Personal and Ubiquitous Computing
https://doi.org/10.1007/s00779-021-01605-5

ORIGINAL PAPER

Offensive keyword extraction based on the attention mechanism of BERT and the eigenvector centrality using a graph representation

Gretel Liz De la Peña Sarracén[1] · Paolo Rosso[1]

Received: 2 December 2020 / Accepted: 12 July 2021

Abstract
The proliferation of harmful content on social media affects a large part of the user community. Therefore, several approaches have emerged to control this phenomenon automatically. However, this is still a quite challenging task. In this paper, we explore the offensive language as a particular case of harmful content and focus our study in the analysis of keywords in available datasets composed of offensive tweets. Thus, we aim to identify relevant words in those datasets and analyze how they can affect model learning. For keyword extraction, we propose an unsupervised hybrid approach which combines the multi-head self-attention of BERT and a reasoning on a word graph. The attention mechanism allows to capture relationships among words in a context, while a language model is learned. Then, the relationships are used to generate a graph from what we identify the most relevant words by using the eigenvector centrality. Experiments were performed by means of two mechanisms. On the one hand, we used an information retrieval system to evaluate the impact of the keywords in recovering offensive tweets from a dataset. On the other hand, we evaluated a keyword-based model for offensive language detection. Results highlight some points to consider when training models with available datasets.

Keywords Unsupervised keyword extraction · Offensive language detection · Attention mechanism · Graph representation

1 Introduction

Automatic keyword extraction (AKE) is a technique of text analysis which consists of automatically extracting the most relevant words in a text. In general, it can be used to identify topics in a text, summarize its content, index data, or generate tag clouds with the most representative words. In this paper, we aim to apply the idea of AKE to obtain words that best describe the offensive language as a particular case of harmful content. Therefore, in the scope of this work, we define keywords as words that are relevant to identify offensive content. There are different approaches and available tools for keyword extraction, but they have been designed with a general purpose. Those methods extract keywords from texts with certain criteria, such as frequency. In this sense, we have identified some limitations to extract keywords of our interest, since we cannot make a distinction between offensive and non-offensive texts. This is a problem because a relevant word in non-offensive texts should not be selected as a keyword. A shallow solution would be to analyze only offensive texts, but relevant words in offensive texts that are also relevant in non-offensive texts should not be selected as keywords. Therefore, we need to solve the keyword extraction problem for our particular case, i.e. how to select words that are relevant in offensive tweets and at the same time very little relevant in non-offensive tweets.

Our methodology consists of three stages: (i) weighting pairs of words by their relationship in the tweets, (ii) building a graph where the vertices are words and the edges are weighted with the values obtained in the previous stage, and (iii) reasoning on the graph to identify the most relevant words. In the first stage, we consider the class of the tweet from which each word pair is taken. If the tweet is non-offensive, the corresponding weight is penalized. In that way, we address the aforementioned limitation. In order to obtain the weights for each word pair, the method we propose is based on the multi-head self-attention mechanism of BERT [10]. Although there are other strategies that can be adapted, we are motivated by the state-of-the-art results that BERT has obtained in several tasks, including offensive language detection and related tasks. The attention

✉ Gretel Liz De la Peña Sarracén
gredela@posgrado.upv.es

[1] Universitat Politècnica de València, 46022, Valencia, Spain

Published online: 27 August 2021

mechanism is precisely one of the strengths of BERT. This mechanism allows capturing relationships among words in a context, while a language model is learned.

The main contributions of our work are the following:

i. We propose a method for extracting keywords from datasets composed of offensive and non-offensive tweets. The method distinguishes between tweets from different classes.
ii. We use an unsupervised method which does not need annotated datasets for automatic keyword extraction, that is, datasets with gold-keywords of the texts. Instead, our method only uses a set of tweets tagged as offensive or non-offensive.
iii. We present a way to exploit the multi-head self-attention mechanism of BERT to weight word pairs from tweets.
iv. We use a method to represent the words and their relationships in a graph for extracting relevant words by using the eigenvector centrality.
v. We analyze the extracted keywords and evaluate their impact in the offensive language identification. As results, we give insights about points to consider when training models with available datasets.

An important advantage of our proposal lies in the facility to be adapted to related phenomena, such as hate speech, misogyny, and sexism [2, 11]. These are similar phenomena with common characteristics and similar datasets. In fact, there are no clear boundaries among them, although each one has particular characteristics [25]. For instance, offensiveness includes rude or vulgar language that does not represent hatred. However, our method can be applied to hate speech or other related tasks by varying the type of datasets and fixing the hyper-parameters of the model.

The rest of this paper is organized as follows. Section 2 summarizes the related work and Section 3 presents the problem formally. Our keyword extraction method is proposed in Section 4. Section 5 describes the experiments and Section 6 presents a discussion of the results. Finally, Section 7 concludes the paper.

2 Related work

This section presents a summary of some widely used keyword extraction techniques. Then, we provide a brief overview of offensive language detection, both in the sense of keyword-based strategies and in the sense of BERT-based approaches. Finally, this section introduces a synopsis of textual graph representation, with focus on techniques used for keyword extraction.

2.1 Automatic keyword extraction

AKE has been developed with different approaches [13, 16, 20]. Using statistics is one of the simplest mechanisms for selecting keywords within a text. This approach includes well-known techniques such as word frequency, term frequency-inverse document frequency (TF-IDF), word collocations, and co-occurrences. Roughly, they consist of listing the words according to some criterion and selecting the top ones. For instance, the word frequency technique looks for the most common words occurring within a collection of texts. The advantage of this kind of approach relies on that they do not need training data in order to extract keywords. However, they may ignore some relevant words that are mentioned only once but are indeed relevant. Linguistic approach is another type of mechanism which considers linguistic information about texts. Some strategies involve morphological, syntactic, or semantic information about the words, such as part-of-speech or the relations between words in a dependency grammar. This kind of information provides an important tool for keyword extraction [14]. Moreover, AKE is also addressed by employing machine learning techniques which are usually supervised approaches. A well-known method that transforms AKE into a binary classification task was presented in [36]. Other methods include models such as support vector machines, conditional random fields and deep learning strategies [12]. The authors of [28] proposed a keyphrase extraction as a sequence labeling task. They used BERT to obtain contextual embeddings, although they required manually annotated keyphrases.

Some of the previous techniques are combined in a hybrid mechanism in order to obtain better results. In this paper, we exploit this idea.

2.2 Keywords in offensive language detection

Regarding offensive language, keywords have been mainly used to build datasets. The data collection for the construction of the Offensive Language Identification Dataset (OLID) [37], used in OffensEval 2019 shared task [39], was based on searching for keywords and constructions that are often included in offensive messages. Initially, a set of words was used to collect tweets, and then some keywords that were not frequent in offensive content were excluded during the trail annotation. Similarly, for the dataset of the HASOC track [18], the data were acquired using hashtags and keywords with offensive content. Here, we aim to use a keyword-based technique to evaluate our keyword extraction method by analyzing how these keywords can influence the detection of offensive language. However, it is important to consider that the keyword-based strategies have been found to be biased and problematic for offensive

language detection and related tasks. They overlook cases in which no profane nor offensive words are used but the text actually conveys an offense. Moreover, these strategies can cause non-offensive texts that contain some keywords to be misclassified. That is why we only employ a keyword-based strategy to evaluate the extracted keywords, not to improve the offensive language detection.

