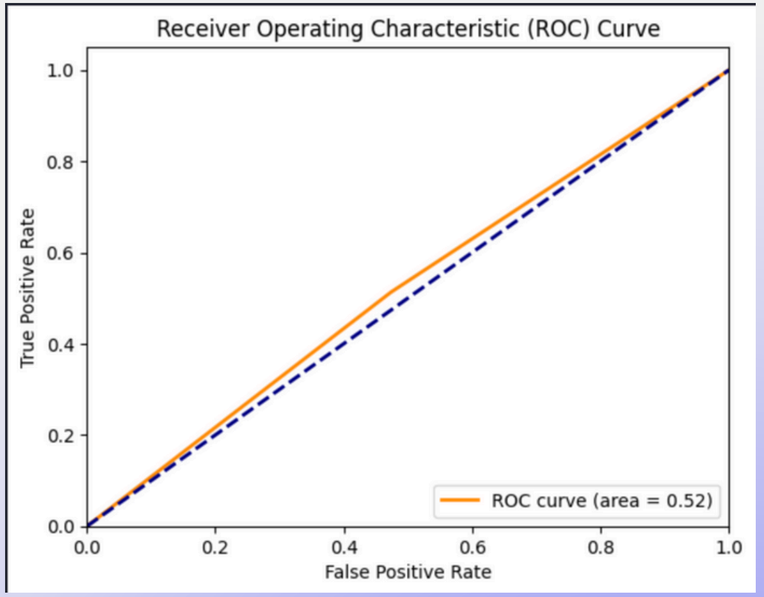


Overfitting!

Accuracy data obtained (val_max = 57% V/S Train_max = 99%).

Ideal V/S Obtained ROC Curve





WHATS THE SOLUTION?

NOVELTY: Improvements to base model

- Adding one more Convolutional layer:-
  a. helps in deeper feature capture and enables complex pattern learning
  b. But it comes with its own problems.
    i. more layers - more training data needed, if not Overfitting is inevitable
    ii. Training time is a big issue here!

It seems like we have been caught in an endless loop!
AND
problem boils down to overfitting!

Problem :- Overfitting

Solutions:-

- Use a regulariser in each convolutional layer. We have used an L2 regulariser.

```
model.add(Conv1D(filters=128, kernel_size=5, activation='relu', kernel_regularizer=l2(0.001)))
```

- Use EarlyStopping Callback function. Early stopping monitors the validation loss. If it stops improving and training accuracy starts increasing (overfitting) then it stops the training process at that epoch. We simultaneously increase the number of epochs as well.

```
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)
```

NOVELTY: Improvements to base model


```python
# Define the model architecture
model = Sequential([
    Conv1D(filters=128, kernel_size=7, activation='relu', input_shape=(X_train.shape[1], 1), kernel_regularizer=l2(0.01)),
    BatchNormalization(),
    MaxPooling1D(pool_size=2),
    Dropout(0.3),
    Conv1D(filters=128, kernel_size=5, activation='relu', kernel_regularizer=l2(0.01)),
    BatchNormalization(),
    MaxPooling1D(pool_size=2),
    Dropout(0.3),
    Conv1D(filters=256, kernel_size=3, activation='relu', kernel_regularizer=l2(0.01)),
    BatchNormalization(),
    MaxPooling1D(pool_size=2),
    Dropout(0.4),
    Flatten(),
    Dense(512, activation='relu', kernel_regularizer=l2(0.01)),
    Dropout(0.4),
    Dense(y_train.shape[1], activation='softmax')
])
```

