

RESEARCH ARTICLE

Identifying Hate Speech Through Syntax Dependency Graph Convolution and Sentiment Knowledge Transfer

XIAOCHAO FAN¹, JIAPENG LIU¹, JUNJIE LIU¹, PALIDAN TUERXUN¹, WENJUN DENG¹, AND WEIJIE LI²

¹School of Computer Science and Technology, Xinjiang Normal University, Ürümqi 830000, China

²School of Software, Xinjiang University, Ürümqi 830000, China

Corresponding author: Palidan Tuerxun (pldtrs@xjnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62066044 and Grant 62167008, in part by the Natural Science Foundation of Xinjiang Uygur Autonomous Region under Grant 2022D01A99 and Grant 2021D01B72, in part by the National Natural Science Foundation Young Investigator Grant Program under Grant 62006130, and in part by the Xinjiang Normal University's 2022 Young Top-Notch Talent Program under Grant XJNUQB2022-23.

ABSTRACT In recent years, hate speech spread on the Internet has seriously affected society's harmony, stability, and development. A way to quickly identify hate speech from the vast amount of data on the Internet is urgent. In this paper, different from previous traditional methods, we explore a novel scenario of constructing a syntax dependency graph for each instance based on the syntactical information retrieved from an external tool. We propose a model called the Dependency Graph Convolutional and Sentiment Knowledge Transfer (DGCSKT) network. DGCSKT utilizes syntactic dependency graphs and dependency graph convolutional operations to enhance the model's ability to perceive contextual information. Additionally, we introduce sentiment resources that are data-homogeneous as an auxiliary task at the bottom level of the model to share effective sentiment features and improve recognition performance. Then, we propose the Dynamic Normalized Weighting (DNW) method to weight the training information of different tasks and thus improve the model's generalization ability. Compared to the current state-of-the-art methods, our proposed approach improves the Macro-F1 by 3.88% and 0.54% in OLID and HateEval respectively.

INDEX TERMS Hate speech detection, multi-task optimization, dependency graph convolution, dynamic normalized weighting.

I. INTRODUCTION

With the prevalence of mobile Internet and social media, bullying has gradually spread from the real world to the Internet, and the malicious spread of hate speech has caused great harm to society and families. Therefore, preventing the abuse of hate speech is of great importance to maintaining social harmony. How to quickly and accurately automatically detect hate speech and then better intervene to prevent it has become one of the hot research issues in the field of natural language processing.

The associate editor coordinating the review of this manuscript and approving it for publication was Shadi Alawneh¹.

However, hate speech is speech that attacks individuals or groups based on attributes, such as gender, ethnicity, disability, etc. [1] The semantic features of different types of hate speech vary greatly, and it is difficult to identify all types of hate speech with a unified model. In addition, due to the diversity of social media culture and the openness of words, many spans or words are abbreviated or rewritten when users make comments, such as "wtf" - "what the fuck"; "fuck" - "fffffuck". Therefore, the task is quite challenging.

Some early studies attempted to identify hate speech around linguistic rules or manual feature extraction [2], [3], [4], [5], [6]. Chen et al. [2] find that syntactical structures are beneficial for identifying hate speech, for example, when a second person pronoun and a derogatory

word appear together in a sentence, it was considered hate speech. But the rules required domain experts and had weak generalization ability. Davidson et al. [3] constructed part-of-speech(pos) features, sentiment score features, etc., The experimental results show that adding part-of-speech and sentiment features further improves the detection effect of the classifier. However, traditional machine learning methods ignore the sequence relationships between words, and these artificial features only reflect the shallow features of the text and are powerless in the face of hate speech text.

With word vector technology as the basis, numerous results have emerged based on deep learning methods. Badjatiya et al. [7] used convolutional neural network (CNN) and long-short term memory (LSTM) for hate speech detection on Twitter. The experimental results showed that the use of CNN for extracting window-level context or LSTM to extract sentence sequence information significantly improved the model's performance compared to traditional machine learning methods. In recent times, some researchers have fine-tuned large-scale pre-trained Transformer based models, such as BERT [11], GPT [12], and XLNet [13], using attention mechanisms to capture contextual information in hate speech, which has further improved the performance of the models [14], [15], [16]. Kennedy et al. discovered that in detecting hate speech, existing deep learning models exhibit strong biases towards certain words, such as "nigger", "blamed", and "homosexuality", among others. These words possess strong negative sentiment connotations, suggesting that their presence in a sentence is likely to indicate hate speech [17]. Zhou et al. also recognized that sentiment features may be a strong predictor of hate speech detection and designed a multi-task model consisting of both a hate speech detection task and a sentiment analysis task. By introducing sentiment information through shared layers, they achieved good results [19]. Previous research has demonstrated that contextual, sequential, and sentiment features of hate speech, such as the context in which it occurs, the sequence of statements, and the associated sentiment, can significantly enhance the ability to detect hate speech.

The recurrence of certain words in texts pertaining to specific topics may result in the presence of strong biases towards them in hate speech detection models. A model that exhibits strong biases towards certain core keywords is not a desirable phenomenon. The presence of these words in a sentence does not necessarily indicate hate speech and must be contextualized in the sentence to confirm. In this work, we depart from traditional methods by basing our approach on the idea that a sentence's meaning is conveyed through its core keyword and other words that are syntax connected to it. Our goal is to have the model reduce its attention to irrelevant words and increase its focus on semantically related words to enhance the detection of hate speech. Lou et al. utilized syntactic analysis tools to identify the occurrence of long-range incongruity patterns and inconsistent expressions in irony [18]. Inspired by Lou et al. and Zhou et al., we explore a novel scenario of constructing a dependency graph for

each instance based on the syntactical information retrieved from an external tool(spacy¹). Furthermore, we introduce homologous short-text sentiment data as auxiliary task to provide effective underlying sentiment features as hate speech always manifests negative sentiment. Based on it, we propose Dependency Graph Convolution and Sentiment Knowledge Transfer (DGCSKT) model.

DGCSKT is a multitask learning model that utilizes syntactic dependency graph and dependency graph convolution to enhance the model's contextual perception ability. Additionally, both sentiment analysis and hate speech recognition tasks obtain data from the same domain, and both tasks require classification of words or phrases with strong sentiment while considering the semantic similarity of these words or phrases. Therefore, effective sentiment features can be shared at the bottom layer to enhance the model's generalization ability.

Finally, we propose the Dynamic Normalized Weighting (DNW) method to weight the training information of different tasks so the auxiliary task can assist the primary task learning. The main contributions of our work can be summarized as follows:

- We are the first to exploit the Dependency Graph Convolution Network(DGCN), variants of Graph Convolutional Networks (GCN) [21], to extract syntactic information in hate speech detection.
- We propose a DNW method to weight the training information of different tasks so that they can share information and complement each other.
- We propose a new multi-task framework (DGCSKT) to capture contextual sentiment dependencies in sentences so as to improve hate speech detection performance. Experimental results on the SemEval-2019 task-5 and SemEval-2019 task-6 demonstrate that our method achieves state-of-the-art performance in hate speech detection.

