Article

Improving Crisis Events Detection Using DistilBERT with Hunger Games Search Algorithm

Hadeer Adel ¹, Abdelghani Dahou ² , Alhassan Mabrouk ³ , Mohamed Abd Elaziz ^{4,5,6} , Mohammed Kayed ⁷ , Ibrahim Mahmoud El-Henawy ⁸, Samah Alshathri ^{9,*} and Abdelmgeid Amin Ali ¹⁰

¹ Department of Computer Science, Faculty of Computer Science, Nahda University, Beni Suef 62511, Egypt; hadeeradeleid@gmail.com

² Mathematics and Computer Science Department, University of Ahmed DRAIA, Adrar 01000, Algeria; dahou.abdghani@univ-adrar.edu.dz

³ Mathematics and Computer Science Department, Faculty of Science, Beni-Suef University, Beni Suef 62511, Egypt; alhassanmohamed@science.bsu.edu.eg

⁴ Faculty of Computer Science and Engineering, Galala University, Suez 435611, Egypt; abd_el_aziz_m@yahoo.com

⁵ Artificial Intelligence Research Center (AIRC), Ajman University, Ajman P.O. Box 346, United Arab Emirates

⁶ Department of Mathematics, Faculty of Science, Zagazig University, Zagazig 44519, Egypt

⁷ Computer Science Department, Faculty of Computers and Artificial Intelligence, Beni-Suef University, Beni Suef 62511, Egypt; mskayed@gmail.com

⁸ Department of Computer Science, Faculty of Computer Science, Zagazig University, Zagazig 44519, Egypt; henawy2000@yahoo.com

⁹ Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

¹⁰ Faculty of Computer Science and Information, Minia University, Minia 61519, Egypt; abdelmgid@yahoo.com

* Correspondence: sealshathry@pnu.edu.sa

Citation: Adel, H.; Dahou, A.; Mabrouk, A.; Abd Elaziz, M.; Kayed, M.; El-Henawy, I.M.; Alshathri, S.; Amin Ali, A. Improving Crisis Events Detection Using DistilBERT with Hunger Games Search Algorithm. *Mathematics* **2022**, *10*, 447. <https://doi.org/10.3390/math10030447>

Academic Editor: Amir Mosavi

Received: 5 January 2022

Accepted: 27 January 2022

Published: 30 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Abstract: This paper presents an alternative event detection model based on the integration between the DistilBERT and a new meta-heuristic technique named the Hunger Games Search (HGS). The DistilBERT aims to extract features from the text dataset, while a binary version of HGS is developed as a feature selection (FS) approach, which aims to remove the irrelevant features from those extracted. To assess the developed model, a set of experiments are conducted using a set of real-world datasets. In addition, we compared the binary HGS with a set of well-known FS algorithms, as well as the state-of-the-art event detection models. The comparison results show that the proposed model is superior to other methods in terms of performance measures.

Keywords: event detection; deep learning; hunger game search; DistilBERT; feature selection optimization algorithms

1. Introduction

In the past decade, as social media grows in popularity and the number of users grows, more and more studies are considering the exploitation of the crowdsourced events [1,2]. When a crisis occurs, individuals often use social media to express their concerns and expectations about specific agents, such as events, persons, targets, or policy proposals. These posts allow event organizers to know what is going on around them and to be aware of this fact in a fast and effective manner [3]. Social media demonstrated its usefulness as a valuable spacial information source during many recent crisis events, such as the transmission of infectious disease, volcanic eruptions, tropical storms, tornadoes, earthquakes, river flooding, forest fires, and nuclear accidents awareness and disaster monitoring.

Event detection (ED) is a crucial task for extracting information to discover event trigger points (words or phrases that elicit events in text) and identify event types. A crucial part of ED utilizing social media is detecting and describing crisis-related events when the type of event of concern is unknown ahead of time [4]. Although there is much content on social networks, relatively little of it is related to crisis events and offers

meaningful information. Informative material is frequently overshadowed by unrelated and needless noise in most social media postings. Some prior study aims to obtain effective content utilizing text classifiers to process and translate these large amounts of social media data into corporate data [5]. Previous work specifically on event detection was centered on developing domain text classifiers [6]. Numerous recent studies looked at natural disasters such as floods and storms, as well as man-made disasters such as acts of terrorism and bombs [7,8]. These studies concentrate on binary classifications for various crisis characteristics, such as determining source type, forecasting tweet-crisis relation, and determining information quality and relevance. Many such research [9,10], on the other hand, presented multi-classifiers for affected people, infrastructural facilities, deaths, donors, warn, and guidance. Moreover, the recognition of crisis forms, such as typhoons, floods, and flames, is also carried out [11]. In recent years, deep learning (DL)-based models have been frequently used to carry out the tasks noted above.

A DL structure consists of multiple layers, each one of which relates to a different aspect of the brain. Thus, every layer has its own set of neurons and outcomes and its own set of input and activation functions. While there is a trade-off between generalization and high computation, before actually selecting the method and training weights, the manner of the feature map is explained and the parameter values define all impact the performance of DL techniques. Due to its importance, the DL techniques have recently demonstrated excellent performance in several fields, such as the Internet of Things [12], sentiment analysis [13], and toxicity classification on social media [14]. Moreover, several papers proposed DL models for crisis-related knowledge classification and detection. For example, utilizing domain-specific and GloVe embeddings, Alrashdi and O’Keefe [15] looked into two DL structures: Bidirectional Long Short-Term Memory (BiLSTM) and CNN. In addition, a similar CNN method was suggested which uses Twitter posts to identify disaster-related events [16]. The DL architectures, particularly recurrent neural networks (RNNs), have recently gained popularity for detecting crisis events due to their ability to represent sequence data [17,18].

Nevertheless, there are a few drawbacks to just using RNN for data modeling. Initially, while RNN could indeed grab sequential data using the recurrence strategy, it could not encode metadata both from the left and proper contexts within every occurrence in a sequential manner. When detecting targeted activity relying on disaster event texts, it is critical to notice the complete contextual information rather than just the data from previous stages. Following that, existing RNN-based event detection methods are utilized to understand styles in common patterns by predicting every following event document derived from previous crisis texts. This learning goal is primarily concerned with acquiring the significant relation between crisis event texts in standard sequence data. The RNN model cannot accurately estimate the following sample based on the previous event message when a check sequence’s connection is broken. Finally, the sequence will then be classified as anomalous. Moreover, utilizing only the forecasting with the following event text as the objective function does not allow for the precise encoding of the patterns distributed by all regular patterns.

Technological improvements in language pre-training have markedly increased the level of a variety of Natural Language Processing (NLP) jobs, with notable fine-tuning concepts including such BERT, ALBERT, XLNet, RoBERTa, and DistilBERT [19]. Due to the wide availability and improved user experience among those techniques, multiple structures relying on fine-tuning have emerged. In this paper, we use a Distilled Bidirectional Encoder Representations from Transformers (DistilBERT) [19], which surpassed many methods, to overcome the drawbacks of RNN-based methods shown above [11]. For event detection datasets, which have huge feature sizes, these transformer models suffer from local optimality difficulties due to the big solution space. To overcome this problem, feature selection (FS) used in classification tasks which reduces noisy features in order to optimize the method performance. Recently, researchers used metaheuristic optimization algorithms to select the irrelevant features to reduce the high-dimensional datasets, as in [20,21]. Therefore, we used the recent metaheuristic algorithm, Hunger

Games Search (HGS) [22], in our approach. The reasons for employing HGS approaches to optimize the FS challenge in this paper are as follows: we want to examine the most recent HGS optimizer, and when the HGS method is compared to complex, modern, and high-efficiency algorithms, it is revealed that the HGS optimizer has the optimal solution for the problems examined, with typically greater classification performance (i.e., fewer iterations and execution time).

