Clickbait Detection

YOU WILL NEVER BELIEVE HOW WE DETECT CLICKBAITS...

Similarity-Aware Deep Attentive Model for Clickbait Detection

Final presentation

POINTS OF DISCUSSION

Introduction

Conceptual Discussion

Dataset Description

Methodology

Results and Analysis

Limitations and Challenges

Conclusion

Introduction

Clickbaits lead to articles that are either misleading or non-informative, making their detection essential. Most recent work handles the clickbait detection problem with deep learning approaches to extract features from the metadata of the content. In this work, we explore the relationship between misleading titles and the target content as an important clue for clickbait detection. **We propose a deep similarity-aware attentive model to capture and represent such similarities with better expressiveness. We evaluate our model on two benchmark datasets and compare it with other baseline methods.**

Conceptual discussion

1. **Recurrent Neural Networks**
2. **Long Short-Term Memory**
3. **Gated Recurrent Unit**
4. **Bidirectional GRU**
5. **Deep Semantic Similarity Model**

Deep Semantic Similarity Model

DSSM is a latent semantic model. It uses deep neural networks to learn the latent representations. DSSM first maps the input features x to the latent semantic space l,

$$\begin{aligned} layer_1 &= W_1 x \\ layer_i &= f(W_i layer_{i-1} + b_i), i = 2, \ldots, N-1 \\ l &= f(W_N layer_{N-1} + b_N) \end{aligned}$$

where layer_i is the i_th intermediate hidden layer, W_i is the i_th weight matrix, b_i is the i_th bias matrix, and f is the activation function, e.g., sigmoid function. From this, the semantic relevance score between say query Q and document D is measured as

$$R(Q, D) = cosine(l_Q, l_D) = \frac{l_Q^T l_D}{\| l_Q \| \| l_D \|}$$

Learning the DSSM is equivalent to maximising this similarity score for matching documents from the entire collection.

Deep Semantic Similarity Model

DSSM is a latent semantic model. It uses deep neural networks to learn the latent representations. DSSM first maps the input features x to the latent semantic space l,

$$\begin{aligned} layer_1 &= W_1 x \\ layer_i &= f(W_i layer_{i-1} + b_i), i = 2, \dots, N-1 \\ l &= f(W_N layer_{N-1} + b_N) \end{aligned}$$

where layer_i is the i_th intermediate hidden layer, W_i is the i_th weight matrix, b_i is the i_th bias matrix, and f is the activation function, e.g., sigmoid function. From this, the semantic relevance score between say query Q and document D is measured as

$$R(Q, D) = cosine(l_Q, l_D) = \frac{l_Q^T l_D}{\| l_Q \| \| l_D \|}$$

Learning the DSSM is equivalent to maximising this similarity score for matching documents from the entire collection.

Data Pre-Processing

Data Cleaning

- Firstly the data is cleaned by removing all the NAN and invalid datatypes.
- Then all the stop words are removed from all the sentences.
- After which stemming is performed to shorten the word length for better performance.

Data Encoding

- Every sentence has been encoded by assigning each word a unique index via Tokenizer and a wordCounter.

Padding

- Finally all the encoded sentences are padded to have equal lengths of 200 words