2.3 BERT for offensive language detection

Most of the strategies used for offensive language detection are based on traditional machine learning and deep learning [24, 29, 30, 33]. Among them, BERT and other transformers-based models are state-of-the-art in the latest results, specially in shared tasks such as OffensEval 2020 [40]. The best team used RoBERTa-large, which was fine-tuned on the dataset by using the masked language modeling objective [34]. The second team used an ensemble which combined XLM-RoBERTa-base and XLM-RoBERTa-large [32]. In general, the top teams used BERT, RoBERTa, or XLM-RoBERTa [7, 9, 23].

2.4 Text representation based on graph

A graph-based text representation allows exploration of the relationships and structural information into a text very effectively. Then, AKE is often performed by selecting vertices or groups of vertices with a search on a graph. TextRank [19] is a model widely used in this type of approach. It is derived from PageRank [5] which scores each vertex taking into account the importance of its neighborhood. Ao et al. [1] recently proposed a new keyword extraction algorithm based on TextRank. However, Boudin [4] compares various centrality measures for graph-based AKE and the experiments on datasets in English and French show that the simple degree centrality achieves results comparable to TextRank. We test our proposal with different types of degree centralities to select vertices from a word graph. Finally, we use the eigenvector centrality as it is explained later.

3 The problem

The problem we address in this work can be formally described as follows: Let O and N be two sets of offensive and non-offensive tweets respectively, for which the following holds: $\{O \cap N = \emptyset\}$. Let W be the set of words from $\{O \cup N\}$. The problem is to identify a set of words $K \subset W$ such that each $k \in K$ is in the top ranking of the words highly relevant in O and little relevant in N. In modeling this problem, we represent W in a graph with weighted edges from which we rank the words. In the graph each vertex is a word of W and each edge (w_1, w_2), $w_1, w_2 \in W$, indicates the weight between the words w_1 and w_2. We aim to calculate the weights considering the context of the words in each tweet, as well as whether the tweet is offensive or not. For that, BERT can be suitable since its self-attention mechanism analyzes each word looking at other words in the context. Thus, the research questions we address in this work are:

> **RQ1:** How can we leverage the attention mechanism of BERT to weight pairs of words in the context of a text?
> **RQ2:** How can we effectively extract words that are relevant in offensive tweets and little relevant in non-offensive tweets from a dataset?

4 Keyword extraction based on BERT

In this section we first introduce our methodology for keyword extraction from a dataset. Then, we explain in detail each of the stages of the methodology and comment how our method can be extended to deal with longer texts.

Figure 1 illustrates the elements in which our proposal is based on. The methodology is composed of three stages. The first stage consists of obtaining a relationship between words in the dataset. In this sense, we obtain a weight for each pair of words by relaying on BERT and specifically on the multi-head self-attention mechanism. In the second stage we generate a graph where the vertices are the words from the datasets, and the edges are weighted according to the relationship between words. The weight of each edge is updated every time the corresponding word pair appears in a text. For each text, the weight calculated for each pair is added to the weight of the corresponding edge in the graph if the text is offensive, and is subtracted otherwise. In this way, we penalize the non-offensive texts. Finally, we obtain a keyword list in the third stage by identifying the most relevant vertices in the graph with the eigenvector centrality.

4.1 Attention from BERT

In text classification, some parts of the input can often be more relevant compared to others. Attention mechanisms incorporate the notion of relevance by allowing a model to dynamically pay attention to only certain parts of the input. The assumption is that the higher attention weights correlate with how relevant a specific region of input is [8]. The following example shows three texts from the OffensEval 2019 dataset [38]. An attention mechanism can help to classify the second example as offensive and identify *fu**ing* in that text as more meaningful to determine offensive.

He is such a good ad for conservatives. ||| She is fu**ing delusional. ||| I quite enjoy these tweets you are liking.

We use this idea to obtain the relationship between words in the texts of a dataset. Concretely, we leverage the multi-head self attention mechanism from BERT.

The first step is fine-tuning a pre-trained BERT model for the offensive language detection task as the first part of the Fig. 1 shows. For this, we use a dataset with offensive and non-offensive texts (1 or 0 respectively). Thus, while the model is trained, the parameters of the attention mechanism in each layer of the BERT model are updated according to the data.

In this step of fine-tuning, the input is text and a softmax is added on the top of the last layer of BERT. We only retain non-padding tokens to feed the softmax by multiplying the output with a mask. Cross entropy is used as the loss function (1), where y_i is the true classification of a text i, and $\hat{y}_i$ its predicted value.

$$L = -\Sigma_i \; y_i * log(\hat{y}_i) \tag{1}$$

Now, it is important to understand what happens inside BERT. With the self-attention mechanism, each position t in the input (token) is processed by looking at other positions to obtain a good encoding for t. Thus, this mechanism is used to capture related and important words. Basically, self-attention creates three vectors for each word by multiplying the input embedding by three matrices which are fitted in the training process. The vectors are known as query (q), key (k), and value (v), and the matrices of parameters are W^q, W^k, and W^v respectively. Then, a score is calculated for each word against each of the other words. The calculation is done by a normalized dot product of the q vector of the current token t and the k vector of the other tokens. Thus, a vector is obtained for each token where the components determine how much focus to put on the other parts of the input at this position. Next, this vector is multiplied by the v vector to keep the values of the original token t. Equation 2 recaps this process for the matrix calculation for all words at once [31]). Where d_k is the dimension of q, k, v, and Q, K, and V are the matrix representations respectively for the text.