II. RELATED WORK

Hate speech is usually offensive, discriminatory, and inflammatory. For minors with weak information screening ability, it may harm the formation of their three views. Attacked individuals and groups may develop negative sentiments and a counterattack psychology which is not conducive to the harmonious development of society. In addition, due to its limited textual information, insufficient data samples, and the significant differences in semantic features of different categories, researchers have conducted extensive research in recent years. In this section, we review related works on multi-task learning-based methods of hate speech detection, applications of graph convolutional networks on text, and optimization methods for multi-task learning.

Multi-task learning (MTL) can improve specific task performance by simultaneously training on multiple related tasks. In recent years, there have been some achievements

¹We employ spaCy toolkit to derive dependency tree of the sentence: <https://spacy.io>

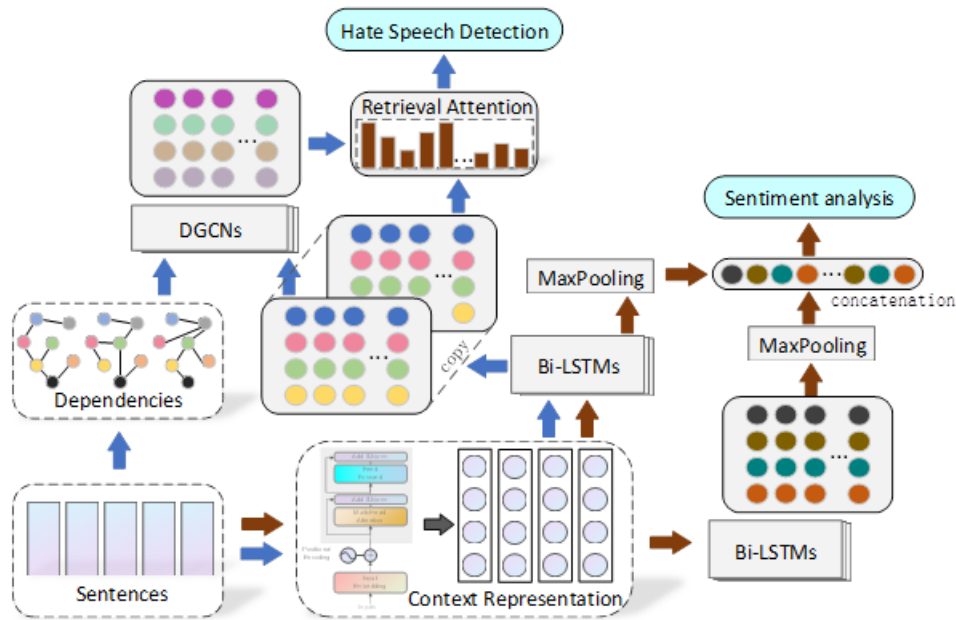


FIGURE 1. The overall framework of our proposed Hate Speech Detection based on Dependency Graph Convolution and Sentiment Knowledge Transfer Network(DGCSKT).

in the field of hate speech detection. Farha and Magdy [22] tested a simple CNN-BiLSTM MTL model based on the intuition that hate speech always expresses negative emotions toward the target, adding sentiment information to perform the task of hate speech identification in Arabic tweets. Their results showed that sentiment information is correlated with hate speech identification. Rajamanickam et al. [23] were the first to introduce emotional features into MTL models to gain auxiliary knowledge to detect abuse in English tweets. They proposed different MTL models, and the best model was the Gated Double Encoder model based on BiLSTM encoders. Their experiments showed that emotion detection is beneficial for abusive language detection. Plaza-Del-Arco et al. [24] and Zhou et al. [19] utilized sentiment analysis tasks to introduce sentiment knowledge into the model, and the results showed that it helped improve the accuracy of hate speech detection.

Previous studies have demonstrated that auxiliary tasks related to the target task can effectively improve the performance of the target task. However, they all ignore the syntax dependence information of words and sentences. What makes a comment hate speech is not determined by individual words alone, but by the keywords in conjunction with their context.

To the best of our knowledge, there has been no research related to the application of GCN to hate speech detection tasks so far, but there has been some research applying GCN to tasks related to natural language processing. Yao et al. [25] constructed an entire corpus as a giant graph in which both words and documents were nodes. They utilized GCN to model the graph and transform the text classification problem into a node classification

problem. Experimental results showed that this approach requires only a small fraction of labeled documents to achieve robust classification performance. In an emotion-cause pair extraction task, Chen et al. [26] designed a PairGCN network to model dependency relations among local neighborhood candidate pairs so as to facilitate the extraction of pair-level contextual information. Liu et al. [28] proposed the Affective Dependency Graph Convolutional Network (ADGCN) framework to address the phenomenon of incongruous expression in ironic texts. They used external tools to construct affective and dependency graphs for each instance, replacing the degree matrix of the input matrix in GCN so as to leverage the contextual affective dependencies of incongruous expressions in sarcasm detection.

Previous research has demonstrated that GCN has a powerful feature extraction capability in text as well, and making appropriate adjustments can significantly improve the performance of the target task. In the above work, Lou et al. are closest to our idea in GCN, but their approach relies too much on two external tools and has a low synchronization rate. This means that the affective and dependency graphs probably cannot be successfully constructed together.

Although multi-task learning will have advantages over single-task learning, it is not easy to learn successfully. One of the critical factors is the multi-task loss optimization problem. It is intuitive to dynamically weight losses for different tasks as opposed to fixed weights or simple summation of losses, and a lot of related work has sprung up in recent years. Chen et al. [27] and Liu et al. [28] wanted different tasks to be learned at similar speeds, and proposed the GradNorm algorithm and the Dynamic Weight

Average (DWA) algorithm, respectively. In comparison to GradNorm, DWA does not take into account the magnitude of loss for different tasks. Guo et al. [29] wanted to give higher weights to more difficult learning tasks and proposed the Dynamic Task Prioritization (DTP) algorithm. These dynamic weighting methods have different starting points but share the common goal of the best learning for each task. However, the relevance of each task is finite, and under tough competition, gains on one side necessarily entail losses on the other. The dynamic adjustment of weights while allowing for prioritization of tasks is what we want, and none of the above methods can do this.

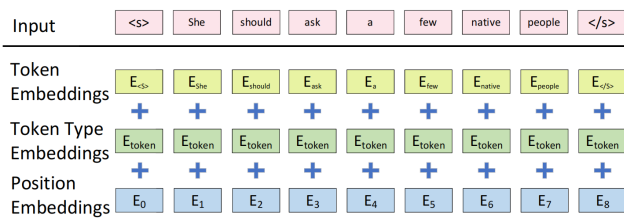


FIGURE 2. Input to the RoBERTa model.

III. METHODOLOGY

In this section, we introduce our model (DGCSKT) and the Dynamic Normalized Weights (DNW) method in detail.