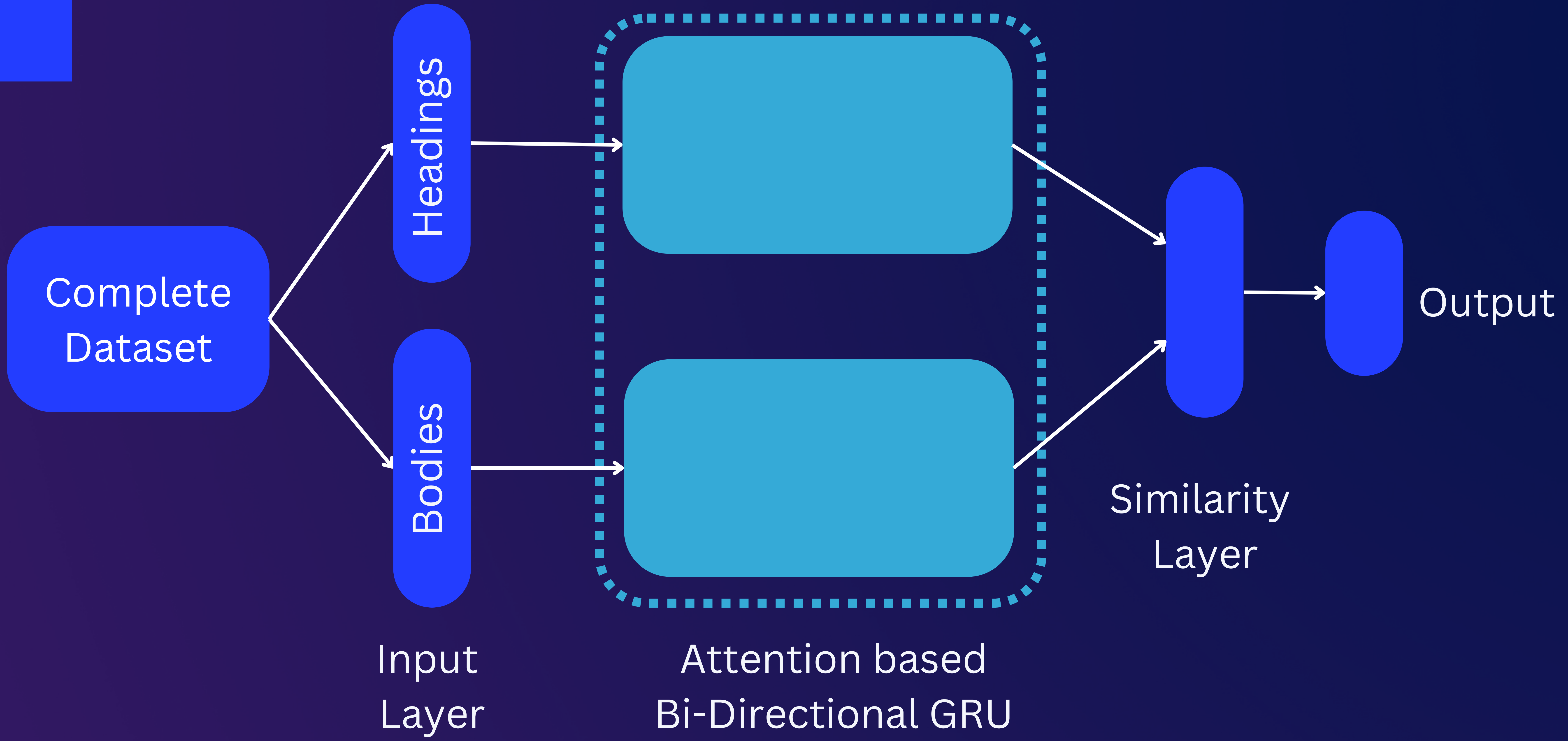
Methodology

Given a set of titles $H=\{h1,h2,...hn\}$ and their bodies $B = \{b1, b2,...,bN \}$, the goal is to predict a label $Y = \{y1, y2,...,yN \}$of these pairs, where $Yi = 1$ if headline i is a clickbait.

The paper implements this task by following the framework of three parts:

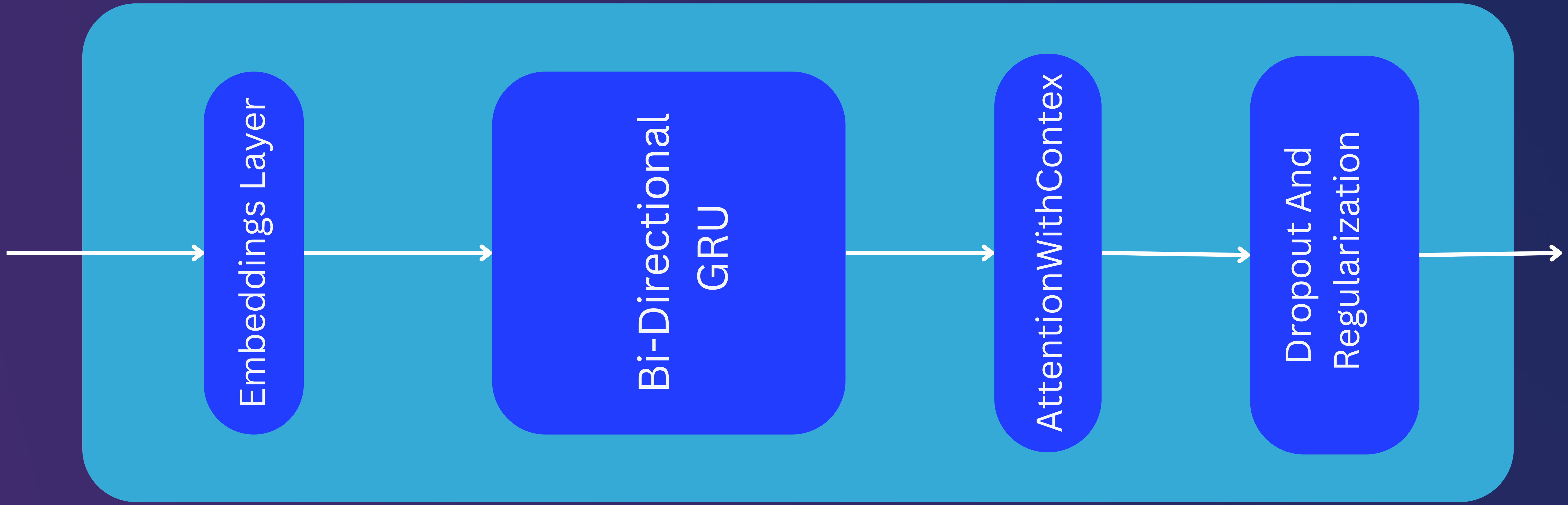
- Learning latent representations,
- Learning the similarities,
- Using the similarity for further predictions.

Complete Model Arhitecture



Bi-Directional Gru

Both of the bi-directional GRU have the following architecture



Used for converting words to vectors

An RNN network which is at the heart of our model

Extract the meaningful features

Done to reduce overfitting

Methodology

DATA PREPROCESSING

The data is preprocessed by removing the stop words and performing stemming using the nltk library.

SEQUENCE ENCODING

The words of the dataset are encoded to single and unique integers and the word sequences are all normalized such that they are of equal size of 200 words.

VECTORIZATION

The vectorization of encoded data is handled by an Embedding layer provided by the Keras API, The Layer is configured to convert every word index to an embedding vector of size 100.

