भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Jargon/Terminology Detection

(AAAI-23 Workshop on Scientific Document Understanding)

Debika Samanta (CS22MTECH12001)
Tuhin Dutta (AI21MTECH02002)

Under the guidance of :

Dr. Maunendra Sankar Desarkar

Contents

Motivation & Problem Statement

Technical terminology expressions are unique to a particular field and have implied meanings that do not conform to common expectations.

Non-experienced individuals may fail to understand technical terms or interpret common words in a different sense than intended, creating a significant entry barrier to reading scholarly writing.

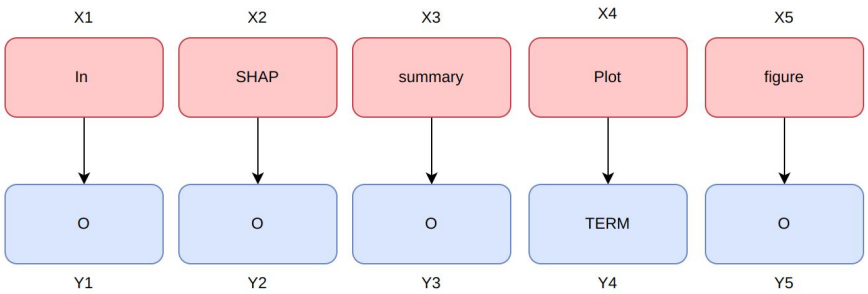
- *Task aims to identify domain-specific technical terminology or jargon used in scientific research papers.*
- *Each sentence is treated as a sequence of words and the model assigns a label to each word in the sequence based on whether it is a technical term or jargon or not. The goal is to identify and tag all the technical terms or jargon in each sentence, which makes it a sequence labeling problem.*

Literature review

- Jargon Detection deals with the detection of special terminology used by a particular group of the people in that profession and this word may not be regularly used by the people of other domain.

- This comes under the Sequence Labeling task where we assign, to each word x_i in an input word sequence, a label y_i , so that the output sequence Y has the same length as the input sequence X.

- Jargon Detection is important as it enables us to identify the context sensitive terminologies used in a specific domain.



Dataset Analysis

The provided data contains three columns:

- ***token_id*** contains the domain, document ID, sentence ID, and token ID, in order, separated by /.
- ***token*** contains the cased tokens.
- ***label*** contains either "O" or "TERM"

The total data is divided into Train, Validation, and Test sets in three different files.

The distribution of labels in this datasets are as follows.

