# Schneider Electric's Technical Project

**Topic: Netflix Model**

**Presented By:** Anmol Srivastava

# Dataset - Overview

```
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       7787 non-null   object
 1   type          7787 non-null   object
 2   title         7787 non-null   object
 3   director      5398 non-null   object
 4   cast          7069 non-null   object
 5   country       7280 non-null   object
 6   date_added    7777 non-null   object
 7   release_year  7787 non-null   int64
 8   rating        7780 non-null   object
 9   duration      7787 non-null   object
 10  listed_in     7787 non-null   object
 11  description   7787 non-null   object
dtypes: int64(1), object(11)
memory usage: 730.2+ KB
```

**No. of Rows:** 7787

**No. of Columns:** 12

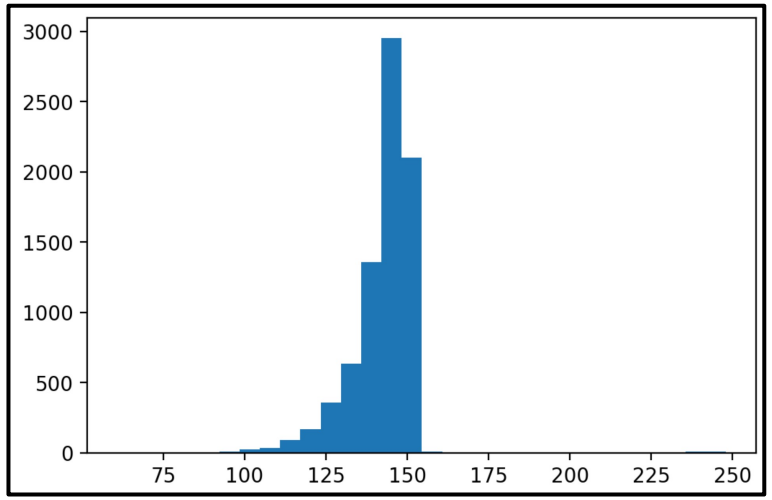
```
show_id          0
type             0
title            0
director      2389
cast           718
country        507
date_added      10
release_year     0
rating           7
duration         0
listed_in        0
description      0
dtype: int64
```