The overall architecture of DGCSKT is shown in Figure 1. DGCSKT consists of three main layers: 1) Input layer. To obtain a contextual representation that more closely matches the sentence’s meaning, we use the large-scale pre-trained model RoBERTa [30]. The input is to be tokenized accordingly. On the other hand, we use the external parsing tool spaCy to obtain a syntax dependency graph for each sentence; 2) Sentiment knowledge sharing layer. We use an MTL framework to model task relationships and learn task-specific features to leverage shared sentiment knowledge based on an intuition that sentiment analysis and hate speech detection are highly correlated. Learning context representation, which learns the vector representations of the context with RoBERTa and bidirectional LSTMs(Bi-LSTM); 3) Private task layer. In DGCSKT, the entire private layer is divided into the primary task private and auxiliary task private layers. We added a private feature extraction layer Bi-LSTM to the auxiliary task. On the other hand, the primary task private layer consists of multiple layers of DGCNs and a retrieval attention layer.

A. INPUT LAYER

Regardless of the specific task associated with the text data, tokenization is the initial step. We use the open-source RoBERTa model tool from HuggingFace [31], given a sentence s consisting of n words, $s = \{w_1, w_2, \dots, w_n\}$. As shown in Figure 2, the input text sequence starts with the special symbol "<s>" and uses the special symbol "</s>" to indicate the end of the sentence. Then it transforms into a word embedding representation $[e_1, e_2, \dots, e_n]$.

The word representation is derived through the summation of three components: Token Embeddings, Token Type Embeddings, and Position Embeddings. Among them, Token Embeddings is the word embedding vector independent of context representation; Token Type Embeddings distinguishes the sentence part and the padding part in the input sequence; Position Embeddings represent the position information of the corresponding word in the word sequence.

In addition, intuitively, the expression of the meaning of a sentence usually depends on a specific syntactic structure. Therefore, intuitively, if the model can obtain word-word syntactic dependency information, it can significantly enhance the classification effect. In this work, to leverage the dependencies of the context, we explore a novel scenario of constructing a dependency graph for each sentence. This aims to remove redundant information and retain the overall structural information of the sentence in hate speech detection at the same time. We constructed a dependency graph for each sentence based on the dependency tree parsed by the external tool, spaCy:

$$A_{i,j}^d = 1 \quad \text{if } \tau(w_i, w_j) = \text{true} \quad (1)$$

where $A^d \in \mathbb{R}^{(n \times n)}$ whose remaining elements are 0. $\tau(w_i, w_j) = \text{true}$ represents that there is a relation between w_i and w_j in the dependency tree of the sentence. It is noted that this syntactic dependency is bidirectional, so $A_{(i,j)} = A_{(j,i)}$, and we set a self-loop for each word: $A_{(i,i)} = 1$.

B. SENTIMENT KNOWLEDGE SHARING LAYER

This layer includes a shared word embedding model (RoBERTa) and a shared feature extraction model (Bi-LSTMs). RoBERTa is used to train and generate word embeddings adapted to all task data, and Bi-LSTM is used to extract standard and fixed features among all tasks.

There is a considerable similarity between the hate speech detection task and the sentiment analysis task. Also, the sources of the datasets are similar, so there is no problem of training failure due to too much difference in data domains in the shared word embedding layer joint training data representation. Moreover, sharing word embedding training for different tasks enables the final contextual representations to contain representation information adapted to multi-task learning. RoBERTa is trained from a sizeable unsupervised corpus, which provides more complementary information to the data and makes the final contextual representation richer. Fine-tune the parameters of RoBERTa to make the model more suitable for multiple tasks. The output of the last hidden layer is selected as the word embedding vector, denoted as E , $E \in \mathbb{R}^{(n \times h)}$, where n is the input sequence length, and h is the word vector dimension.

With the contextual representation, feature extraction and subsequent training are required. This is where the shared feature extraction model comes into play, and the Bi-LSTM is used here. Hate speech detection tasks focus on sequence information and extraction of information from a limited context, and Bi-LSTM’s most significant advantage is that

it can simultaneously learn sequence features from context information. Its output consists of two parts, the forward output vector, and the reverse output vector. The calculation formula is as follows:

$$h_t = o_t * \tan h(f_t * C_{t-1} + i_t * \check{C}_t) \quad (2)$$

$$\vec{h}_t = LSTM(x_t, \vec{h}_{t-1}), \overleftarrow{h}_t = LSTM(x_t, \overleftarrow{h}_{t-1}) \quad (3)$$

where $h_t \in \mathbb{R}^{d_h}$ is the output of the LSTM and denotes the hidden representation of input in time step t . d_h is the word embedding dimension. o_t , f_t , i_t are the output gate weight, forget gate weight and input gate weight, respectively. In addition, C_{t-1} is the output of the previous unit, and \check{C}_t is the content that needs to be updated in the current unit. Since it is a bidirectional model, it includes forward and reverse outputs. The final output of each time step t is obtained by directly contacting these two parts, which is represented by H_t :

$$H_t = \vec{h}_t \oplus \overleftarrow{h}_t \quad (4)$$

The contextual representations obtained in this layer will respectively be used as part of the input of the two private layers. All tasks share the parameters of the trained model.

C. PRIVATE TASK LAYER

The private task layer is layer-specific to each task. It contains the task-specific feature extractor and the task output layer. Different from traditional hate speech detection methods, which treats a sentence as a sequence of words and extracts hate information purely from textual or semantic content, we explore a novel dependency graph convolutional network framework to feed sentence syntax dependency into a multi-layer DGCN architecture to leverage the syntactic dependency information. Each node in the l -th DGCN layer is updated according to the hidden representations of its neighborhoods in the adjacency matrices of the dependency graphs. The process is defined as:

$$g^l = ReLU(\tilde{A}^d g^{l-1} W^l + b^l) \quad (5)$$

where $g^{l-1} \in \mathbb{R}^{n \times 2d_h}$ is the hidden graph representation evolved from the preceding DGCN layer, and the first DGCN's input is the output of the shared layer's Bi-LSTMs: $g^0 = (h_0, h_1, \dots, h_n)$, $h_i \in \mathbb{R}^{1 \times 2d_h}$. Normalization of A^d to get \tilde{A}^d : $\tilde{A}_i = \frac{A_i}{E_i + 1}$. $E_i = \sum_{j=1}^n A_{i,j}$ is the degree A_i . $W^l \in \mathbb{R}^{2d_h \times 2d_h}$, $b^l \in \mathbb{R}^{2d_h}$ are the trainable parameters of the l -th DGCN layer.

In addition, instead of the conventional attention mechanism, we employ a retrieval-based attention mechanism to capture sentiment dependency-oriented features from contextual sentiment representations in hate speech detection:

$$\beta_t = \sum_{i=1}^n H_t^T g_i^L \quad (6)$$

$$\delta_t = \frac{\exp(\beta_t)}{\sum_{i=1}^n \exp(\beta_i)} \quad (7)$$

$$r_n = \sum_{t=1}^n \delta_t H_t \quad (8)$$

where T represents matrix transposition, and g^L is the output of the final DGCN layer. r_h is the output of attention mechanism. Here, intuitively, the difference between retrieval-based and public attention is that Q and V are the same matrices. However, from another perspective, the H_t and g^L dot product in Eq. (6) is used to measure the sentiment-dependency relatedness between word and sentence. Eq. (7) then converts this connection into an attention score, based on which it is possible to make Eq. (8)'s r_h focus on each word's syntactic dependency information.

