

# proBE—A BERTweet-Attentive Prototype Few-shot Model for Event Detection in OSN

Sielvie Sharma      Muhammad Abulaish, *SMIEEE*      Tanvir Ahmad  
*Department of Computer Engineering*    *Department of Computer Science*    *Department of Computer Engineering*  
*Jamia Millia Islamia*      *South Asian University*      *Jamia Millia Islamia*  
New Delhi, India      New Delhi, India      New Delhi, India  
sielvie@outlook.com      abulaish@ieee.org      tahmad2@jmi.ac.in

**Abstract**—In the domain of digital communication, a significant transition have occurred, with social networks overtaking traditional news media outlets to become the primary source of information on a global scale. This evolution has given rise to a dynamic environment, where user-generated content escalates at an exceptional pace. However, managing this vast and ever-changing data poses significant challenges. Annotating such voluminous and dynamic information is not only financially burdensome but also impractical in terms of feasibility. In addition, current learning techniques face significant challenges when it comes to identifying unseen or novel occurrences in social network data, while also demanding a huge amount of training data to operate effectively. This entails a new era that emphasis on the need for less data and more generalisation, similar to how humans interpret information. To this end, in this research, we developed the model *proBE*, which formalises event detection in online social networks as a few-shot learning problem and offers a novel perspective on it. The suggested method encodes the tweet messages (*aka* tweets) with *BERTweet*, to capture context with respect to inbuilt features of Twitter like hashtags, emoticons and then employed an attentive prototype model where, tweet attention and feature attention is applied to highlight the contextually rich key tweets and the prominent features, respectively. *proBE* is evaluated on real world benchmark Twitter datasets *CrisisLexT26* and *CrisisLexT6* and performs significantly better when compared to various baseline methods in terms of accuracy, F-score, precision and recall. To the best of our knowledge, this is the first study that employs few-shot learning in event detection in online social networks to overcome the sparsity of labelled data due to the volatile dynamics of online social networks and the inability of existing approaches to detect unseen events even after enormous amount of training data.

**Index Terms**—Social Network Analysis, Few-shot Learning, BERTweet, Event Detection, Online Social Networks, Attention Mechanisms

## I. INTRODUCTION

The popularity of online social networks (OSNs) is growing over time. The ease of availability and access to information changed the entire concept of news distribution. In this day and age, users have more access to every event around the world than in the past, when

the next day's gazette is the only source to find out what is going on around? At present, the availability of news is not limited to designated news sources. Users are also contributing to make news available by posting it online in the shortest time possible and further all other users have the access and right to share their thoughts and opinions about it. There are many existing online social networks like Twitter, Instagram and Facebook, and yet Twitter is most popular among all with millions of monthly active users and millions of tweets per day. Added features like hashtags and mentions changed the dynamics and dissemination of information among users. Such vast and dynamic data inspires a wide range of research opportunities to extract useful information such as social bot detection, hate speech detection, and many others. One such unfolding study is event detection from online social networks. In Natural Language Processing (NLP), event detection is a task of information extraction to educe useful findings from the vast amount of available data.

Event detection is dealt with different perspectives by various researchers. Initial efforts employ approaches that explored linguistic features and adopted statistical methods [1], [2] and [3]. Moving on to supervised approaches, the authors focused on neural networks (such as CNN and RNN) to automatically learn required features from large scale datasets, resulting in significantly improved performance. However, fail to detect unseen events. In real-world scenarios, such as online social networks, labelling all the available data can be a challenging and laborious task. Due to above limitations such as inability of labelling a large amount of data and not able to detect unseen events, the authors focused on unsupervised techniques [4], [5], which have their own set of challenges ranging from tedious task to handle copious data to computation intensive tasks.