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2}$$

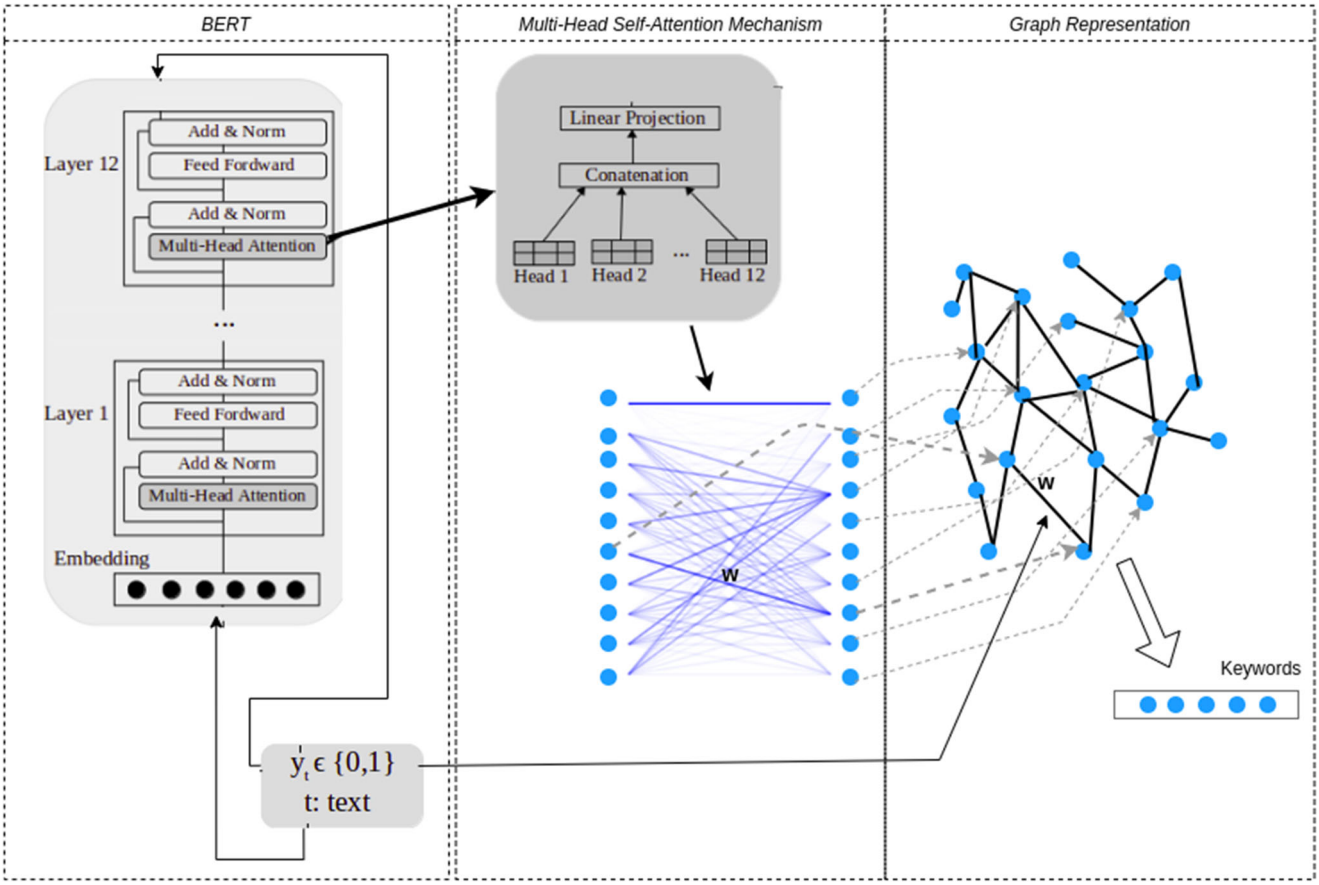

Fig. 1 Illustration of our keyword extraction model

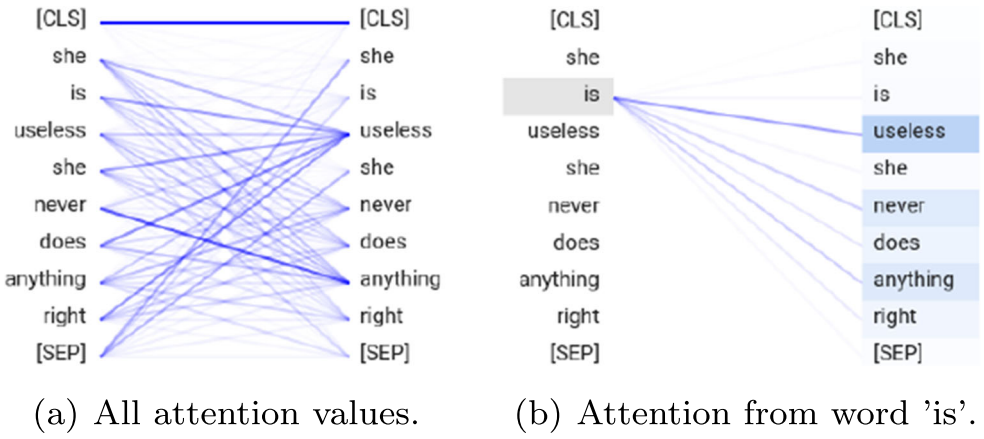

Fig. 2 Attention visualization for a sample text

(a) All attention values. (b) Attention from word 'is'.

Figure 2 shows the visualization of the weights between pairs of words for the text '*she is useless she never does anything right*' using the pre-trained BERT_base model. This is an offensive text taken from the dataset of the OffensEval 2019 shared task. On the left (Fig. 2(a)), we can see the higher weights with a more intense color, which indicates relevant parts of the text for each term. For example, the word '*anything*' seems to be quite relevant when the word '*never*' is analyzed. On the right, Fig. 2(b) shows the particular case of the attention values (weights) between the word '*is*' and the other words in the text.

Furthermore, BERT incorporates a multi-head attention which expands the ability to focus on different positions by giving the attention layer multiple representation subspaces from multiple sets of $Q/K/V$ weight matrices. Thus, there are $L \cdot H$ self-attention patterns in the model, where L is the number of layers in the model and H the number of heads in each layer.

Figure 3 shows 12 patterns for the text analyzed previously. They correspond to the three last layers of the model, which are usually the most used layers for obtaining the output. The first row corresponds to the layer #12 (the last one of the model), the second row corresponds to the layer #11, and the third row corresponds to the layer #10. For each layer, we show the first four heads from left to right. That is, the first column corresponds to head #1. We can see interesting patterns in these layers. For example, the fourth head in the layer #12 represents a pattern where the attention for each word is focused on the previous word in the text. This makes sense because adjacent words are often relevant for predicting the next word. On the other hand, the second head of the same layer matches

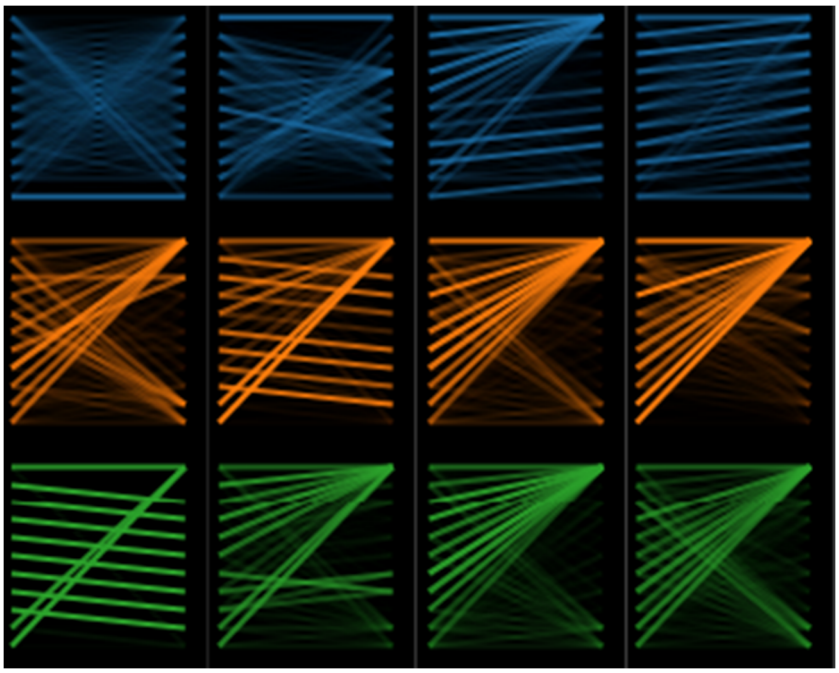

Fig. 3 Visualization of 12 heads of the attention mechanism

the one shown in Fig. 2. Other heads, like the fourth one in the layer #11, represent a null pattern where almost all the attention is focused on the token CLS. This probably indicates that those heads did not find a linguistic phenomenon. However, with the multi-head mechanism different strategies are combined to analyze the relationship between words.