This paper focuses on identifying crisis events introduced or discussed on social media (such as Twitter and Wikipedia) to identify events. As there is so many data to consider, the only way to analyze it would be to use the DistilBERT method in automated extracting features effectively. Next, the Hunger Games Search (HGS) optimization approach has been used for selecting features because it is a crucial stage due to the “curse of dimensionality” challenge. There have been no studies using HGS for event detection problems as far as we are aware. Finally, for the classification, Support Vector Machine (SVM) and K-Nearest Neighbor (KNN) are the most frequently used in FS literature [23,24] for the following reasons: The SVM is used to tackle binary problems and seeks to optimize the margin between positive and negative classes around a hyperplane. As a result, the best hyperplane with the greatest distance to the closest training location of any category is obtained, resulting in acceptable class discrimination. Furthermore, one of the most widely used Machine Learning (ML) and pattern categorization approaches is KNN. It is widely used due to its simplicity and ease of implementation compared to more complex supervised ML approaches. The proposed model’s contributions can be summarized as follows:

- A transformer-based model named DistilBERT has been used to learn and automatically extract meaningful and complex text representation from the input data.
- The pre-trained DistilBERT has been fine-tuned on crisis detection data with the aim of maximizing the detection accuracy and perform feature extraction.
- A feature selection algorithm named binary Hunger Games Search (HGS) was combined with DistilBERT in a framework to perform feature selection and dimensionality reduction on the extracted features from DistilBERT.
- The overall proposed framework has been evaluated on various real-world datasets to assess the crisis detection accuracy and compare the performance of the framework to several state-of-the-art techniques.

The remainder of this paper is structured as follows. Section 2 describes a review of recent work on detecting crisis events. Section 4.2 offers a description of the technique. Experimental results and the case study are reported and discussed in Section 4, followed by the conclusion in the last section.

2. Related Works

Several articles have reported how to retrieve beneficial information from websites in the event of a disaster [25–27]. These techniques are mainly based on the characteristics used to identify crisis-related information. Rudra et al. [25] classified posts on Twitter into contextual and non-situational information using lexicon characteristics such as reduced lexical and grammatical attributes. They tested their methodologies on various disaster datasets and matched them to the Bag-of-Words (BOW) approach for English and Hindi twitter posts. For accurate classification of eyewitnesses, Zahra et al. [26] applied linguistic features and context features to the classification method. They divided eyewitnesses into three types: direct independent witnesses, indirect witness accounts, and susceptible direct eye witness accounts. They conducted experiments with quakes, flood, storm, and wildfire datasets. In order to detect emergency tweets, Kejriwal and Zhou [27] utilized low-supervision and transfer learning-based methods. They conducted experiments with datasets from the earthquakes in Kerala, Macedonia, and Nepal. They demonstrated that their technique is useful, particularly when the labeled data is limited, and that it surpasses existing baseline techniques. Every one of these studies, even so, starts concentrating on disaster-related Twitter posts rather than sub-categories of tweets such as the destruction of infrastructure, human destruction, source of information need, and availability. Little

research has concentrated on classifying tweets throughout a crisis [28,29]. Madichetty and Sridevi [30] concentrated on a design to detect the accessibility and scarcity of resources. They have been using the re-ranking feature selection method to extract information from Twitter posts and send the yield of the optimization technique to the classification model to identify the accessibility and necessity of resources. For trying to extract resource requirements and availability from crisis Twitter posts, Basu et al. [31] had been using data gathering methods, individual word encoding, and a mixture of term and character word embedding. They utilized earthquake risk datasets from Italy and Nepal. Dutt et al. [32] discovered a method for deciphering the semantic meaning of need on the accessibility tweets, such as what resource is accessible or required and where the source of information is located. They also devised a method for correlating knowledge posts on Twitter accessibility and demand regarding resource correlation and position. Alam et al. [33] looked at tweets from three hurricanes (such as Harvey, Irma, and Maria) to see what people were saying. They used a Random Forest (RF) depending on the bag-of-words method for multi-class categorization in their trials. It has categories such as affected individuals, infrastructural facilities, and utility injury, warnings, and guidance. They demonstrated that perhaps the tweet's content and photo provide additional data. Purohit et al. [29] created a social-EOC model to determine and rate community service demands shared on social media. They reduced duplication by combining related applications and saved time by using the linguistic clustering strategy. They utilized data from the Alberta floods, Hurricane Sandy, Hurricane Harvey, and the earthquake in Nepal, among other things. Madichetty and Sridevi [28] developed a new qualified majority adaptable framework to describe healthcare resource twitter posts during an emergency. Relying on the relevant feature directly applicable to the medicine resource posts on Twitter, they used classification techniques such as support vector machine (SVM), AdaBoost, Random Forest, and gradient boosting methods. Even so, throughout an emergency, they did not depend on final inspection from social media. However, these approaches were still unable to achieve high degree of efficiency. Thus, the researchers recently used transformers, which are reported to perform well in understanding and inference tasks.

Transformers have unquestionably been the most popular type of Natural Language Processing (NLP) system in recent years [34]. BERT [35], the movement's "golden child", was the first method to relate a transformer's bi-directional learning to a language modeling assignment. BERT is prepared with such a masked language modeling purpose: nonsense syllables in the context of the sentence have been modified by a [MASK] token, and the method tries to forecast the masked token dependent on the circumstances. To follow BERT's accomplishment, many comparable models in bio-medicine NLP have been proposed, each proposing a different version of in-domain learning such as using a various corpus, as introduced in [36]. SpanBERT [37] is another BERT for action recognition that could be helpful to put to the librarian of BERT. Instead of single words, irregular adjacent spans of tokens are masked throughout SpanBERT's learning, trying to force the method to forecast the entire span from tokens at its edges. Recently, Liu et al. [11] proposed CrisisBERT, a transformer that depends on categorization method that outperformed traditional linear and DL techniques in terms of stability and performance. The CrisisBERT method was introduced for crisis detection and recognition disaster tasks. However, in the recent approaches, fine-tuning the pre-trained transformer for classification tasks produced irrelevant features [38] which reduce the model performance. To overcome this problem, a transformer is integrated with meta-heuristic optimization to create our methodology in order to improve performance.

3. Proposed Method

3.1. Distilled BERT for Feature Extraction

The architecture of the proposed feature extraction model based on DistilBERT is shown in Figure 1. DistilBERT receives as an input X which represents a tweet from the dataset (word sequence). The inputted sequences to DistilBERT are converted to a set of embedding vectors where each vector is mapped to each word in a sequence (S_1).

DistilBERT uses the transformer encoder to learn the contextual information for each word. The transformer encoder uses a self-attention mechanism to generate the contextual embeddings (S2). The extracted contextual embeddings for each word are concatenated into a single vector to represent the semantic information presented in the tweet (S3). S3 is the input of a fully connected layer that outputs a vector of size d where d is the number of neurons. Later, a classification layer is placed at the end of the feature extractor model to fine-tune the pre-trained DistilBERT on the event detection task and predict the corresponding event class for each inputted sequence (tweet). In what follows, we detail the model fine-tuning and feature extraction processes.

Figure 1. The proposed feature extraction model.

3.1.1. Lexicon Encoder

Each tweet is represented with a set of tokens (words) of s -length vector. Thus, the input $X = x_1, \dots, x_s$ will be fed to a multi-layered RNN to map each token to its corresponding embedding vectors representing the word, segment, and positional embeddings. Based on the encoding proposed by Devlin et al. [35], the special token [CLS] is placed as the first token in the sequence (x_1), whereas the [SEP] token is placed at the end of the sequence. To generate the embedding vectors for X , the lexicon encoder sums up the word, segment, and positional embeddings for each token in X .

