MAGNET: Multi-Label Text Classification using Attention-based Graph Neural Network

Team 32

‘The Matrix’

Members

- Santanu Biswas 2022201031
- Aman Motwani 2022201077
- Ayush Lakshakar 2022201051

The Problem

01

Paper

Multi-Label Text Classification using Attention-based Graph Neural Network

Ankit Pal, Muru Selvakumar and Malaikannan Sankarasubbu

02

Problem Statement

Multi-label text classification assigns zero or more labels to a text document without considering dependency among the labels.

Scope :

- Explore the datasets available with some preliminary data analysis.
- Implement standard techniques for multi label classification on datasets and find desired results.
- Implement Graph Attention Network
- Experimentation with Hyperparameters like number of heads and embedding methods etc.
- Analysis of the experiments done.

Datasets

01

Toxic Comment

Train - 27384
Test - 31915
Dev - 16383
Labels - 7

02

Reuters-21578

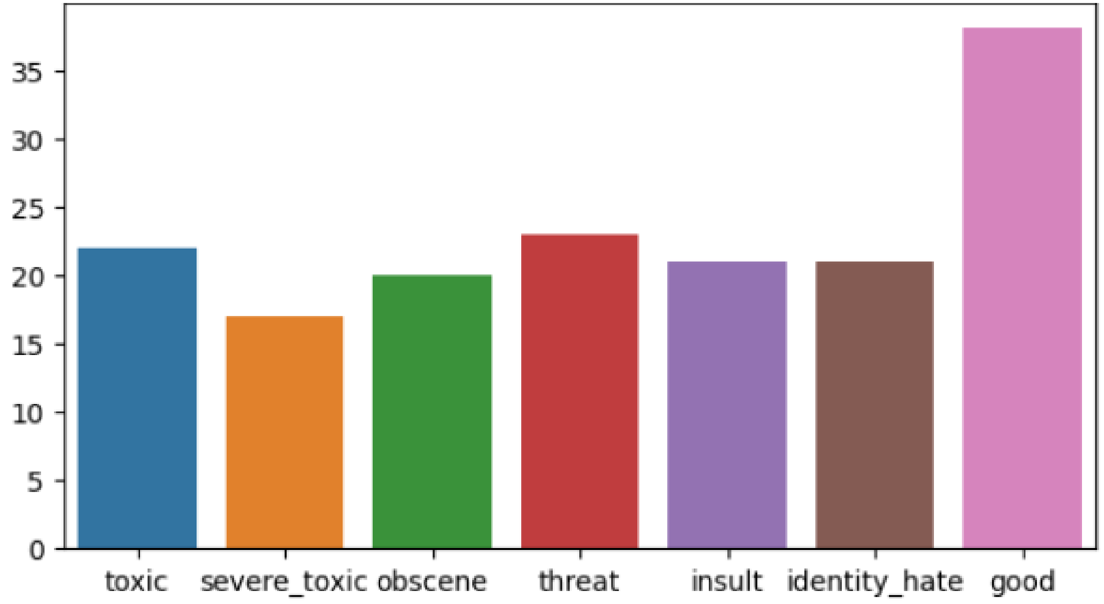
Train - 6215
Test - 3019
Dev - 1554
Labels - 90