Once the parameters of BERT are learned in the fine-tuning step, the texts feed the model again to obtain the attention values for each pair of words in the dataset. These attention values are obtained by the condensation of the pattern of each head in the last layer of BERT as (3), where h is the number of heads in the layers. That is, first all attention heads are concatenated and then, it is projected into a new space by multiplying for a matrix W, which is also fitted in the training step.

$$\begin{aligned} Att_i &= Attention(Q_i, K_i, V_i),\ i = \overline{1,h} \\ \text{MHA} &= Concat(Att_1, \ldots, Att_h)W \end{aligned} \tag{3}$$

As result, we obtain a matrix $A \in \mathcal{M}_{|\mathcal{T}|}(\mathbb{R})$ with the relativity of words, where $\mathcal{T}$ is the set which contains the words, $|\cdot|$ denotes the size of a set, and $\mathcal{M}_n(\mathbb{R})$ represents the set of square matrices of size n with inputs in the field $\mathbb{R}$.

We consider a word as the terms that are not stopwords and that represent English words. That is, the special tokens CLS and SEP are not selected into $\mathcal{T}$. They do not have a meaning in the human language, therefore they cannot be selected as keywords. Moreover, these tokens tend to get high scores due to the null patterns in the attention mechanisms. Furthermore, the tokens starting with the characters '##' are not considered as words because they only represent parts of words.

Earlier, we have seen how to get the matrix A for a set of text, now we explain how to update this matrix with new texts. Let $\mathcal{T}_t = \{w : w \in t\}$ be a set of words for a sample text t from the dataset, then $\mathcal{T}$ is modified as $\mathcal{T} = \mathcal{T} \cup \mathcal{T}_t$. Moreover, let A_t be the attention matrix obtained given t, that is $A_t = \text{MHA}$ for t. The attention matrix A is updated with t by a function $\theta : \{0,1\} \times \mathcal{M}_{|\mathcal{T}_t|}(\mathbb{R}) \times \mathcal{M}_{|\mathcal{T}|}(\mathbb{R}) \to \mathcal{M}_{|\mathcal{T}_t \cup \mathcal{T}|}(\mathbb{R})$ as (4) where y_t is the label of t and ϵ is a parameter to control the change in A given t.

$$\begin{aligned} A'_t &= (2 \cdot y_t - 1) \cdot \epsilon \cdot A_t \\ \theta(y_t, A_t, A) &= \begin{cases} (A)_{i,j} + (A'_t)_{i,j} & \textbf{if } \exists (A)_{i,j} \\ add((A'_t)_{i,j}) & \textbf{if } \nexists (A)_{i,j} \end{cases} \\ & i, j = \overline{1, |\mathcal{T}_t|} \end{aligned} \tag{4}$$

The function $add(\cdot)$ incorporates a raw and a column in A for each word in $\mathcal{W}_t = \{w : w \in \mathcal{T}_t \wedge w \notin \mathcal{T}\}$. Then, for each pair of words that is out of A'_t, that is a pair (w_1, w_2) such that $w_1 \in \mathcal{W}_t$ and $w_2 \in \{w : w \in A \wedge w \notin \mathcal{W}_t\}$ or vice versa, the function puts 0 in the corresponding cell of A; otherwise, the value into A'_t for the pair is taken.

Notice that for an offensive text (label 1) the new attention value of each word pair is added according to the magnitude of ϵ. In contrast, the new attention values for a non-offensive text (label 0) are subtracted. In this way, we control the score of each word pair, penalizing those extracted from non-offensive texts.

4.2 Graph representation

We use a graph as a mathematical model to represent the relation among pairs of words from the matrix A of attention values. Formally, we build a directed graph $G = (V, E_A)$ as a set of vertices V that matches with the set $\mathcal{T}$ and a set of edges $E_A \subseteq \{(w_1, w_2) : (w_1, w_2) \in \mathcal{T} \times \mathcal{T}\}$. Also, we define a function $val : E_A \to \mathbb{R}$ such as $val(w_1, w_2) = (A)_{w_1,w_2}$ to assign a weight to each edge. Thus, the vertices represent words and the edges represent the relation between pairs of words (attention values).

In the construction of the graph, we define the function $\Phi : \mathcal{M}_{|\mathcal{T}|}(\mathbb{R}) \to E_A$ such that $\Phi(A) = \{(w_1, w_2) : (w_1, w_2) \in \mathcal{T} \times \mathcal{T}, val(w_1, w_2) > 0\}$. Hence, we only represent the relationships between words with a positive attention value. Furthermore, we use this function Φ to update the graph G once new texts are incorporated in the analysis.

4.3 Keyword extraction from graph

The candidate keyword list $\Gamma(G)$ is obtained by selecting the words associated to the most relevant vertices into the graph G. To carry it out, we rank the vertices using a measure which assigns a score to each vertex considering the weights of the edges in G.

The eigenvector centrality (EC) is the measure we use for the ranking [21]. EC measures the influence of a vertex in a graph scoring each vertex as a function of the centralities of its neighbors as (5), where λ is a constant called eigenvalue and $\mathcal{N}(w) = \{w_j : (w, w_j) \in E_A\}$ is the neighborhood of the vertex w.

$$EC(w_i) = \frac{1}{\lambda} \sum_{w_j \in \mathcal{N}(w_i)} val(w_i, w_j) \cdot EC(w_j) \tag{5}$$

Alternatively, we use some centrality measures based on the degree of the vertices. Unlike EC, they do not take into account the weight of the edges. Instead, they just use the information of the neighborhood of each vertex. These measures have sense since vertices with a high degree indicate words with relevance. However, the comparison among all the strategies allows to analyze the importance of considering the weight of edges estimated from the attention

mechanism. We used the following alternative centrality measures:

- Degree (DC): $DC(v)$ is calculated by dividing the amount of vertices that v is connected to, by the maximum possible degree in G.
- In degree (IC): $IC(v)$ is calculated by dividing the amount of incoming edges at v, by the maximum possible degree in G.
- Out degree (OC): $OC(v)$ is calculated by dividing the amount of outgoing edges from v, by the maximum possible degree in G.

Selection criterion based on part-of-speech Once the ranking of vertices is obtained, we select as keywords those candidates that are one of the parts of speech: noun, adjective, or verb. This idea has been developed in some approaches where usually only nouns and adjectives are taken into account [15]. We consider that verbs also can express offenses, like for instance '*fu***'.