**Missing Data**

**No Duplicate Entries !!**

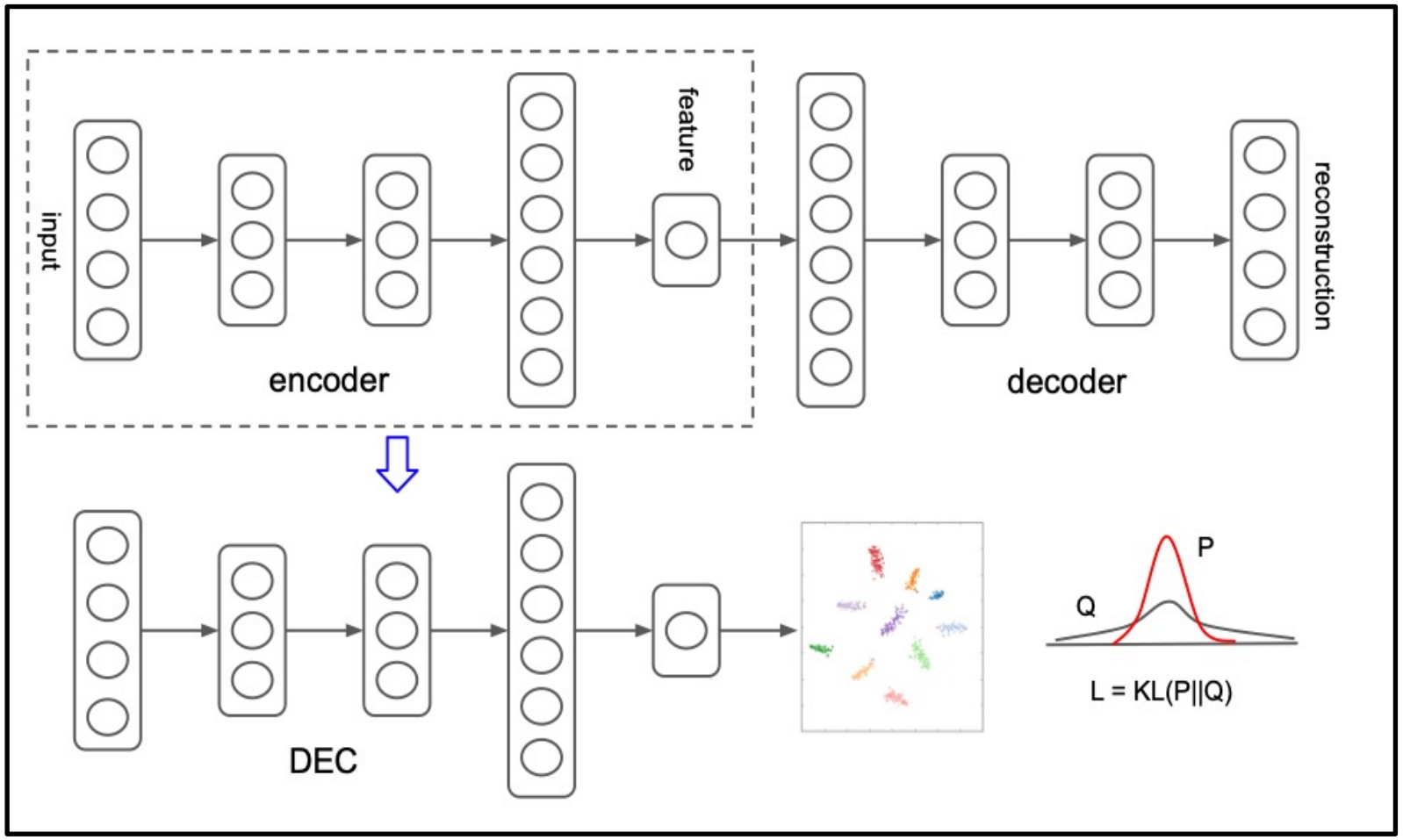
# Text - Processing

- Using ‘description’ text of movies for clustering purposes.
- Tokenizing data with 10000 most frequent words
- Padding resulting sequence to 1500 length.
- Scaling features to the range between 0 and 1




*Distribution of Sequence Length*

# Clustering - Framework

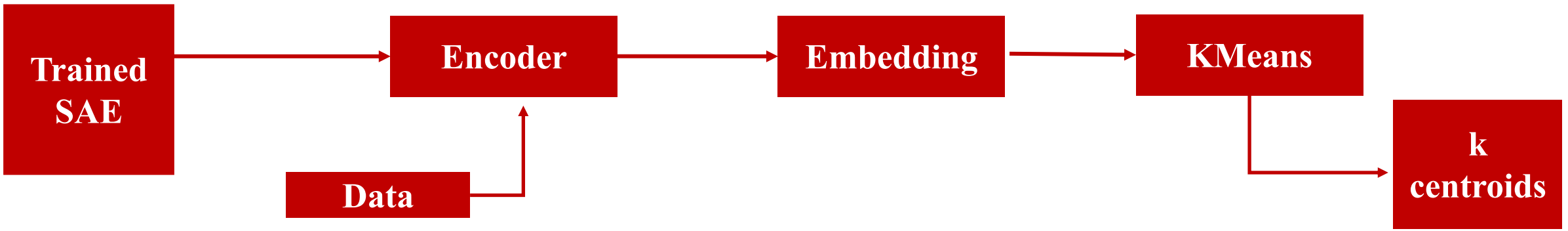



- $f_\theta: X \rightarrow Z$
- Clusters data by learning k clusters center in Z
- Parameter initialization with a deep autoencoder
- Parameter Optimization

Xie, Junyuan et al. "Unsupervised Deep Embedding for Clustering Analysis." *ArXiv*abs/1511.06335 (2016): n. pag.

# Clustering – Parameter Initialization

- DEC initialized by stacked Auto Encoder
- AE trained by minimizing the MSE loss
- Discard ***decoder*** after training to use ***encoder*** for initial mapping: $\boldsymbol{X} \longrightarrow \boldsymbol{Z}$
- ***k*** initial centroids obtained by ***kmeans*** on embedded data points $\mathbf{Z}$