Methodology

LEARNING LATENT REPRESENTATIONS

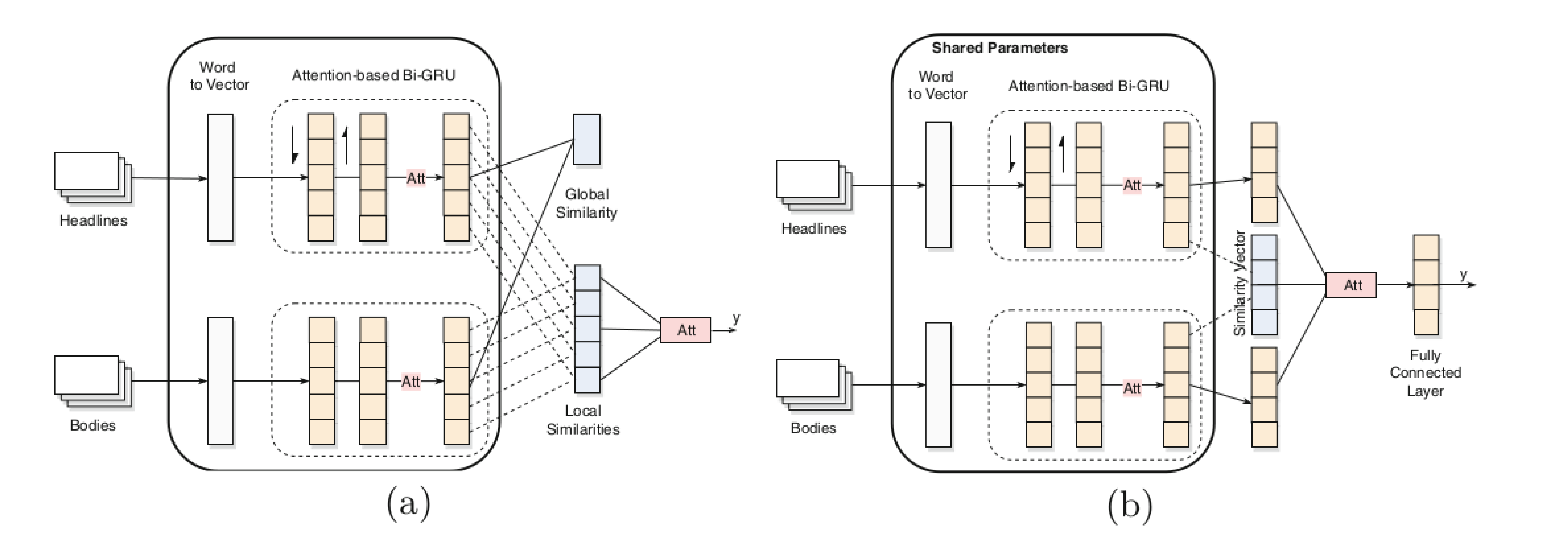
We first preprocess the text and convert it into a vector by using word2vec and then apply the attention-based bidirectional GRU to obtain hidden representations

LEARNING THE SIMILARITIES

We have defined the similarity of two vectors by calculating the cosine similarity between the latent representation of the bodies (LB) and the latent representation of the heading(LH)

LEARN FOR PREDICTION

For training our model, we use the classification method which combines the features with the similarities, for this, we define a concatenation layer and use them to get a combination layer L'', which then is fed into multi-layer perceptron to get the output.


Word
to Vector
Attention-based Bi-GRU
Att
Att
Headlines
Bodies
Global
Similarity
Local
Similarities
Att
y
(a)
Shared Parameters
Word
to Vector
Attention-based Bi-GRU
Att
Att
Headlines
Bodies
Similarity Vector
Att
y
Fully
Connected
Layer
(b)

Dataset Description

CLICKBAIT CHALLENGE

Clickbait Challenge is a benchmark dataset for clickbait detection that released in 2017. The dataset contains over 20,000 labelled pairs of posts for training and validation. A higher score in 0 to 1 stands for the higher probability of a post being clickbait. A post with a mean clickbait score over 0.5 is considered to be clickbait.

FNC DATASET

FNC dataset is from the Fake News Challenge in 2017. The data describe pairs of titles and bodies and are labeled as ‘agree’, ‘disagree’, ‘discuss’ and ‘unrelated’. We regard data with label ‘unrelated’ as clickbait. The dataset contains 49,972 pairs of titles and bodies for training and 25,413 pairs for the testing.

Dataset Description

The Datasets are preprocessed by removing the stop words and lemmetization
We further vectorize the data using word-embedding techniques

CLICKBAIT CHALLENGE

The processed dataset has an average of 10 words in the titles and 50 words in the bodies.

FNC DATASET

The FNC dataset has an average of 8 words in the titles and 200 words in the bodies.