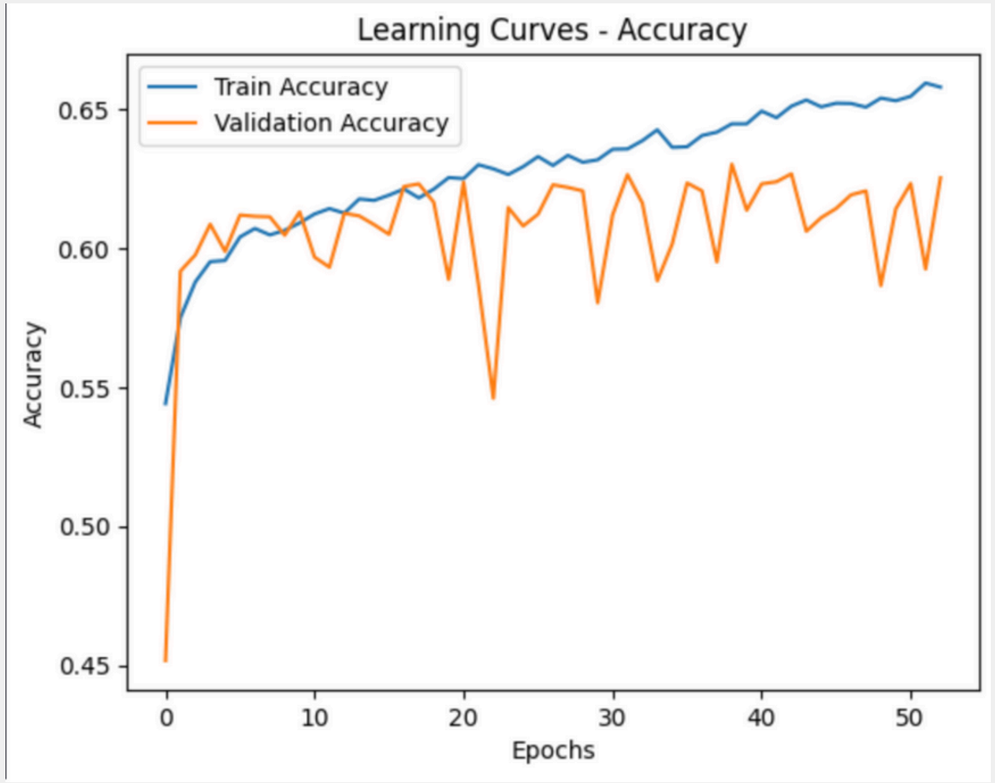
- Convolutional Layer:-
  a. increased filters (more features captured)
  b. increased kernels(more patterns captured)
  c. L2 regulariser.

- BatchNormalisation Layer:- reduces the internal covariate shift and helps accelerate the learning process.

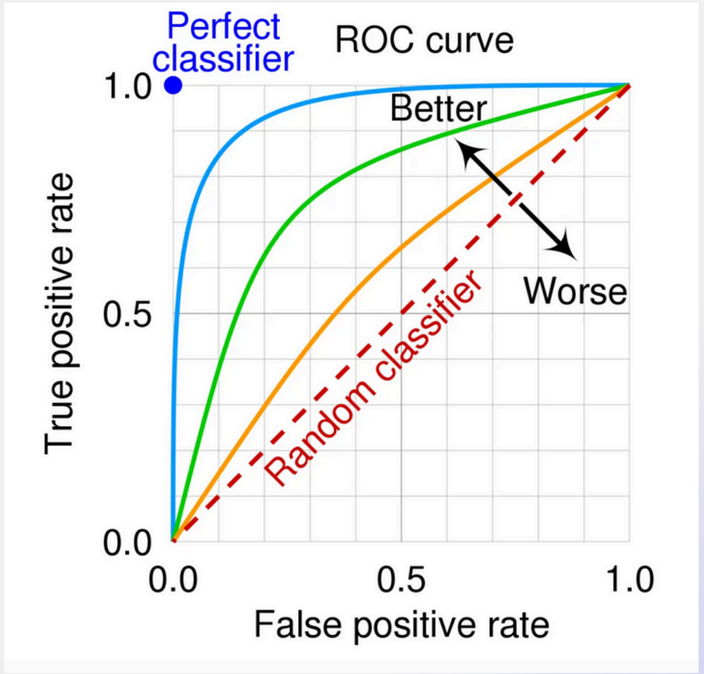
- Dropout layers before each convolutional layer (except the first one ofc).