In the sentiment analysis task, we used Bi-LSTMs as its task-specific feature extractor and employed the pooling mechanism to get the final representation:

$$r_s = MaxPooling(H_t^s) \oplus MaxPooling(H_t) \quad (9)$$

where H_t^s is the output of the private Bi-LSTMs and H_t is the output of the knowledge sharing layer. Afterward, r_s in Eq. (9) and r_h in Eq. (8) are fed to a fully-connected layer with a SoftMax normalization activation function, respectively to capture a probability distribution y of sentiment analysis and hate decision space:

$$y = softmax(W_o r + b_o) \quad (10)$$

where $y \in \mathbb{R}^{d_p}$ is the predicted probability for the input sentence, d_p is the dimensionality of hate labels or sentiment labels. W_o and b_o are trainable parameters.

D. DYNAMIC NORMALIZED WEIGHTING

The simplest way to obtain the overall loss for a multi-task is to directly sum the losses between tasks to get the overall loss. However, the unreasonableness is obvious. The magnitude of the loss is likely to be different for different tasks, and the direct summation may lead to multi-task learning dominated by one task. An alternative approach is to configure a fixed weight parameter for each task's loss which allows us to manually adjust the importance level compared with the direct summation approach; however, the fixed weights will remain throughout the training cycle. The difficulty of learning varies from task to task, and such fixed weights can limit the task learning at some stage. In summary, a better weighting approach in multi-task learning would be dynamic, adjusting for the stage of learning, the difficulty of learning, and even the effectiveness of learning for different tasks.

Due to the diversification of text sentence patterns and the significant differences in semantic features, multi-task text training has always been difficult. Whether GradNorm, Dynamic Weight Average, or Dynamic Task Prioritization Algorithm are proposed based on image tasks, because of the complexity of the text, the training loss fluctuates wildly, which is not suitable for this work. Therefore, inspired by Guo et al., we propose the Dynamic Normalized Weighting

method:

$$W_i(t') = \frac{-\lambda_i(1 - k_i(t')) \log k_i(t')}{\sum_{j=1}^m -\lambda_j(1 - k_j(t')) \log k_j(t')} \quad (11)$$

where $W_i(t')$ is the loss weight of each task i and t' denotes the training step. $k_i(t')$ represents the metric for the task i , and m denotes the total number of tasks. Intuitively, the DNW method will increase the weight of hard-to-learn tasks and vice versa. In addition, a normalized strategy ensures that losses do not get out of hand.

E. LEARNING OBJECTIVE

We minimize the cross-entropy loss by the Adam algorithm to train the model:

$$\min_{\Theta} Loss = - \sum_i \sum_j y_i^j \log \hat{y}_i^j + \lambda \|\Theta\|^2 \quad (12)$$

where i is the index of sentences and j is the index of class. y_i and \hat{y}_i respectively represent the ground-truth and estimated label distribution of instance i . λ is the L_2 regularization parameter. Θ denotes all trainable parameters.

IV. EXPERIMENTS

A. EXPERIMENTAL DATA AND SETTINGS

Next, to evaluate our proposed model, we experiment with two public datasets from a well-known competition, Semeval and one benchmark sentiment dataset:

- **Offensive Language Identification Dataset (OLID)** [32]: OLID comes from SemEval 2019 task 6, which has three subtasks. We only focused on subtask A: Offensive Language Detection. The test set had 820 sentences, and the training set had 13420 sentences.
- **HateEval Dataset** [33]: HateEval from SemEval 2019 task 5 is a multilingual dataset on hate speech against immigrants and women. It contains English and Spanish, and we take only 13,000 English data points from it. Among them, there were 10,000 sentences in the training set, and 3000 sentences in the test set.
- **Sentiment140 Dataset (S140D)**: The S140D dataset, which contains 1.6 million English tweets labeled as either positive or negative, was publicly released by the Kaggle community for sentiment analysis.

The statistics of the experimental data are reported in Table 1. The datasets OLID, HatEval, and S140D were sourced from Twitter. OLID and HateEval are both imbalanced datasets. Data imbalance is fairly common in ecology, and it usually reflects an unequal distribution of classes within a dataset. Models built on imbalanced datasets will be constrained by its ability to predict rare and minority points. On the other hand, the predictive capabilities of the model are better by balancing the data compared with imbalanced data. Therefore, whether to train on a balanced dataset does not affect the experimental results. To facilitate comparison with the baseline method, we used standard Accuracy (ACC) and F-measure (F1) as evaluation metrics in our hate speech detection.

In our experiments, the random seed was set to 1998; For the input layer, all non-Roberta models, we used the static pre-trained word vector Glove [35] as the embedding layer with each word as a 300-dimensional embedding. For the RoBERTa-based models, we used RoBERTabase with 768-dimensional embedding. In addition, the number of layers of the DGCN was set to 3, and the Bi-LSTM was only set to 1 layer. The maximum sentence sequence length was 128 for the hate speech task and sentiment analysis task. Using Adam as our optimizer, the learning rate and the coefficient λ of L_2 regularization were 10^{-3} and 10^{-5} , respectively, to train the model. The mini-batch size was 16 for the hate speech task, and the sentiment analysis task's mini-batch size varied with the proportion of the two data volumes. For example, at this time, the latter's training set size was double that of the former, so the latter's mini-batch size was 32. To prevent overfitting, we used the learning rate decay and early stop in the training process.

TABLE 1. Statistics of datasets used in the experiment.

Dataset	Train		Test	
	Hate	Non	Hate	Non
OLID	4400	8840	240	620
HateEval	4155	5845	1248	1752
S140D	790000	790000	10000	10000

B. COMPARISON MODELS

We designed model comparison experiments. All models were trained and tested on the same dataset, and the best results were selected to evaluate model performance.

SVM. Zhang et al. [36] used a linear support vector machine as a classifier, introduced ngram, derogatory word features to the model, and trained on OLID. We used the same model and trained it again on HateEval.

Bi-LSTM. Glove is used as a word embedding input to the bidirectional long-short-term memory network to extract semantic information and detect hate speech [37].

Bi-LSTM_Att. In addition to Bi-LSTM and Glove, this method adds an attention mechanism to focus keywords and detect hate speech [38].

Bi-GRU_Stacked. This method uses stacked Bidirectional Gated Recurrent to encode sentence information [39].

BERT. This method uses a fine-tuning BERT for hate speech detection [40], [41].

UE_SVM. This method pretrained Universal Encoder sentence embeddings to transform the input and SVM (with RBF kernel) for classification, scoring the first position on the leaderboard on the test set [42].

MTL_GE. MTL GE was proposed by Rajamanickam et al. [23] and used sentiment analysis as an auxiliary task, multilayer Bi-LSTM as a shared word embedding layer, and the final sentence representation was obtained by combining weighted shared representation and task private representation.

SKS. This was proposed by Zhou et al. [19] They used multiple feature extraction units to extract semantic features and then combine them with a gated attention mechanism to feed both hate speech detection and sentiment analysis tasks.

DGCSKT. DGCSKT is our proposed model, which detects hate speech based on the combination of sentiment knowledge sharing and syntax dependency.