03

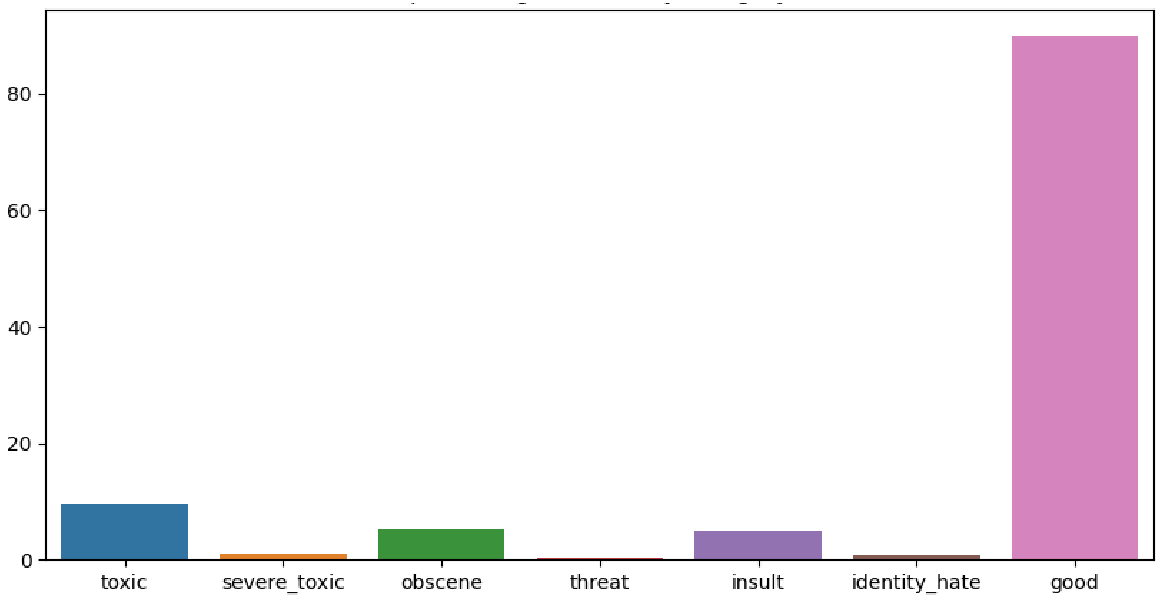
RCV1-V2

Train - 611354
Test - 160883
Dev - 32177
Labels - 103

EDA – Toxic comment



Median text length



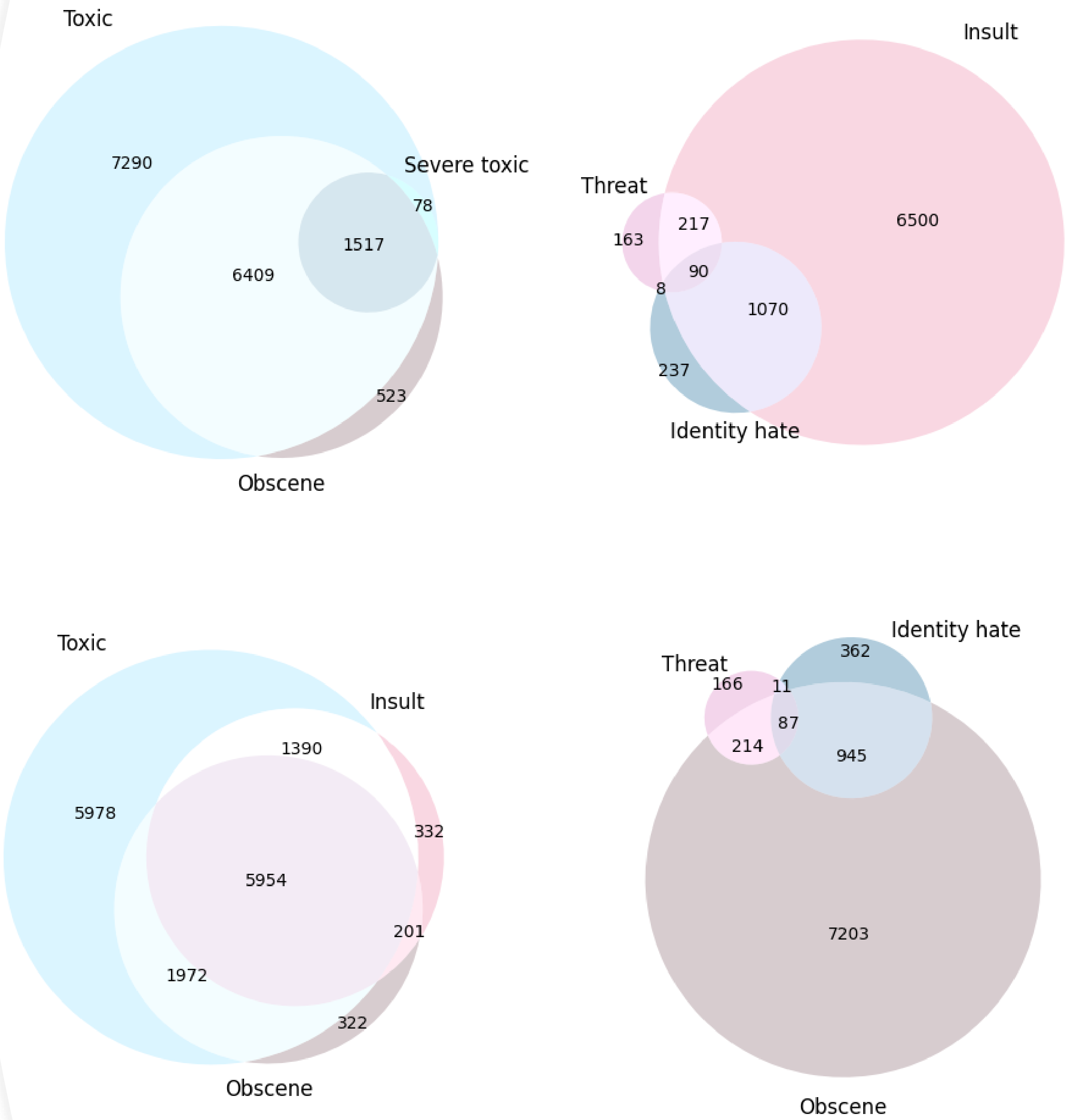
Percentage records by category

EDA

Correlation between toxic categories

EDA

Correlation between toxic categories



Evaluation Metric

- F1 micro score is the harmonic mean of precision and recall calculated globally across all labels.

$$F1-Score_{micro} = \frac{\sum_{j=1}^{L} 2tp_j}{\sum_{j=1}^{L} (2tp_j + fp_j + fn_j)}$$

$$Precision_{micro} = \frac{\sum_{j=1}^{L} tp_j}{\sum_{j=1}^{L} tp_j + fp_j}$$

$$Recall_{micro} = \frac{\sum_{j=1}^{L} tp_j}{\sum_{j=1}^{L} tp_j + fn_j}$$

Pre-processing

Tokenization and cleaning

Word2index mapping

Creating
Data Loader

Baseline Models

Model	Micro F1 score
OneVsRest	0.70358
Binary Relevance	0.79971
Classifier Chains	0.72687
Label Powerset	0.68389

OneVsRest

Binary Relevance

Classifier Chains

Label Powerset

Models

- MLP
- Bi-LSTM
- BERT
- MAGNET

MLP Model

Embedding layer

Maps input sentences to its dense vector representation using pre-trained word embeddings

Fully connected layer

Linear transformation to the sentence embedding to produce a hidden representation

Output layer

Provide a probability distribution over the possible classes using a sigmoid activation function