4.4 Analyzing longer texts

Notice that our method is designed to work with tweets, that are small texts. However, our proposal can be extended based on the way that BERT can be generalized to deal with longer texts. BERT can handle input sequences up to 512 tokens long. However, there are some strategies that can be adopted. Among them, the strategy presented in [22] is a good one. The idea is to divide each large text into segments and feed BERT with each of them. The pooled output and the logits are used as representations for each segment. Then, they are passed along to either an LSTM recurrent neural network model (RoBERT variant) or a lightweight transformer (ToBERT variant). Thus, our method can be used in the same way.

5 Experiments

This section presents our experiments and results. We first describe the datasets and the evaluation methods we used. Then, we detail the experimental setup and other models used to compare our proposal. Finally, we present the results and analysis.

Datasets We used the datasets released for the OffensEval shared task in its two editions: OffensEval 2019 [39] and OffensEval 2020 [40]. Both tasks focus on the identification of offensive language in tweets.

OffensEval 2019 (OFF19) This is a dataset which contains English tweets. The labels are organized in a hierarchical tag set [37]. We used the tags at the level of the binary classification, such that we worked with the two labels offensive (4640 tweets) and non-offensive (9460 tweets).

OffensEval 2020 (OFF20) This is a multilingual dataset with tweets in five different languages [27]. We randomly selected 30,000 tweets from the nine million of English tweets, keeping the proportion between offensive (4782 selected) and non-offensive (25,218 selected) tweets in the original set. In this dataset we had access to the average of ,the confidence in the offensive class of several supervised models. In order to work with binary labels, we transformed the averages values by considering as offensive a tweet with a confidence average greater than or equal to 0.5, otherwise we considered it as non-offensive.

Evaluation methods An AKE model is usually evaluated with a set of 'gold' keywords that constitute the references. The idea is to compare these references with the extracted keyword list. In this case, we do not have this kind of labeled datasets. Therefore, we use an extrinsic evaluation method with two relevant applications. Thus, we evaluate the keywords by means of their impact in other two tasks: information retrieval and offensive language detection.

Information retrieval (IR) We evaluated the keywords by searching tweets in a collection of indexes tweets. The idea is to analyze how suitable the keywords are to extract offensive tweets. In this regard we defined an IR task as finding offensive tweets from a large set of tweets given a keyword list as query. We created a model which indexes the documents (tweets) by concepts. Then, we use the BM25 algorithm [26] to retrieve tweets given a set of keywords. As the evaluation measure we use the Precision@K (P@K) and F@K which compute the precision and F1-score respectively over the top-K retrieved tweets [6]. Here, a true positive is a retrieved tweet that is offensive.

Offensive language detection (OLD) In order to study how the keywords can impact offensive language detection, we evaluated a keyword-based model. We have to point out that keyword-based approaches can be very misleading for detecting offensive language, since they overlook many cases in which no offensive words are used but the text still conveys extremely offensive content [35]. In this sense, our aim is not to improve the state-of-the-art in this task, but rather to analyze how relevant words in the datasets used to train models can influence the task. We used a LSTM recurrent neural network and its output is concatenated with a vector of similarity values between the input (tweet) and each of the keywords. Finally, the classification is obtained

with a softmax.[1] As the evaluation measure we used the macro-averaged F1-score (F1).

Experimental setting In order to obtain the keywords, we used the pre-trained BERT_base[2] model which has 12 layers and 12 heads per layer. Moreover, we used *NLTK*[3] to obtain the part-of-speech tags of each word, since our method only considers nouns, adjectives and verbs, as we explained before. We trained BERT for 4 epochs with minibatches of size 16. The optimizer we used was Adam [17] and we set the learning to 5e-5. The parameter ϵ of our proposal was fixed to 0.1 after a parameter setting. For OLD we relied on the stratified 5-fold cross-validation technique and the paired permutation test with p-value < 0.05 for the analysis of statistical significance. For IR we used the t-test with p-value < 0.05 For the reproducibility of the experiments we set the random seed to 5.

Furthermore, in our experiments we only used the datasets introduced before. Both OFF19 and OFF20 are composed of tweets, but they are different in the sense of data collection and annotation. OFF19 was collected by retrieving tweets with keywords that are common in offensive texts, while OFF20 was collected by searching the 20 most common English stopwords to ensure a variety of random tweets. Moreover, a semi-supervised labeling was used for OFF20. Thus, OFF19 can be more biased towards the keywords used in the collection, and OFF20 towards the way it was annotated. In this sense, we followed the two possible cross-evaluations: (i) obtain the keywords from OFF19 and then use OFF20 to evaluate the OLD and IR tasks considering the keywords from OFF19 (OFF19-OFF20) and (ii) vice versa (OFF20-OFF19).

In addition, we carried out two in-domain evaluations OFF19-OFF19 and OFF20–OFF20 to study how the bias in the datasets can influence the extraction of keywords. For example, we suppose that the keywords from OFF19 should reflect the bias in this dataset. Therefore, the use of these keywords should affect the detection of offenses.

Benchmarks As part of our experiments, we used some well-known models to extract keywords. The objective is to analyze the relevance of our proposal by comparing it with methods that are not tailored to our concern. We adapted term frequency-inverse document frequency (TFIDF) by calculating TF in the offensive tweets and IDF in the non-offensive tweets. Moreover, we used TF, TEXTRANK (TRANK), RAKE [3], and YAKE.[4] This last method reports state-of-the-art results. For each of these last five methods, we only used the set of offensive tweets, since they are not thought to discriminate between classes.

Table 1 F1-score with BERT. Each column corresponds to the layer(s) used to feed the classifier in the fine-tuning

	Layers		
Dataset	[12]	[11]	[9–12]
FF19	0.789	0.770	0.779
OFF20	0.895	0.894	0.885

5.1 Results

Table 1 shows the performance of BERT in offenses detection after fine-tuning with each of the datasets. We evaluated different layers as output to feed the classifier on the top of BERT, including the concatenation of some layers. Specifically, we evaluated the last layers and their concatenation. That is why Table 1 shows the results for the two last layers, as well as for the concatenation of the four last layers. In general, the results are similar among them. Hence, in the rest of the experiments, we used the output of the last layer (layer #12) to feed the classifier for fine-tuning. Furthermore, in the experiments, we used lists of keywords of different sizes. However, we only report the results taking into account 20 keywords.[5]

A. Implication of the parameter ϵ In this section, we present the results obtained in the setting of the parameter ϵ. This is the parameter that our method uses to update the weights of the word pairs in the graph. Since one of our objectives is to discriminate between offensive and non-offensive tweets to select the keywords, this parameter is relevant in the model. Table 2 illustrates the results for different values of ϵ in IR. Although there are no relevant differences among the results, 0.1 seemed to be an appropriate value considering F@50. That is, we obtained more suitable weighted graphs with this value of ϵ. From these graphs we obtained keywords that allowed us to retrieve offensive tweets from datasets with 0.550 of F@50 in OFF19 and 0.732 in OFF20.

B. Implication of the parameters related to the attention mechanism We also evaluated different parameters regarding the multi-head self-attention mechanism of BERT that we used in our method. In each cas, we fixed ϵ to 0.1.

[1] We also use this model without concatenating the vectors with the keyword information. We refer to this model as the base model in this paper.