Train

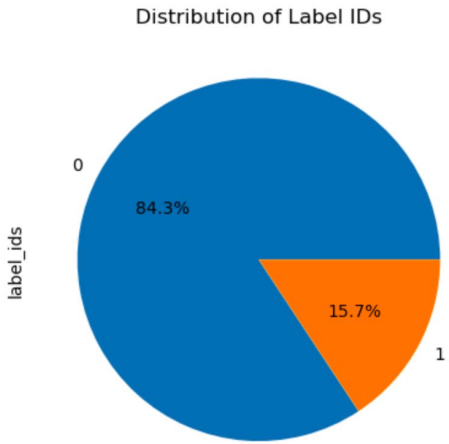
TERM	54624
O	520286

Validation

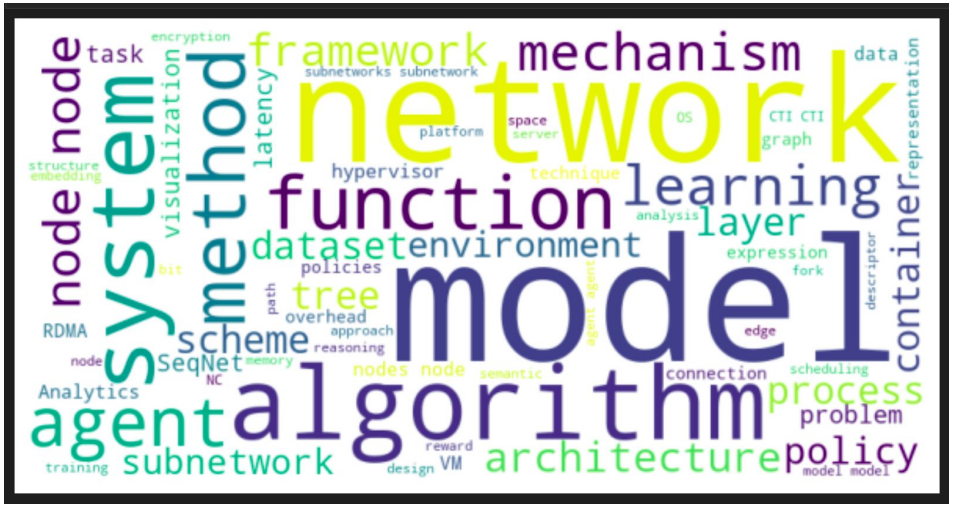
TERM	3664
O	33479

Test

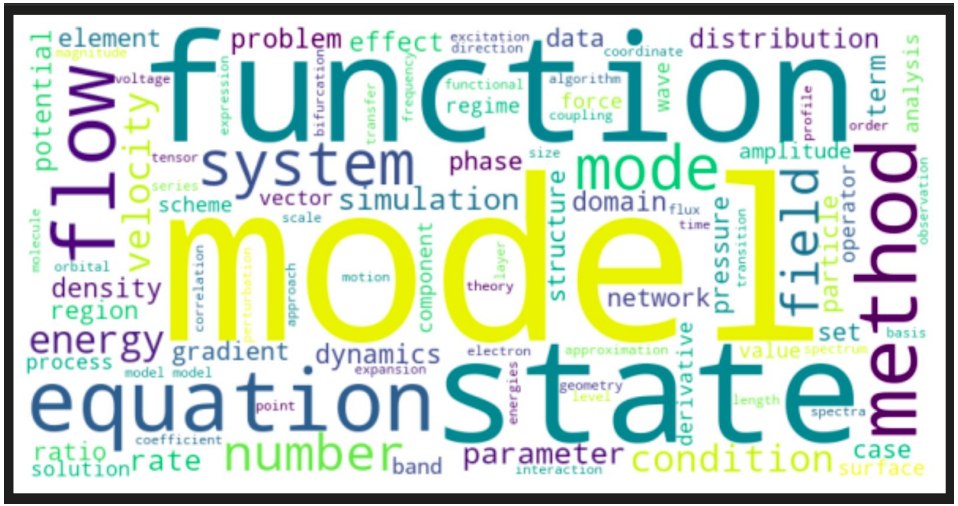
TOTAL ENTRIES 42358



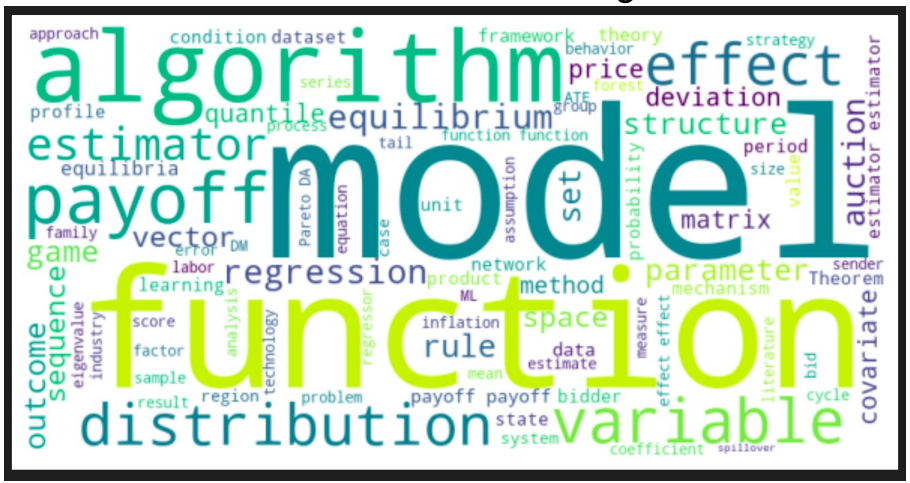
Computer terminologies



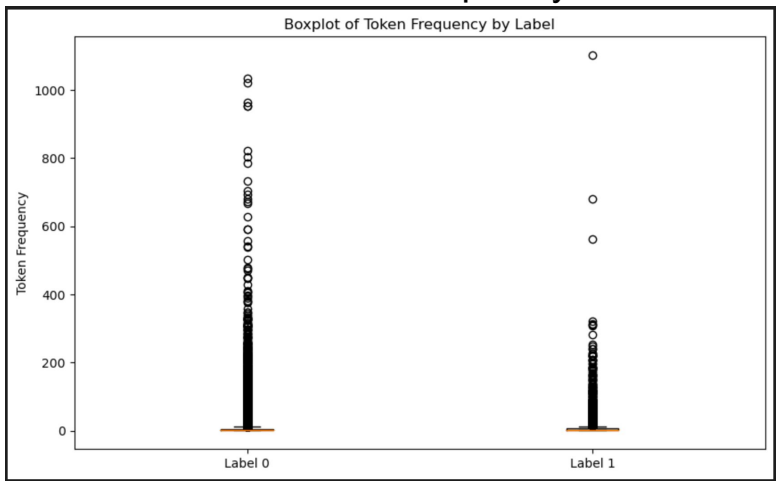
Physics Terminology



Economics terminologies



Label Frequency



Baseline Algorithm

