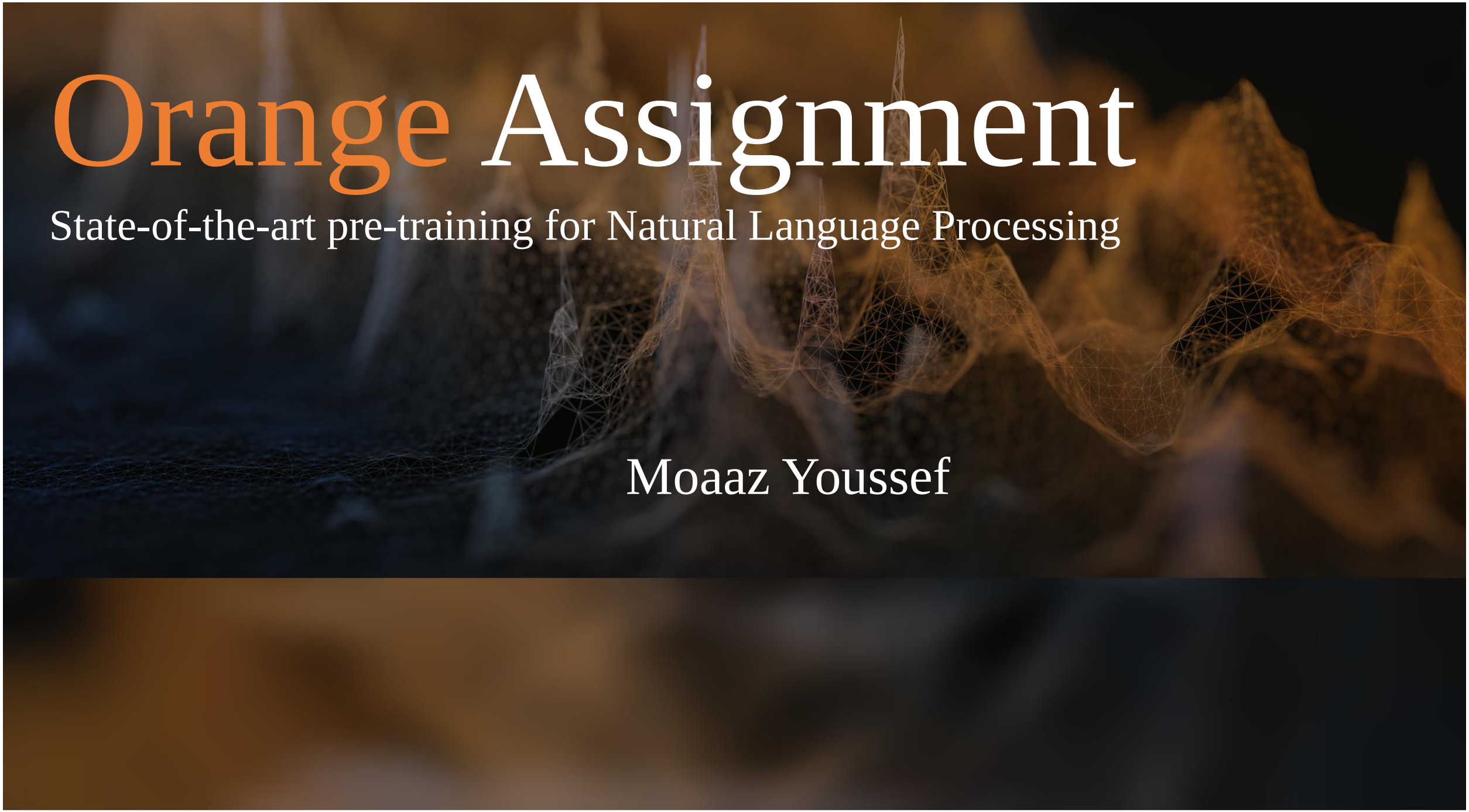
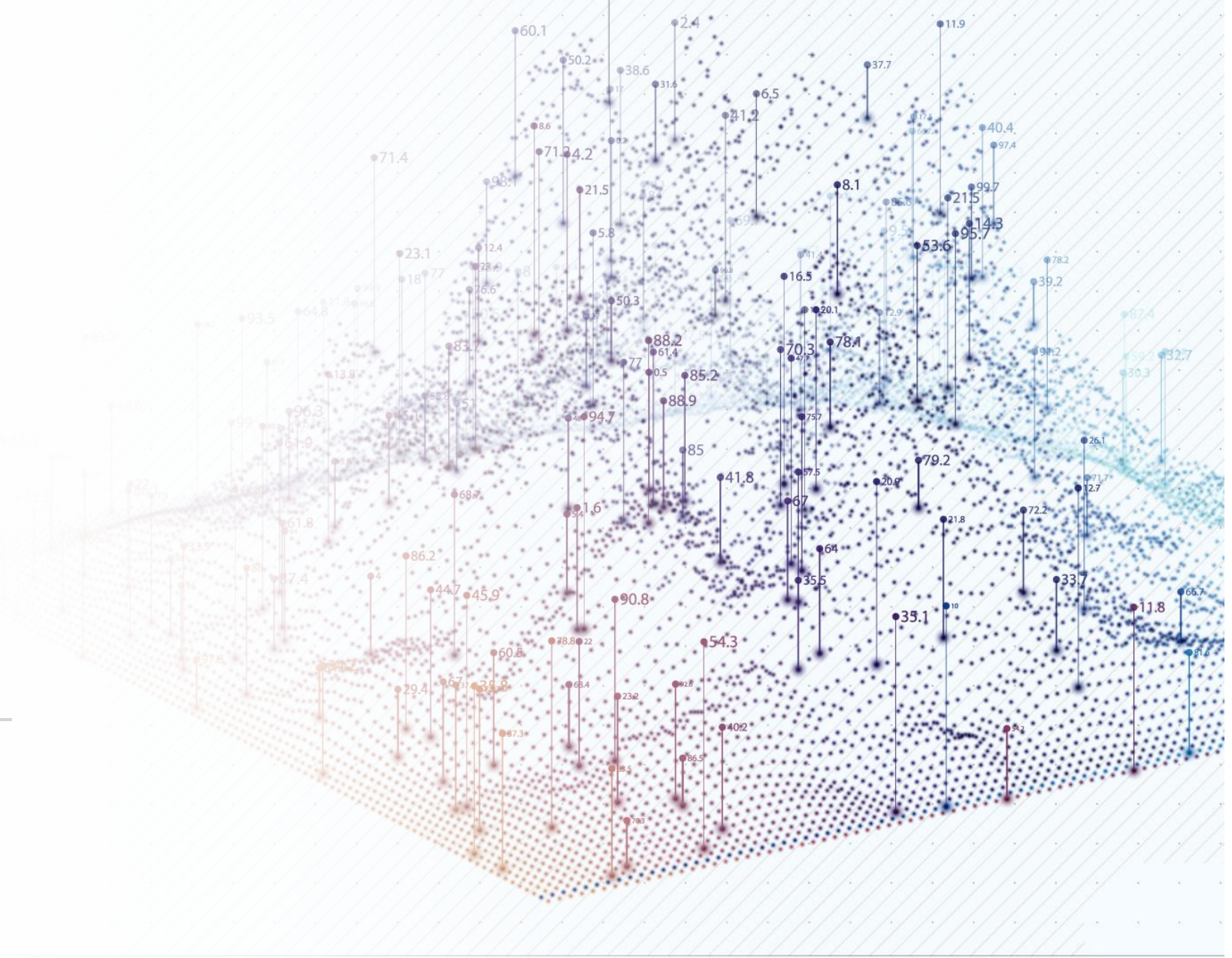