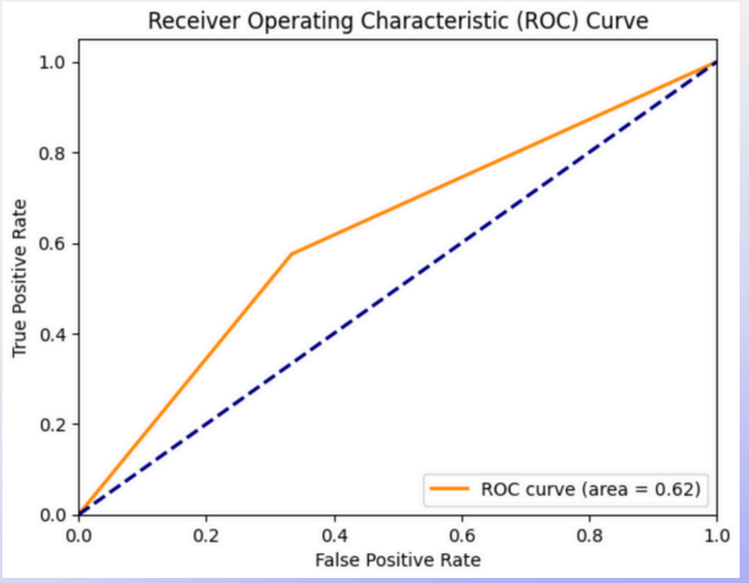
- Large dense layer.