The main s used for the project are as follows :

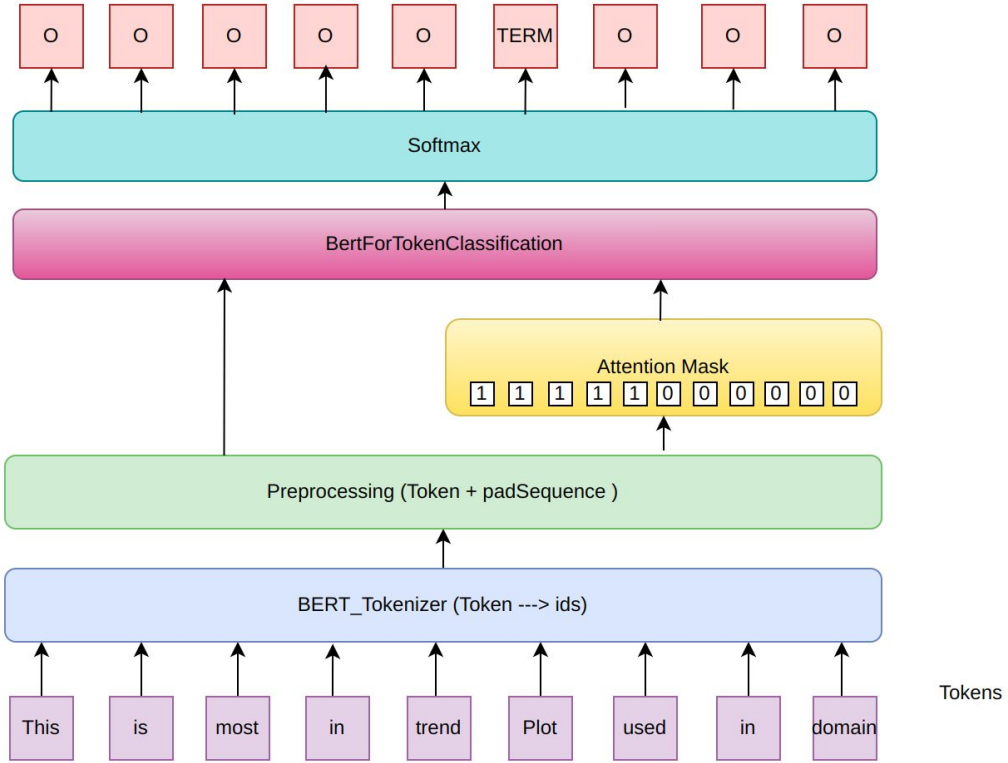
1. BERT (Bidirectional Encoder Representations from Transformer)
2. CRF (Conditional Random Fields)
3. ELMo + Bi-LSTM (Bidirectional Long-Short Term Memory)

BERT(Bidirectional Encoder Representation for Transformer)

- This model was pre-trained on unsupervised Wikipedia and Bookcorpus datasets using language modeling.