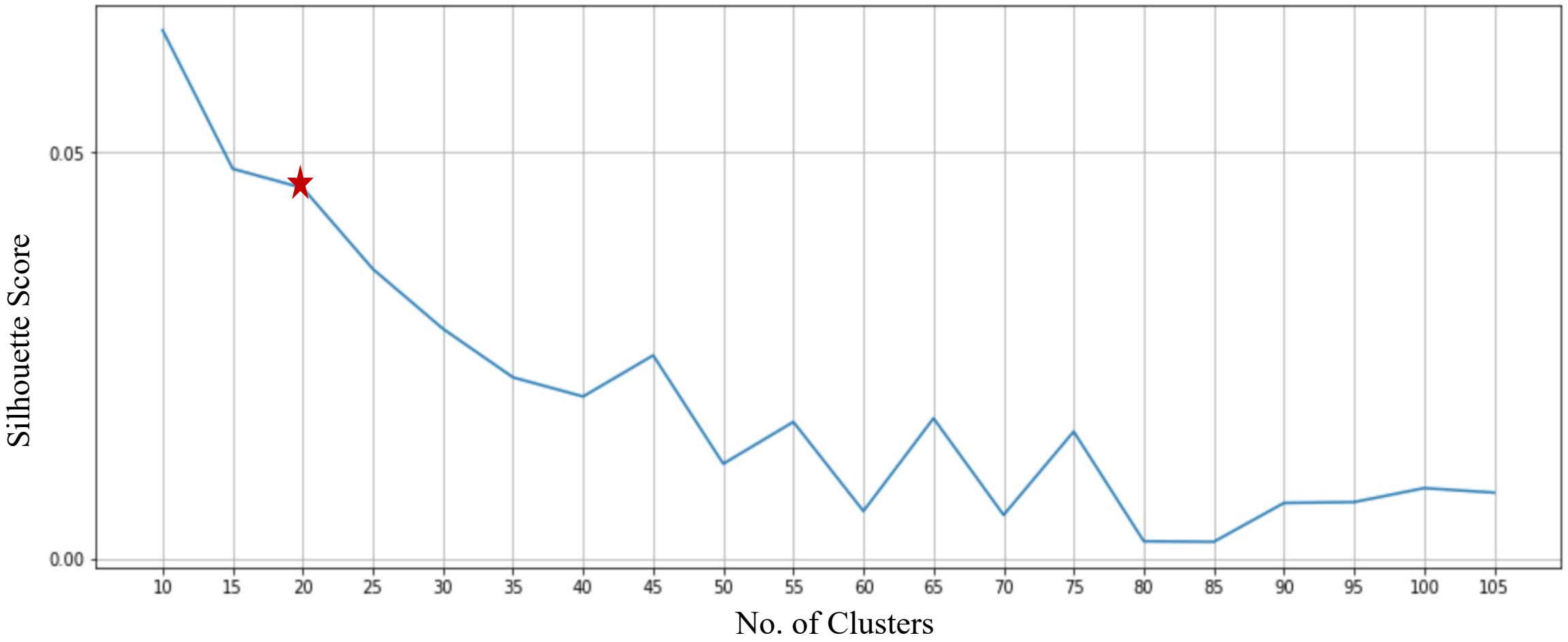
# Clustering – Parameter Optimization

- Parameter Optimization by alternating two steps:
  - ***Soft Assignments*** between embedded points and cluster centroids
  - ***Refinement*** of embedding and cluster centroids by learning target distribution
- ***Soft Assignments:*** Probability of assigning sample *i* to cluster *j*

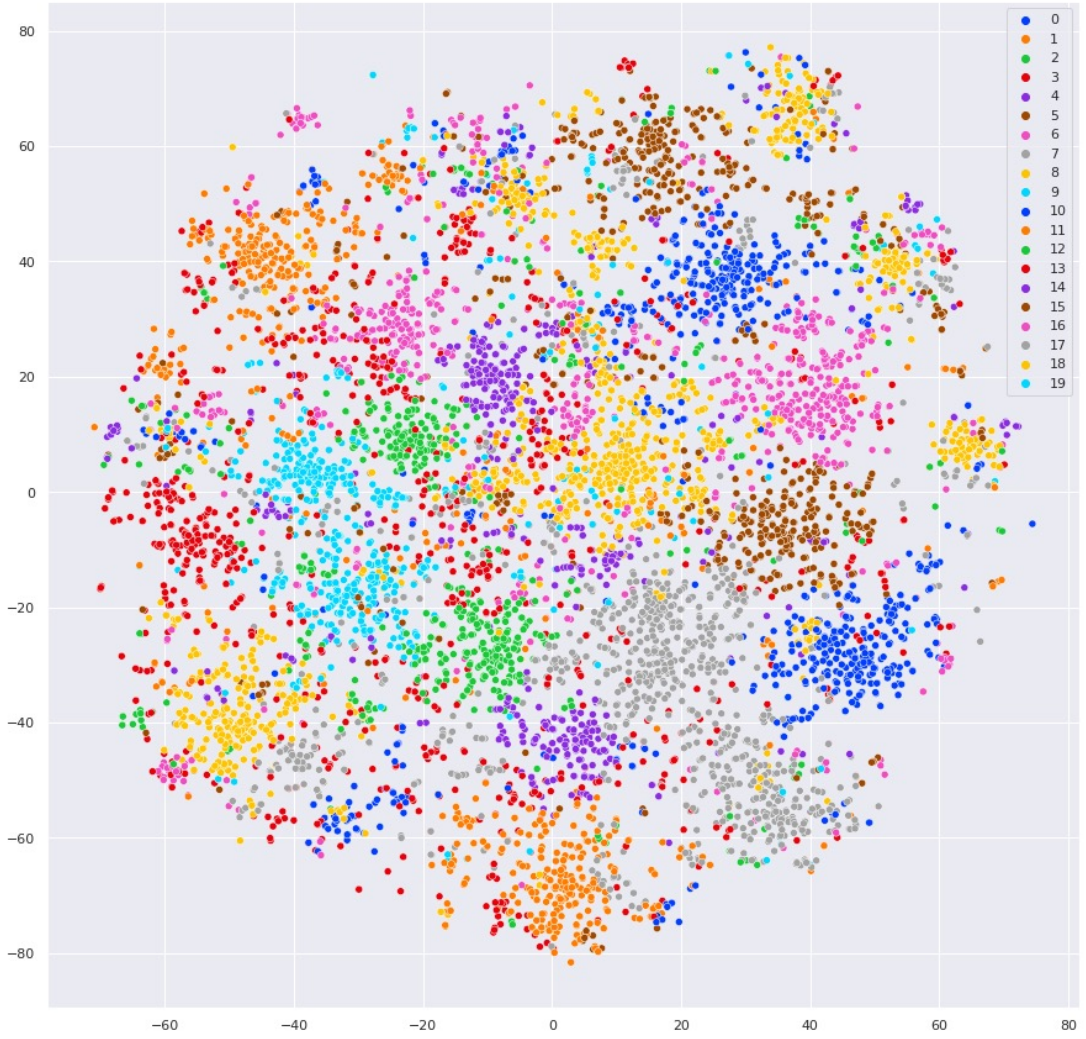
$$q_{ij} = \frac{(1 + \|z_i - \mu_j\|^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'}(1 + \|z_i - \mu_{j'}\|^2/\alpha)^{-\frac{\alpha+1}{2}}}$$