[2] https://tfhub.dev/google/bert_uncased_L-12_H-768_A-12/1

[3] https://www.nltk.org/

[4] https://github.com/LIAAD/yake

[5] Similar results are obtained with other numbers of keywords.

Table 2 IR results for different values of ϵ

	OFF20-OFF19		OFF19-OFF20	
ϵ	P@50	F@50	P@50	F@50
0.01	0.640	0.464	0.518	0.675
0.1	0.620	0.550	0.527	0.732
0.5	0.633	0.458	0.514	0.672
1.0	0.627	0.456	0.513	0.670

Each column corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation

First, we compared the results by varying the layer of BERT from which we leverage the attention mechanism. Table 3 shows the results for different layers, specifically for the two last layers and the two first ones. We could see that using a specific BERT layer to obtain the attention values (weights of word pairs) does not seem to be significant. However, the last layer seems to be better. That is why we used the layer #12 to study other parameters.

Then, we evaluated different heads into the attention mechanism of the layer #12. That is, we obtained the weights of the word pairs for a specific head in that layer, and compared the results with our variant of taking the combination of all the heads. Table 4 shows the comparison for the second and penultimate layers. We obtained worse results with the first and last layers. They seemed to be null patterns. In general we noticed that one of the heads obtained better results than the others, when they are used individually. It can be for the type of pattern represented in each particular head. The 11th head was the best in the experiments for both datasets. However, our proposal uses the combination of all the heads which obtained slightly better results according to F@50.

Moreover, we analyzed the centrality measure (CM) used to look for relevant vertices in the graph of words. Table 5 shows the comparison among the use of EC and other alternatives based on degree of vertices. As we expected, the best results are obtained with the keywords obtained by using EC which takes into account the weight of edges in the graph.

Table 3 IR results when varying the attention layer

	Off20-OFF19		OFF20-Off19	
Layer	P@50	F@50	P@50	F@50
12	0.620	0.550	0.527	0.732
11	0.637	0.468	0.527	0.684
2	0.630	0.460	0.521	0.675
1	0.637	0.468	0.516	0.676

Each column corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation

Table 4 IR results for different heads in the layer #12

	Off20-OFF19		Off19-OFF20	
Head	P@50	F@50	P@50	F@50
All	0.620	0.550	0.527	0.732
2	0.593	0.437	0.505	0.614
11	0.415	0.537	0.527	0.684

Each column corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation

C. Comparison with other keyword extraction methods We fixed the parameters of our method according to the previous results. That is, we used the combination of all the heads in the attention mechanism of the layer #12 of BERT. Moreover, we set ϵ to 0.1 and used the EC for extracting the keywords from the graph. With this configuration we compare our results with other keyword extraction methods.

On the one hand, Table 6 illustrates the results in IR. As we expected, the results with our method are higher than those obtained with other methods that have shown good performance in general purpose keyword extraction.

On the other hand, Table 7 shows the results in OLD. Once again, our proposal outperformed other general purpose keyword extraction methods. However, it is important to consider that the keyword-based models might lead to skewed results according to the bias in the datasets. Therefore, we analyze this problem later, where we compare the results with those obtained with a method that is not based on keywords.

5.2 Bias analysis

Along with the cross-validation, we included both evaluation OFF19-OFF19 and OFF20-OFF20 (in-validation). The idea is to illustrate how the extracted keywords can reveal

Table 5 IR results for different centrality measures (CM)

	Off20-OFF19		Off19-OFF20	
CM	P@50	F@50	P@50	F@50
EC	0.620	0.550	0.527	0.732
DC	0.607	0.442	0.514	0.662
IC	0.600	0.438	0.512	0.672
OC	0.597	0.436	0.516	0.677

Each column corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation

Table 6 F@50 in IR

Approach	Off20-OFF19	Off19-OFF20
Our	0.550	0.732
TF	0.399	0.581
TFIDF	0.490	0.605
RAKE	0.541	0.440
YAKE	0.325	0.483
TRANK	0.382	0.495

Each column corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation

the bias in the datasets. The three last rows in Table 8 show the results when we used OFF20 in the evaluation. Among these three rows, the first one corresponds to the results obtained with a model based on the keywords extracted from OFF19. The second row corresponds to the results obtained with a model based on the keywords extracted from OFF20, while the third row shows the results obtained without considering keywords.

First, we can see that the use of keywords can improve slightly the results. However, it depends on the characteristics of the keyword list. The base method (LSTM) obtained 0.7958 and this results increased to 0.8071 when the keywords from OFF20 were added. On the other hand, the results were considerably on decline when the keywords from OFF19 were added instead. This makes sense, since the first keyword list is from the same domain of the dataset used for the evaluation. Nevertheless, let us analyze the results in the evaluation with OFF19.

The three rows corresponding to the evaluation with OFF19 show a different performance. In this case, the base method obtained 0.5864 and the inclusion of keywords did not improve it, neither the keywords from OFF20 nor the keywords from OFF19. It can be explained by the bias in OFF19 that affects not only the performance when a keyword-based method is used, but also the list of keywords extracted from this dataset. I.e. the keywords

Table 7 F1-scores in OLD

Approach	Off20-OFF19	Off19-OFF20
Our	0.5687	0.5798
TF	0.3747	0.4108
TFIDF	0.5047	0.5588
RAKE	0.4707	0.4588
YAKE	0.5327	0.5548
TRANK	0.4287	0.4718

Each column corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation

Table 8 Cross-validation and in-validation in OLD

Evaluation with		F1-score
OFF19	OFF19-OFF19	0.5651
	OFF20-OFF19	0.5687
	*-OFF19	0.5864
OFF20	OFF19-OFF20	0.5798
	OFF20-OFF20	0.8071
	*-OFF20	0.7958

Each row corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation. A is * in the method that does not use keywords

are biased according to the characteristics of the dataset. Thus, conforming our results, the more skewed the dataset (from which the keywords are extracted), the worse the generalization of the keyword-based models (based on the extracted keywords).

6 Discussion

Regarding the use of the attention mechanism of BERT to weight the word pairs (RQ1), we first fine-tuned BERT with a set of texts for the offensive language detection task. Then, with the learned weights of BERT, we captured the attention each word assigns to other words in its context to estimate the weights of the corresponding pairs. In the experimentation we varied the parameters of the attention mechanism, i.e. the layer and heads. The variation does not seem to be significant. However, the experimental results suggest the use of the last layer of BERT and the combination of all the heads in the selected layer.

With respect to the distinction between the offensive and non-offensive texts (RQ2), we designed the method to update the weights between words according to the class of each tweet. That is, for each offensive text, the weight of each word pair updates in a positive sense the corresponding edge in the word graph, while for each non-offensive text, the update is in a negative sense. Thus, we penalize the words that can be relevant to the non-offensive texts. Moreover, we used a parameter ϵ to control the update. The higher the value of ϵ, the greater the increase or decrease of the weight of the word pairs. In the experiments, we varied the value of ϵ and saw that small variations in this parameter are not relevant. However, we verified the suitability of our proposal for keyword extraction by the IR and OLD tasks. Besides, we point out that the proposed method is unsupervised, in the sense that it does not require a dataset with a set of keywords as reference, instead it effectively only uses a set of offensive and non-offensive texts.