- Fine-tuning BERT for token classification involves training the pre-trained BERT model on a specific token classification task, such as recognition or tagging task using a labeled dataset.
- The input to the model consists of *input_ids* and *attention_mask*. *input_ids* is a tensor of token ids for each input sequence, while *attention_mask* is a tensor of the same shape as input_ids that indicates which tokens should be attended to by the model (1 for tokens to attend to, 0 for tokens to ignore).

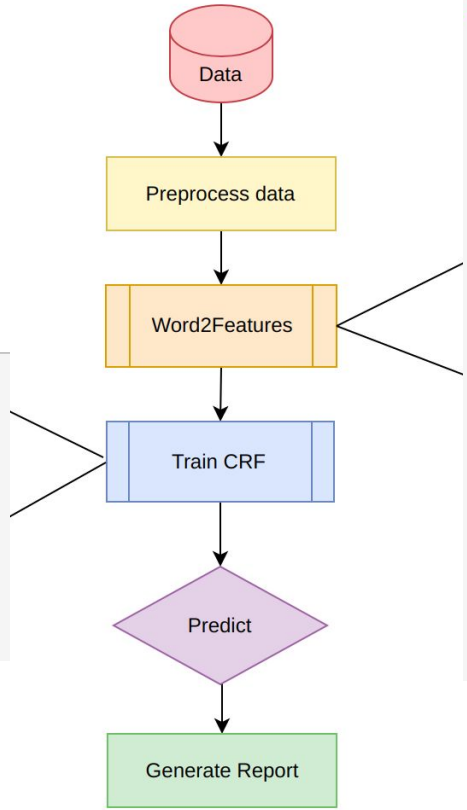
- BERTForTokenClassification assigns a label to individual tokens in a sentence.



Pure CRF

```python
# Train the CRF model
crf = sklearn_crfsuite.CRF(
    algorithm='lbfgs',
    c1=0.1,
    c2=0.1,
    max_iterations=100,
    all_possible_transitions=True
)
```

Fig: Model definition with hyperparameters




```python
# Define feature functions
def word2features(sent, i):
    word = sent[i]
    features = {
        'bias': 1.0,
        'word.lower()': word.lower(),
        'word.isupper()': word.isupper(),
        'word.istitle()': word.istitle(),
        'word.isdigit()': word.isdigit()
    }
    if i > 0:
        prev_word = sent[i-1]
        features.update({
            '-1:word.lower()': prev_word.lower(),
            '-1:word.istitle()': prev_word.istitle(),
            '-1:word.isupper()': prev_word.isupper()
        })
    else:
        features['BOS'] = True

    if i < len(sent)-1:
        next_word = sent[i+1]
        features.update({
            '+1:word.lower()': next_word.lower(),
            '+1:word.istitle()': next_word.istitle(),
            '+1:word.isupper()': next_word.isupper()
        })
    else:
        features['EOS'] = True

    return features
```

Fig: Function to generate features for CRF model

Bi-LSTMs

Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies.

A sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and the other in a backwards direction.