3.1.2. Transformer Encoder

Using DistilBERT, the representation is learned via pre-training. We employ a pre-trained multilayer bidirectional transformer encoder to map the input vectors (S1) into contextual embedding vectors, one for each word. DistilBERT uses knowledge distillation to minimize the BERT base model (bert-base-uncased) parameters by 40%, making the inference 60% faster as shown in Figure 2. The main idea of distillation is to approximate the full output distributions of the BERT model using a smaller model such as DistilBERT. Thus, the number of transformer layers (encoders) in the BERT base (12 layers) has been reduced to six. The pre-trained model contains 66 million trainable parameters, compared

to the BERT base model with 110 million parameters. In terms of training time, DistilBERT was trained in 3.5 GPU ($8 \times V100$) days compared to 12 GPU days ($8 \times V100$) for the BERT base. The DistilBERT is trained on 16 GB of data collected from Toronto books corpus and English Wikipedia (same as BERT base training data). During the training process of DistilBERT, a large batch size (400) with gradient accumulation is used where the accumulation is performed locally using the gradients from multiple mini-batches before updating the parameters in each step. In addition, next sentence prediction (NSP) and segment embeddings learning objectives are omitted in the training process. The static masking used in the BERT base model is replaced by dynamic masking applied during inference.

Figure 2. The DistilBERT model architecture and components.

3.1.3. Fine-Tuning on Event Detection Task

Assuming that S_3 is the contextual embedding learned by the token [CLS], which serves as the semantic representation of input tweet X . The task is formulated as a multi-class classification problem. Thus, the probability of X being classified as class c (i.e., the event) is predicted as the Softmax function used in Equation (1).

$$P_r(c|X) = \text{Softmax}(W^T \cdot X) \quad (1)$$

where W is the weight matrix learned during the fine-tuning of the pre-trained model used during the initialization of the feature extractor model. r is the number of classes. It is worth noting that the first five transformer layers in the pre-trained model are not trainable. We only fine-tuned the last transformer layer (encoder) of the pre-trained model and replaced the classification layer with two fully-connected layers for feature extraction and classification, respectively.

3.1.4. Feature Extraction Layer

As mentioned in the previous paragraph, a fully connected layer is placed on top of the DistilBERT pre-trained model, which will serve as our feature extraction point rather than retrieving a large vector of size 768. The generated output vector S_3 from the last transformer layer, DistilBERT, will be fed the fully connected layer out of size 128 to reduce the feature space dimensionality and later inputted to the classification layer. At this stage, an activation function of type GELU [39] is used with the fully connected layer followed by a Dropout regularizer to prevent over-fitting. The GELU activation function is defined as follows:

$$\text{GELU}(m) = x * \Phi(m) \quad (2)$$

where m is the output of the fully-connected layer and $\Phi(m)$ represents the Cumulative Distribution Function for Gaussian Distribution.

3.2. Hunger Games Search

Yang et al. [40] suggested the Hunger Games Search (HGS) algorithm (Algorithm 1) as an optimization approach for modeling animal behavior and hunger. Hunger's ability to become one of the most important homeostatic reasons for decisions, behaviors, and actions in the animal's existence characterizes HGS. HGS mathematical modeling begins with a population of N solutions, X , and proceeds to the objective function values for solutions, Fit_i . The following equations are used to accomplish the modernization phase:

$$X = \begin{cases} X(t) \times (1 + rand), & r_1 < l \\ W_1 \times X_b + R \times W_2 \times |X_b - X(t)|, & r_1 > l, r_2 > E \\ W_1 \times X_b - R \times W_2 \times |X_b - X(t)|, & r_1 > l, r_2 < E \end{cases} \quad (3)$$

r_1 and r_2 are arbitrary numbers, and the variable $rand$ generates numbers from a normal distribution, and R is a variable whose value is determined by the interval $[-a, a]$ and can depend on the number of iterations as follows:

$$R = 2 \times s \times rand - s, \quad s = 2 \times \left(1 - \frac{t}{T}\right) \quad (4)$$

While, the parameter E , in Equation (3), denotes the control parameter that is defined as:

$$E = \text{sech}(|Fit_i - Fit_b|) \quad (5)$$

Fit_b represents the finest value of the objective function and Sech corresponds to the hyperbolic function where $\text{sech}(x) = \frac{2}{e^x + e^{-x}}$.

Furthermore, W_1 and W_2 represent the hunger weights given in Equations (6) and (7).

$$W_1 = \begin{cases} H_i \times \frac{N}{SH} \times r_4, & r_3 < l \\ 1, & r_3 > l \end{cases} \quad (6)$$

$$W_2 = 2 \left(1 - e^{(-|H_i - SH|)}\right) \times r_5 \quad (7)$$

r_3 , r_4 , and r_5 represent random numbers whose values are in the interval $[0, 1]$, and the variable SH corresponds to the solution of the hunger feeling summation given as follows:

$$SH = \sum_i H_i \quad (8)$$

Furthermore, the variable H_i corresponds the solution hunger H_i given by:

$$H_i = \begin{cases} 0, & Fit_i = Fit_b \\ H_i + H_n, & otherwise \end{cases} \quad (9)$$

The best value for the objective is supplied by Fit_b , and the current solution X_i has an objective given by Fit_i , and the new hunger is given by the variable H_n :

$$H_n = \begin{cases} LH \times (1 + r), & TH < LH \\ TH, & otherwise \end{cases} \quad (10)$$

$$TH = 2 \frac{Fit_i - Fit_b}{Fit_w - Fit_b} \times r_6 \times (UB - LB) \quad (11)$$

Fit_w gives a lower value to the objective function, and $r_6 \in [0, 1]$ is a random variable that can indicate if hunger has positive or harmful effects depending on numerous aspects.

Algorithm 1 Steps of HGS [40]

```

1: Start with the iterations number defined by  $T$ , the solutions number defined by  $N$ 
2: initialize the position of solutions  $X$ .
3: while  $t \leq T$  do
4:   Compute the objective value for the solutions  $X_i$ .
5:   Identify the finest solution  $X_b, Fit_b, Fit_W$ 
6:   Enhance  $H_i$  using Equation (9)
7:   Iterate and update  $W_1$  and  $W_2$  based on Equations (6) and (7), respectively.
8:   for  $doi = 1 : N$ 
9:     Modernize  $R$  utilizing Equation (4)
10:    Modernize  $E$  utilizing Equation (5)
11:    Modernize  $X_i$  utilizing Equation (3)
12:    $t = t + 1$ 
13: Return  $X_b$ .

```

3.3. Proposed Framework

When utilizing approaches for extracting features, such as DistilBERT, the obtained features were not given directly to the classification stage because they required additional computing time to run. Feature Selection (FS) algorithms remove unnecessary or superfluous features from an extracted crisis text as a data reduction technique. This means the FS approach reduces the amount of data transmitted. Hence, the effective feature selection mechanism was adopted, in which the majority of essential features are determined to use the optimization method, i.e., Hunger Game Search (HGS).

In general, the extracted features are divided into training and testing sets, where the training set are used to learn the model to detect the relevant features. The steps of the binary HGS as FS approach are presented in this section. Figure 3 depicts the general steps of the developed FS approach, dubbed HGS. The developed HGS's first phase is to build a set of N agents X that reflect the FS problem's solutions. The following equation is used to perform that method:

$$X_i = rand * (U - L) + L, i = 1, 2, \dots, N, j = 1, 2, \dots, Dim \quad (12)$$

Dim is the dimension of the provided problem in Equation (12) (i.e., the number of features). The random search limitations are U and L . The next step is to find the Boolean version of each X_i , which may be achieved with the formula below:

$$BX_{ij} = \begin{cases} 1 & \text{if } X_{ij} > 0.5 \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

The fitness value of each X_i is then calculated using the objective function below, based on the binary BX_i and the classification error.

$$Fit_i = \lambda \times \gamma_i + (1 - \lambda) \times \left(\frac{|BX_i|}{Dim} \right), \quad (14)$$

where $\left(\frac{|BX_i|}{Dim} \right)$ represents the ratio of determined relevant features. The classification error of training using KNN with $K = 5$ is denoted by γ_i . KNN is commonly used because it is more stable than other classifiers and has fewer parameters, while λ is a parameter used to balance the ratio of selected features and classification error.