- ***Refinement*** of clusters involves matching of soft assignment to target distribution via minimizing KL divergence loss.

# Clustering – Optimal Number of Clusters



# Clustering – Results



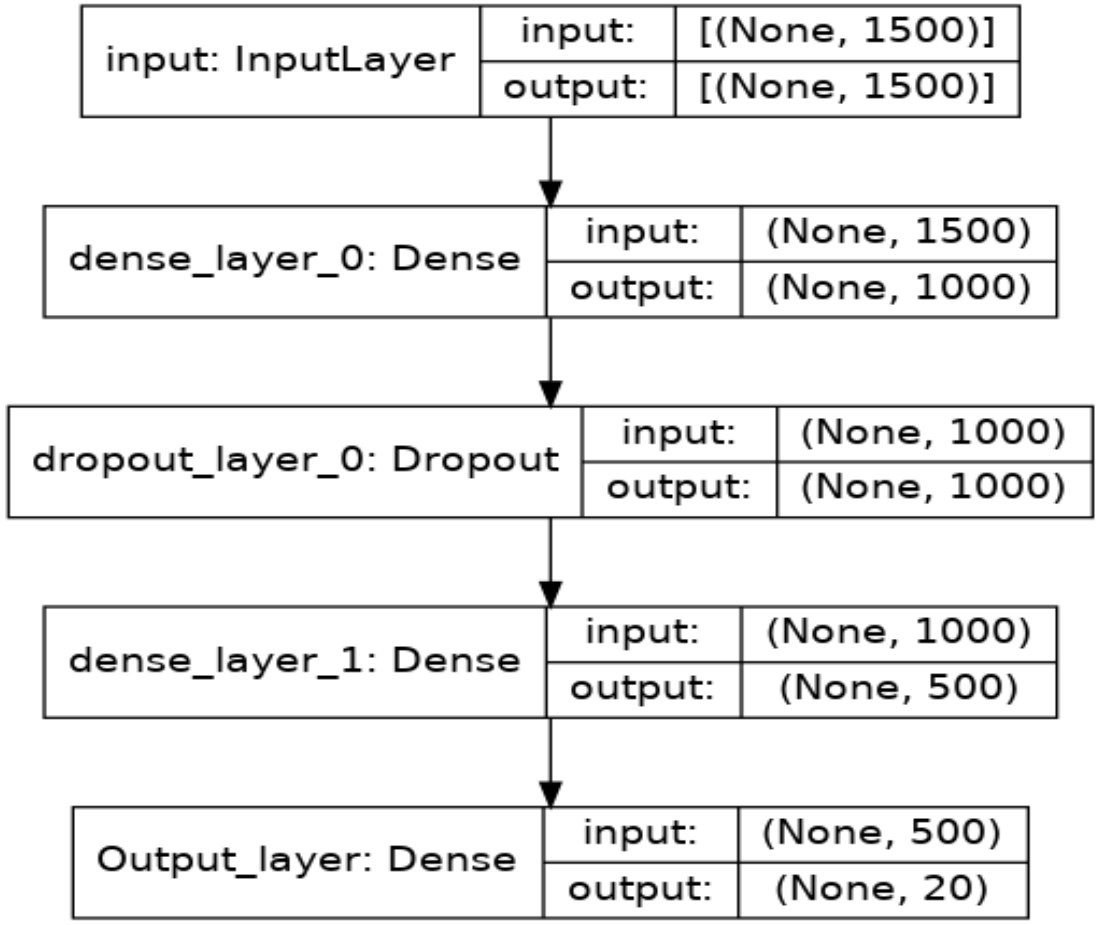

# Classification – Data Splitting


**Training Set:** X {6229, 1500}, Y {6229, 20}

**Validation Set:** X {779, 1500}, Y {779, 20}

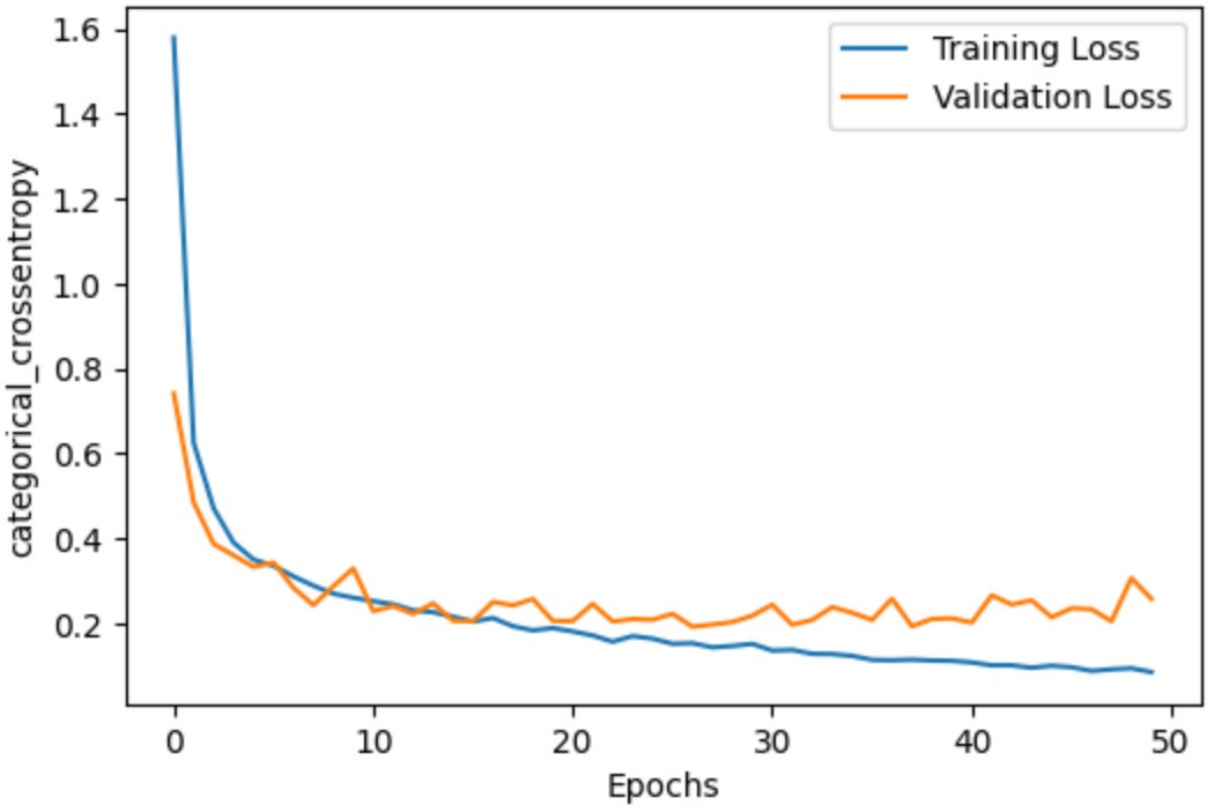
**Test Set:** X {779, 1500}, Y {779, 20}