BiLSTMs effectively increase the amount of information available to the network, improving the context available to the algorithm

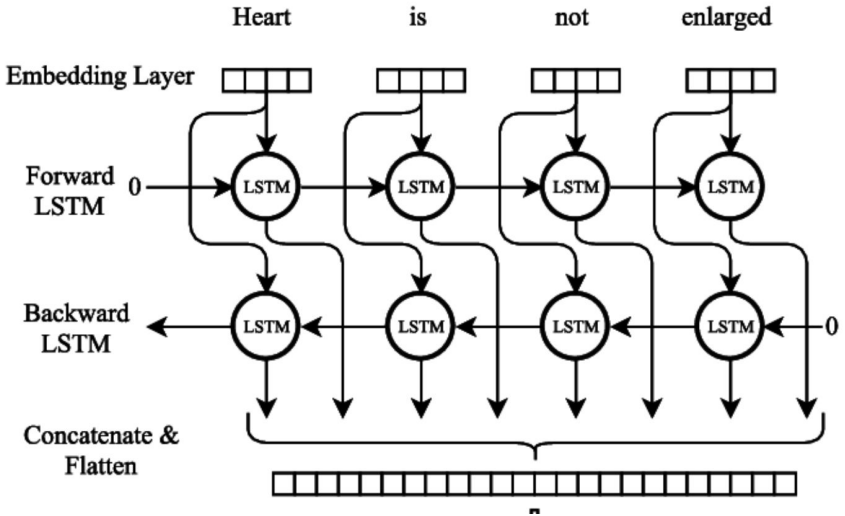
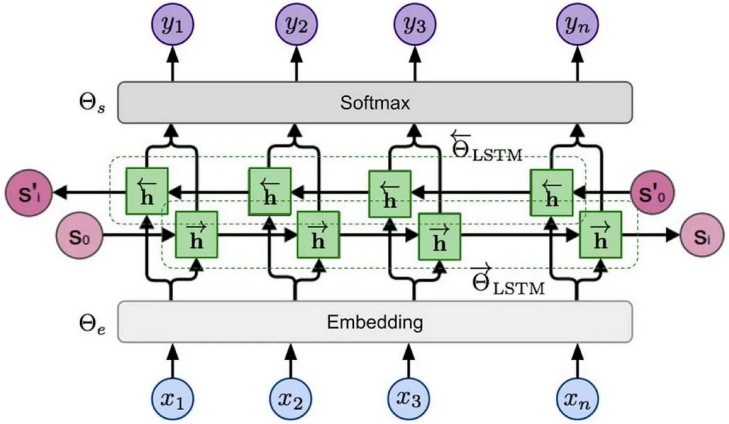


Image Source: Modelling Radiological Language with Bidirectional Long Short-Term Memory Networks, Cornegruta et al

ELMo sentence embedding

- ELMo which stands for Embedding from Language Model, uses bi-directional deep LSTM network for producing vector representation.
- ELMo considers words within which context they have been used rather than creating dictionary of words with its vector form.
- The model can represent the unknown or out of vocabulary words into vector form as it is character based.
- ELMo representations are:
  - Contextual: The representation for each word depends on the entire context in which it is used.
  - Deep: The word representations combine all layers of a deep pre-trained neural network.
  - Character based: ELMo representations are purely character based, allowing the network to use morphological clues to form robust representations for out-of-vocabulary tokens unseen in training.
- The model consists of 2-layer of bi-directional LSTM and between both the layers, there is a residual connection which without any non-linear activation function allows gradient to traverse in the network. And this allows deep models to be trained effectively.