BiLSTM Model

Embedding layer

Maps input sentences to its dense vector representation using pre-trained word embeddings

BiLSTM layer

Single BiLSTM layer that produces a hidden representation of the embeddings

Output layer

Linear transformation to the sentence embedding to provide a probability distribution over the possible classes using a sigmoid activation function

BERT Model

BERT Embedding layer	A pretrained BERT model is used for generating embeddings for the input text
Classifier layer	Single fully connected layer for classification task
Output layer	Provide a probability distribution over the possible classes using a sigmoid activation function

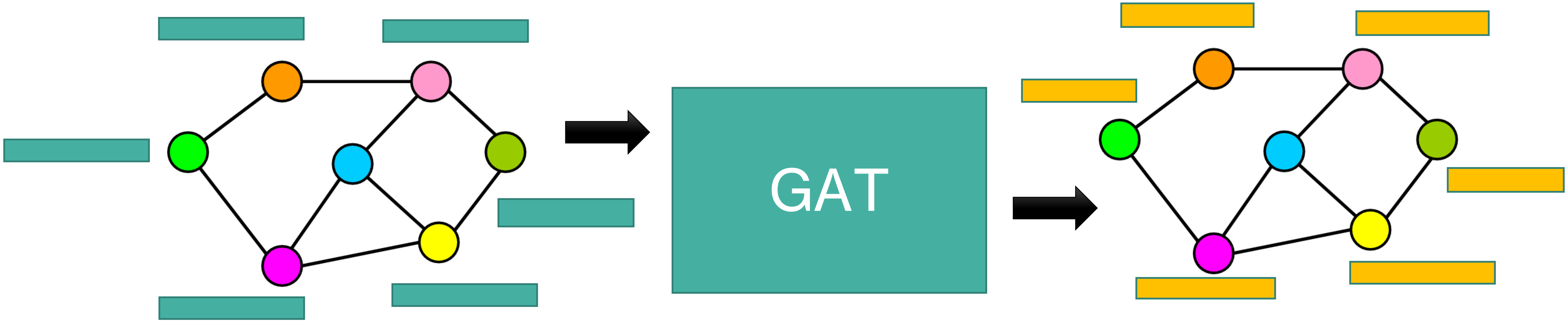
MAGNET Model

- **Graph Representation:**

Node feature description M {n x d} and adjacency matrix A {n x n}

GAT (Graph Attention Network) takes node features and adjacency as input. Model will learn the adjacency matrix.

Model correlation among labels. Adjacency Matrix and attention weight represents correlation.



Node Update Mechanism

- For any node **i** in **(L+1)** th layer (***without attention***) – $$\mathbf{H}^{(\ell+1)} = \sigma\left(\mathbf{A}\mathbf{H}^{\ell}\mathbf{W}^{\ell}\right)$$

where

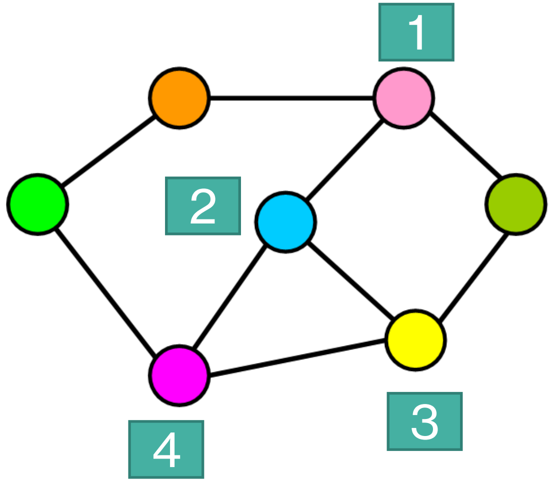
A : Adjacency Matrix (***so that only neighbour will contribute in update mechanism***) ,

W: Weight Matrix

H: Node feature matrix in **Lth** layer.

(***without attention***)

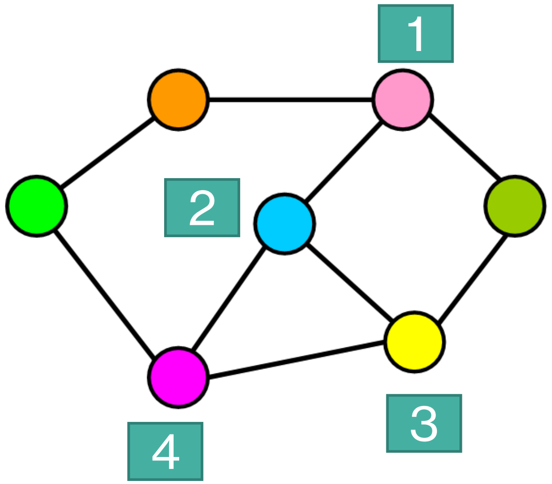
$$\mathbf{H}_2^{(\ell+1)} = \sigma\Big(\mathbf{H}_2^{(\ell)}\mathbf{W}^{(\ell)} + \mathbf{H}_1^{(\ell)}\mathbf{W}^{(\ell)} + \mathbf{H}_3^{(\ell)}\mathbf{W}^{(\ell)} + \mathbf{H}_4^{(\ell)}\mathbf{W}^{(\ell)}\Big)$$



Node Update Mechanism Continued...