# Classification – Deep Neural Network



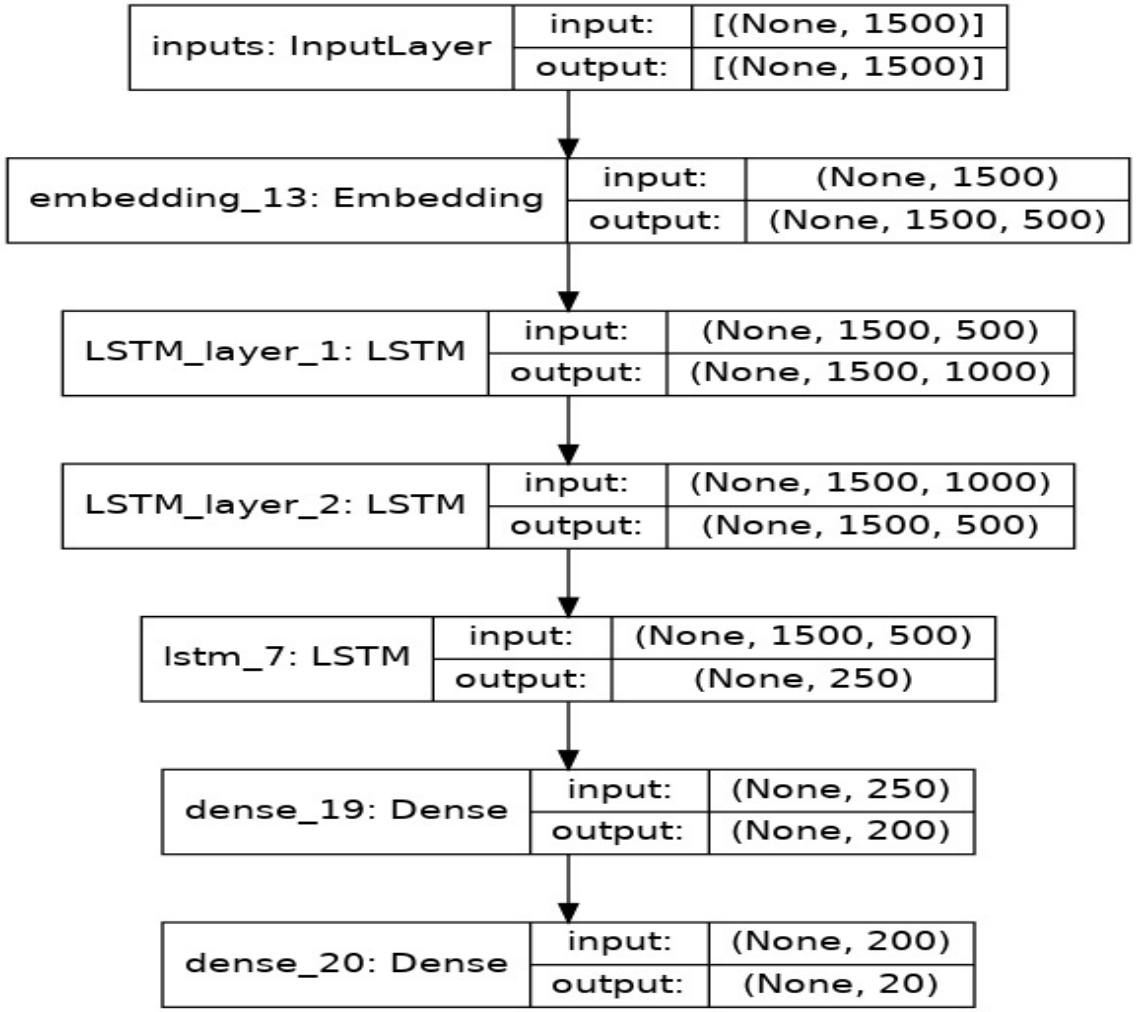

- **Activation Function:** ReLU
- **Activation Function (Output):** Softmax
- **Batch Size:** 128
- **Epochs:** 50
- **Optimizer:** RMSprop
- **Learning Rate:** 0.001
- **Momentum:** 0.9