Architectures of ELMo

Ref: https://medium.com/@kashyapkathrani/all-about-embeddings-829c8ff0bf5b

ELMo + Bi-LSTM

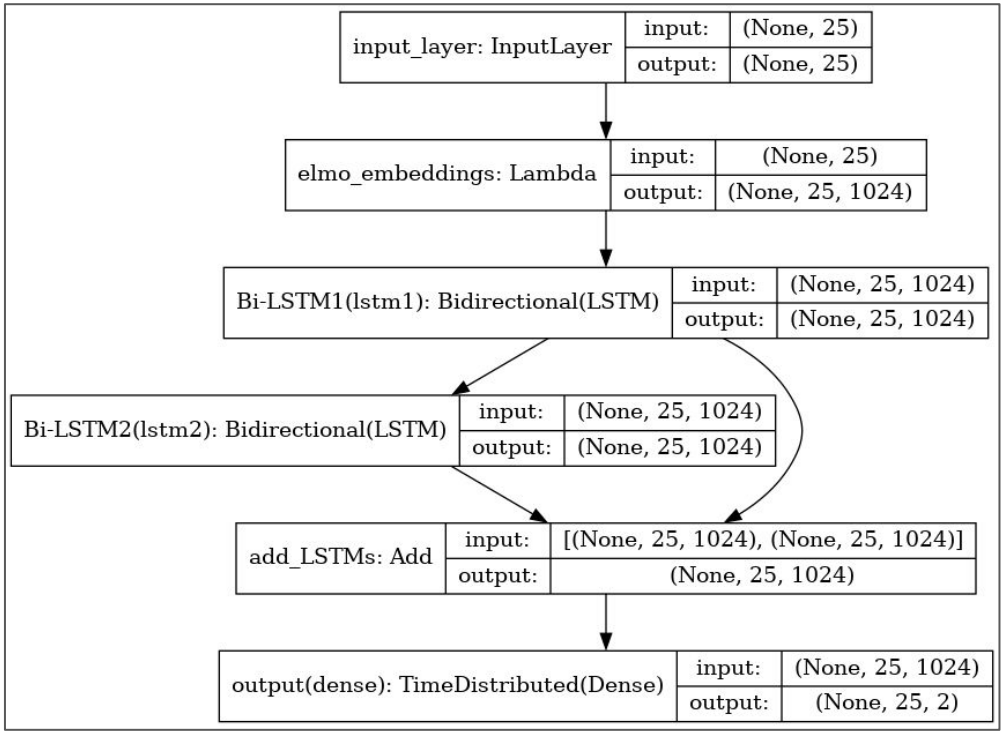
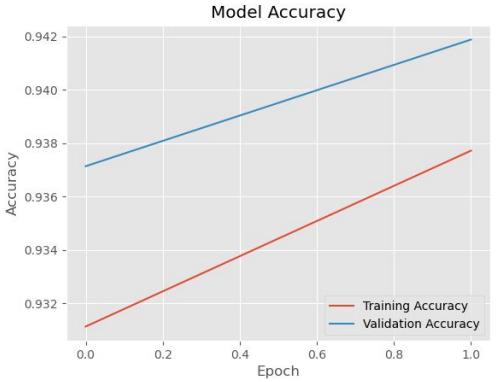


Fig: model description



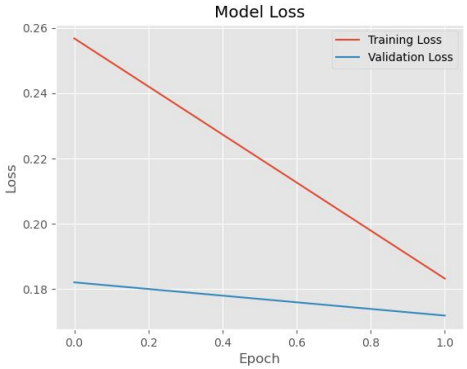


Fig: Training accuracy and loss graph

Hybrid model:

BiLSTM + CRF

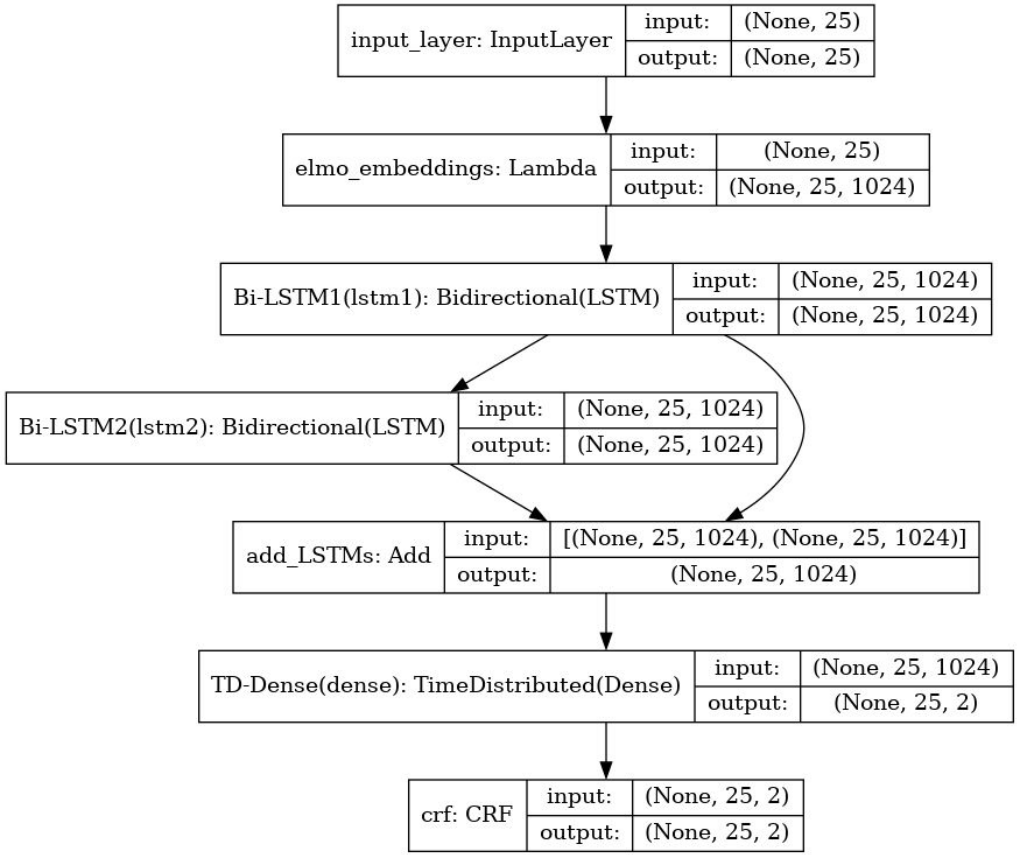
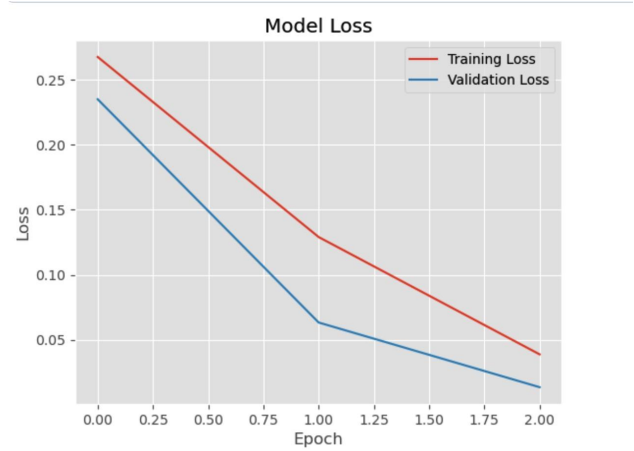


Fig: model description



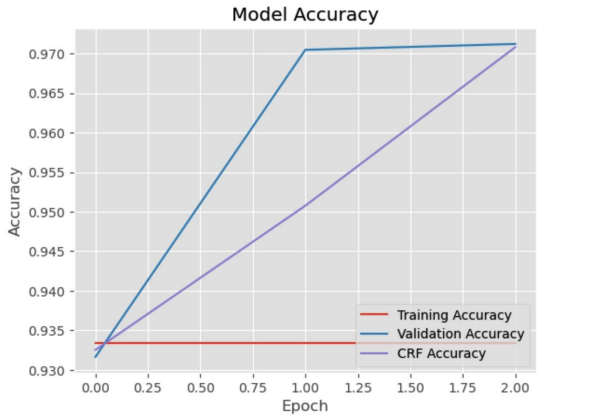


Fig: Training accuracy and loss graph

Improvement in the hybrid model over baseline

- Combined features: The model combines the Elmo embeddings with Bi-LSTM layers, which allows the model to capture both contextualized word embeddings from Elmo and sequential dependencies through the Bi-LSTM layers.