(***with attention***)

$$\mathbf{H}_2^{(\ell+1)} = \mathbf{ReLU}\Big(\alpha_{22}^{(\ell)}\mathbf{H}_2^{(\ell)}\mathbf{W}^{(\ell)} + \alpha_{21}^{(\ell)}\mathbf{H}_1^{(\ell)}\mathbf{W}^{(\ell)} + \alpha_{23}^{(\ell)}\mathbf{H}_3^{(\ell)}\mathbf{W}^{(\ell)} + \alpha_{24}^{(\ell)}\mathbf{H}_4^{(\ell)}\mathbf{W}^{(\ell)}\Big)$$



Where $\alpha_{ij}^{(\ell)}$ is the attention coefficient : importance of **j**th node in updating **i**th node with

$$\alpha_{ij}^{(\ell)} = f\left(\mathbf{H}_i^{(\ell)}\mathbf{W}^{(\ell)}, \mathbf{H}_j^{(\ell)}\mathbf{W}^{(\ell)}\right)$$

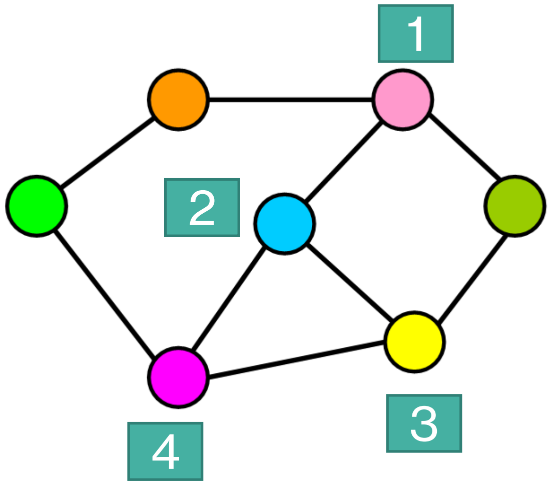
where

F can be any function. It can be Neural Network also.

$\mathbf{H}_i^{(\ell)}\mathbf{W}^{(\ell)}, \mathbf{H}_j^{(\ell)}\mathbf{W}^{(\ell)}$ are the transformed node features embeddings.

Node Update Mechanism Continued...

For Multiple Attention Head :- $\mathbf{H}_{\mathbf{i}}^{(\ell+1)} = Tanh\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N(i)}\alpha_{ij,k}^{\ell}H_j^{\ell}W^{\ell}\right)$