The results of the comparisons are listed in Table 2 and Table 3. Table 2 shows the comparison of DGCSKT with existing methods. All data are from the literature. Furthermore, to further demonstrate the superiority of our method, in Table 3, we reproduce some strong baseline methods and our method trained in the same environment to remove the effects of other factors such as hardware devices and hyperparameters, etc., and select the best results to evaluate model performance.

TABLE 2. Comparison with existing methods. The results with underlined ‘_’ are imported from the literature.

Model	OLID		HateEval	
	Acc.(%)	Macro-F1(%)	Acc.(%)	Macro-F1(%)
SVM*	-	69.00	49.20	45.10
Bi-LSTM*	78.00	73.00	53.50	51.90
Bi-LSTM_Att*	82.60	76.80	-	-
Bi-GRU_Stacked*	-	-	56.00	54.60
UE_SVM*	-	-	65.30	65.10
BERT*	83.90	79.80	-	48.80
MEL_GE*	-	79.55	-	-
SKS*	-	-	65.90	65.20
DGCSKT(ours)	87.28	83.68	68.32	65.74

From Table 2, we can see that:
as follow:

1. The performance of traditional machine-learning methods based on the bag-of-words model was not satisfactory. Even incorporating n-grams and shallow sentiment features does not represent hate speech well. The experimental results show that the performance of SVM is very different from the mainstream deep learning methods, and its generalization ability is weak.
2. Deep learning has more advantages than traditional machine learning methods, but it is not absolute. Adding appropriate strategies or mechanisms, machine learning methods can also have outstanding classification performance, such as UE SVM. The hybrid neural network is better than the single neural network. For example, Bi-LSTM Att and BiGRU Stacked improved by about 3% compared with the single Bi-LSTM.
3. The performance of BERT is quite different on the two datasets. This may be because compared with OLID, which has more simple sentences with insulting words, HateEval has more language that hides the real meaning between the lines. If there is no suitable strategy for these sentences, the deep network structure will not achieve good results. In HateEval, the BERT performance is lower than that of the traditional deep learning method Bi-LSTM.
4. We observe that our proposed DGCSKT consistently outperforms all models on both datasets. To be specific,

the best-improved results of Acc. and F1 respectively are 3.38% and 3.88% compared with the previous state-of-the-art performance.

TABLE 3. Comparison with strong baseline methods. G and R denote Glove and RoBERTa, respectively.

Model	OLID		HateEval	
	Acc.(%)	Macro-F1(%)	Acc.(%)	Macro-F1(%)
Bi-LSTM+G	84.19	79.48	61.12	60.61
Bi-LSTM_Att+G	84.42	80.11	56.83	56.43
MTL_GE	84.92	80.43	59.62	58.37
SKS	85.02	81.84	64.90	63.20
DGCSKT+G(ours)	85.35	82.01	65.23	63.56
Bi-LSTM+R	84.26	80.01	62.37	62.16
Bi-LSTM_Att+R	85.81	81.48	60.60	60.69
DGCSKT(ours)	87.28	83.68	68.32	65.74

From Table 3, we can see that: as follow:

1. Multi-task models generally have more advantages than single-task models. SKS and DGCSKT perform better than the single-task model Bi-LSTM and Bi-LSTM Att on both datasets. MTL GE is slightly lower than Bi-LSTM on HateEval, higher than Bi-LSTM Att, and higher than both on OLID. is weak.
2. In the model with RoBERTa as embedding, all models' performance improved significantly compared with Glove embedding under the same conditions. Bi-LSTM and BiLSTM Att improve macro-F1 by 0.53% and 1.37% in OLID, respectively, and by 1.55% and 4.26% in HateEval. In addition, DGCSKT improves F1 by 1.57% and 2.18%, and Acc improves by 1.93% and 3.09% more than DGCSKT+G on both datasets. These show that the large-scale pre-trained model RoBERTa can better represent sentence semantics compared with the pre-trained static word vector Glove.
3. Under the same conditions, our proposed multi-task framework DGCSKT is more competitive than MTL GE and SKS. DGCSKT+G improves macro F1 by 1.% and 5.19% over MTL GE on both datasets. Compared with SKS, it has increased by 0.17% and 0.36%.

Based on the above analysis, our proposed DGCSKT model has demonstrated significant superiority over other strong baseline models. The effectiveness of our approach can be attributed to two main factors. First, our model leverages the sharing of effective features at a lower level between the two tasks. Second, the utilization of Dependency Graph Convolution and syntactic dependency graph enhance the model's ability to capture the contextual information of sentences.

C. ABLATION STUDY

To analyze the impact of different components of DGCSKT on performance, we conducted an ablation study and report the results in Table 4. Where '-degn' denotes ablation of dependency graph construction and convolution, '-sa' denotes ablation of sentiment analysis task and '-dnw' denotes ablation of DNW methods. Based on the results in Table 4, we observe that: as follow:

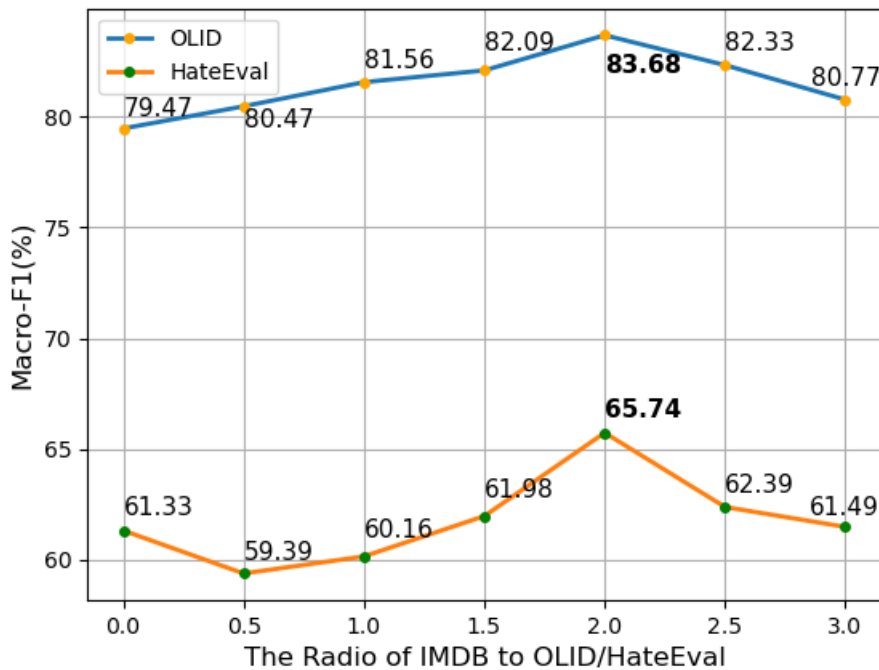


FIGURE 3. The influence of the scale of sentiment dataset.

TABLE 4. The results of ablation experiment.