# Classification – Deep Neural Network Performance



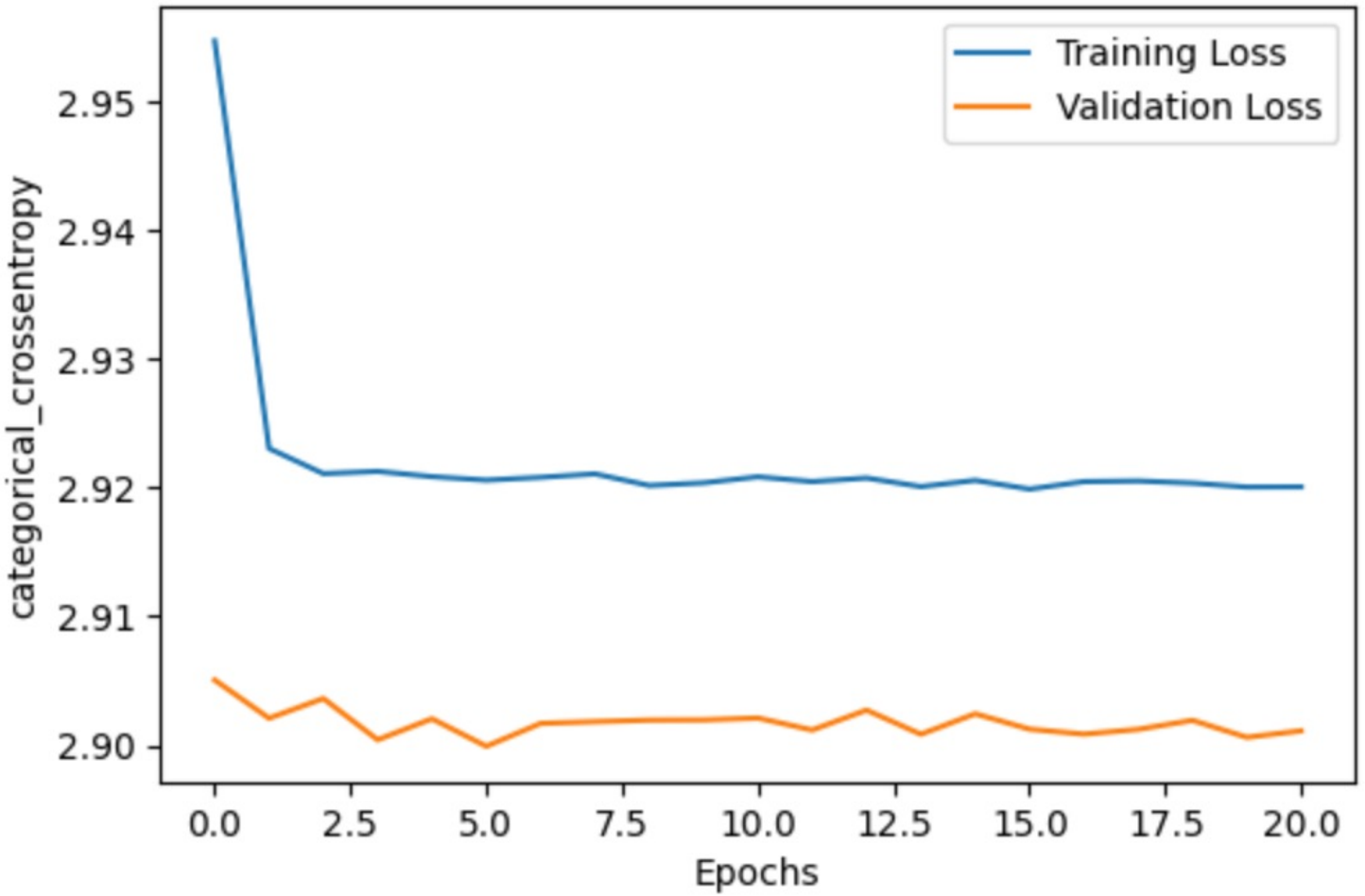

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.97 | 0.88 | 0.92 | 32 |
| 1 | 0.97 | 0.80 | 0.88 | 41 |
| 2 | 0.82 | 1.00 | 0.90 | 23 |
| 3 | 0.96 | 0.92 | 0.94 | 50 |
| 4 | 0.85 | 0.81 | 0.83 | 21 |
| 5 | 0.96 | 0.82 | 0.89 | 33 |
| 6 | 0.87 | 0.92 | 0.89 | 36 |
| 7 | 0.94 | 0.93 | 0.94 | 72 |
| 8 | 0.94 | 0.83 | 0.88 | 36 |
| 9 | 0.88 | 0.95 | 0.91 | 22 |
| 10 | 1.00 | 0.94 | 0.97 | 48 |
| 11 | 0.90 | 1.00 | 0.95 | 37 |
| 12 | 0.88 | 0.93 | 0.90 | 30 |
| 13 | 0.93 | 1.00 | 0.96 | 37 |
| 14 | 0.86 | 0.86 | 0.86 | 29 |
| 15 | 0.86 | 0.98 | 0.92 | 50 |
| 16 | 0.95 | 0.93 | 0.94 | 40 |
| 17 | 0.89 | 0.94 | 0.91 | 33 |
| 18 | 0.89 | 0.92 | 0.90 | 83 |
| 19 | 0.96 | 0.88 | 0.92 | 26 |
| accuracy | | | 0.92 | 779 |
| macro avg | 0.91 | 0.91 | 0.91 | 779 |
| weighted avg | 0.92 | 0.92 | 0.91 | 779 |