Where

K is the total number of heads

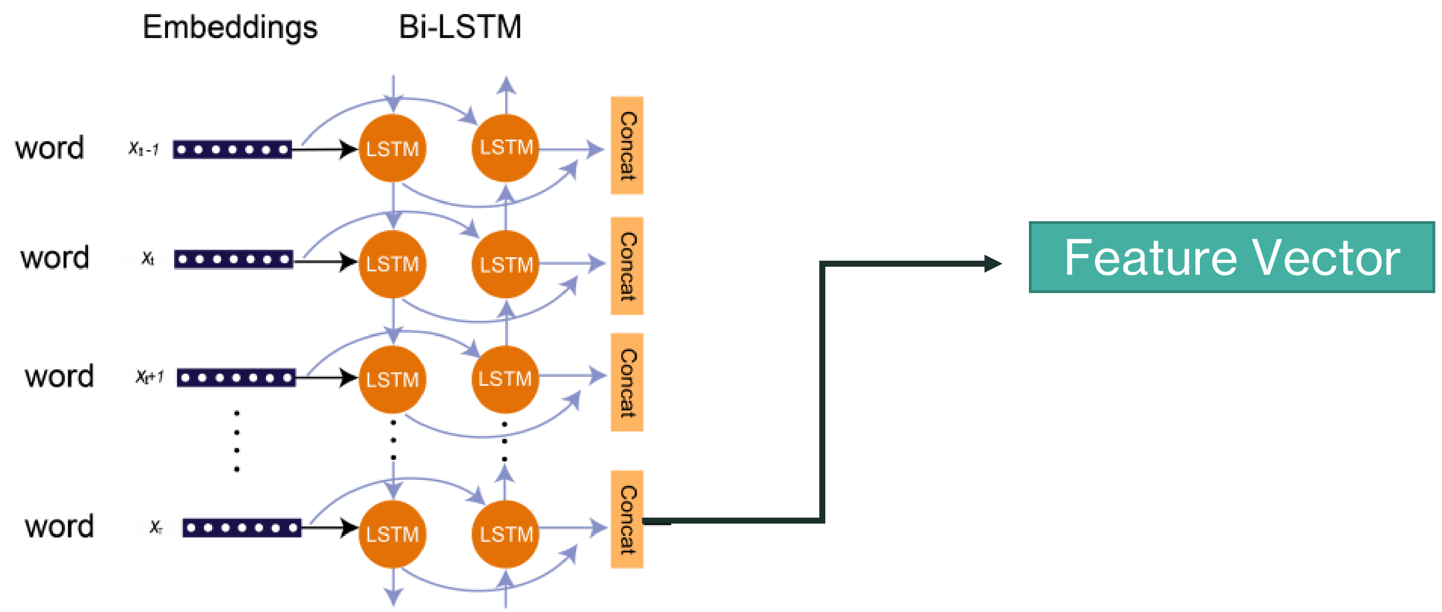
N(i) is the neighbours of **i**th node

H is the node feature matrix

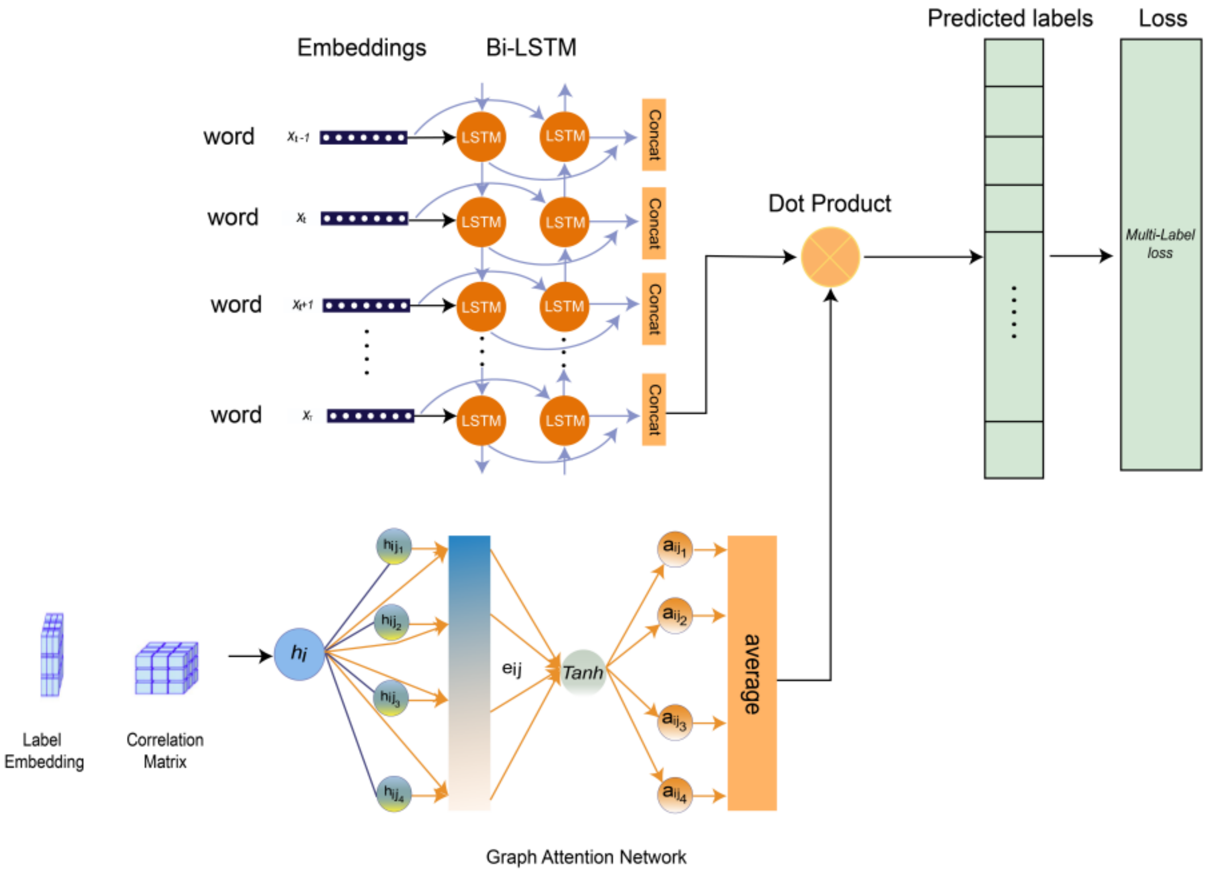
L is the layer no, for **L** = 1, **H** = Adjacency Matrix

Feature Vector Generation

- Bidirectional LSTM is used for feature vector generation of a sentence.
- Forward and Backward pass will capture both forward and previous context of a sentence
- Feature vector is the concatenation of output hidden states of forward and backward pass.



Overall Solution



- Loss will be calculated on dot product of Feature Vector from Bi-LSTM and final node feature embedding from last layer of GAT.

Our Implementation

- Feature Vector is generated using **Bi-LSTM**.
- For attention coefficient –

$$\alpha_{ij} = \frac{\exp\left(LeakyReLU\left(\overrightarrow{w_a^T}\,[Wh_i \| Wh_j]\right)\right)}{\sum_{k\epsilon N(i)} \exp\left(LeakyReLU\left(\overrightarrow{w_a^T}\,[Wh_i \| Wh_j]\right)\right)}$$