Table 9 List of keywords

Dataset	Keywords
OFF19	'trump', 'hate', 'gun', 'anti', 'mag', 'liberals', 'stupid', 'sick', 'black', 'government', 'violence', political', 'wrong', 'bad', 'election', 'violent', 'woman', 'conservatives', 'control', 'vote', 'stop', 'country', 'people', 'president', 'law', 'white'
OFF20	'bitch', 'hate', 'bad', 'ass', 'trump', 'girl', 'stop', 'black', 'last', 'someone', 'real', 'season', 'game', 'world', 'little', 'guess', 'school', 'hard', 'person', 'god', 'old', 'twitter', 'sad', 'fun', 'white', 'work'

Table 10 Percentage of misclassified tweets

Evaluation	% of tweets with keywords	% of tweets without keywords
OFF20-OFF19	28.8	74.2
*-OFF19	24.2	75.8
OFF19-OFF20	21.3	78.7
*-OFF20	19.7	80.3

Each row corresponds to an evaluation A-B: A is the dataset used to obtain the keywords and B is the dataset used in the evaluation. A is * in the method that does not use keywords

6.1 Error analysis

In order to gain deeper insight on our method performance, we conducted an error analysis. First, we manually analyzed some keywords extracted for each of the datasets. Then, we analyzed the presence of the keywords in the instances misclassified.

Table 9 illustrates examples of extracted keywords per dataset.[6] Some of them can be easily recognized as offensive words, like for example '*stupid*' in OFF19 and '*bitch*' in OFF20. However, others are non-offensive in a general sense. For instance, the word '*liberals*' was selected as an offensive keyword from OFF19, but we do not consider it as an offensive word. We checked on the original paper where the dataset was proposed and realized that '*liberals*' was one of the terms used to filter tweets. Nevertheless, other terms that were also used in the filtering of tweets as '*antifa*', were not extracted as keywords by our method. Therefore, we calculated the percentage of occurrence of the words used to collect the dataset, discriminating between offensive and non-offensive tweets. As we expected, these words are very frequent in the dataset. In the case of '*liberals*', the percentage of occurrence in offensive tweets is higher. Thus, errors can arise from those non-offensive words which are relevant only in the offensive tweets. On the other hand, errors can appear due to those non-offensive tweets that contain offensive words.

Furthermore, we conducted an error analysis on the experimental results in OLD. We observed that most of the errors were in tweets that do not contain keywords. That is, tweets which do not contain at least one of the keywords from the list we extracted. Table 10 illustrates a statistic related to the tweets misclassified with both the keyword-based models and the models which do not consider the keywords (base model). In each case, it is shown the percentage of misclassified tweets without keywords (last column) and the percentage of misclassified tweets with at least one keyword. With the keyword-based method, 74.24% of the errors came from tweets without keywords in OFF19, and 78.7% in OFF20. These percentages increased to 75.76% and 80.28% respectively, with the base models. Thus, the probability of error is higher in tweets which do not contain keywords.

Regarding OFF20-OFF19, 91.3% of the misclassified tweets that do not contain keywords (74.2% of the total of misclassified tweets), corresponds to offensive tweets. This amount represents 74.7% of the total of misclassified offensive tweets. This data suggests that a large part of the errors come from offensive tweets that do not contain offensive keywords. The rest 8.7% of misclassified tweets without keywords are non-offensive that represents the 58.5% of all the misclassified non-offensive tweets. Therefore, a large percentage (41.5%) of errors in non-offensive tweets is due to the presence of keywords. This suggests a possible bias in the dataset in relation to some keywords. In the case of OFF19-OFF20, all the misclassified tweets without keywords correspond to offensive tweets, that represent the 80% of the total of the misclassified offensive texts.

6.2 Limitations of our work

One limitation of our keyword extraction method is that it does not consider the tokens starting with ##. BERT uses this symbol to identify parts of unknown words in the tweets. We intend to include this information in coming works. Moreover, we attempt to extend the method for phrase extraction. The idea is to define some patterns to identify phrases in the tweets and extract those that contain closed keywords. One way to measure closeness among keywords is the sum of the weights of all the edges on the path between the words in the word graph.

Another limitation is the characteristic of the phenomenon that we aim to address. Since offensive language

[6] Some examples can represent offensive content. They are not the views of the authors.

can be expressed in a subtle manner, many offensive tweets may simply not have words that are considered offensive. Thus, our method can extract lists of words that do not generalize an offensive content. Moreover, our method depends on the distribution of words between the classes in the dataset. The words that are relevant in non-offensive tweets will not be extracted as keywords, even when they are offensive. Therefore, the quality of the extracted keywords is data-dependent. Anyway, this can be useful, because it helps us characterize the dataset used by our method.

7 Conclusion and future work

In this paper, we proposed an unsupervised method for extracting keywords from datasets with offensive content. The aim is to study the offensive language as a particular case of online harmful content. Our approach provides a way to extract keywords from datasets without the need of a tagged dataset with reference keywords. Instead the method only uses a set of tweets tagged as offensive or non-offensive. In this sense, the extracted keywords can be used to explain the offensive language within a dataset, since they are relevant words in the offensive tweets. An important contribution lies in the exploitation of BERT. We designed the method by leveraging the abilities of the multi-head self-attention mechanism of BERT to assign attention values among word pairs in a context. In the proposal, we calculate a weight for each word pair from the tweets as the attention value obtained with BERT for this pair. Then, the weight is updated when processing each tweet containing the pair. The proportion of the update of the weight is controlled by a parameter ϵ, and the weight increases if the processed tweet is offensive and decreases otherwise. Thus, we penalized the word pairs from non-offensive tweets for distinguishing between offensive and non-offensive tweets. Then, the weights are used to represent the edges of a graph where the vertices are the words from all the tweets. This representation finally allows to select the keywords by using the eigenvector centrality. We extrinsically evaluated the quality of the generated keyword list in two ways. On the one hand we tested an information retrieval system to extract offensive tweets taking the keywords as queries. On the other hand, we evaluated the performance of a model for offense detection as a classification task. Firstly, we made experiments to find a good configuration for the parameters of our method. Then, we evaluated the suitability of our method to extract keywords for our particular purpose over other general purpose AKE techniques. Moreover, we evaluated how our method can detect some characteristics in the datasets that can influence the performance of offenses detection. As future work we aim to expand our method for dealing with multilingual datasets.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This research work was partially funded by the Spanish Ministry of Science and Innovation under the research project MISMIS-FAKEnHATE on Misinformation and Miscommunication in social media: FAKE news and HATE speech (PGC2018-096212-B-C31). The authors thank also the EU-FEDER Comunitat Valenciana 2014–2020 grant IDIFEDER/2018/025.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Ao X, Yu X, Liu D, Tian H (2020) News keywords extraction algorithm based on textrank and classified TF-IDF. In: 2020 international wireless communications and mobile computing (IWCMC). IEEE, pp 1364–1369
2. Basile V, Bosco C, Fersini E, Debora N, Patti V, Pardo FMR, Rosso P, Sanguinetti M et al (2019) Semeval-2019 task 5: multilingual detection of hate speech against immigrants and women in twitter. In: 13th international workshop on semantic evaluation. Association for Computational Linguistics, pp 54–63
3. Berry MW, Kogan J (2010) Text mining: applications and theory. John Wiley & Sons, New York
4. Boudin F (2013) A comparison of centrality measures for graph-based keyphrase extraction. In: Proceedings of the sixth international joint conference on natural language processing, pp 834–838
5. Brin S, Page L (1998) The anatomy of a large-scale hypertextual Web search engine. In: Proceedings of the seventh international conference on World Wide Web, pp 107–117
6. Büttcher S, Clarke CL, Cormack GV (2016) Information retrieval: implementing and evaluating search engines. Mit Press, Cambridge
7. Casula C, Aprosio AP, Menini S, Tonelli S (2020) Fbk-dh at semeval-2020 task 12: using multi-channel bert for multilingual offensive language detection. In: Proceedings of the fourteenth workshop on semantic evaluation, pp 1539–1545
8. Chaudhari S, Polatkan G, Ramanath R, Mithal V (2019) An attentive survey of attention models. arXiv:1904.02874
9. Dai W, Yu T, Liu Z, Fung P (2020) Kungfupanda at semeval-2020 task 12: Bert-based multi-task learning for offensive language detection. arXiv:2004.13432
10. Devlin J, Chang MW, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805