Orange Assignment

State-of-the-art pre-training for Natural Language Processing

Moaaz Youssef

Problem Statement



Problem Statement

- To predict, using Bert and XlNet, which category-out of five-BBC news articles fall into
- Categories are sport, business, politics, entrainment and technology.
- To compare the classification accuracy and time complexity of both algorithms

Methodology

Data Preprocessing

Nulls check : There is no nulls.

Duplicates check: There is no duplicates.

Balance check : By creating pie-chart to detect categories imbalance. Categories are imbalanced, with sport category dominates the distribution.

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Data Preprocessing

Label encoding: encode categories to one-hot-encode representation

Sample distribution: Population is assumed to have same distribution as the sample.

Data splitting: splitting the data into 80% for training and 20% for validation. Using sampling stratification method, the training dataset maintains the same imbalanced distribution as the sample.

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Data Preprocessing

Tokenization

1- BERT tokenizer: is a WordPiece tokenizer.

Example:

BertTokenizer.from_pretrained("bert-base-uncased")

tokenizer.tokenize("I have a new GPU!")

Output: ["i", "have", "a", "new", "gp", "##u", "!"]

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Data Preprocessing

Tokenization

2- Xlnet tokenizer: is a SentencePiece tokenizer.

Example:

```
tokenizer = XLNetTokenizer.from_pretrained("xlnet-base-cased")
tokenizer.tokenize("Don't you love ☺ Transformers? We sure do.")
```

Output:

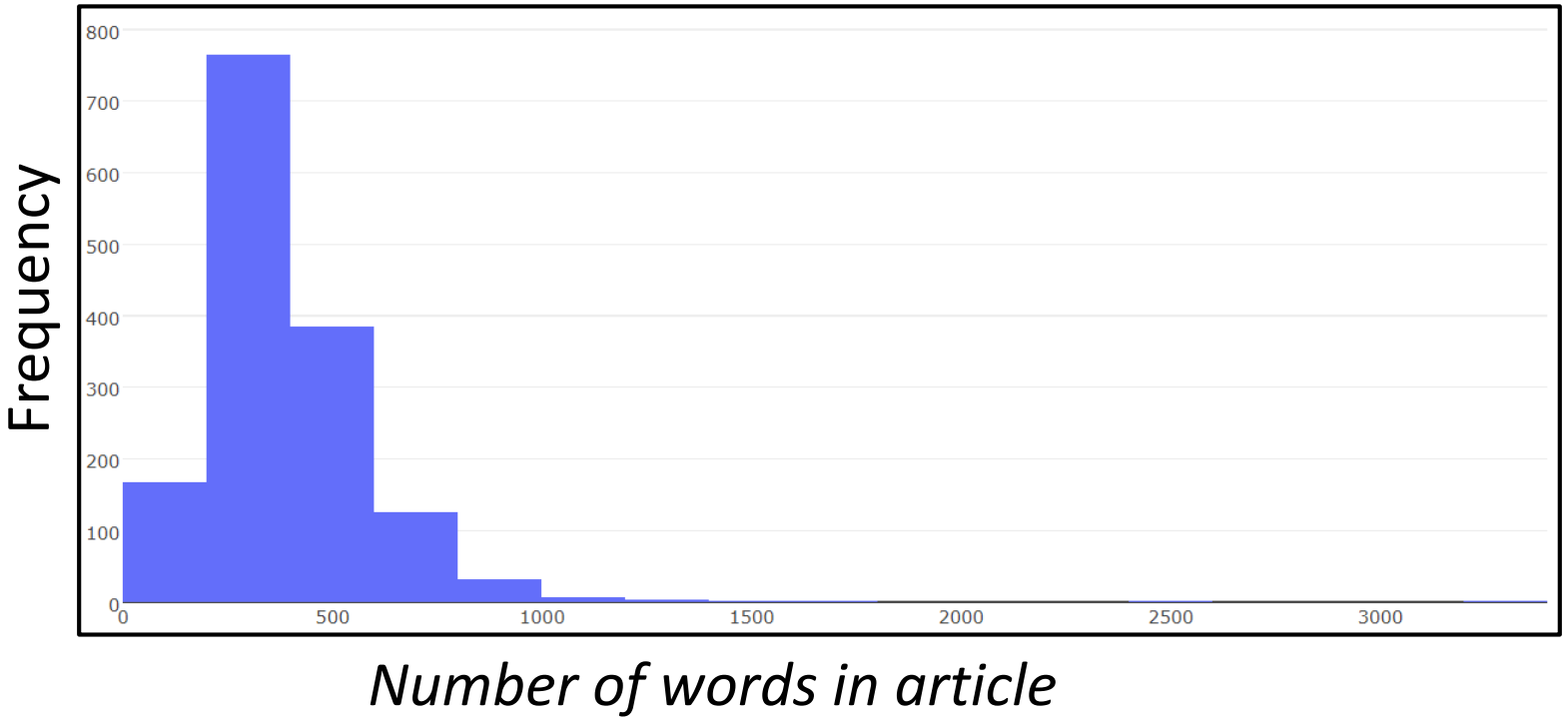
```
["▁Don", "’", "t", "▁you", "▁love", "▁", "☺", "▁", "Transform",
"ers", "?", "▁We", "▁sure", "▁do", "."]
```

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Data Preprocessing

Words distribution in articles: to determine mean and mode of number of words per article, and to choose efficient tokenization length.



Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Data Preprocessing

Model Inputs: Using tokenizer length of 512 to capture part of distribution tail, inputs ID's and attention masks were created.

Inputs ID's: are numerical representations of tokens building the sequences that will be used as input by the model.

Example:

tokenizer("A Titan RTX has 24GB of VRAM")

Output:

[101, 138, 18696, 155, 1942, 3190, 1144, 1572, 13745, 1104, 159, 9664, 2107, 102]

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Data Preprocessing

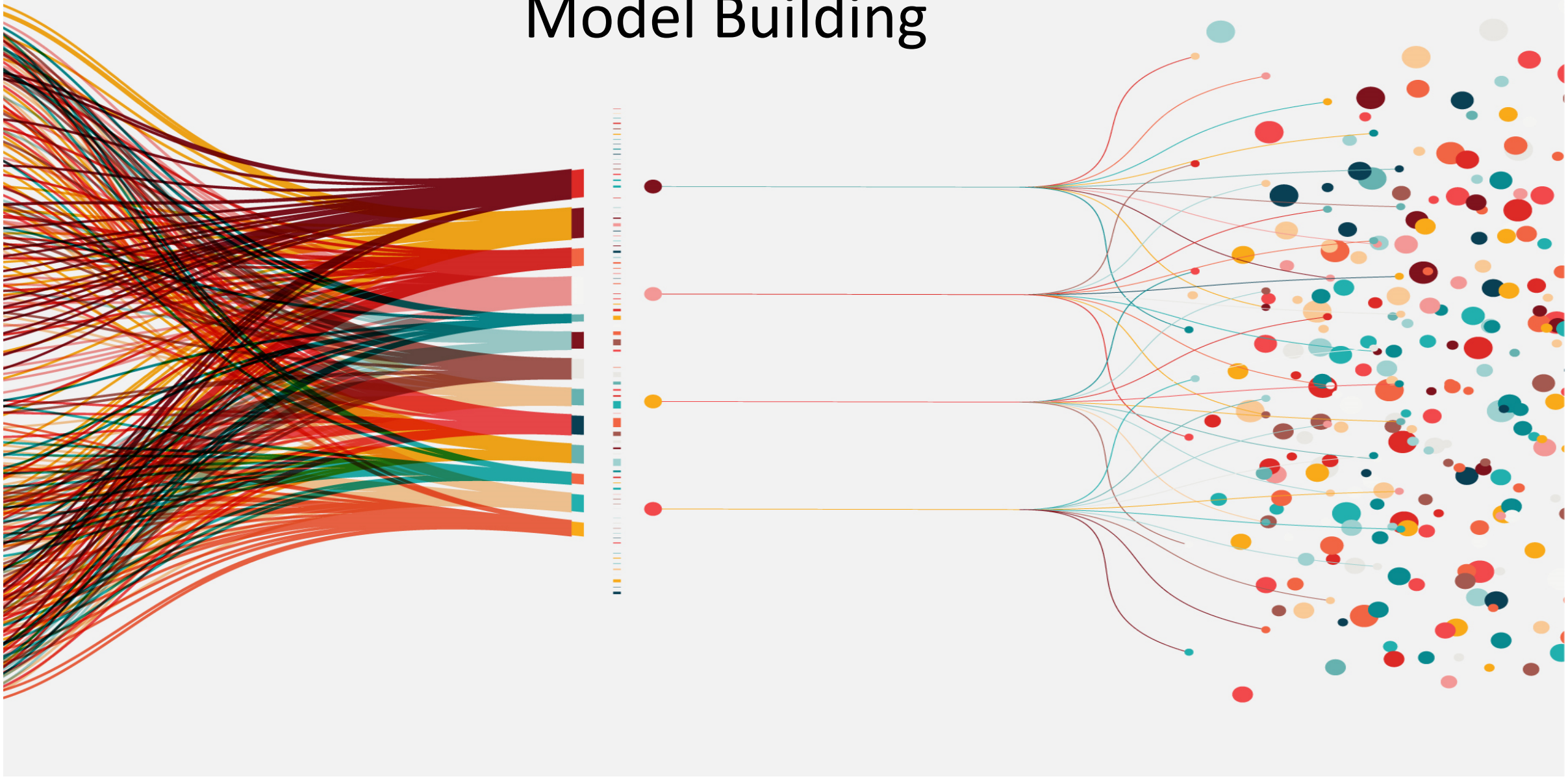
Attention mask : is an argument indicates to the model which tokens should be attended to, and which should be padded. 1 indicates a value that should be attended to, while 0 indicates a padded value.