Model	OLID		HateEval	
	Acc.(%)	Macro-F1(%)	Acc.(%)	Macro-F1(%)
DGCSKT(ours)	87.28	83.68	68.32	65.74
-degn	84.57	81.57	60.63	60.58
-sa	83.95	79.47	61.37	61.33
-dnw	85.58	82.56	65.83	64.32

1. Removing dependency graph convolution sharply degrades the performance, which indicates that whether the model can capture the syntax dependency of words is an important factor in improving hate speech detection. Macro-F1 on both datasets is reduced by 2.11% and 5.16%, respectively. It is not surprising that the decrease in HateEval is more pronounced because it has more hate speech sentences without prominent insulting words. The recognition of such instances depends more on syntax knowledge. is weak.
2. As an auxiliary task, the sentiment analysis task substantially improved the model's generalization ability. Experiments showed that removing the sentiment analysis task reduced the Macro-F1 by approximately 4% on both datasets.
3. Our proposed DNW method can slightly improve the model performance by about 1.2%. This is because DNW does not focus on the loss of the training process, but only on the Key Performance Indicator, i.e., Macro-F1. The lower the F1 value, the more weight the loss accounts for, so the model does not dynamically adjust the weight balance of the two tasks while biasing the focus on the sentiment analysis task in the training process.

D. IMPACT OF THE SCALE OF SENTIMENT DATASET

Hate speech detection and sentiment analysis are highly correlated, so sentiment knowledge transfer can improve hate speech detection performance. But there is one issue that is overlooked here: the impact of sentiment analysis dataset size on hate speech detection.

As shown in Figure 3, the point with 0 on the horizontal axis means that the sentiment analysis task is eliminated. The performance is worst when the size of S140D is only half of OLID and HateEval. The performance peaks when the S140D is twice the size of OLID and HateEval. In addition, when the S140D data size is too small, for the complex dataset HateEval, not only does it not achieve the effect of helping hate speech detection task, but it also increases the noise, leading to performance degradation. However, as the size of S140D keeps increasing, the performance will gradually improve. After reaching a peak, if the data size of IMDB continues to expand, the performance also shows different procedures of degradation.

E. THE EFFECT OF DYNAMIC NORMALIZED WEIGHTING

The DNW method can guarantee that the model is learning properly even when dynamically adjusting the weight relationship between multiple tasks. We recorded the two task's loss changes and weight changes separately during the training process.

As shown in Figure 4. We average the loss and weights for each epoch. In a complete training, nine epochs were trained on the OLID, and four were trained on HateEval. Whether it is the OLID or HateEval dataset, the weight changes of the hate

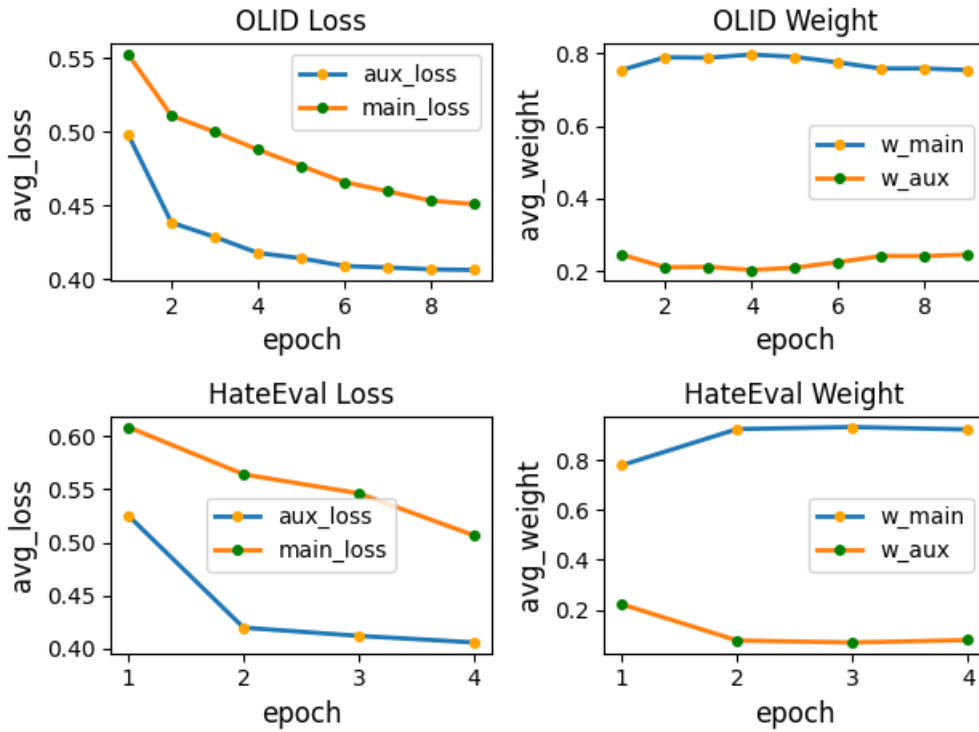


FIGURE 4. Loss change and weight change for hate speech and sentiment analysis tasks on OLID and HateEval datasets.

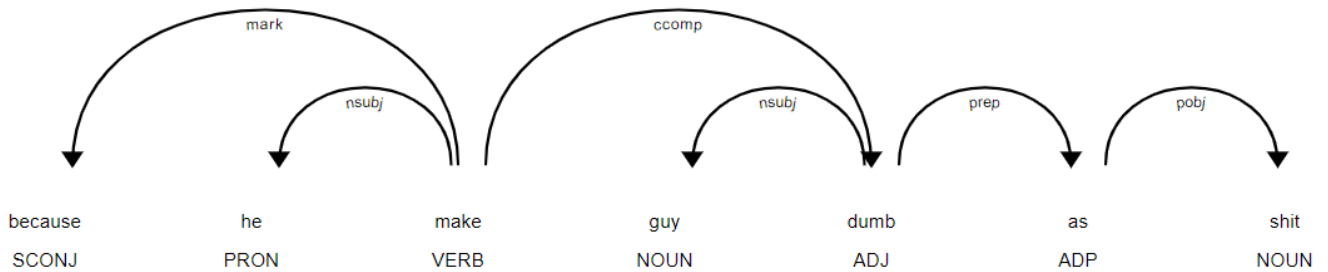


FIGURE 5. Syntax dependency parsing.

speech detection task and the sentiment analysis task always keep the hate speech detection task occupying the main body. The model is not biased towards the sentiment analysis task in the middle of training. On the other hand, the loss of the two tasks gradually decreases in the iteration, indicating that the model is learning typically. The addition of the sentiment analysis does not bring much noise to the model but adds compelling features to promote hate speech learning.

F. ATTENTION VISUALIZATION AND COMPARISON

To qualitatively demonstrate how dependency graph convolution and retrieval attention mechanisms can improve the performance of hate speech detection, we performed a visual analysis, as shown in Figure 5 and Figure 6. We first visualize the syntactic dependency graph of a typical hate speech example in Figure 5 to visualize the composition of our dependency graph. In this, the textual

information in each arc indicates the syntax relations. Second, we visualize the dependency attention and the retrieved attention learned by our proposed DGCSKT in Figure 6 to analyze how the proposed DGCSKT draws sentiment dependencies in hate speech expressions learning by the retrieved attention mechanism. The dependency attention is calculated by Equation (7), while the retrieval attention is obtained by normalizing the output of the retrieval attention layer.