To this end, few-shot learning is introduced in this work with an attempt to address the aforementioned limitations without relying on a large training dataset and detect existing events in social network data. For each class,

most classification algorithms require a large amount of training data. In contrast to that, few-shot learning requires a very small number of training instances of each class. Furthermore, unlike traditional classification, where training and testing examples are drawn from the same set, classes for training and testing are drawn from separate or disjoint sets, which can also aid in the resolution of the covariant shift issue that exists widely in online social network data. The notion behind few-shot learning is to explicitly train models to handle novel classes which were never seen during training with limited labelled examples. This work aims to learn a more robust and generalisable representation that can adapt to unseen classes during testing. The ability to learn patterns and discriminate between features across different classes lead to more accurate prediction despite of having limited data. To this end, the ability of few-shot learning to generalise to new classes with limited amount of data is precisely what event detection in online social networks requires. It allows the model to adapt and perform well on unseen data by leveraging the knowledge gained from the support set, even when the distribution of the data changes between training and testing phases.

The main contributions of the paper are:

- In light of the dynamic nature of social network data, event detection in online social networks is modelled as a few-shot learning problem.
- Extending the event detection in online social networks to unseen events.
- Emphasis is on low resource learning because labelling enormous diverse and dynamic social network data is unrealistic.
- Addressing the inability of other existing approaches of detecting unseen events despite requiring a considerable amount of training data.

Few-shot learning is a well-studied concept in image classification, but it has received little attention in the field of text (NLP). To this end, an attempt is made to detect events utilising the benefits of few-shot learning on unstructured informal online social network data. To our best knowledge, this is the first work addressing the problem detecting novel events in OSN with an attempt of utilising low resource learning. The rest of the paper is organised as follows. Section II presents a brief overview of the existing literature on event detection, with a focus on online social media. Section IV presents the functioning details of the proposed approach. Section V presents a detailed description of the experimental setup and evaluation results. Finally, section VI concludes the paper and presents future directions of research.

## II. RELATED WORK

With the goal of detecting and locating events in the stream of broadcast news, event detection research was introduced [6]. With the time, social networks gained popularity and so the inspiration to dig relevant information which can be useful to people in many ways. A detailed survey by the authors of [7], [8] and [9] discussed different formulations of event detection by various authors with respect to unsupervised, supervised and semi supervised learning. Authors also discussed the lack of benchmark datasets and no defined evaluation measures in the area of event detection in online social networks.

Initial efforts of [10], [6] and [11] mainly focused on feature engineering, statistical and linguistic methods [10], [1] to find events that are unknown to us prior the detection. With the development of deep neural networks [12] [13], many researchers opted it for solving event detection problem by formulating it as a classification problem [14], [13] and [15]. It also significantly improves performance, but it fails to detect unknown events that are not in training. Furthermore, a large amount of data is required to train and the system still fails to detect unseen events. Labelling social network data for a task is impractical because data on online social network sites is diverse and vast. When dealing with such massive amounts of data, heavy computation is required, and time is also a constraint. To detect unseen events, data augmentation approaches are also introduced [16].

Few-shot learning makes it easier for the model to pick up useful features without needing a lot of labelled data. Early research concentrated on models of generative transfer learning that take the target task from pre-trained models. These techniques, however, are challenging to use for real-world applications [17] since they demand subject designated designs and are unable to adequately capture the distribution's characteristics. Matching networks were used by the authors of [18] to map a small labelled support set and an unlabeled example to their labels without the requirement for fine tuning in order to accommodate new class types. Furthermore, by calculating the distance between the query and the prototype representations for each class, prototypical networks [19] learn a metric space in which the model can perform well. Then, categorisation of the query is done to the prototype class that is the closest to the query. Metric learning computes the distance between the observed classes [19] [20] [18].

For a newly discovered few shot task like intention classification, the authors of [18] recently proposed an adaptive metric learning strategy that automatically chooses the best weighted combination from a set of metrics gathered through meta-training tasks. FewRel,

a relation classification dataset, is presented by [21], which also adapts the most state-of-the-art few-shot learning techniques for it. These approaches do not, however, take semantic data mining or more precise noise reduction into account. The concept of a quick learner, who can generalise a new concept quickly, was recently presented in the meta-learning [22]. The meta-learning based on a deep neural network with fine tuning has been improved by recent work by [23]. Among all others, FSL (Few-shot Learning) -metric learning has been demonstrated to be efficient and explicable for a variety of reasons, including its well-researched theory of the distance function and the ease of its implementation in practise.