Results From NOVELTY 1 model implementation



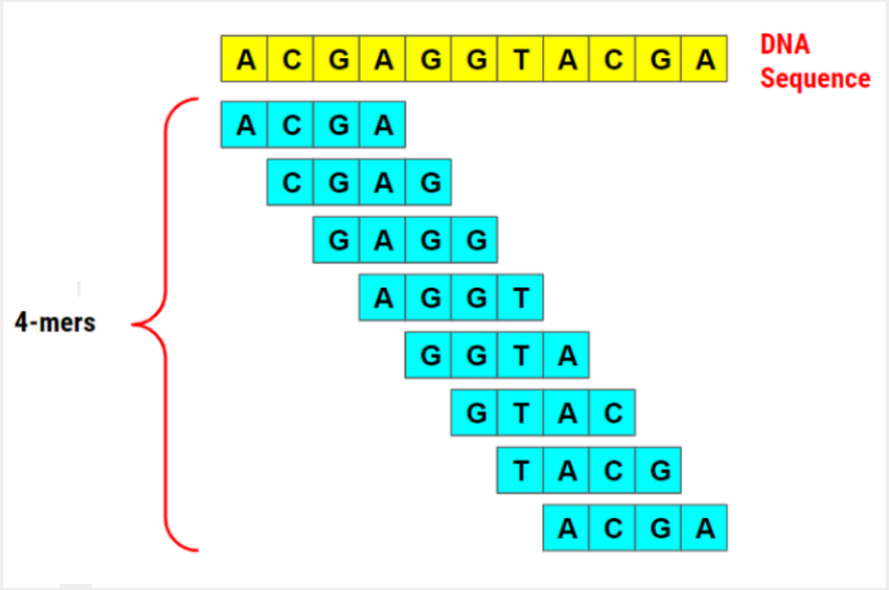
Ideal V/S Obtained ROC Curve



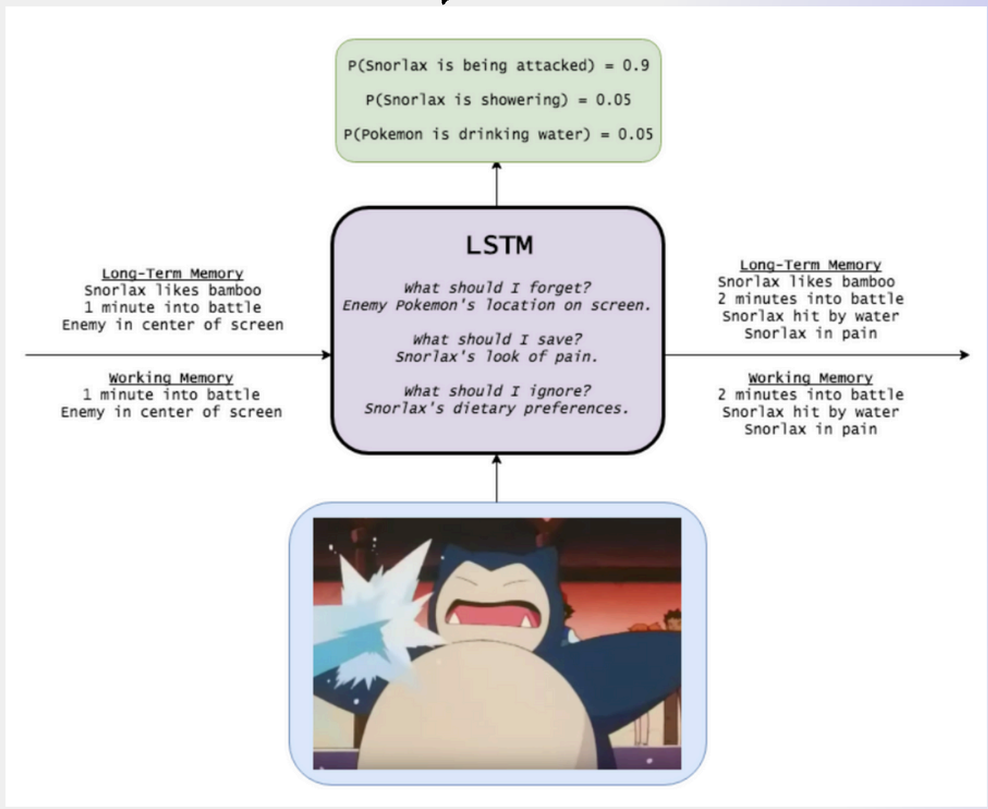


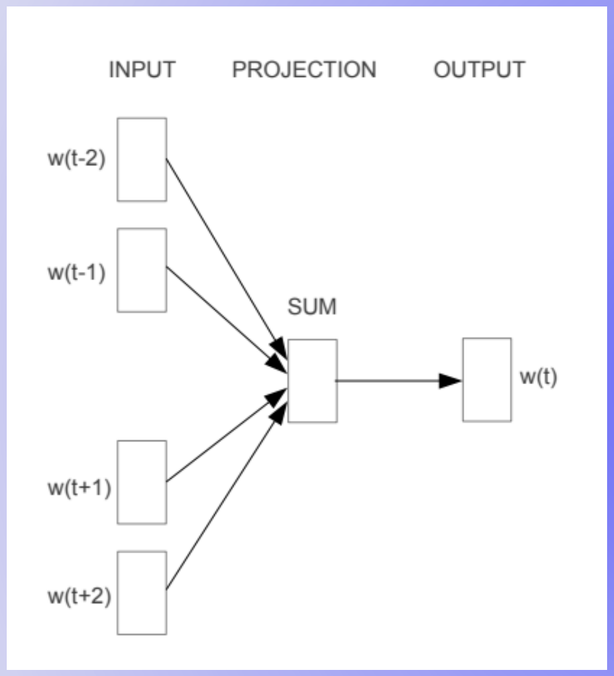
A Significant improvement over the base model!

Experimentation with Sequence Representation, Embedding and Hybrid Models



4 mer or 3 mer?





More complex/hybrid models?

A different approach: Word2Vec or One-hot encoding ?

- **One hot encoding:-**
  a. easy to understand and simple to implement embedding technique.
  b. **but,** it does'nt capture the semantic relationships between characters.
  c. Vectors are orthogonal like [1,0,0], [0,1,0], [0,0,1]

This is basically how computers learned to talk like us!

- **Word2Vec:-**
  a. Uses a neural network to learn dense vector representations for items based on their context within a sequence.
  b. **Thus,** capture the semantic relationships between characters.
  c. Vectors are like [-0.71, 0.45, -0.64].

Suitable for large vocabularies !

Hybrid models?

- Since we have established that the order of elements, (here, nucleotides) matter in the sequence, we need a model that an capture the Temporal order or the chronology.

- CNN's capture the local/spatial feature well, they can effectively learn patterns that are important for classification. But they do not fare well with Temporal feature extraction.