CLICKBAIT CHALLENGE

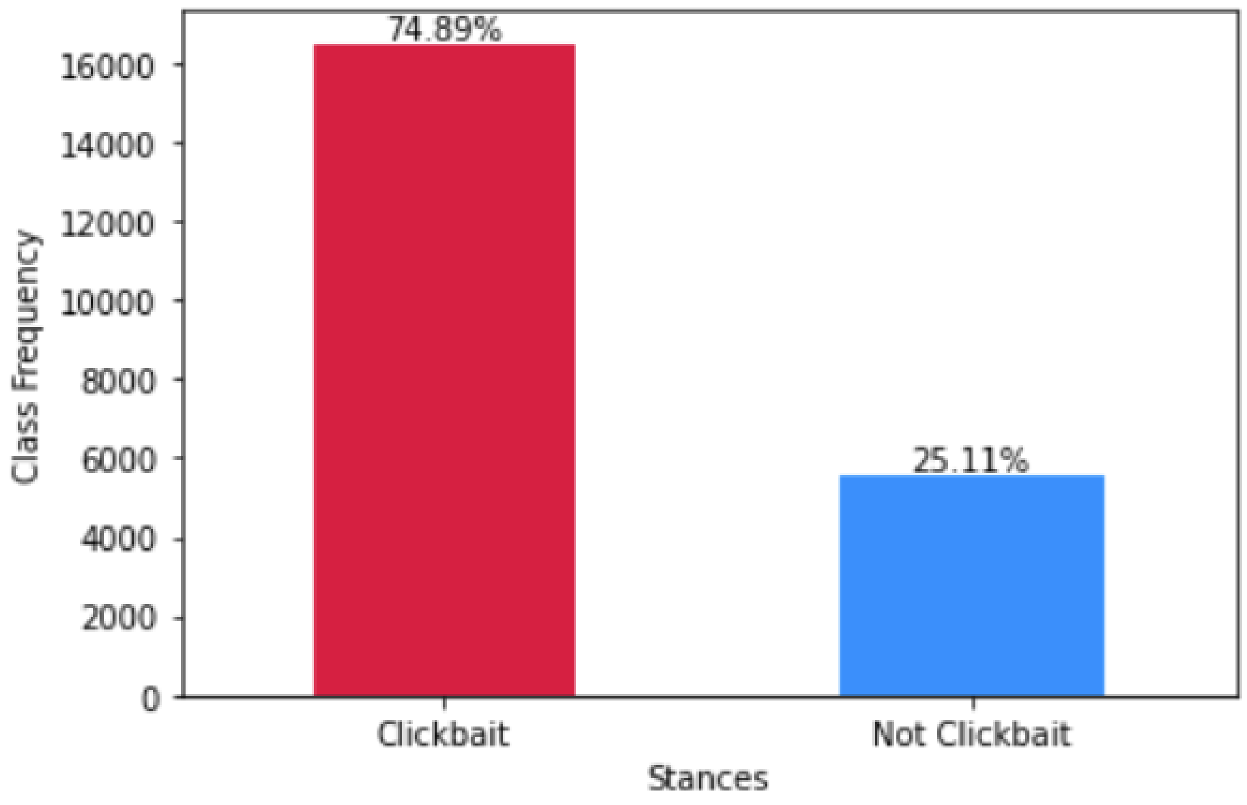


Figure 8: Clickbait Data Class Label Distribution

FNC DATASET

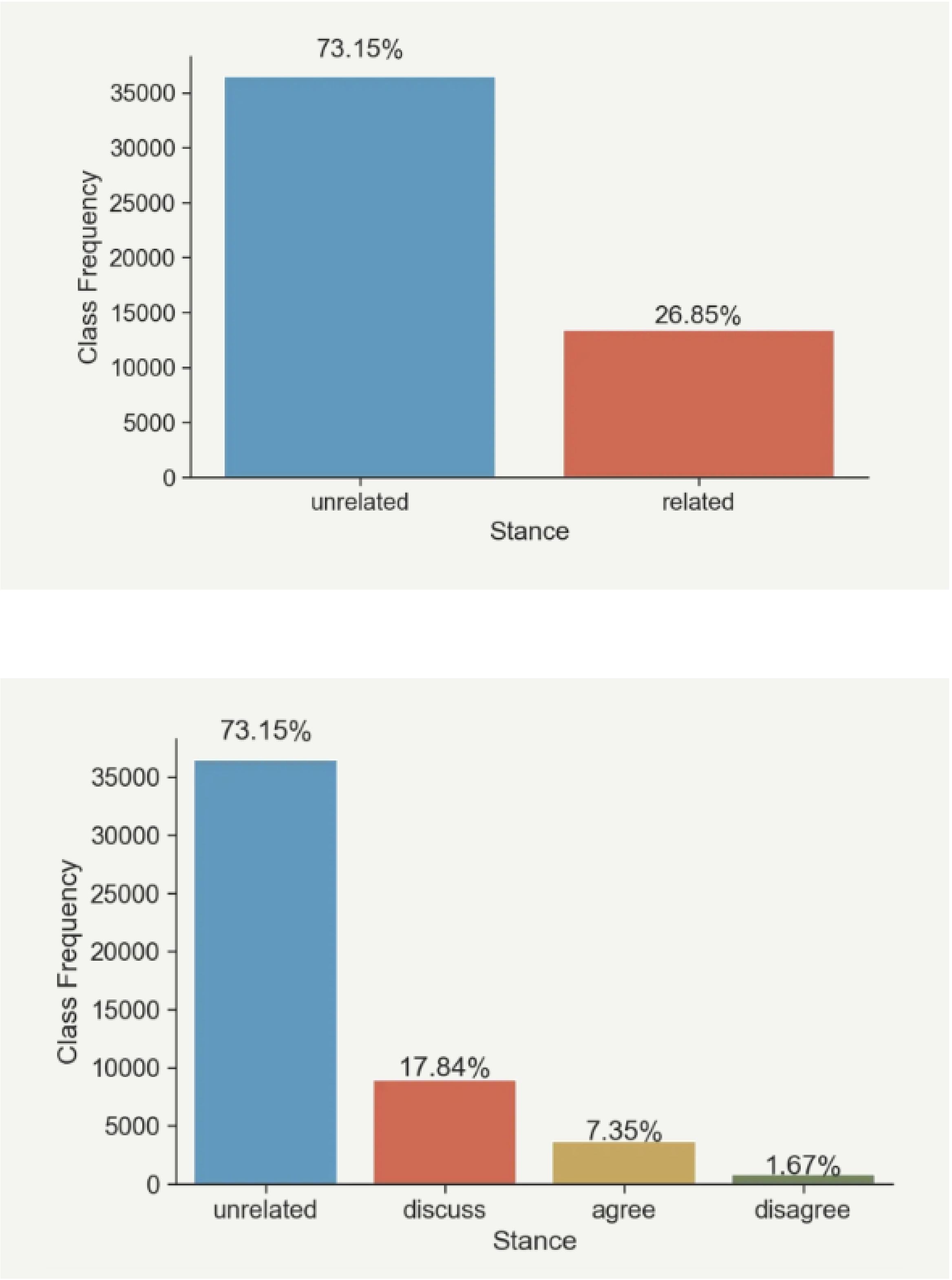
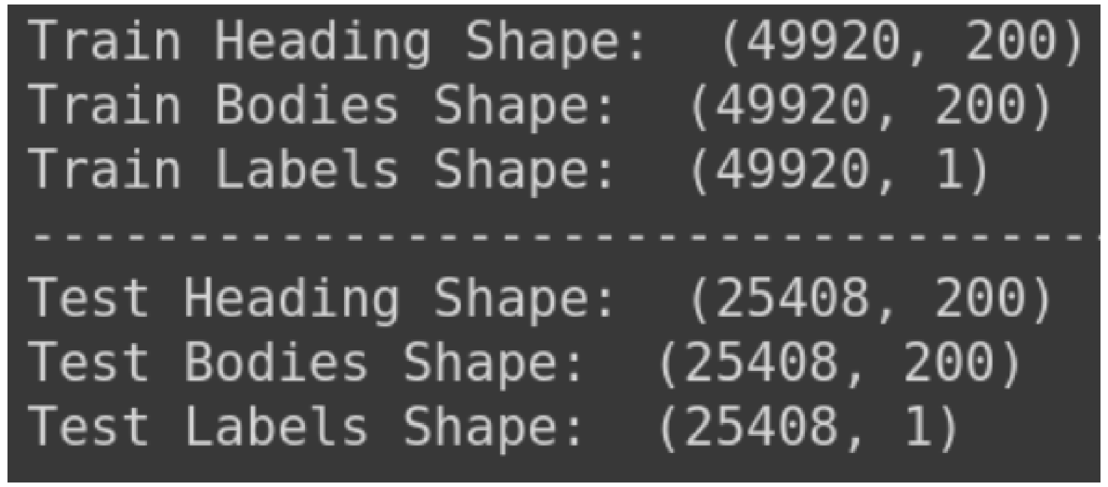