Where

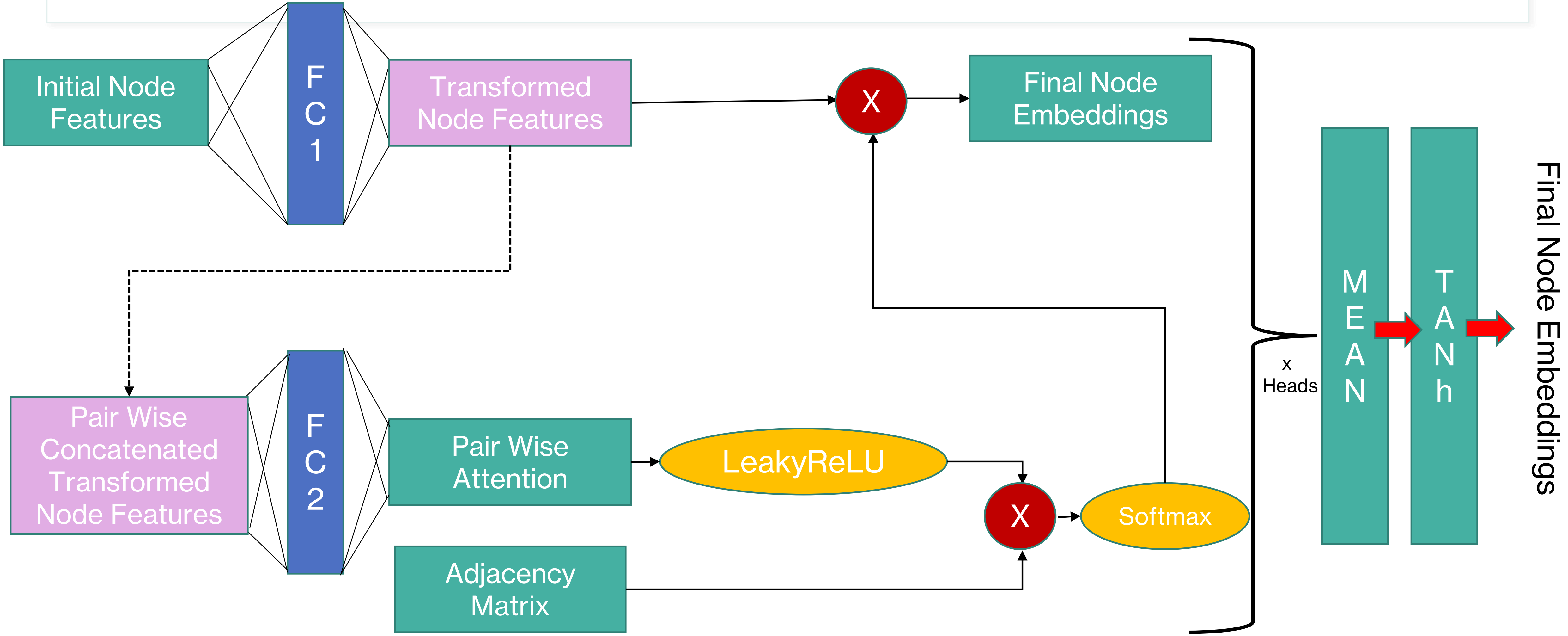
Whi & **Whj** are transformed node features, **W** – Learnable parameter

Wa – Learnable parameter for calculating attention

- For multiple heads, **mean of node features generated from all heads is taken**, followed by **Tanh** activation function over mean.
- Loss is calculated over **dot product** of final Node embeddings and Feature vector.

Our Implementation

- For Node feature updation



Sample Input Output Inference -

MAGNET on Reuters-21578 dataset

Sentence	Output
The stock market rallied after the Fed announced a new interest rate policy.	Interest, money-fx
The pharmaceutical company received approval for a new drug to treat a rare disease.	acq
The energy sector experienced a surge in demand for renewable energy sources.	crude
The U.S. Federal Reserve is expected to raise interest rates next month, according to analysts.	interest

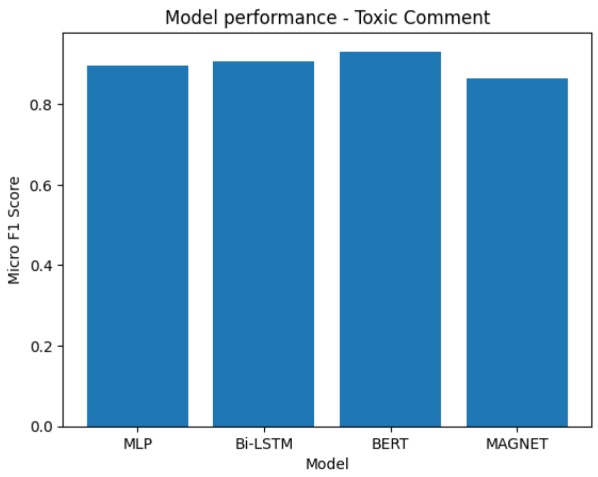
Results

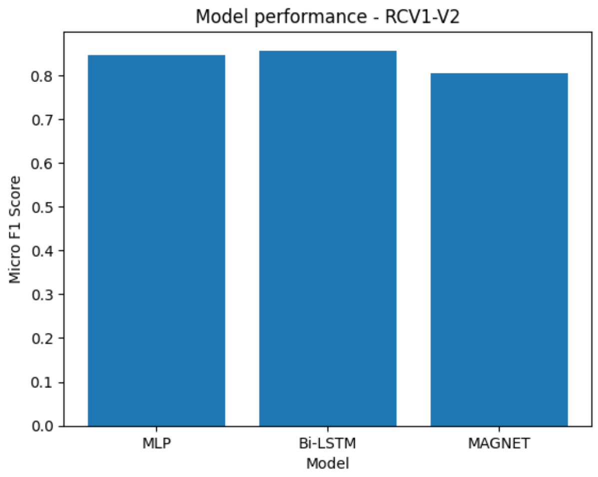
Comparisons of **Micro F1-score** for various models on three benchmark datasets