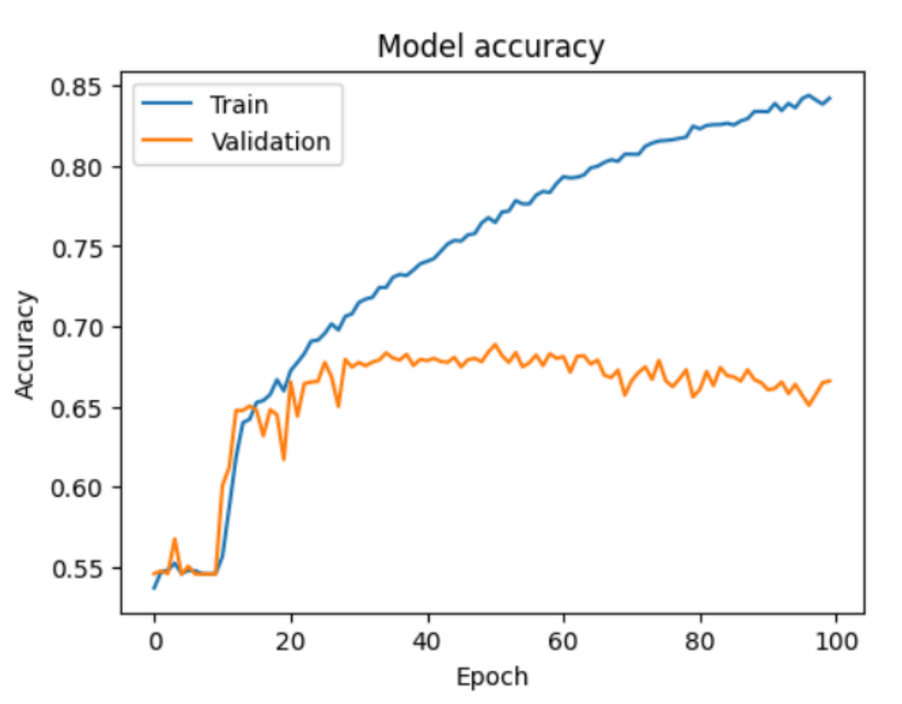
- **LSTM's (long short-term memory)** models capture the dependencies across the sequence, **in a way LSTM+CNN hybrid models will complement one another.**

- **CNN's can learn specific motif's and which can be then processed by LSTM layers to capture complex patterns (sequence of motifs).**

LET'S TEST THIS PROPOSTION!