Figure 6: FNC Training Data Class Label Distribution

Results and Analysis

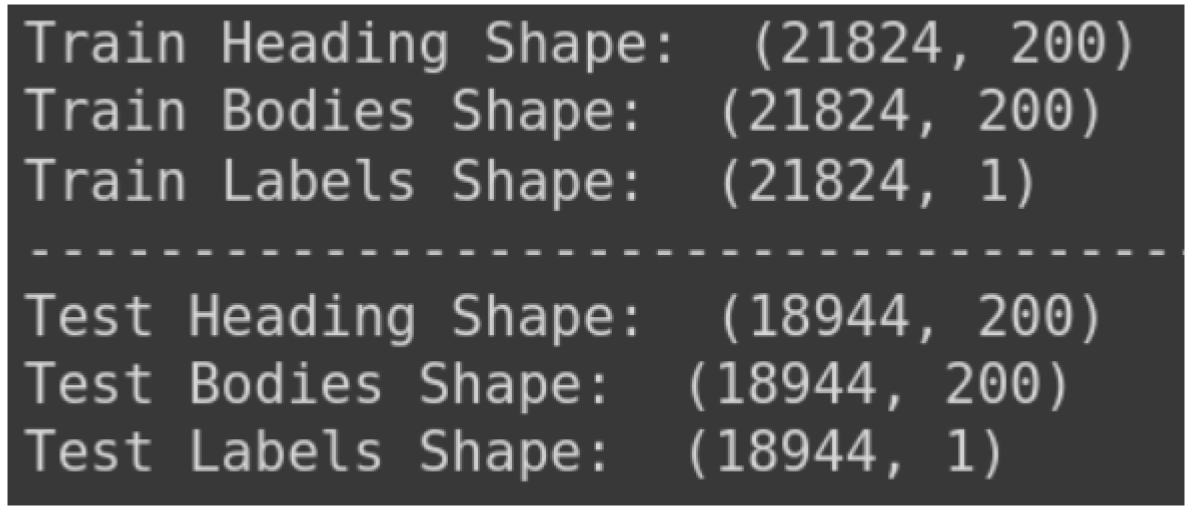
We initialize the weight and bias parameters with random variables.

- Word embedding dimension = 100
- Hidden size = 50
- Number of epochs = 5
- Block size = 50
- Learning rate = 0.001
- Lasso regularization, rate = 0.05



```
Train Heading Shape:  (49920, 200)
Train Bodies Shape:  (49920, 200)
Train Labels Shape:  (49920, 1)
----------------------------------
Test Heading Shape:  (25408, 200)
Test Bodies Shape:  (25408, 200)
Test Labels Shape:  (25408, 1)
```

Figure 12: Number of training and test samples in the FNC dataset



```
Train Heading Shape:  (21824, 200)
Train Bodies Shape:  (21824, 200)
Train Labels Shape:  (21824, 1)
-----------------------------------
Test Heading Shape:  (18944, 200)
Test Bodies Shape:  (18944, 200)
Test Labels Shape:  (18944, 1)
```

Figure 11: Number of training and test samples in the Clickbait Challenge dataset