# Classification – Deep LSTM Network



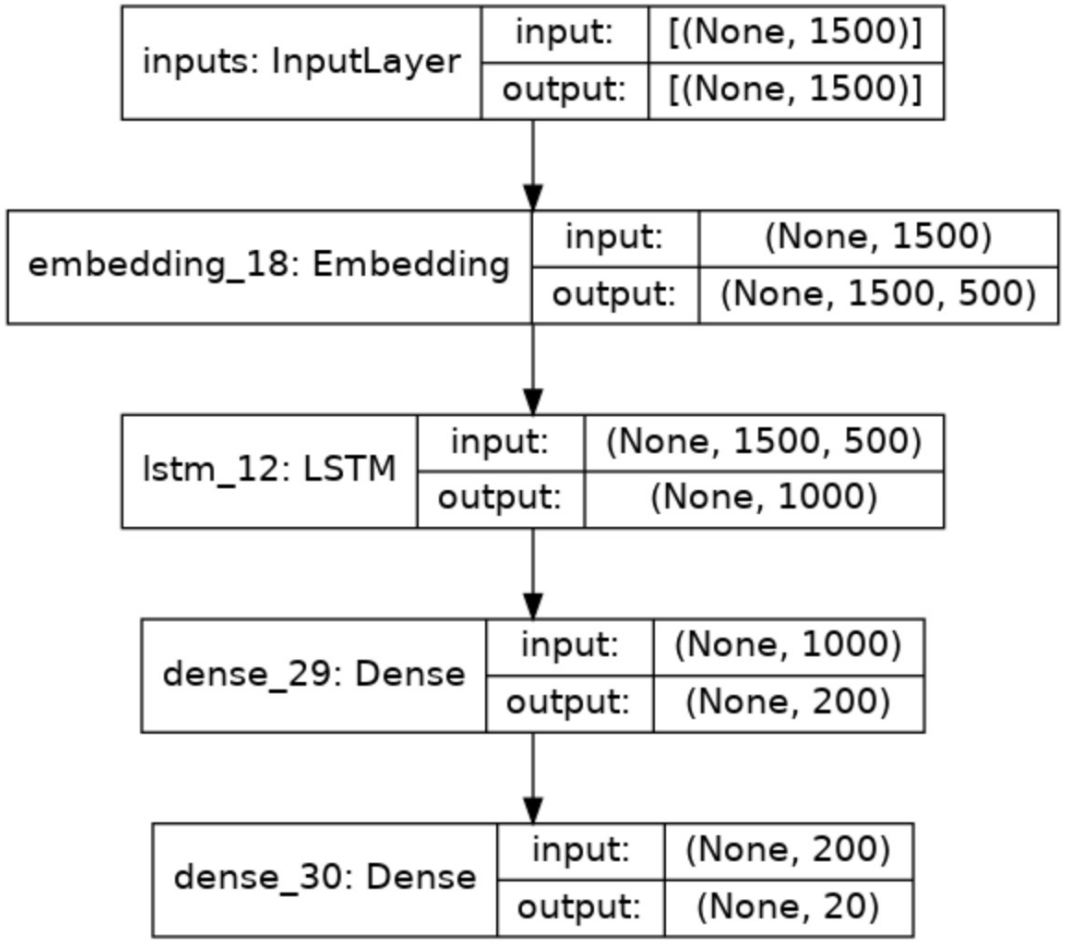

- **Activation Function:** ReLU
- **Activation Function (Output):** Softmax
- **Batch Size:** 128
- **Epochs:** 100
- **Optimizer:** SGD
- **Learning Rate:** 0.01
- **Momentum:** 0.9

# Classification – Deep LSTM Performance



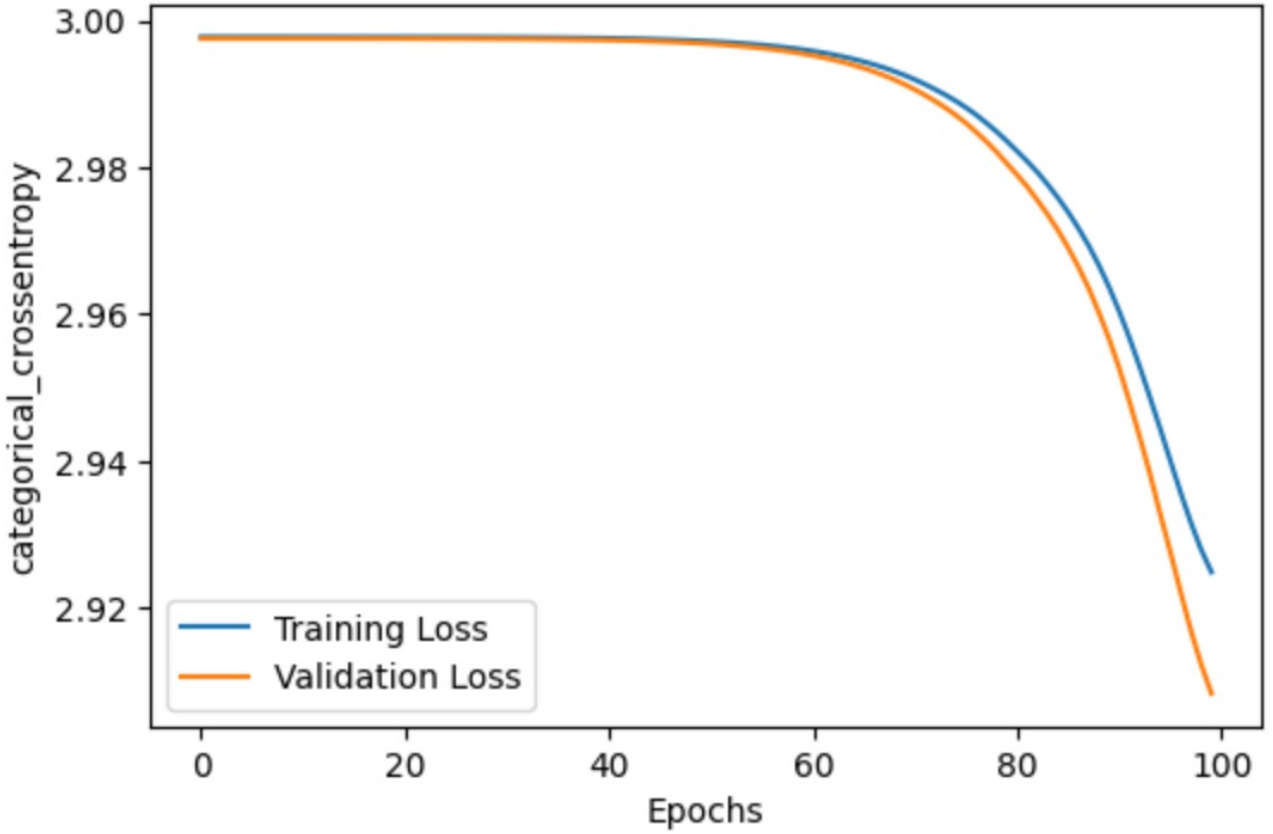

High Bias !!

# Classification – Shallow LSTM Network




- **Activation Function:** ReLU
- **Activation Function (Output):** Softmax
- **Batch Size:** 128
- **Epochs:** 100
- **Optimizer:** SGD
- **Learning Rate:** 0.1
- **Momentum:** 0.9

# Classification – Shallow LSTM Performance




## Slow Training !!

- Requires more training iterations.
- Losses tend to decrease significantly from 60$^{th}$ epoch. But stopped early due to no further improvements.

# THANK YOU!!