Figure 3. Flowchart of the proposed methodology.

The best option X_b with the smallest fitness value Fit_b is then determined. The next step is to update the solution X_i , which is achieved with the HGS operators defined in Equations (9)–(3).

Following that, the stop conditions are checked, and if they are matched, the best solution is returned. Otherwise, the steps for upgrading are repeated.

The last step is to reduce the testing set based on the best solution and then evaluate the performance of the output using different measures.

4. Experiments and Results

In this section, we show and analyze a variety of experimental tests designed to evaluate the performance of our proposed technique. Section 4.1 provides a full explanation of the datasets used in our research. The metrics used to evaluate the performance of our HGS algorithm and other scheduling methods in the trials are explained in Section 4.2. Section 4.3 concludes by summarizing the results achieved and making some final observations.

4.1. Datasets

To implement crisis classification tasks and validate the proposed framework, three datasets of labeled crisis-related tweets are used, including C6 [41], C36 [11] (combination of C6, C8 [42], and C26, as shown in Tables 1–2), and MAVEN, which present an event schema. It is worth mentioning that this is the first time that the MAVEN dataset has been used and validated for crisis event classification based on the sentence level.

The statistics of the used datasets are presented in Table 1. The C6 and C36 datasets are divided into 95% training and 5% test sets.

Table 1. Dataset description.

Dataset	# Classes	# Samples	# Training	# Testing
C6	6	60 k	29,458	3004
C36	36	91.6 k	57,811	3043
MAVEN	90	4480	3623	857

Table 2. C6, C8, and C26 classes distributions.

S#	Classes	Dataset	# Labelled Tweets per Class
1	2013_Boston_crisis	C6	5648
		C26	929
2	2013_Queensland_Floods	C6	5414
		C26	919
3	2013_West_Texas_Explosion	C6	5246
		C26	911
4	2012_Sandy_Hurricane	C6	6138
5	2013_Alberta_Floods	C6	5189
		C26	986
6	2013_Oklahoma_Tornado	C6	4827
7	Ferguson	C8	1198
8	Sydneysiege	C8	1164
9	Charliehebd	C8	1163
10	2013_Russia_meteor	C26	1133
11	2013_NY_train_crash	C26	999
12	2013_Spain_train_crash	C26	991
13	2013_Bohol_earthquake	C26	969
14	2013_Lac_Megantic_train_crash	C26	966
15	2012_Colorado_wildfires	C26	953
16	2013_Brazil_fire	C26	952
17	2013_Australia_bushfire	C26	949
18	2012_Italy_earthquakes	C26	940
19	2013_Typhoon_Yolanda	C26	940
20	2012_Guatemala_earthquake	C26	940
21	2012_Venezuela_refinery	C26	939
22	2013_Singapore_haze	C26	933
23	2013_Sardinia_floods	C26	926
24	2013_Colorado_floods	C26	925
25	2013_Manila_floods	C26	920
26	2013_Glasgow_helicopter_crash	C26	918
27	2013_LA_airport_crisis	C26	912
28	2013_Savar_building_collapse	C26	911
29	2012_Costa_Rica_earthquake	C26	909
30	2012_Typhoon_Pablo	C26	907
31	2012_Philippines_floods	C26	906
32	ottawa_crisis	C8	786
33	germanwings-crash	C8	286
34	prince-toronto	C8	100
35	putinmissing	C8	64
36	ebola-essien	C8	34

4.2. Performance Measures

To detect events, the extraction classifier pair is evaluated using confusion matrices. The confusion matrix is shown in Figure 4. True Positives (TP) and False Negatives (FN) represent the number of events of a specific class that were correctly classified and wrongly classified, respectively, in the analysis.

True Negatives (TN) are the number of events defined as not belonging to a specific class. False Positive (FP) levels are the number of events wrongly categorized as belonging to a specific category.

In the formula for accuracy Equation (15), precision Equation (16), recall Equation (17), and F1-Score Equation (18), the parameters TP, FN, TN, and FP are defined.

		Prediction	
Label		TP	FN
		FP	TN

Figure 4. The structure of the Confusion Matrix.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (15)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (16)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (17)$$

$$\text{F1-Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (18)$$

4.3. Results and Discussion

This section introduces the proposed HGS task scheduling strategy's outcome analysis and discussion of experimental results.

4.3.1. Comparison with FS Methods

We assessed the HGS method against five well-known algorithms to objectively assess its effectiveness, namely, Particle Swarm Optimization (PSO) [43], Multi-Verse Optimizer (MVO) [44], Whale Optimization Algorithm (WOA) [45], Firefly Algorithm (FFA) [46], and Bat Algorithm (BAT) [47]. Table 3 outlines the different parameters that remain in each algorithm.

Table 3. Parameters settings of FS algorithms.

Optimizer	Parameters
PSO	WMax = 0.9, VMax = 6, WMin = 0.2
MVO	WEPMIn = 0.2, WEPMMax = 1
WOA	a1 = 2 to 0, a2 = −1 to −2
FFA	BetaMin = 0.2, Alpha = 0.5, Gamma = 1
BAT	QMax = 2, QMin = 0
HGS	VC2 = 0.03; shrink ∈ [2, 0]

The performance of each algorithm was evaluated in terms of Recall, Precision, F1 Score, and Accuracy to evaluate the algorithms. The four metrics of KNN and SVM classifiers with C6, C36, and MAVEN are depicted in Tables 4–6, respectively. The best accuracy results are highlighted in bold. According to the results in these tables, the HGS

using SVM as the classifier typically outperforms the other studied scheduling algorithms, including PSO, MVO, WOA, FFA, and BAT.

Table 4. Results of each algorithms on C6 dataset.

Algorithm	Model	Recall	Precision	F1 Score	Accuracy
PSO	KNN	0.9796	0.9798	0.9796	0.9796
	SVM	0.9796	0.9799	0.9796	0.9796
MVO	KNN	0.9796	0.9797	0.9796	0.9796
	SVM	0.9800	0.9802	0.9800	0.9800
WOA	KNN	0.9800	0.9801	0.9800	0.9800
	SVM	0.9800	0.9802	0.9800	0.9800
FFA	KNN	0.9796	0.9797	0.9796	0.9796
	SVM	0.9803	0.9805	0.9803	0.9803
BAT	KNN	0.9793	0.9794	0.9793	0.9793
	SVM	0.9803	0.9806	0.9803	0.9803
HGS	KNN	0.9800	0.9801	0.9800	0.9800
	SVM	0.9806	0.9809	0.9806	0.9806

Table 5. Results of each algorithms on C36 dataset.