Tensorflow data: created tensorflow data, from inputs ID's and attention masks, for training and evaluation to return (features, labels) pairs, as expected by keras.Model.fit

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Building



Model Building

BERT blocks

i. **Inputs layer**
   i. Input ID's
   ii. Attention mask
ii. **Bert Base Uncased layer**
   i. 12 layers (transformer blocks), 768 hidden layer, 12 attention heads and 110 million parameters
iii. **Dense layer**
   i. GELU activation function with 1024 neurons which takes pooled out as an input from Bert layer
iv. **Dropout layer**
   i. With 0.1 dropout rate to prevent overfitting

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Building

BERT blocks

I. **Dense layer**
    I. Softmax activation function which gives probability of each category as an (model output).

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Building

XLNet blocks

i. **Inputs layer:**
   i. Input ID's
   ii. Attention mask
ii. **XLNet Base cased layer:**
   i. 12 layers (transformer blocks), 768 hidden layer,12 attention heads and 110 million parameters
iii. **Dense layer:**
   i. GELU activation function with 1024 neurons which takes last hidden state as an input from XLNet layer
iv. **Dropout layer:**
   i. With 0.1 dropout rate to prevent overfitting

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Building

XLNet blocks

I. **Dense layer:**
   I. Softmax activation function which gives probability of each category (model output).

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Building

Metrics

i. **Categorical accuracy:**
    i. Calculates the percentage of predicted values that match with actual values for one-hot-encode labels
ii. **Precision:**
    i. Calculates the percentage of relevant instances among the retrieved instances
iii. **Recall:**
    i. Calculates the percentage of relevant instances that were retrieved