Significantly, state-of-the-art performance on a number of FSL (Few-shot Learning) benchmarks is attained by the prototype networks in metric learning. Although few-shot learning is a well-researched job in image classification, very few studies have looked at its application to NLP. To this end, an effort is made to address event detection in online social networks using a low-resource learning idea, namely few-shot learning, in order to free ourselves from computation-intensive activities, suffer from a scarcity of labelled data, and extend current event detection to new event categories.

### III. PROBLEM DEFINITION

This work presents event detection in Twitter data exploiting few shot learning data, often known as few-shot learning. The problem under consideration is an event categorisation problem with the purpose of learning a function  $f: f(\text{dataset}, \text{supportset}, \text{supportsample}) \rightarrow \text{Query}$ . Labeled dataset is divided into  $\text{Dataset}_{\text{train}}$  for training the data and optimising parameters,  $\text{Dataset}_{\text{validation}}$  for selecting the best hyper-parameters and  $\text{Dataset}_{\text{test}}$  to evaluate the efficacy of the model.

In this work, episodic training strategy [18] is opted that has proved to be effective in few-shot learning scenario as depicted in figure1. A label set (*Label*) is sampled from the training dataset for each training episode, followed by a support set and a query set. Finally, the support and query sets are fed into the model to minimise the loss. Support set in an episode contains  $N_i$  tweet samples for each class  $C_i$ . Then, there is unlabelled tweets to classify, and query  $\in \text{Label}$  is the output label followed by the prediction of the function  $f$ . Important terminology regarding few-shot learning is: N-way K-shot. N-way represents the number of classes (events) in the support set and number of samples (tweets) in a particular class are considered as K-shot. In this work, model performance is evaluated on  $N = 3$  or 4 and  $K = 5$  or 10.

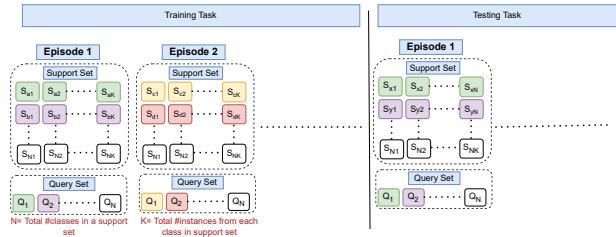


Fig. 1: Episodes in few-shot learning that contain support sets and query sets

### IV. PROPOSED MODEL

This section presents a detailed description of our proposed approach, `proBE`, which consists of various steps such as *Tweet Encoding*, *Prototype Model*, *Attentive Prototype*. The overall architecture is shown in figure 2. Further, details of individual functional component is presented in the following sub-sections.

#### A. Tweet Encoding

Tweet encoding consists of an embedding layer where, `BERTweet` is used to capture the contextual information of a tweet contained in a support set, keeping in mind the need to grasp the context of input owing to small training data sets. Given a tweet of input support set  $t$  contains  $t = w_1, w_2, \dots, w_m$  words. The raw text will be encoded to vector embeddings using an embedding layer.

1) *Embedding Layer*: Recently, BERT has gained immense popularity and success in NLP tasks as it is based on Transformer architecture and capable of capturing contextualised representations with consideration of context of the words in a sentence from both forward and backward directions. We are largely dealing with unstructured data when it comes to online social networks, and one of the obstacle is transforming unstructured information into contextual representations. `BERTweet` model is specially designed for dealing with Twitter data. It addresses all the challenges associated with amorphous data available on Twitter, due to casual use of words, short text and inbuilt Twitter features such as hashtags and emoticons, all of which contribute significantly to the interpretation of an event. Hashtags are sometimes responsible for the event propagation, and for the most part, they are user-created, with little significance if not addressed with context to the tweet. Hashtags are not eliminated during the pre-processing step since they play a significant role in Twitter data, and `BERTweet` is capable of recognising the Twitter structure, which leads to the identification of special linguistic and contextual information unique to Twitter data.