Algorithm	Model	Recall	Precision	F1 Score	Accuracy
PSO	KNN	0.9690	0.9693	0.9689	0.9690
	SVM	0.9681	0.9687	0.9681	0.9681
MVO	KNN	0.9694	0.9697	0.9692	0.9694
	SVM	0.9694	0.9697	0.9693	0.9694
WOA	KNN	0.9683	0.9688	0.9682	0.9683
	SVM	0.9694	0.9701	0.9694	0.9694
FFA	KNN	0.9685	0.9688	0.9684	0.9685
	SVM	0.9710	0.9713	0.9709	0.9710
BAT	KNN	0.9683	0.9686	0.9681	0.9683
	SVM	0.9690	0.9692	0.9689	0.9690
HGS	KNN	0.9679	0.9684	0.9677	0.9679
	SVM	0.9710	0.9714	0.9710	0.9710

Analyzing Table 4, it is clear that the HGS method plays a critical role in feature selection when employing an SVM classifier, as the results are still successful; this is evident across all metrics. The best results occur when using the SVM classifier; HGS can classify 98.06% of the test samples on accuracy metric, more than PSO, MVO, WOA, FFA, and BAT results. In detail, the HGS scored 98.06%, followed by the BAT and FFA in the second level, with 98.03%. WOA was equal in accuracy result with MVO, with 98%. Lastly, the PSO had the worst outcome (i.e., 97.96%). The HGS achieved 98.09% on precision metric, the highest result on the SVM algorithm, followed by BAT and FFA, which achieved 98.06% and 98.05%, respectively. In the same precision result, WOA and MVO had 98.02%. PSO again scored the lowest performance, with 97.99%. In addition, the recall metric for the SVM classifier was 98.06%, 98.03%, 98.03%, 98%, 98%, and 97.96% for HGS, BAT, FFA, WOA, MVO, and PSO, respectively. Finally, for the F1-score metric, our HGS algorithm obtained the best performance, with 98.06%. Next, the BAT and the FFA algorithm had 98.03%. They were followed by the WOA and the MVO, which achieved 98.00%. In the last algorithm, the PSO has the lowest performance. On the other hand, merging the six optimization algorithms with KNN achieved the worst performance among all metrics.

Table 6. Results of each algorithms on MAVEN dataset.

Algorithm	Model	Recall	Precision	F1 Score	Accuracy
PSO	KNN	0.8281	0.8036	0.8126	0.8281
	SVM	0.8329	0.8073	0.8156	0.8329
MVO	KNN	0.8269	0.8036	0.8118	0.8269
	SVM	0.8353	0.8073	0.8146	0.8353
WOA	KNN	0.8281	0.8011	0.8105	0.8281
	SVM	0.8317	0.8012	0.8117	0.8317
FFA	KNN	0.8281	0.7972	0.8085	0.8281
	SVM	0.8341	0.8037	0.8136	0.8341
BAT	KNN	0.8281	0.8091	0.8135	0.8281
	SVM	0.8353	0.8066	0.8158	0.8353
HGS	KNN	0.8269	0.8038	0.8123	0.8269
	SVM	0.8413	0.8125	0.8214	0.8413

In Table 5, according to the results of Recall, Precision, F1-score, and Accuracy metric, the proposed HGS outperformed other optimization algorithms on the C36 dataset. The results combined the six optimization algorithms and traditional machine learning classifiers (i.e., SVM and KNN). From the results, we noticed that SVM has the best results on several optimization algorithms. For the SVM classifier, the accuracy of HGS and FFA algorithms achieved 97.10%, which is the best performance. In contrast, the WOA and MVO were in the second level, with 96.94%, followed by BAT, 96.90%. The worst result, PSO, achieved 96.81%. For the precision metric, 97.14% was the best result achieved in our proposed HGS algorithm. Next, the FFA has 97.13%. The WOA followed them, with 97.01%. The previous three algorithms are followed by the MVO, BAT, and PSO, which have 96.97%, 96.92%, and 96.87%, respectively. The recall metric compared 97.10%, 97.10%, 96.94%, 96.94%, 96.90%, and 96.81% of the test samples using HGS, FFA, WOA, MVO, BAT, and PSO algorithms, respectively. On F1-Score, the proposed HGS also outperformed other algorithms, with 97.10%, followed by FFA achieving 97.09%. Next, the other algorithms WOA, MVO, and BAT, scored 96.94%, 96.93%, and 96.89%, respectively. Finally, the PSO achieved 96.81%, the worst result.

The results of HGS and other optimizers for the MAVEN dataset are presented in Table 6. We merged the SVM classifier and KNN classifier in the table on six optimizers (e.g., MVO, PSO, WOA, HGS, FFA, and BAT). The SVM achieved better results than the KNN. Thus, analysis of the six optimization algorithms on SVM are presented in this section. From the table on accuracy metric, we noticed that merging the HGS algorithm with SVM outperformed other algorithms, which achieved 84.13% for the accuracy, followed by the BAT and the MVO, which each have the same result (83.53%). Furthermore, the FFA has 83.41%. The worst performance algorithms were PSO and WOA, which achieved 83.29% and 83.17%, respectively. Regarding the results on the precision metric, the HGS also achieved the highest results, with 81.25%. The 80.73 percent were the second-best results, which belonged to the PSO and MVO. The other algorithms (i.e., BAT, FFA, and WOA) had the lowest performance, with 80.66%, 80.37%, and 80.12%, respectively. On the recall, the best results belonged to the proposed HGS algorithm followed by the BAT and the MVO which share a score (83.53%). They are followed by the FFA and the PSO, with 83.41% and 83.29%, respectively. Finally, 83.17% was the worst value, which belonged to the WOA algorithm.

From another point of view, Figure 5 shows the average accuracy of each FS method on the three datasets C6, C36, and MAVEN. By discussing each dataset separately, for the C6 dataset, the HGS outperformed the other FS methods, as shown in Figure 5a, where its overall average value on the two classifiers (i.e., KNN and SVM) is nearly 98.03%, followed by the WOA method in the second position, with 98.00%. The FFA provides results that are better than that of BAT and MVO, at 97.99%. Finally, the PSO is a minor performance. The

average accuracy of the C36 dataset on both classifiers (i.e., SVM and KNN) is displayed in Figure 5b. In the figure, we note that the HGS, FFA, and MVO optimizers outperformed other algorithms, with 96.95%, 96.98%, and 96.94% for accuracy, respectively. These results are close to each other. They are followed by the WOA, with 96.89%. Next, The BAT algorithm achieved 96.87%. Finally, The PSO has the lowest average accuracy, which achieved 96.86%. As shown in Figure 5c, the average accuracy of the C6 dataset was introduced. The HGS optimization algorithm shows clear superiority, comparing other algorithms. This algorithm produced 98.03%, followed by the WOA and the FFA, with the same result, 98.00%. Next, the BAT and the MVO have the same value, with 97.98%. Lastly, the PSO algorithm achieved the lowest performance, 97.96%.

Figure 5. Average accuracy of each FS method: (a) C6, (b) C36, and (c) MAVEN.

The Friedman (FD) test is a nonparametric, bidirectional analysis of differences by rank, in which the statistical value is calculated and ranked. In [48], the FD test is used to check whether there is an important difference between other algorithms and numerous datasets. The superior approach is low (high) if the smallest (biggest) method is regarded to be the best. Furthermore, Figure 6 displays the HGS algorithm's mean rank in terms of Recall, Precision, F1-measure, and Accuracy when compared to the five optimization algorithms on the three datasets. When analysing the HGS' behaviors on the four measures,

it can be seen that the HGS algorithm outperforms the others. For accuracy measurement, we noticed that the HGS is in the best mean rank of 5.83, and the FFA is in the mean rank of 4.33. BAT and MVO have almost the same mean level. WOA averages 2.33. Lastly, PSO is less than others, with a mean rank 1.33. According to the FD test findings for the F1-score, we also observed that HGS is better than others, with a mean rank of 6. BAT and FFA have the same mean level, with 3.83. followed by MVO, with a mean level of 2.83. Finally, the lowest mean ranking is WOA and PSO. Furthermore, in precision metric, we noticed that the HGS is in the best mean rank of 6 and the FFA is in the mean rank of 3.67. BAT and MVO have almost the same mean level (i.e., 3.33). Lastly, PSO is less than others, with a mean rank 2.17. Finally, the difference between the HGS and the PSO, MVO, WOA, FFA, and BAT optimization algorithms on the recall measure is averaged nearly 1.33, 3.5, 2.33, 4.33, and 3.67, respectively.

Figure 6. The mean rank of FD test to compare among FS optimization algorithms.