11. Fersini E, Rosso P, Anzovino M (2018) Overview of the task on automatic misogyny identification at IberEval 2018. IberEval@ SEPLN 2150:214–228
12. Firoozeh N, Nazarenko A, Alizon F, Daille B (2020) Keyword extraction: issues and methods. Nat Lang Eng 26(3):259–291
13. Hasan KS, Ng V (2014) Automatic keyphrase extraction: a survey of the state of the art. In: Proceedings of the 52nd annual meeting of the association for computational linguistics (Volume 1: Long Papers), pp 1262–1273
14. Hu X, Wu B (2006) Automatic keyword extraction using linguistic features. In: Sixth IEEE international conference on data mining-workshops (ICDMW'06). IEEE, pp 19–23
15. Kathait SS, Tiwari S, Varshney A, Sharma A (2017) Unsupervised key-phrase extraction using noun phrases. Int J Comput Appl 162(1)
16. Kaur J, Gupta V (2010) Effective approaches for extraction of keywords. Int J Comput Sci Issues (IJCSI) 7(6):144
17. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv:1412.6980
18. Mandl T, Modha S, Majumder P, Patel D, Dave M, Mandlia C, Patel A (2019) Overview of the HASOC track at FIRE 2019: hate speech and offensive content identification in indo-european languages. In: Proceedings of the 11th forum for information retrieval evaluation, pp 14–17
19. Mihalcea R, Tarau P (2004) Textrank: bringing order into text. In: Proceedings of the 2004 conference on empirical methods in natural language processing, pp 404–411
20. Nasar Z, Jaffry SW, Malik MK (2019) Textual keyword extraction and summarization: state-of-the-art. Inf Process Manag 56(6):102088
21. Newman ME (2008) The mathematics of networks. New Palgrave Encycl Econ 2(2008):1–12
22. Pappagari R, Zelasko P, Villalba J, Carmiel Y, Dehak N (2019) Hierarchical transformers for long document classification. In: 2019 IEEE automatic speech recognition and understanding workshop (ASRU). IEEE, pp 838–844
23. De la Pena Sarracén GL, Rosso P (2020) Prhlt-upv at semeval-2020 task 12: Bert for multilingual offensive language detection. In: Proceedings of the fourteenth workshop on semantic evaluation, pp 1605–1614
24. Pitsilis GK, Ramampiaro H, Langseth H (2018) Detecting offensive language in tweets using deep learning. arXiv:1801.04433
25. Poletto F, Basile V, Sanguinetti M, Bosco C, Patti V (2020) Resources and benchmark corpora for hate speech detection: a systematic review. Lang Resour Eval pp 1–47
26. Robertson SE, Walker S, Jones S, Hancock-Beaulieu MM, Gatford M et al (1995) Okapi at trec-3. Nist Spec Publ 109:109
27. Rosenthal S, Atanasova P, Karadzhov G, Zampieri M, Nakov P (2020) A large-scale semi-supervised dataset for offensive language identification. arXiv:2004.14454
28. Sahrawat D, Mahata D, Kulkarni M, Zhang H, Gosangi R, Stent A, Sharma A, Kumar Y, Shah RR, Zimmermann R (2019) Keyphrase extraction from scholarly articles as sequence labeling using contextualized embeddings. arXiv:1910.08840
29. Uglow H, Zlocha M, Zmyślony S (2019) An exploration of state-of-the-art methods for offensive language detection. arXiv:1903.07445
30. Vashistha N, Zubiaga A (2020) Online multilingual hate speech detection: experimenting with Hindi and English social media
31. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. In: Advances in neural information processing systems, pp 5998–6008
32. Wang S, Liu J, Ouyang X, Sun Y (2020) Galileo at semeval-2020 task 12: multi-lingual learning for offensive language identification using pre-trained language models. arXiv:2010.03542
33. Wani AH, Molvi NS, Ashraf SI (2019) Detection of hate and offensive speech in text. In: International conference on intelligent human computer interaction. Springer, pp 87–93
34. Wiedemann G, Yimam SM, Biemann C (2020) Uhh-lt at semeval-2020 task 12: fine-tuning of pre-trained transformer networks for offensive language detection. In: Proceedings of the fourteenth workshop on semantic evaluation, pp 1638–1644
35. Wiegand M, Ruppenhofer J, Kleinbauer T (2019) Detection of abusive language: the problem of biased datasets. In: Proceedings of the 2019 conference of the North American Chapter of the Association for Computational Linguistics: human language technologies, volume 1 (long and short papers), pp 602–608
36. Witten IH, Paynter GW, Frank E, Gutwin C, Nevill-Manning CG (2005) KEA: practical automated keyphrase extraction. In: Design and usability of digital libraries: case studies in the asia pacific. IGI Global, pp 129–152
37. Zampieri M, Malmasi S, Nakov P, Rosenthal S, Farra N, Kumar R (2019) Predicting the type and target of offensive posts in social media. In: Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics (NAACL), pp 1415–1420
38. Zampieri M, Malmasi S, Nakov P, Rosenthal S, Farra N, Kumar R (2019) Predicting the type and target of offensive posts in social media. arXiv:1902.09666
39. Zampieri M, Malmasi S, Nakov P, Rosenthal S, Farra N, Kumar R (2019) Semeval-2019 task 6: identifying and categorizing offensive language in social media (offenseval). arXiv:1903.08983
40. Zampieri M, Nakov P, Rosenthal S, Atanasova P, Karadzhov G, Mubarak H, Derczynski L, Pitenis Z, Çöltekin Ç (2020) Semeval-2020 task 12: multilingual offensive language identification in social media (offenseval 2020). arXiv:2006.07235