`BERTweet` is a pre-trained masked language model

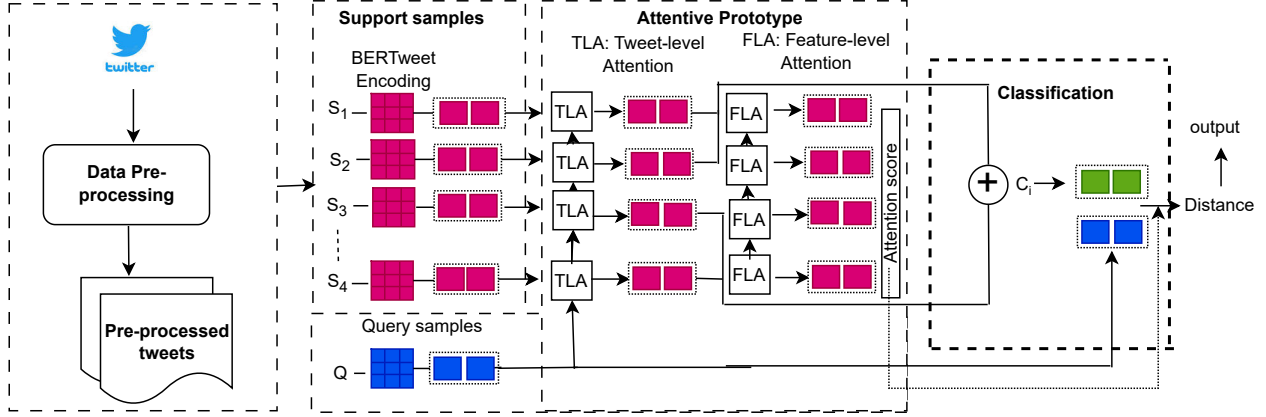


Fig. 2: Workflow of the proposed proBE framework

[24] utilizing BERT having the same architecture as  $BERT_{base}$  and trained using RoBERTa pre-training procedure. A tweet ( $t$ ) with raw tweet text with  $m$  words will be represented in vector form with the help of BERTweet as mentioned in equation 1 and 2.

$$E_t = BERT_{tweet}(tweet) \quad (1)$$

$$tweet(t) = w_1, w_2, \dots, w_m \quad (2)$$

Where,  $E_t$  is embedded tweet sample  $\in \mathbb{R}^d$  where,  $d$  is the dimension of the vector.

Finally, to get the meaningful representation of  $t$ , encoded representations from [CLS] token, which is a special token that captures contextual information from the entire input text, is extracted as it encapsulates the aggregate information from the entire input sequence and will be passed to the next layer.

### B. Prototype Model

Prototype encoder for few shot learning is first introduced by the authors of [19], where it is described as a concept to compute a representative vector, also called a Prototype (Proc) for all the samples of the support set and which can be generated by averaging the vectors of the samples for each class  $C \in C_{all}$  as calculated in equation 3.

$$Proc_i = \frac{1}{m_i} \sum_{j=1}^{m_i} t_i^j \quad (3)$$

Where,  $Proc_i$  is the prototype for a class  $C_i$  that is computed for the samples of support set ( $s$ ) of the class.  $m_i$  are the total number of tweets in the support set of the class  $C_i$  and ( $t_i^j$ ) is tweet embedding with the help of BERTweet. Further, probabilities for the query tweet with the help of distance function can also be computed with the help of equation 4.

$$\begin{aligned} \text{Prob}(y = C_i | (q, p), \text{Support}) \\ = \frac{\exp(-\text{dist}((q, p), \text{Proc}_i))}{\sum_{k=1}^{|C_{all}|} \exp(-\text{dist}((q, p), \text{Proc}_k))} \quad (4) \end{aligned}$$

Where  $\text{dist}(\dots)$  is the distance function between two provided vectors, namely the class prototype and the query tweet. Euclidean distance is utilised for distance computation since the authors of [19] discovered that it outperforms all other existing distance measures with respect to a prototype network.

In the traditional setup of prototype networks, all the samples of the support set are treated identically. However, in real world scenario, not all the samples are contributing to classification equally. Specifically, when it comes to Twitter, not every tweet is equally informative and due to the difference in representation of different support samples, the average (prototype) can be far away from the representation of each of the prototype. Likewise, some features are more prominent in discriminating the class than the other not so important features. To this end, an attention mechanism is applied to highlight all the contributing features as described in following sub-sections.