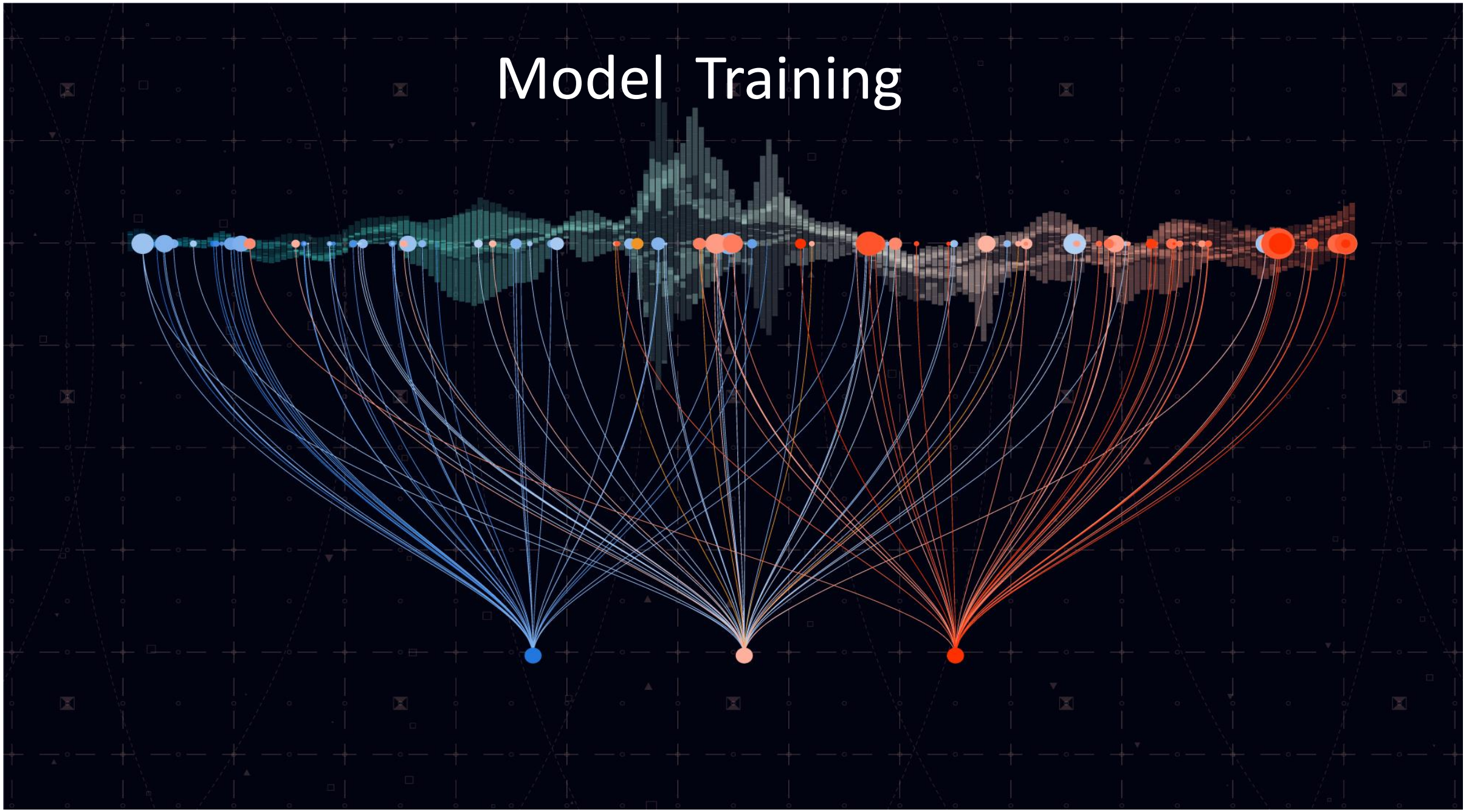
Loss

I. Opted for categorical cross-entropy loss function since the case is multi-class classification .

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Training



Model Training

Hyperparameters

i. Literature recommends the following range of hyperparamters to work well across all tasks:

   i. Batch size: 8,16,32
   ii. Learning rate (Adam): 5e-5, 3e-5, 2e-5
   iii. Number of epochs: 2, 3,4

ii. After trying this range of hyperparameters, the following hyperparameters yielded best metrics

iii. However, hyperparameters tuning (Random, Grid or Bayesian search) is the recommended approach in time abundance scenarios.

Models	Epochs	Batch size	Learning rate	Decay
Bert	4	16	$5e^{-5}$	e^{-6}
XLNet	4	8	$2e^{-5}$	e^{-6}

Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Training

Classes weights

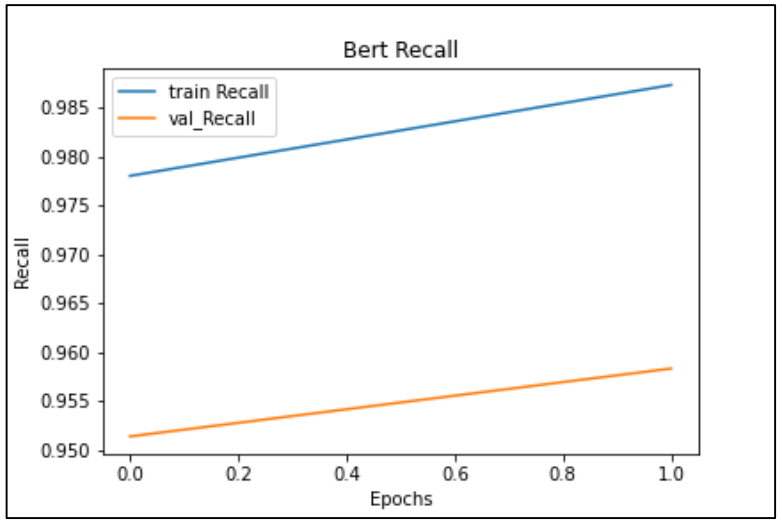
- Classes weights tell the model to pay more attention to samples from an under-represented classes.
- Calculated the classes weights to deal with imbalanced data.
- Feed the classes weights dictionary as a parameter in Model.Fit method