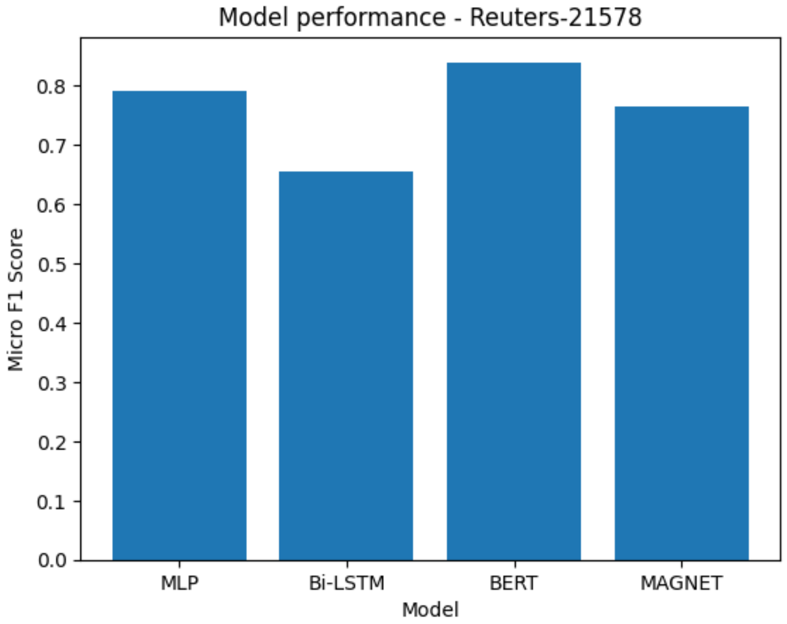
Model	**Toxic Comment**	**Reuters-21578**	**RCV1-V2**
MLP	0.89656	0.79069	0.84490
Bi-LSTM	0.90575	0.65613	0.85611
BERT	0.93092	0.83994	NA
MAGNET	0.86315	0.76419	0.80379

Results

Comparisons of **Micro F1-score** for various models on three benchmark datasets

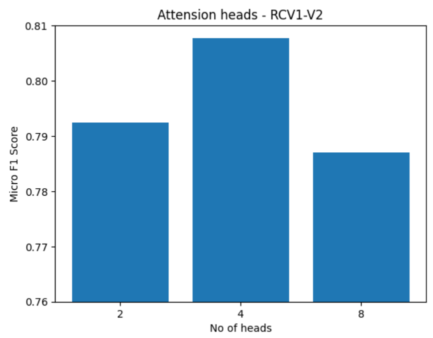


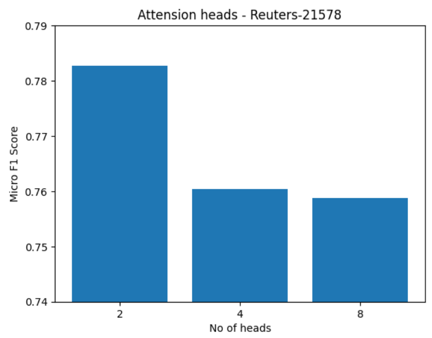


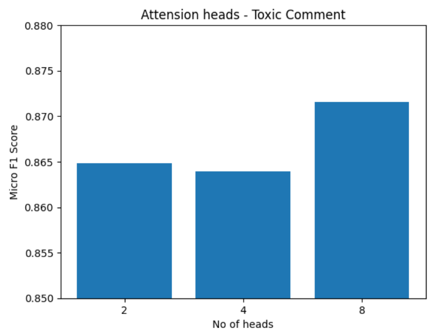


Experiments

Comparisons of **Micro F1-score** for various attention heads on three benchmark datasets



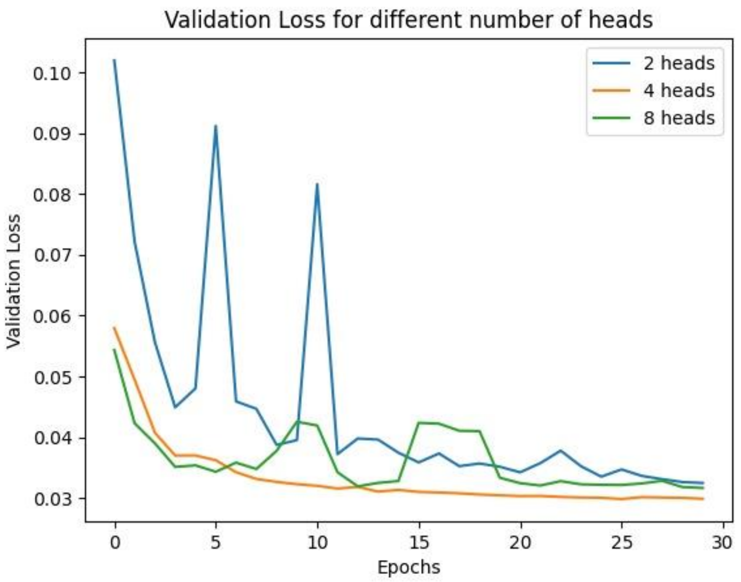




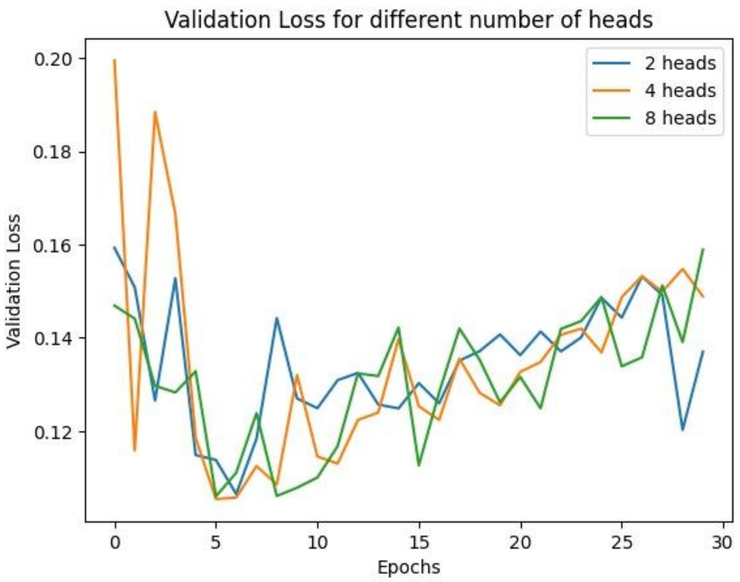
No of Attention Heads	**Toxic Comment**	**Reuters-21578**	**RCV1-V2**
2	0.86486	0.78276	0.79248
4	0.86391	0.76044	0.80767
8	0.87157	0.75870	0.786971