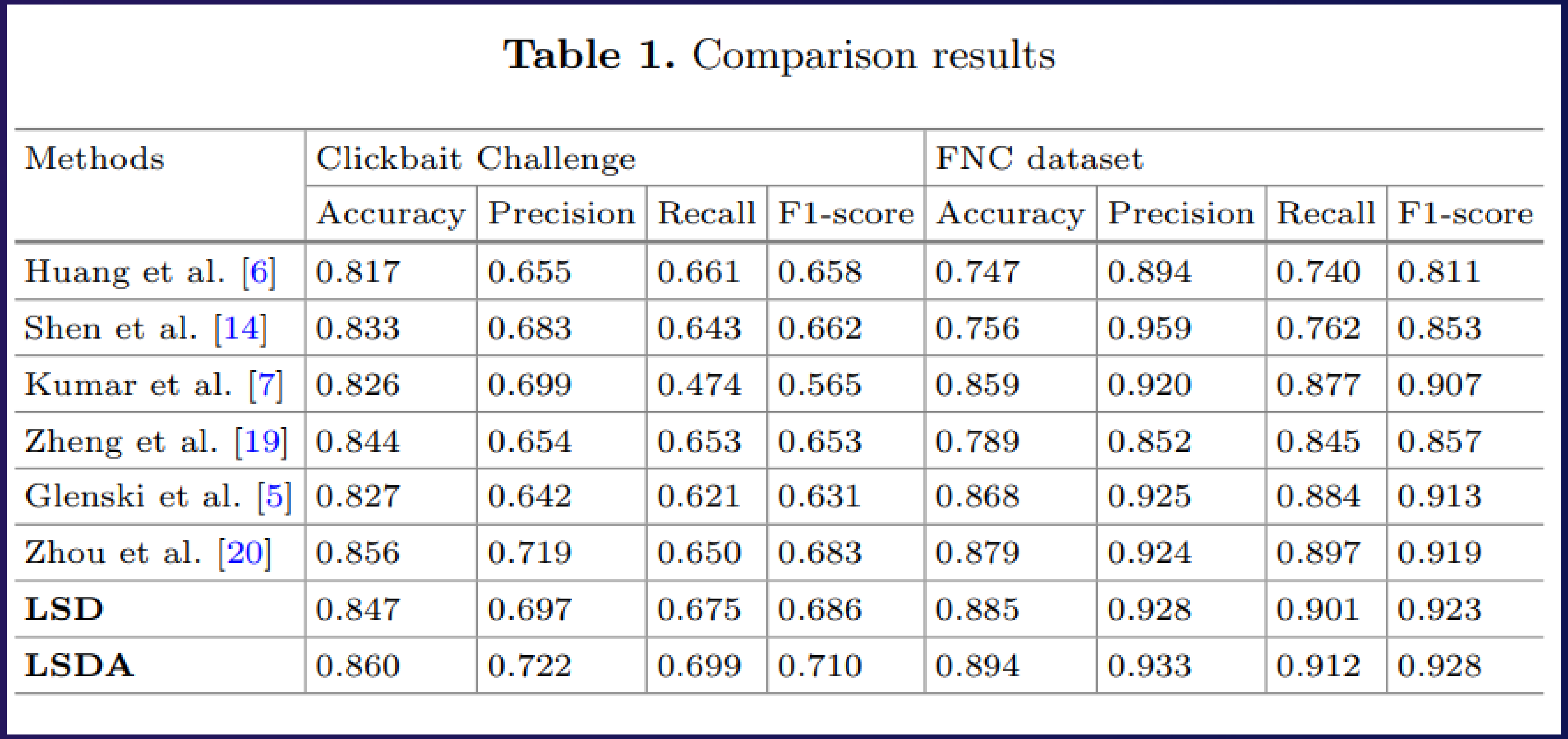
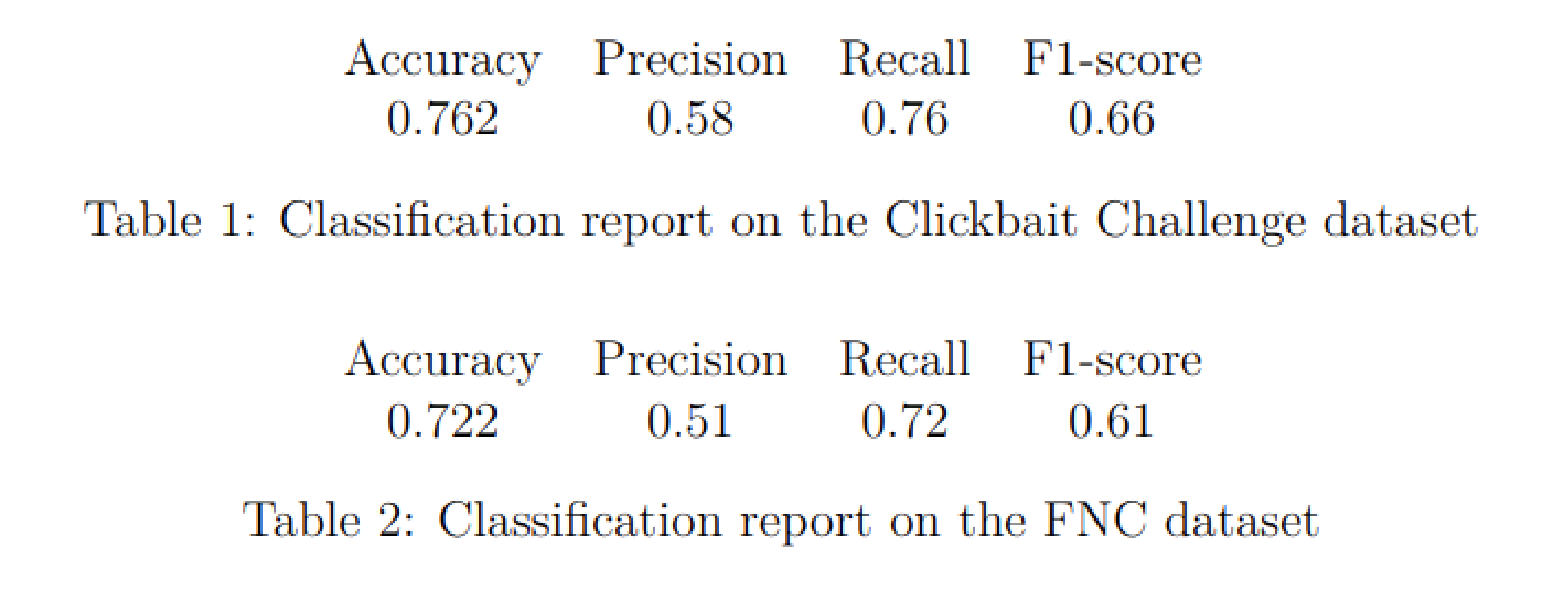