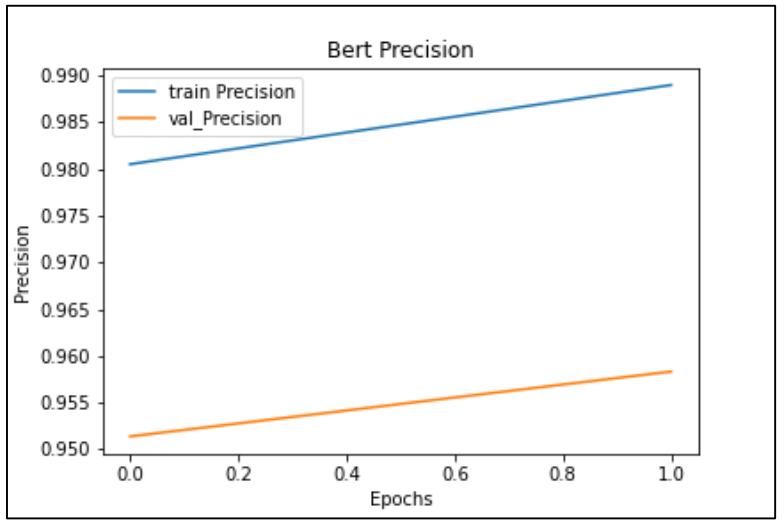
Methodology

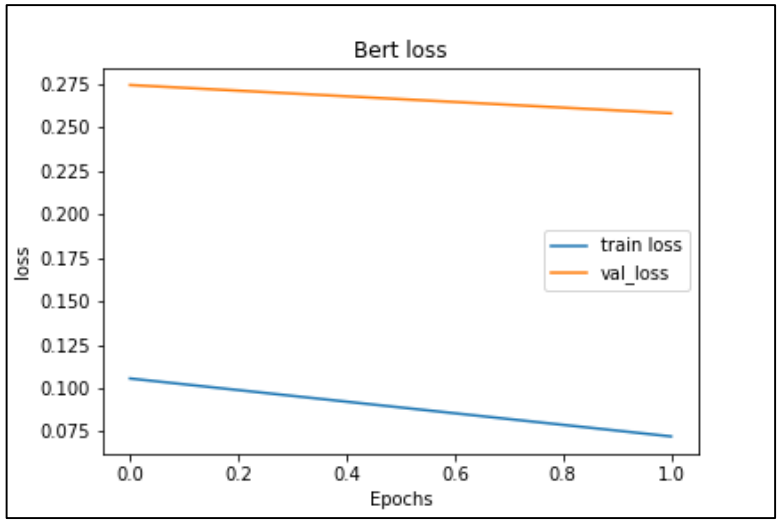
- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