4-mer with Word2Vec Embedding fed to CNN+LSTM

```python
# Convolutional layer
model.add(Conv1D(filters=32, kernel_size=3, activation='relu', input_shape=input_shape))
model.add(MaxPooling1D(pool_size=2))

# LSTM layer
model.add(LSTM(50, return_sequences=True))
model.add(Dropout(0.5))
model.add(LSTM(50))

# Fully connected layer
model.add(Dense(50, activation='relu'))
model.add(Dropout(0.5))

# Output layer
model.add(Dense(num_classes, activation='softmax'))

# Compile the model
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```



Achieved an accuracy of 68.87% !

Continuing Discussion on Novelty: Hybrid models

FROM THIS POINT, HOW DO WE IMPROVE?

Since we have got good results with CNN+LSTM, Lets think along Similar lines.

What's better than searching for Temporal dependencies in only 1 direction?

SIMPLE! searching in 2 directions along a sequence.

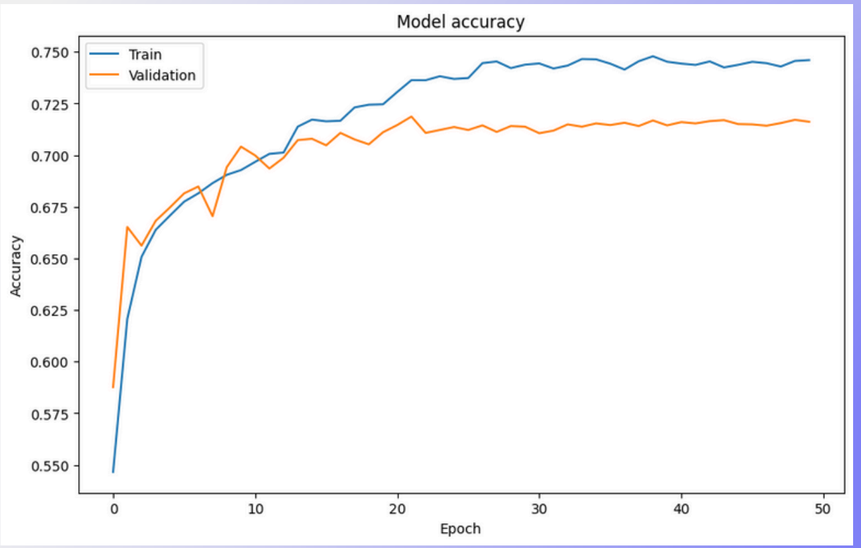
CNN + BiLSTM models.

- **Word2Vec encoding.**
- **One-hot encoding.**

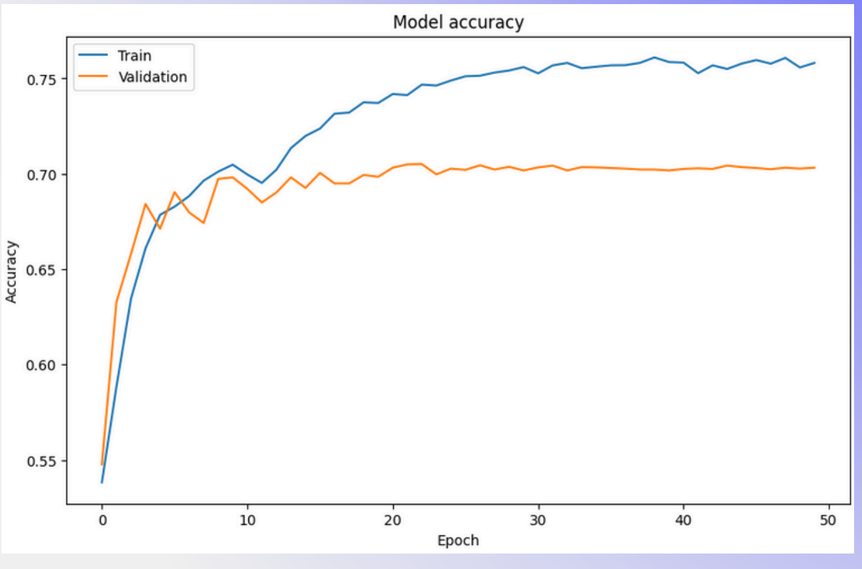
} **Lets's Test them.**

One hot encoding and Word2Vec with CNN+BiLSTM

```
model.add(Conv1D(filters=32, kernel_size=3, activation='relu', input_shape=input_shape))
model.add(BatchNormalization())
model.add(MaxPooling1D(pool_size=2))
model.add(Dropout(0.25))  # Added dropout layer

model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling1D(pool_size=2))
model.add(Dropout(0.25))  # Added dropout layer

model.add(Bidirectional(LSTM(32, return_sequences=True)))
model.add(Dropout(0.5))  # Added dropout layer

model.add(Bidirectional(LSTM(32)))
model.add(Dropout(0.5))  # Added dropout layer

model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))

model.add(Dense(num_classes, activation='softmax'))

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
return model
```



w2v



One-hot

FINAL RESULTS

Representation	Embedding	Model	Accuracy
3-mer	One-hot encoding	Base CNN	57%
3-mer	One-hot encoding	Fine tuned CNN	64%
4-mer	Word2Vec	CNN+LSTM	68.87%
4-mer	One-hot encoding	CNN+BiLSTM	70.51%
4-mer	Word2Vec	CNN+BiLSTM	71.87%

Final reflections and future steps

Models can be developed for more number of datasets.

Hybrid models can be furthur developed to give higher accuracy scores like SVM's.

Different encoding techniques can be utilised like BERT and Fasttext.

Thank You!

May your DNA be safely wrapped with Histone :)

Paper links:- https://www.scirp.org/journal/paperinformation?paperid=65923
https://www.ijitee.org/wp-content/uploads/papers/v11i10/J927309111022.pdf