In comparison to the time usually needed by a customer, the completion time for the total process is limited. Therefore, as shown in Figure 7, the average execution times of the proposed HGS algorithm for the C6, C36, and Maven, respectively, were 1.3733, 7.1789 s, and 1.6756 s. These time results are lower than other compared algorithms. For more accurate analysis, the execution time for the C6 dataset was 449.42 ms, the time was 2.3698 s, 1.7429 s, 1.5632 s, 1.8180 s, 1.3812 s, and 1.3733 s for PSO, MVO, WOA, FFA, BAT, and HGS, respectively. From these results, we note that HGS has the lowest time. For the C36 dataset, we also observed that the proposed algorithm was achieved in less time than other methods followed by BAT, which has 7.1789 s. Next, the WOA algorithm has 7.3930 s. They are followed by the BAT and FFA, executed in 8.3395 s and 11.5831 s, respectively. The other algorithms, MVO and PSO, have the most significant time (i.e., the worst time). Finally, the MAVEN dataset was also evaluated as the same as experiments in another dataset. Subsequently, the average execution time for our proposed HGS is the best. In addition, FFA, MVO, and WOA followed our algorithm, which has the same level (i.e., they achieved 2.2951 s, 2.3741 s, 2.4659 s, respectively).

Figure 7. Average execution time of each FS method.

In terms of the reduction rate, Figure 8 shows the average number of the selected features. The HGS selects the smallest number of features over all three event datasets. This choosing, nevertheless, has a beneficial impact on classifying performance, as evidenced by its accuracy, as seen in the Figure 5. Furthermore, because it chooses well, almost 51, 41, and 44 features among the C6, C36, and Maven data sources, respectively, the HGS has an increased opportunity to choose the fewest portion from the feature set without impacting of event detection across all tested corpora.

However, by analyzing the behavior of FS methods at each dataset, it can be observed that, in the C6 dataset, HGS and WOA have the same number of features (i.e., 51). The MVO, BAT, and FFA follow, choosing 58, 60, and 61 features, respectively. Finally, the PSO selects the most significant number of features, which has 74 features. For the C36 dataset, PSO has the highest number of features, which means this algorithm is the worst. This selector has 88 features, followed by BAT and MVO, with 58 features and 57 features, respectively, while WOA has 52 and FFA has 49, which is a lower number of features than before. In contrast, the HGS is the best algorithm, because it has a lower feature number than others (i.e., 41 features). In addition, it has higher accuracy, as mentioned above. In the Maven dataset, our proposed HGS optimization has 44 features. The other algorithms, MVO, FFA, and WOA, are approximately equal in the number of features (i.e., nearly 65 features). The worst algorithms in terms of feature number are BAT and PSO, with 75 for BAT selector and 87 for PSO selector.

Of the three selected datasets, the average accuracy of the SVM classifier and the KNN classifier was introduced in Figure 9 on several optimization algorithms (i.e., the six optimizers, which were introduced before). In the figure, we note that the SVM outperformed another classifier on different metrics. In detail, regarding accuracy, the SVM achieved 92.83%, while the KNN has 92.53%. SVM achieved 91.90% on the precision metric, which was a higher result than the KNN (i.e., 91.73%). The results on recall were 92.53% and 92.83% for the KNN and the SVM, respectively. On the F1-score, the KNN achieved 91.99%, while the SVM obtained 92.17%.

Figure 8. Number of selected features of each FS method.

Figure 9. Average accuracy of classifiers on the selected datasets.

The distribution of the SVM and KNN classifier on six different optimizers (i.e., PSO, MVO, WOA, FFA, BAT, and HGS) is displayed in Figure 10. From the figure, we note that the HGS achieved the best result on the SVM during the minor performance on the KNN. These results were 93.10% for the SVM algorithm and 92.49% for the KNN algorithm. On the SVM, the FFA followed HGS, with 92.85%. They are followed by the MVO and the BAT, which both obtained 92.82%. The other algorithms, the WOA and the PSO, have the worst results (i.e., 92.70% and 92.69%, respectively). On the other hand, the PSO has the best result, with 92.56%, on the KNN classifier, followed by the WOA, FFA, MVO, and BAT, which achieved 92.55%, 92.54%, 92.53%, and 92.52% respectively. On the average accuracy for both classifiers, our proposed HGS algorithm outperformed the other algorithms, with 92.80%. In addition, the FFA and the MVO followed our algorithm, which has 92.69% and 92.68%, respectively. The BAT achieved 92.67% accuracy. Finally, the worst results belonged to the WOA (92.63%) and the PSO (92.62%).

Figure 10. Distribution of the two classifiers based on FS algorithms.

In summary, for the C36 dataset, the HGS optimization algorithm combined with the SVM classifier had the best classification measurement of any mixture. For the accuracy metric, this collection received 97.10 percent. Furthermore, it achieved 97.10 percent F1-Score, 97.14 percent precision, and 97.10 percent recall. This combined effect further resulted in the highest performance measures values for the C6 data. The accuracy metric yielded a score of 98.06 percent, the F1-Score yielded a score of 98.09 percent, and the recall yielded a score of 98.06 percent. Ultimately, the results for the MAVEN data source were lesser than before, with 84.13 percent, 82.14 percent, 84.13 percent, and 81.25 percent for the accuracy, F1-Score, Recall, and Precision metrics, respectively. In addition, our HGS optimization algorithm outperformed other optimization methods as it has the least number of features among datasets. It achieved the highest performance, as shown before.

4.3.2. Comparison with Previous Studies

This section compares ours with other state-of-the-art crisis event detection techniques. Table 7 shows the results of a few important methodologies. The development of high-accuracy technology for event detection is a major undertaking. It is important to compare our strategy to other models that have been tested on the same datasets. Using C6 and C36 datasets, Table 7 evaluates the performance of several techniques for crisis identification.

For both selected datasets, in [49], the authors used three methods: (1) The combination of a Logistic Regression (*LR*) classifier and pre-trained Word2Vec (*w2v*) formulated in (*LR_{w2v}*); (2) The SVM (*SVM*) classifier with pre-trained Word2Vec embedding coined as (*SVM_{w2v}*); and (3) The pre-trained W2V encoding associated with the Naive Bayes (*NB*) model, which assumes a Gaussian distribution for attributes and benefits, denoted as (*NB_{w2v}*) model. Moreover, in [50], they merged the *CNN* approach with the Global Vector (*gv*) embedding, in which the *CNN* has two convolutional layers of 250 filters, 128 hidden units, a pool size of 2, and a kernel size of 3. Kumar et al. [51] combined a W2V embedding with Long Short-Term Memory (*LSTM*) model with 30 hidden states in two layers. In [11], they suggested Crisis2Vec (*c2v*), a documentation context-specific encoding technique for crisis representation that significantly outperformed traditional methods. In addition, they applied two methods: (1) *LR_{c2v}*: Combining a linear *LR* classifier with the Crisis2Vec embedding, and (2) *LSTM_{c2v}*: a non-linear *LSTM* model merged with the Crisis2Vec. In contrast, our approach employs a transformer-based method for feature extraction, which is carried out using DistilBERT. Moreover, a HGS is used as a feature selection approach in order to exclude unnecessary features from the feature sets in order to improve performance.

Table 7. Comparison results of the existing approaches.