Table 1. Comparison results

Methods	Clickbait Challenge				FNC dataset			
	Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Huang et al. [6]	0.817	0.655	0.661	0.658	0.747	0.894	0.740	0.811
Shen et al. [14]	0.833	0.683	0.643	0.662	0.756	0.959	0.762	0.853
Kumar et al. [7]	0.826	0.699	0.474	0.565	0.859	0.920	0.877	0.907
Zheng et al. [19]	0.844	0.654	0.653	0.653	0.789	0.852	0.845	0.857
Glenski et al. [5]	0.827	0.642	0.621	0.631	0.868	0.925	0.884	0.913
Zhou et al. [20]	0.856	0.719	0.650	0.683	0.879	0.924	0.897	0.919
LSD	0.847	0.697	0.675	0.686	0.885	0.928	0.901	0.923
LSDA	0.860	0.722	0.699	0.710	0.894	0.933	0.912	0.928



Accuracy	Precision	Recall	F1-score
0.762	0.58	0.76	0.66

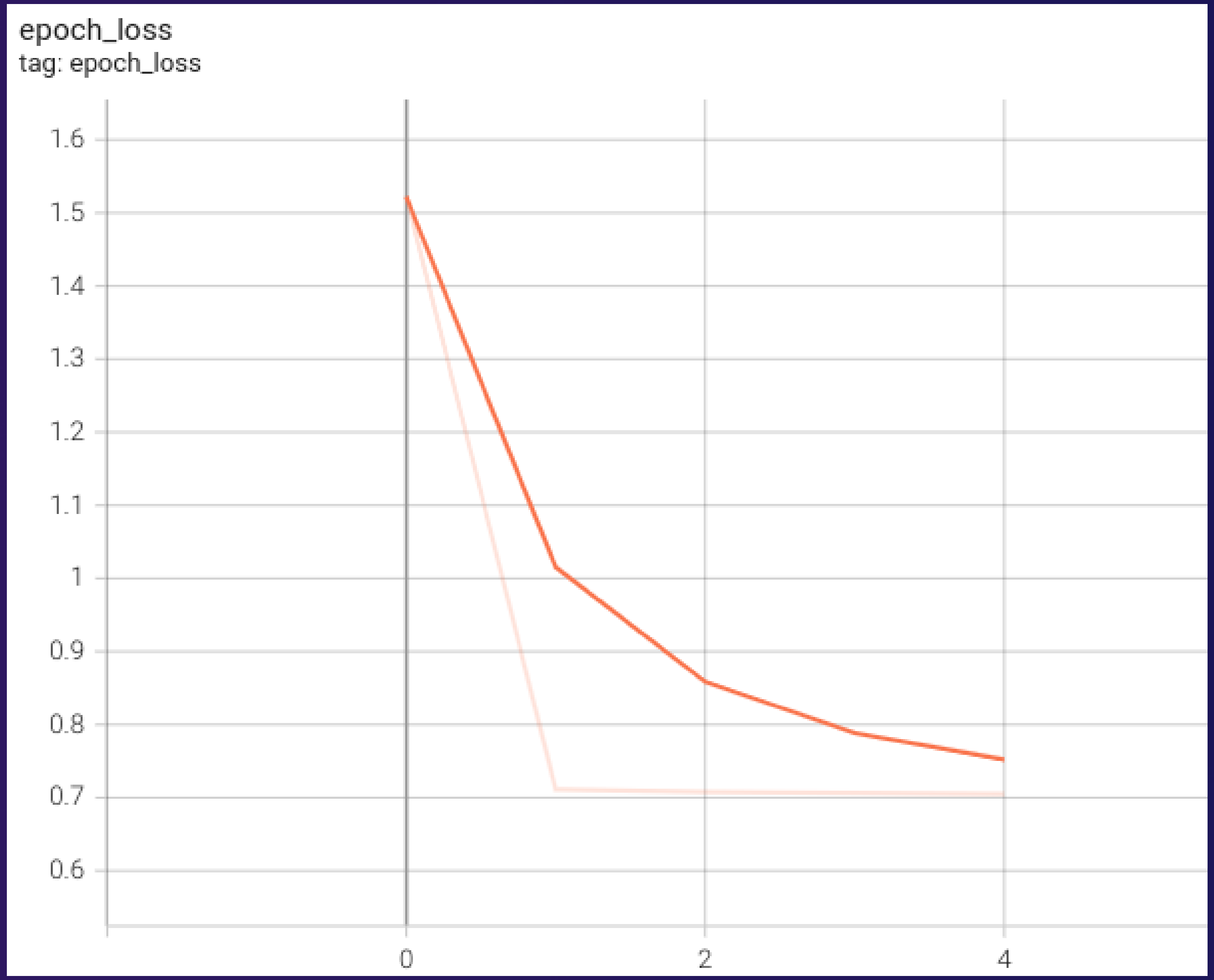
Table 1: Classification report on the Clickbait Challenge dataset