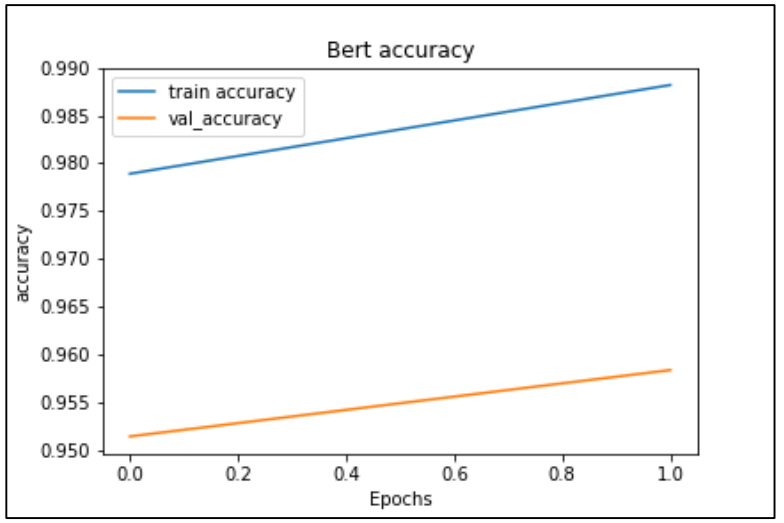
Model Validation

Model Validation-Bert





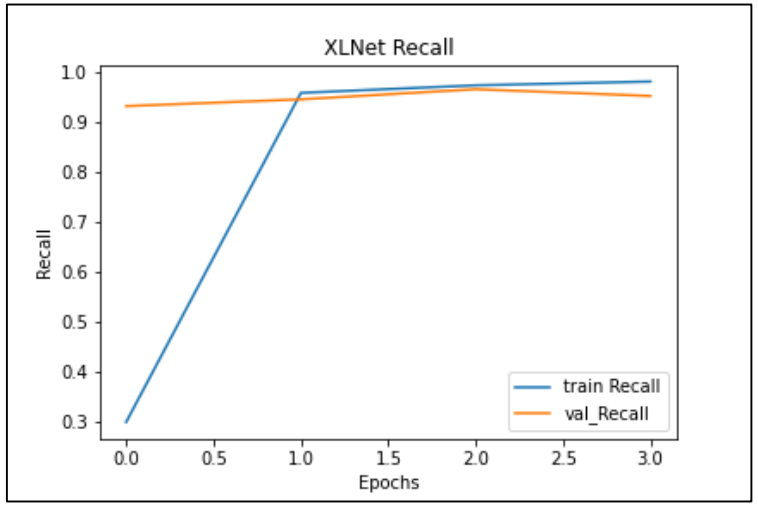


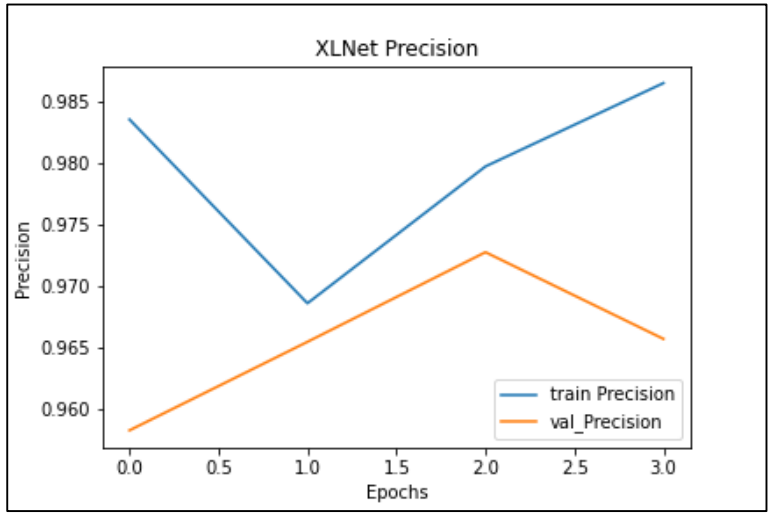


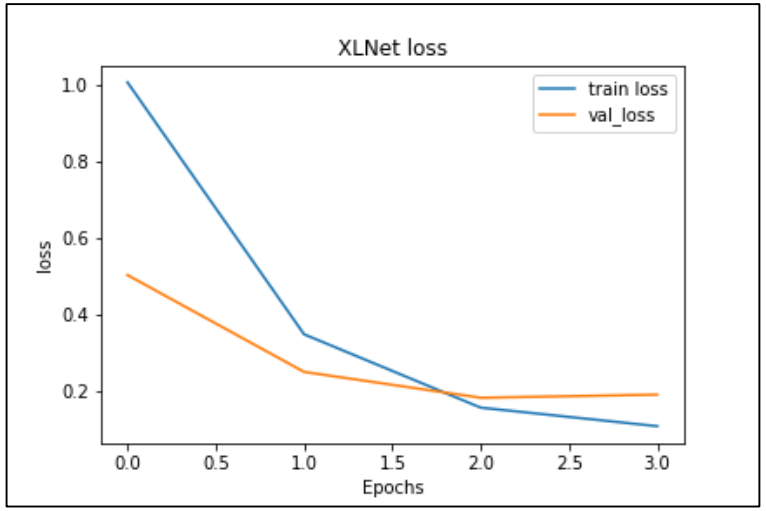
Methodology

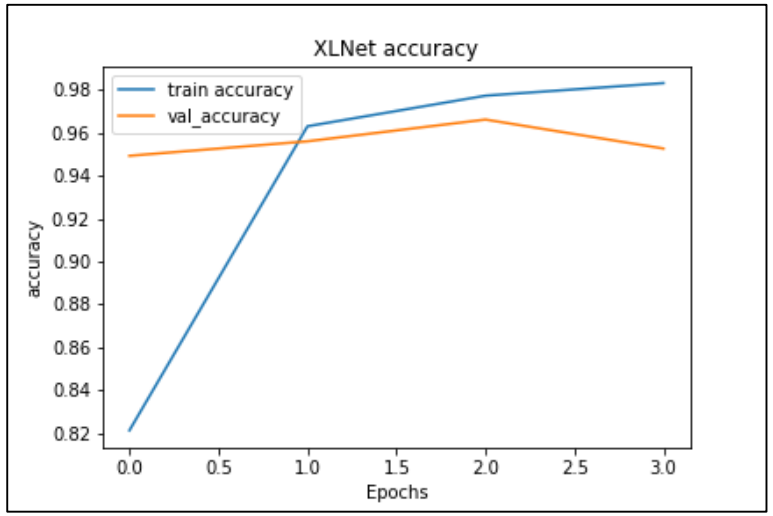
- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Validation-XLNet









Methodology

- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Model Comparison

- **Outputs Evaluation**

Bert model: Scored 97.959% accuracy on unseen data

XLNet: Scored 97.006% accuracy on unseen data

- **Time Complexity**

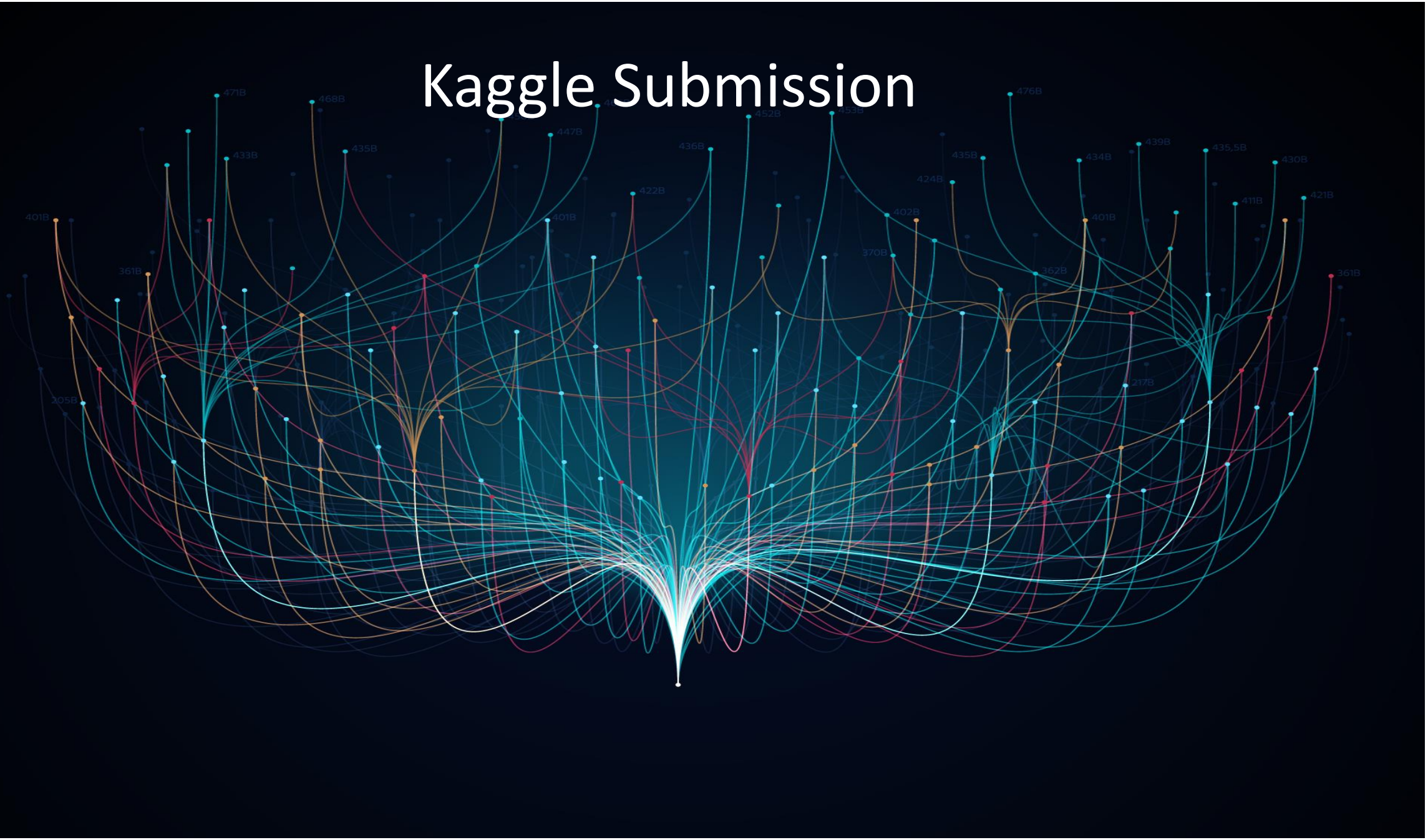
Bert: Took around 58 minutes to train on Cloud Tensor Processing Units (TPUs), and around 30 minutes with early stopping.