Source	Dataset	Year	Classification Model	F1-Score (%)	Accuracy (%)
[49]	C6	2019	NB_{w2v}	78.80	80.50
[49]	C6	2019	SVM_{w2v}	86.60	87.90
[49]	C6	2019	LR_{w2v}	87.30	88.50
[50]	C6	2019	CNN_{gv}	90.50	90.40
[51]	C6	2020	$LSTM_{w2v}$	87.30	87.30
[11]	C6	2021	LR_{c2v}	93.60	93.70
[11]	C6	2021	$LSTM_{c2v}$	97.50	97.50
Our	C6	present	HGS_{db}	98.06	98.06
[49]	C36	2019	NB_{w2v}	47.60	63.50
[49]	C36	2019	SVM_{w2v}	71.6	81.90
[49]	C36	2019	LR_{w2v}	72.10	82.30
[50]	C36	2019	CNN_{gv}	23.30	64.40
[51]	C36	2020	$LSTM_{w2v}$	58.90	72.30
[11]	C36	2021	LR_{c2v}	85.10	90.90
[11]	C36	2021	$LSTM_{c2v}$	88.00	95.60
Our	C36	present	HGS_{db}	97.10	97.10

For the C6 dataset, our proposed HGS_{db} approach accomplishes 98.06 percent for the F1-score and 98.06 percent for the Accuracy, which also outperforms the highest score results of the model, notably CNN with pre-trained GloVe embeddings, by 7.56 percent for the F1-score and 7.66 percent for the Accuracy. Throughout aspects of embedding, LSTM with Crisis2Vec achieves 97.5 percent including both F1-score and Accuracy, representing a 10.2% improved performance over LSTM with Word2Vec. Correspondingly, LR with Crisis2Vec achieves a 93.6 percent F1-score and a 93.7 percent Accuracy, representing a 6.3 percent and 5.2 percent improved performance over LR with Word2Vec, respectively.

For the C36 dataset, our proposed HGS_{db} approach obtains a 97.10 percent F1-score and 97.10 percent Accuracy, outperforming the previous best prediction model by a significant rate of return, such as LR with pre-trained Word2Vec encoding, by 25.0 percent and 14.8 percent, respectively. Through definitions of word vectors, LSTM with Crisis2Vec achieves 88.0 percent F1-score and 95.6 percent Accuracy, which outperforms LSTM with Word2Vec by 29.1 percent and 23.3 percent, respectively. Similarly, LR with Crisis2Vec achieves 85.1 percent F1-score and 90.9 percent Accuracy, outperforming LR with Word2Vec by 13.0 percent and 8.6 percent, respectively.

The bottom line is that we can remove superfluous features from high-dimensional event representations obtained by DistilBERT using our approach. However, this framework's fundamental drawback is its complexity, both in terms of time and memory. Future steps should include reducing complexity and improving the efficiency of our proposed method. In the future, other augmentation procedures can be researched to improve our method's efficiency.

5. Conclusions

This paper demonstrates the hybrid framework that incorporates the pre-trained DistilBERT model and the proposed feature selection algorithm to identify crisis events. For instance, a pre-trained DistilBERT model with a defined architecture for feature extraction was fine-tuned on several real-world datasets to generate sentence embedding for each data sample. Later, the feature selection phase uses a new meta-heuristic technique named

the Hunger Games Search (HGS) in its binary form to select the most relevant features from the extracted tweets embeddings to maximize the classification model performance reduce the dimensionality of the features representation space. Experiments and comparisons of the proposed framework show superiority in terms of events identification accuracy and feature reduction compared to other state-of-the-art feature selection techniques and event identification methods. The DistilBERT is a relatively small model compared to existing state-of-the-art models. Thus, exploring the other transformer-based model may reveal different and valuable feature sets and improve the overall framework performance. In addition, the DistilBERT covers a single language, while multilingual language models are worth investigating.

As a future work, the developed framework can be extended to cover different NLP tasks such as sentiment analysis, offensive detection, and question answering. In addition, grouping related tasks to the main task in a multi-task learning framework may help boost the performance.

Author Contributions: Conceptualization, H.A., A.D., A.M., and M.A.E.; methodology, H.A., A.D., A.M., S.A., and M.A.E.; software, A.D., A.M., and M.A.E.; validation, A.D., A.M., and M.A.E.; formal analysis, H.A., A.D., A.M., S.A., and M.A.E.; investigation, H.A., A.D., A.M., S.A., and M.A.E.; writing—original draft preparation, H.A., A.D., A.M., and M.A.E.; writing—review and editing, H.A., A.D., A.M., S.A., and M.A.E.; visualization, H.A., A.D., A.M., S.A., and M.A.E.; supervision, M.K., I.M.E.-H., and A.A.A.; project administration, H.A., A.D., A.M., S.A., and M.A.E.; and funding acquisition, S.A. All authors have read and agreed to the published version of the manuscript.

Funding: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R197), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia

Acknowledgments: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R197), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

1. Dinkel, H.; Wu, M.; Yu, K. Towards duration robust weakly supervised sound event detection. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 887–900. [\[CrossRef\]](#)
2. Phengsuwan, J.; Shah, T.; Thekkummal, N.B.; Wen, Z.; Sun, R.; Pullarkatt, D.; Thirugnanam, H.; Ramesh, M.V.; Morgan, G.; James, P.; et al. Use of Social Media Data in Disaster Management: A Survey. *Future Internet* **2021**, *13*, 46. [\[CrossRef\]](#)
3. Huang, L.; Liu, G.; Chen, T.; Yuan, H.; Shi, P.; Miao, Y. Similarity-based emergency event detection in social media. *J. Saf. Sci. Resil.* **2021**, *2*, 11–19. [\[CrossRef\]](#)
4. Di Girolamo, R.; Esposito, C.; Moscato, V.; Sperli, G. Evolutionary game theoretical on-line event detection over tweet streams. *Knowl.-Based Syst.* **2021**, *211*, 106563. [\[CrossRef\]](#)
5. Mabrouk, A.; Redondo, R.P.D.; Kayed, M. SEOpinion: Summarization and Exploration of Opinion from E-Commerce Websites. *Sensors* **2021**, *21*, 636. [\[CrossRef\]](#)
6. Song, D.; Xu, J.; Pang, J.; Huang, H. Classifier-adaptation knowledge distillation framework for relation extraction and event detection with imbalanced data. *Inf. Sci.* **2021**, *573*, 222–238. [\[CrossRef\]](#)
7. Mohanty, S.D.; Biggers, B.; Sayedahmed, S.; Pourebrahim, N.; Goldstein, E.B.; Bunch, R.; Chi, G.; Sadri, F.; McCoy, T.P.; Cosby, A. A multi-modal approach towards mining social media data during natural disasters—a case study of hurricane irma. *Int. J. Disaster Risk Reduct.* **2021**, *54*, 102032. [\[CrossRef\]](#)
8. Li, L.; Bensi, M.; Cui, Q.; Baecher, G.B.; Huang, Y. Social media crowdsourcing for rapid damage assessment following a sudden-onset natural hazard event. *Int. J. Inf. Manag.* **2021**, *60*, 102378. [\[CrossRef\]](#)
9. Liang, X.; Cheng, D.; Yang, F.; Luo, Y.; Qian, W.; Zhou, A. F-HMTC: Detecting Financial Events for Investment Decisions Based on Neural Hierarchical Multi-Label Text Classification. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20), IJCAI, Yokohama, Japan, 11–17 July 2020; pp. 4490–4496.
10. Kitazawa, K.; Hale, S.A. Social media and early warning systems for natural disasters: A case study of Typhoon Etai in Japan. *Int. J. Disaster Risk Reduct.* **2021**, *52*, 101926. [\[CrossRef\]](#)
11. Liu, J.; Singhal, T.; Blessing, L.T.; Wood, K.L.; Lim, K.H. Crisisbert: A robust transformer for crisis classification and contextual crisis embedding. In Proceedings of the 32nd ACM Conference on Hypertext and Social Media, Virtual Event, 30 August–2 September 2021; pp. 133–141.