The effect is intuitive. In Figure 6, the dependency attention pays more attention to the syntax body: “he make guy dumb”, rather than paying too much attention to the strong negative sentiment words: “dumb” and “shit”. On the other hand, the retrieval attention was focused on negative sentiment words and related words syntax associated with the keywords: “guy” and “make”. Such a result is not unexpected. After the dependency graph of sentences is



FIGURE 6. Attention visualization.

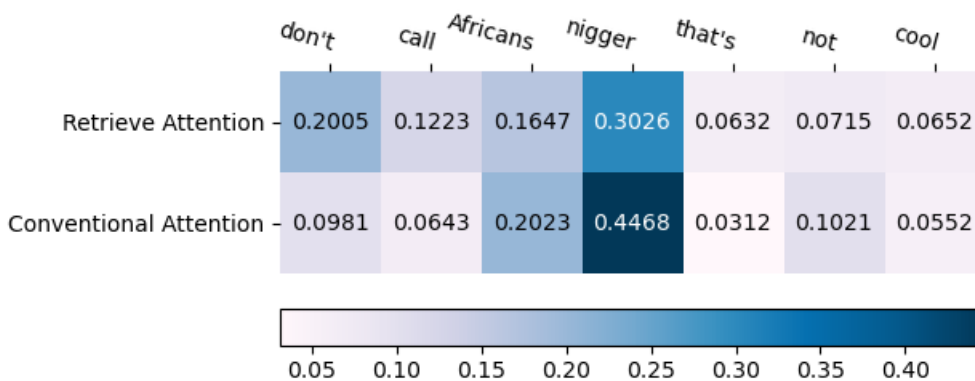


FIGURE 7. Attention comparison.

computed by graph convolution, the model ignores the part of the contextual representation obtained from the sentiment knowledge sharing layer that does not have syntax dependency information. Only the syntax dependency information of Figure 5 is retained, which is then passed to the retrieval attention layer. In addition, dependency attention passes attention on to the syntax body to contextual representations by Equation (8), enhancing attention on words with syntax linkages. This is why the attention scores obtained after the normalization of the output of retrieval attention focus not only on key negative sentiment words but also on related syntax dependent words.

We modified the attention mechanism in the -degen model to a standard attention mechanism and retrained the model in order to observe the effect of the syntax dependency convolution and retrieval attention mechanism on the model. As demonstrated in the example presented in Figure 7, the output of the attention layers for -degen and DGCSKT are normalized to obtain Conventional Attention and Retrieve Attention. -degen exhibits a strong concentration of attention on the tokens 'nigger' and 'Africans', while the weights of the remaining tokens are significantly diminished. This disproportionate distribution of attention results in the misclassification of the non-hateful statement as hateful speech. Our DGCSKT method, which utilizes a

syntax-based dependency graph, graph convolution, and a retrieval attention mechanism, effectively allocates attention to the tokens 'don't' and 'call', thereby reducing the reliance on 'nigger' and 'Africans'. This leads to the successful identification of the non-hateful nature of the statement in question.

In summary, our proposed DGCSKT promotes contextual awareness in the model, reduces dependence on key terms, and enhances the performance of hateful speech detection.

V. DISCUSSION

Our proposed method exhibited a high level of performance in our experimental analysis. In comparison to conventional deep learning approaches, our model effectively attenuates the focus on primary keywords while concurrently allocating attention to both primary keywords and syntactically related words. The most ingenious aspect of our work is the use of existing syntax analysis tools to construct dependency graphs, which are then incorporated as a part of the graph convolutional formula and allow the model to successfully perceive contextual information within the sentence through a retrieval attention mechanism. This work provides a new approach and means for future hate speech detection research. Additionally, we introduce external sentiment data from a homologous source and leverage multi-task learning

to facilitate the sharing of effective sentiment features at the bottom layer of our model, thereby enhancing the generalization capability of the model.

However, our method has the following limitations: 1. The general syntax parsing tool used has general effectiveness. If the adaptation rate of your corpus to the tool is particularly low, the construction of your syntax dependency graph will be unsuccessful. 2. The training data for sentiment analysis tasks should exhibit homology or share a common topic with hate speech recognition tasks, and ideally, have a similar text structure. In cases where the dissimilarity between the two datasets is significant, the positive impact may be limited or, in some cases, lead to adverse effects. 3. Our proposed DNW method will slightly decrease the training speed, resulting in a longer time needed for complete training.

Future work can address these limitations, such as focusing on training a domain-specific syntax analysis tool to further improve the performance of the model. Additionally, finding an appropriate sentiment analysis dataset from existing sources is very difficult. Therefore, incorporating sentiment features into the work is still a precious research direction.

VI. CONCLUSION

In this paper, we develop an efficient multi-task learning framework DGCSKT in a hate speech detection task. DGCSKT utilizes syntactic dependency graph and dependency graph convolution to enhance the model's contextual perception ability. More concretely, we explore a novel scenario of constructing a dependency graph for each instance to enhance the model's attention to syntactic dependencies by DGCNs. In addition, we construct the sentiment analysis task as an auxiliary task to bring more valid sentiment features to hate speech detection by exploiting the correlation between the two tasks. Experimental results on two benchmark datasets show that our proposed approach outperforms the current state-of-the-art methods in hate speech detection.