XLNet: Took around 76 minutes to train on Cloud Tensor Processing Units (TPUs).

Methodology

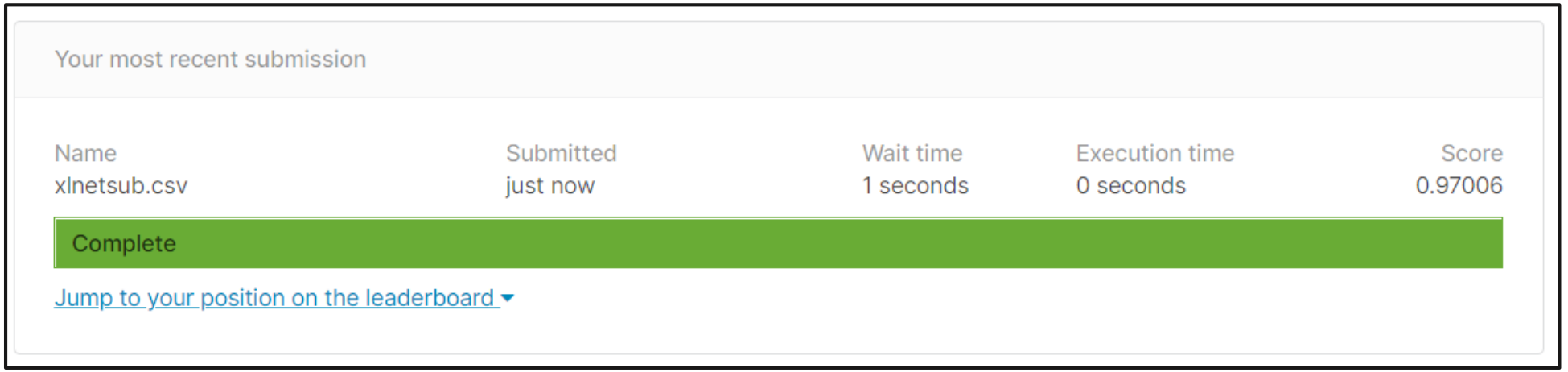
- Data preprocessing
- Model building
- Model training
- Model validation
- Models comparison

Kaggle Submission

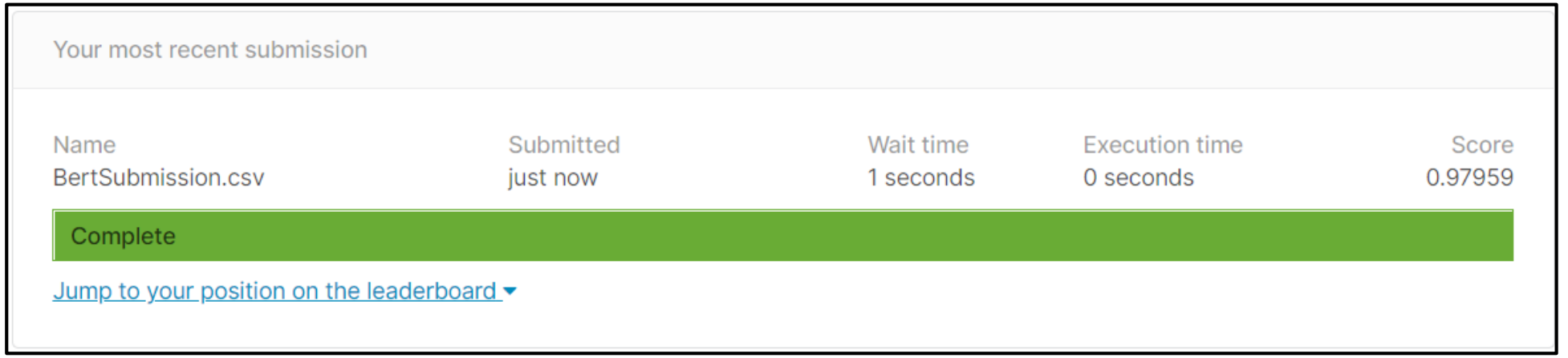


Kaggle Submission

XLNet



Bert



Thanks