- Residual connection: The model includes a residual connection between the two Bi-LSTM layers using the add() function. This allows the model to propagate gradients and information from the first Bi-LSTM layer to the second Bi-LSTM layer, which may help in mitigating the vanishing or exploding gradient problem and improve the overall training stability and performance.

- CRF layer: The model also includes a CRF layer after the TimeDistributed Dense layer. The CRF layer is a popular choice for sequence labeling tasks as it can model the label transitions and dependencies explicitly, which may help in capturing complex patterns in the output sequence.

- Regularization techniques: The model includes dropout in the Bi-LSTM layers, which can help in regularizing the model and prevent overfitting.

Evaluation metrics

Precision : *proportion of true positive of all positive prediction.*
Recall : *proportion of true positive of all actual positives.*
F1 score : *harmonic mean of precision and recall*
Accuracy : *evaluation score (eval dataset) and test score(comparing the kaggle score)*

	precision	recall	f1-score	support
0	0.98	0.99	0.98	49686
1	0.80	0.78	0.79	3664
accuracy			0.97	53350
macro avg	0.89	0.88	0.89	53350
weighted avg	0.97	0.97	0.97	53350

Comparison of the models

bilstm_crf_output_predictions.csv Complete (after deadline) · 8m ago · bi-lstm-crf model	BiLSTM + ELMO + CRF	0.95994	0.96292	☐
bilstm_output_predictions.csv Complete (after deadline) · 1m ago · bi-lstm model	BiLSTM + ELMO	0.95858	0.96103	☐
crf_output_predictions.csv Complete (after deadline) · now · pure crf model	CRF	0.86881	0.88099	☐
bert_large_uncased.csv Complete (after deadline) · 1h ago	BERTForTokenClassificatio(bert-large-uncased)	0.85635	0.87201	☐
bert_base_uncased.csv Complete (after deadline) · 4h ago	BERTForTokenClassificatio(bert-base-uncased)	0.85475	0.86422	☐
bert_base_cased.csv Complete (after deadline) · 4h ago	BERTForTokenClassificatio(bert-base-cased)	0.84702	0.86257	☐
Roberta_base.csv Complete (after deadline) · 3h ago	BERTForTokenClassificatio(bert-large-cased)	0.49295	0.49657	☐
	Roberta-base			

Where model fails?

- Pre-trained language model are trained on generalized dataset, so they fails at domain-specific terminology that the model has not encountered before.
- The model sometimes fails due to ambiguity in the training dataset.

cs/doc_1867/5/1	feature	O
cs/doc_1867/5/2	map	TERM
cs/doc_1867/5/2	.	O
cs/doc_1867/6/0	As	O
cs/doc_1867/6/1	shown	O
cs/doc_1867/6/2	in	O
cs/doc_1867/6/3	Fig.	O
cs/doc_1867/6/4	~	O
cs/doc_1867/6/5	,	O
cs/doc_1867/6/6	the	O
cs/doc_1867/6/7	feature	O
cs/doc_1867/6/8	map	O

- Model might struggle to learn the label of most rare tokens.
- Model may encounter new terms during the testing phase, and find it difficult to classify it.
- Accuracy also gets affected due to Null entries in the dataset.(null entries should be replaced with a substitute like space character)

References

1. Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, & Luke Zettlemoyer. (2018). Deep contextualized word representations.
2. Zhiheng Huang, Wei Xu, & Kai Yu (2015). Bidirectional LSTM-CRF Models for Sequence Tagging. CoRR, abs/1508.01991.
3. Jacob Devlin, Ming-Wei Chang, Kenton Lee, & Kristina Toutanova (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. CoRR, abs/1810.04805.