Accuracy	Precision	Recall	F1-score
0.722	0.51	0.72	0.61

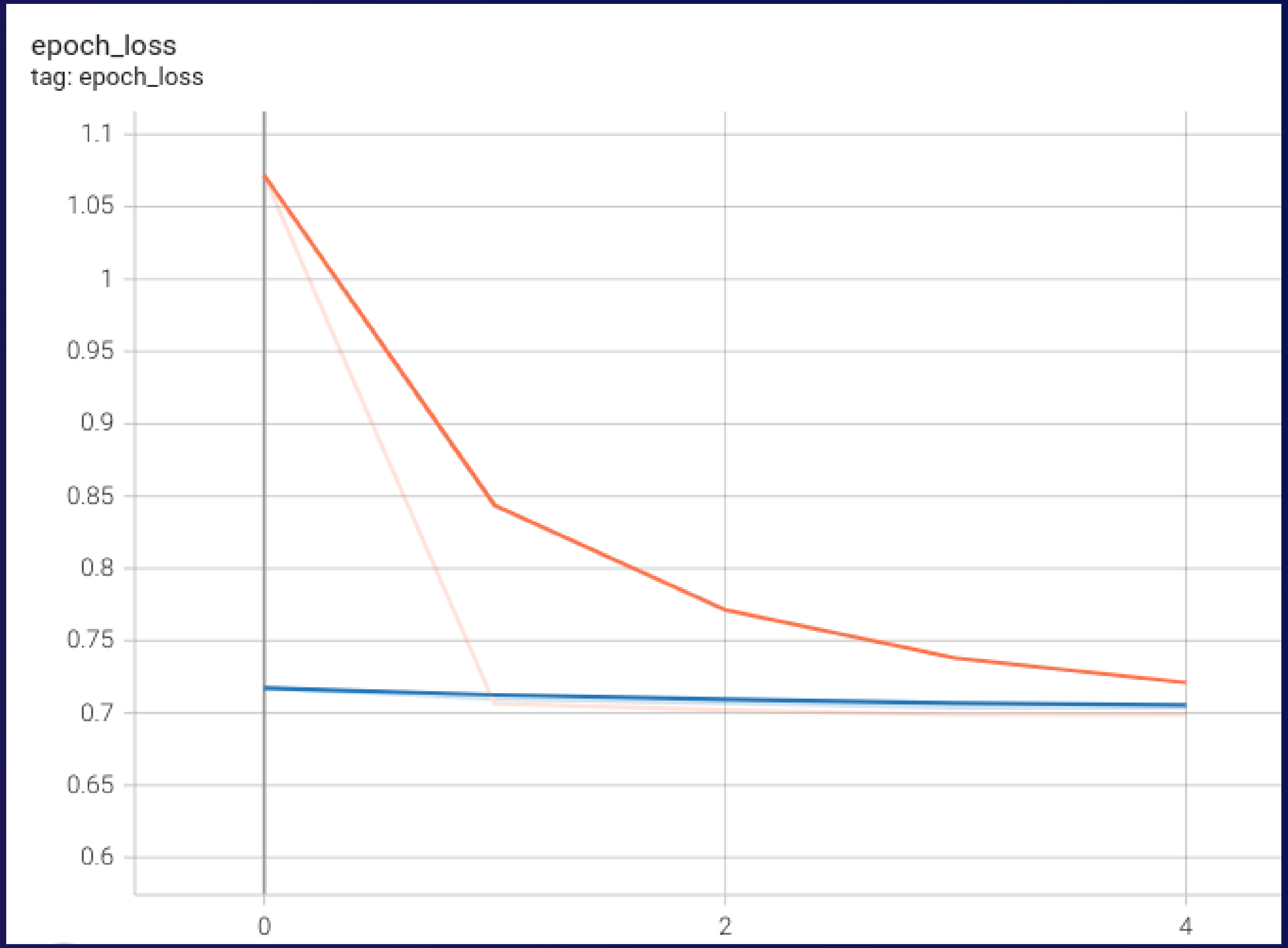
Table 2: Classification report on the FNC dataset

It is observed that both the CNN- and RNN-based models perform better than traditional deep neural networks. The attention-based bidirectional GRU shows superior performance in dealing with textual information. Both the similarity information and the attention mechanism helps with the final prediction

- **Huang et al.** : deep semantic similarity model (DSSM) (deep neural networks, to get the latent representations of the inputs and calculate the similarity in the latent representation space), global similarity. They use N-gram to preprocess the textual features.
- **Shen et al.** : DSSM, CNN
- **Kumar et al.** : Attentive bidirectional RNN based methods for learning the inputs, and then concatenate the latent inputs with the relationship information that learned with Siamese Net for the final prediction.
- **Zheng et al.** : only titles, They first transform the titles into word vectors and then use text-CNN for predicting the labels.
- **Glenski et al.** : LSTM networks
- **Zhou et al.** : attentive bi-GRU model, Two learned hidden representations are concatenated and fed into fully connected layers for the prediction.
- **LSDA**: combination of the local similarities and the raw input features
- **LDA**: deep local similarity but not the attention



Clickbait Challenge dataset



FNC dataset

Limitations and Challenges

Computational challenges: Both the clickbait challenge and FNC dataset had over 20k labelled pairs of headings and bodies. As a result, preprocessing, vectorization and training the data took around few hours.

Extracting Relevant Information from Datasets: For clickbait challenge, we had images in the dataset which made it difficult to download the whole data because of large size. The information was given in a JSONL File format and only some of which was relevant.

Construction of Model: The model required the merging of two bidirectional GRUs such that each of them run parallelly but the output of both affect each other’s loss functions. This design is not very intuitive thus it was a bit difficult to come up with the idea to implement it.

New to TensorFlow: Understanding and getting used to the functions and libraries in the allotted time of the project component has been a challenge.

Conclusion

In this project, we solved the problem of clickbait detection from the similarity perspective which represents the matching information between titles and targeted bodies. We have presented a local similarity-aware deep attentive model that learns global similarity, local similarities and raw input features to make predictions in an attentive manner. The model has a rough accuracy value of 76.2% and yields competitive results against traditional methods on two real world datasets.