REFERENCES

- [1] J. Nockleby, "Hate speech," in *Encyclopedia of the American Constitution*, W. Leonard and L. Kenneth, Eds. Nov. 2020.
- [2] Y. Chen, Y. Zhou, S. Zhu, and H. Xu, "Detecting offensive language in social media to protect adolescent online safety," in *Proc. Int. Conf. Privacy, Secur., Risk Trust Int. Conf. Social Comput.*, Sep. 2012, pp. 71–80.
- [3] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," in *Proc. Int. AAI Conf. Web. Social Media*, vol. 11, no. 1, May 2017, pp. 512–515.
- [4] T. Krause and H. Grassegger, *Facebook's Secret Rules of Deletion*. München, Germany: Süddeutsche Zeitung 2016.
- [5] N. D. Gitari, Z. Zhang, H. Damien, and J. Long, "A lexicon-based approach for hate speech detection," *Int. J. Multimedia Ubiquitous Eng.*, vol. 10, no. 4, pp. 215–230, Apr. 2015.
- [6] E. Bassignana, V. Basile, and V. Patti, "Hurtlex: A multilingual lexicon of words to hurt," in *Proc. 5th Italian Conf. Comput. Linguistics*, vol. 2253, 2018, pp. 1–6.
- [7] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," in *Proc. 26th Int. Conf. World Wide Web. Companion WWW Companion*, 2017, pp. 759–760.
- [8] J. H. Park and P. Fung, "One-step and two-step classification for abusive language detection on Twitter," 2017, *arXiv:1706.01206*.
- [9] S. Zhang, X. Zhang, J. Chan, and P. Rosso, "Irony detection via sentiment-based transfer learning," *Inf. Process. Manage.*, vol. 56, no. 5, pp. 1633–1644, Sep. 2019.
- [10] P. Burnap, O. F. Rana, N. Avis, M. Williams, W. Housley, A. Edwards, J. Morgan, and L. Sloan, "Detecting tension in online communities with computational Twitter analysis," *Technol. Forecasting Social Change*, vol. 95, pp. 96–108, Jun. 2015.
- [11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., (Long Short Papers)*, vol. 1. Minneapolis, MN, USA: Association for Computational Linguistics, 2019, pp. 4171–4186.
- [12] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018. [Online]. Available: <https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf>
- [13] Z. Yang et al., "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 5753–5763.
- [14] P. Liu, W. Li, and L. Zou, "NULI at SemEval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers," in *Proc. 13th Int. Workshop Semantic Eval.*, 2019, pp. 87–91.
- [15] H. S. Alatawi, A. M. Alhothali, and K. M. Moria, "Detecting white supremacist hate speech using domain specific word embedding with deep learning and BERT," *IEEE Access*, vol. 9, pp. 106363–106374, 2021.
- [16] H. Sohn and H. Lee, "MC-BERT4HATE: Hate speech detection using multi-channel BERT for different languages and translations," in *Proc. Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2019, pp. 551–559.
- [17] J. Pavlopoulos, J. Sorensen, L. Dixon, N. Thain, and I. Androutsopoulos, "Toxicity detection: Does context really matter?" 2020, *arXiv:2006.00998*.
- [18] C. Lou, B. Liang, L. Gui, Y. He, Y. Dang, and R. Xu, "Affective dependency graph for sarcasm detection," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2021, pp. 1844–1849.
- [19] X. Zhou, Y. Yong, X. Fan, G. Ren, Y. Song, Y. Diao, L. Yang, and H. Lin, "Hate speech detection based on sentiment knowledge sharing," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, vol. 1, 2021, pp. 7158–7166.
- [20] V. Nahar, S. Unankard, X. Li, and C. Pang, "Sentiment analysis for effective detection of cyber bullying," in *Proc. Asia-Pacific Web. Conf. Cham, Switzerland: Springer*, 2012, pp. 767–774.
- [21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.
- [22] I. A. Farha and W. Magdy, "Multitask learning for Arabic offensive language and hate-speech detection," in *Proc. 4th Workshop Open-Source Arabic Corpora Process. Tools, Shared Task Offensive Lang. Detection*, 2020, pp. 86–90.
- [23] S. Rajamanickam, P. Mishra, H. Yannakoudakis, and E. Shutova, "Joint modelling of emotion and abusive language detection," 2020, *arXiv:2005.14028*.
- [24] F. M. Plaza-Del-Arco, M. D. Molina-González, L. A. Ureña-López, and M. T. Martín-Valdivia, "A multi-task learning approach to hate speech detection leveraging sentiment analysis," *IEEE Access*, vol. 9, pp. 112478–112489, 2021.
- [25] H. Cai, S. Lv, G. Lu, and T. Li, "Graph convolutional networks for fast text classification," in *Proc. 4th Int. Conf. Natural Lang. Process. (ICNLP)*, Mar. 2022, pp. 420–425.
- [26] Y. Chen, W. Hou, S. Li, C. Wu, and X. Zhang, "End-to-end emotion-cause pair extraction with graph convolutional network," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 198–207.
- [27] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 794–803.
- [28] S. Liu, E. Johns, and A. J. Davison, "End-to-end multi-task learning with attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1871–1880.
- [29] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic task prioritization for multitask learning," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 270–287.
- [30] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[31] T. Wolf et al., “Transformers: State-of-the-art natural language processing,” in *Proc. Conf. Empirical Methods Natural Lang. Process., Syst. Demonstrations*, 2020, pp. 38–45.

[32] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar, “Predicting the type and target of offensive posts in social media,” 2019, *arXiv:1902.09666*.

[33] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. M. R. Pardo, P. Rosso, and M. Sanguinetti, “SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter,” in *Proc. 13th Int. Workshop Semantic Eval.*, 2019, pp. 54–63.

[34] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning word vectors for sentiment analysis,” in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2011, pp. 142–150.

[35] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[36] Z. Zhang, D. Robinson, and J. Tepper, “Detecting hate speech on Twitter using a convolution-gru based deep neural network,” in *Proc. Eur. Semantic Web. Conf.* Cham, Switzerland: Springer, 2018, pp. 745–760.

[37] R. Ong, “Offensive language analysis using deep learning architecture,” 2019, *arXiv:1903.05280*.

[38] L. S. M. Altin, A. B. Serrano, and H. Saggion, “LaSTUS/taln at SemEval-2019 task 6: Identification and categorization of offensive language in social media with attention-based bi-lstm model,” in *Proc. 13th Int. Workshop Semantic Eval.*, 2019, pp. 672–677.

[39] Y. Ding, X. Zhou, and X. Zhang, “YNU_DYX at SemEval-2019 task 5: A stacked BiGRU model based on capsule network in detection of hate,” in *Proc. 13th Int. Workshop Semantic Eval.*, 2019, pp. 535–539.

[40] P. Aggarwal, T. Horsmann, M. Wojatzki, and T. Zesch, “LTL-UDE at SemEval-2019 task 6: BERT and two-vote classification for categorizing offensiveness,” in *Proc. 13th Int. Workshop Semantic Eval.*, 2019, pp. 678–682.

[41] M. Benballa, S. Collet, and R. Picot-Clemente, “Saagie at semeval-2019 task 5: From universal text embeddings and classical features to domain-specific text classification,” in *Proc. 13th Int. Workshop Semantic Eval.*, 2019, pp. 469–475.

[42] V. Indurthi, B. Syed, M. Shrivastava, N. Chakravartula, M. Gupta, and V. Varma, “Fermi at SemEval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in Twitter,” in *Proc. 13th Int. Workshop Semantic Eval.*, 2019, pp. 70–74.



JIAPENG LIU received the bachelor’s degree in software from Nanjing Xiaozhuang University, in 2022. He is currently pursuing the master’s degree in computer technology with Xinjiang Normal University. His research interests include hate speech detection, artificial intelligence, and deep learning.



JUNJIE LIU received the master’s degree in computer technology from Xinjiang Normal University, in 2023. His research interests include hate speech detection and natural language processing.



PALIDAN TUERXUN received the Ph.D. degree in computer software and theory from Northwest University. Her research interests include sentiment analysis and multimodal sentiment analysis.



WENJUN DENG received the bachelor’s degree in computer science and technology from the East China University of Technology, in 2022. He is currently pursuing the master’s degree in sentiment analysis. His research interests include sentiment analysis, artificial intelligence, and deep learning.



XIAOCHAO FAN received the Ph.D. degree in computer application technology from the Dalian University of Technology, in 2021. He is currently an Associate Professor with Xinjiang Normal University. His research interests include sentiment analysis, text mining, and text generation.



WEIJIE LI received the bachelor’s degree in resource exploration from the China University of Mining and Technology, in 2020. He is currently pursuing the master’s degree in software engineering with the School of Software, Xinjiang University. His research interests include text generation, sentiment analysis, and knowledge graphs.

...