12. Fatani, A.; Abd Elaziz, M.; Dahou, A.; Al-qaness, M.A.; Lu, S. IoT Intrusion Detection System Using Deep Learning and Enhanced Transient Search Optimization. *IEEE Access* **2021**, *9*, 123448–123464. [\[CrossRef\]](#)
13. Mabrouk, A.; Redondo, R.P.D.; Kayed, M. Deep learning-based sentiment classification: A comparative survey. *IEEE Access* **2020**, *8*, 85616–85638. [\[CrossRef\]](#)
14. Fan, H.; Du, W.; Dahou, A.; Ewees, A.A.; Yousri, D.; Elaziz, M.A.; Elsheikh, A.H.; Abualigah, L.; Al-qaness, M.A. Social Media Toxicity Classification Using Deep Learning: Real-World Application UK Brexit. *Electronics* **2021**, *10*, 1332. [\[CrossRef\]](#)
15. ALRashdi, R.; O’Keefe, S. Deep learning and word embeddings for tweet classification for crisis response. *arXiv* **2019**, arXiv:1903.11024.
16. Huang, X.; Li, Z.; Wang, C.; Ning, H. Identifying disaster related social media for rapid response: A visual-textual fused CNN architecture. *Int. J. Digit. Earth* **2019**, *13*. [\[CrossRef\]](#)
17. Kim, N.K.; Kim, H.K. Polyphonic Sound Event Detection Based on Residual Convolutional Recurrent Neural Network With Semi-Supervised Loss Function. *IEEE Access* **2021**, *9*, 7564–7575. [\[CrossRef\]](#)
18. Chang, H.C.; Wu, H.T.; Huang, P.C.; Ma, H.P.; Lo, Y.L.; Huang, Y.H. Portable Sleep Apnea Syndrome Screening and Event Detection Using Long Short-Term Memory Recurrent Neural Network. *Sensors* **2020**, *20*, 6067. [\[CrossRef\]](#)
19. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:1910.01108.
20. Xue, Y.; Tang, T.; Pang, W.; Liu, A.X. Self-adaptive parameter and strategy based particle swarm optimization for large-scale feature selection problems with multiple classifiers. *Appl. Soft Comput.* **2020**, *88*, 106031. [\[CrossRef\]](#)
21. Xue, Y.; Xue, B.; Zhang, M. Self-adaptive particle swarm optimization for large-scale feature selection in classification. *ACM Trans. Knowl. Discov. Data (TKDD)* **2019**, *13*, 1–27. [\[CrossRef\]](#)
22. AbuShanab, W.S.; Abd Elaziz, M.; Ghandourah, E.I.; Moustafa, E.B.; Elsheikh, A.H. A new fine-tuned random vector functional link model using Hunger games search optimizer for modeling friction stir welding process of polymeric materials. *J. Mater. Res. Technol.* **2021**, *14*, 1482–1493. [\[CrossRef\]](#)
23. Manbari, Z.; AkhlaghianTab, F.; Salavati, C. Hybrid fast unsupervised feature selection for high-dimensional data. *Expert Syst. Appl.* **2019**, *124*, 97–118. [\[CrossRef\]](#)
24. Xue, B.; Zhang, M.; Browne, W.N.; Yao, X. A survey on evolutionary computation approaches to feature selection. *IEEE Trans. Evol. Comput.* **2015**, *20*, 606–626. [\[CrossRef\]](#)
25. Rudra, K.; Ganguly, N.; Goyal, P.; Ghosh, S. Extracting and summarizing situational information from the twitter social media during disasters. *ACM Trans. Web (TWEB)* **2018**, *12*, 1–35. [\[CrossRef\]](#)
26. Zahra, K.; Imran, M.; Ostermann, F.O. Automatic identification of eyewitness messages on twitter during disasters. *Inf. Process. Manag.* **2020**, *57*, 102107. [\[CrossRef\]](#)
27. Kejriwal, M.; Zhou, P. On detecting urgency in short crisis messages using minimal supervision and transfer learning. *Soc. Netw. Anal. Min.* **2020**, *10*, 1–12. [\[CrossRef\]](#) [\[PubMed\]](#)
28. Madichetty, S.; Sridevi, M. Identification of medical resource tweets using majority voting-based ensemble during disaster. *Soc. Netw. Anal. Min.* **2020**, *10*, 1–18. [\[CrossRef\]](#)
29. Purohit, H.; Castillo, C.; Pandey, R. Ranking and grouping social media requests for emergency services using serviceability model. *Soc. Netw. Anal. Min.* **2020**, *10*, 1–17. [\[CrossRef\]](#)
30. Madichetty, S.; Sridevi, M. Re-ranking feature selection algorithm for detecting the availability and requirement of resources tweets during disaster. *Int. J. Comput. Intell. IoT* **2018**, *1*.
31. Basu, M.; Shandilya, A.; Khosla, P.; Ghosh, K.; Ghosh, S. Extracting resource needs and availabilities from microblogs for aiding post-disaster relief operations. *IEEE Trans. Comput. Soc. Syst.* **2019**, *6*, 604–618. [\[CrossRef\]](#)
32. Dutt, R.; Basu, M.; Ghosh, K.; Ghosh, S. Utilizing microblogs for assisting post-disaster relief operations via matching resource needs and availabilities. *Inf. Process. Manag.* **2019**, *56*, 1680–1697. [\[CrossRef\]](#)
33. Alam, F.; Ofli, F.; Imran, M. Descriptive and visual summaries of disaster events using artificial intelligence techniques: Case studies of Hurricanes Harvey, Irma, and Maria. *Behav. Inf. Technol.* **2020**, *39*, 288–318. [\[CrossRef\]](#)
34. Lin, J.; Nogueira, R.; Yates, A. Pretrained transformers for text ranking: Bert and beyond. *arXiv* **2020**, arXiv:2010.06467.
35. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
36. Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C.H.; Kang, J. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **2020**, *36*, 1234–1240. [\[CrossRef\]](#)
37. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [\[CrossRef\]](#)
38. Neggaz, N.; Houssein, E.H.; Hussain, K. An efficient henry gas solubility optimization for feature selection. *Expert Syst. Appl.* **2020**, *152*, 113364. [\[CrossRef\]](#)
39. Hendrycks, D.; Gimpel, K. Gaussian error linear units (gelus). *arXiv* **2016**, arXiv:1606.08415.
40. Yang, Y.; Chen, H.; Heidari, A.A.; Gandomi, A.H. Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Syst. Appl.* **2021**, *177*, 114864. [\[CrossRef\]](#)
41. Olteanu, A.; Castillo, C.; Diaz, F.; Vieweg, S. Crisislex: A lexicon for collecting and filtering microblogged communications in crises. In Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media, Ann Arbor, MI, USA, 1–4 June 2014.

42. Zubiaga, A.; Liakata, M.; Procter, R.; Wong Sak Hoi, G.; Tolmie, P. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PLoS ONE* **2016**, *11*, e0150989. [[CrossRef](#)]
43. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science, MHS'95, Nagoya, Japan, 4–6 October 1995; pp. 39–43.
44. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* **2016**, *27*, 495–513. [[CrossRef](#)]
45. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [[CrossRef](#)]
46. Yang, X.S. Firefly algorithm, Levy flights and global optimization. In *Research and Development in Intelligent Systems XXVI*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 209–218.
47. Yang, X.S. A new metaheuristic bat-inspired algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74.
48. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [[CrossRef](#)]
49. Manna, S.; Nakai, H. Effectiveness of word embeddings on classifiers: A case study with tweets. In Proceedings of the 2019 IEEE 13th International Conference on Semantic Computing (ICSC), Newport Beach, CA, USA, 30 January–1 February 2019; pp. 158–161.
50. Kersten, J.; Kruspe, A.; Wiegmann, M.; Klan, F. Robust filtering of crisis-related tweets. In Proceedings of the ISCRAM 2019 Conference Proceedings—16th International Conference on Information Systems for Crisis Response and Management, Valencia, Spain, 19–22 May 2019.
51. Kumar, A.; Singh, J.P.; Dwivedi, Y.K.; Rana, N.P. A deep multi-modal neural network for informative Twitter content classification during emergencies. *Ann. Oper. Res.* **2020**, 1–32. [[CrossRef](#)]