Experiments

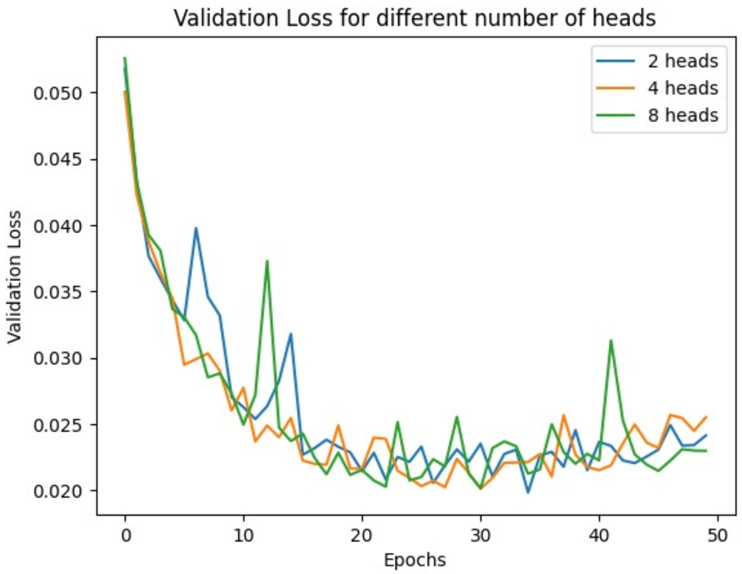
Comparisons of validation loss for various attention heads on three benchmark datasets



Toxic Comment



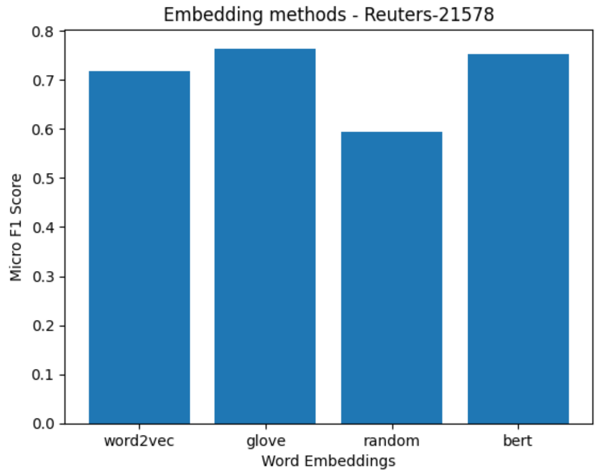
Reuters-21578

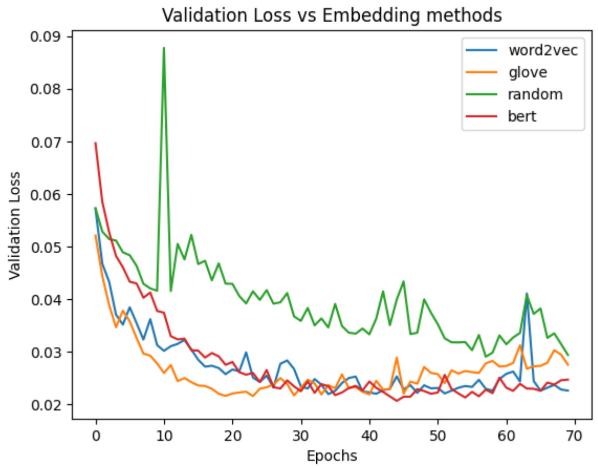


RCV1-V2

Experiments

Comparisons of **Micro F1-score** for various word embeddings for Reuters-21578 dataset





Word embeddings	Micro F1-score
word2vec	0.71873
glove	0.76419
random	0.59519
bert	0.75345

Conclusion:

- The empirical results indicate that the baseline models do not perform satisfactorily in multi-label classification and are unable to capture dependencies.
- MLP and LSTM models are observed to have a better understanding of the task than the baseline models due to their ability to extract complex features from textual data.
- Based on our experiments comparing different models for Multilabel Text Classification, we observed that the basic BERT model outperformed the more complex MAGNET model.
- On an average, across all datasets, 4 heads in the multihead graph attention-based neural network yields relatively better results.
- Comparing different word embeddings for generating feature vectors, both BERT and GloVe showed better performance than other options.

REFERENCES:

- [1] Ashish Vaswani et. al.: Attention Is All You Need
- [2] Devlin et. al.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2018)
- [3] Petar Veliˇckovi´c et. al.: GRAPH ATTENTION NETWORKS
- [4] Jie Zhou et al.: Graph neural networks: A review of methods and applications

THANK YOU!

Contribution

01

Santanu Biswas

- BERT
- Experimenting MAGNET

02

Aman Motwani

- Baseline models
- Implementing MAGNET

03

Ayush Lakshakar

- MLP, BiLSTM
- Implementing MAGNET