1) *Attentive Prototype*: Attention entails focusing on one or more components while neglecting others. Concentrating on a single unit increases machine learning models' intelligence. In general, classification systems describe data as a numeric vector of low-level features, with each feature given the same weight regardless of its ability to conceptualise the data. For attention mechanisms such as words, phrases, and sentences, several granularity levels can be used. The attention layer is made up of an alignment layer, attention weights, and a context vector. The output of the attention mechanism is a weighted sum of all elements in the encoded vector. In attentive prototypical networks, tweet-level and feature level attention are employed with the conventional

prototypical network to enrich the features and incorporate semantics and context to the conventional way. Given that few-shot learning is intended to work with little quantities of training data, semantic and contextual consideration should be prioritised.

2) *Tweet-level Attention*: In tweet-level attention, a sentence level attention is proposed to highlight the important tweets in an support set as not all the tweets of support contribute equally to prototype of a set and due to the lack of support data in FSL (Few-shot Learning) scenario, representation of one tweet is far from another representation create a huge deviation from the prototype. The idea is to highlight and use those tweets that are relevant and informative while ignoring that are not catering to the classification. Following the concept, relevancy based score will be assign tweet-level attention, an attention model [25] is extract contributing tweets and gather them to a more informative vector. The hidden represent of a feature  $j$ ,  $f_j$ , is fed into a feed-forward network to learn  $f'_j$ , an encoded representation in equation 5. Then, similarity will be calculated by applying the dot product between  $f'_j$  and a context vector  $f^l w$  which is randomly initialised and learned during the training process. Finally, softmax function mentioned in equation 6 is used to calculate the attention score  $\alpha^j$  of  $j^{th}$  tweet and attention based representation of a tweet (at) will be calculated with the weighted sum of hidden representations as mentioned in equation 8.

$$f'_j = \tanh(W f_j + b_w) \quad (5)$$

$$\alpha^j = \frac{\exp(f'_j f^l w)}{\sum (\exp(f'_j f^l w))} \quad (6)$$

Finally, equation 3 is replaced with equation 8.

$$Proc_i = \sum_{j=1}^{m_i} \alpha^j t_i^j \quad (7)$$

3) *Feature-level Attention*: Feature-level attention is an attempt to include features only with significant importance and remove the confusing features that can cause hinderance in further classification as some features have the capability to differentiate the class on feature level from other classes. An attention mechanism similar to [26] is applied for a class feature extractor. To this end, max pooling layer is applied over every sample (j) of every class ( $C_i$ ) of support set (s) in order to obtain a fresh feature map. Further, three convolution layer will be applied to get a parameter ( $\beta_i$ ) (score vector of class  $C_i$ ) as described in figure 3. To save the model from the feature scarcity due to the less number of samples in support set from each class, more discriminating dimensions in feature space are extracted

and new distance function with score vector of the class is defined in equation 8

$$distance(C_i, query) = \beta_i \cdot (C_i - query_2)^2 \quad (8)$$

Where,  $\beta_i$  is the score vector of the class is calculated with the help of feature-level attention. (query) is the query vector passed through the tweet-level attention. This attention will help in giving more score for discriminatory dimensions. With the multiplication of attention scores to the squared differences, distance metrics will be changed to fit the given classes and support instances. Finally, the new distance measure will calculate the attentive distance between all prototypes and a target query vector and assign the query to the closest one.

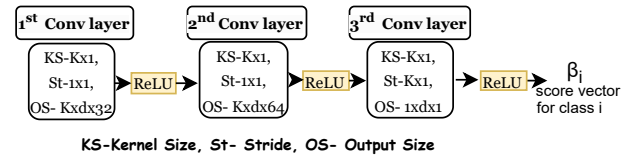


Fig. 3: Feature attention architecture

## V. EXPERIMENTAL SETUP AND RESULTS

This section presents the experimental details of the proBE model, including a brief overview of the dataset, parameters, and evaluation metrics, followed by the evaluation findings, comparison analysis, and ablation study.

### A. Dataset and Parameters

The proposed approach, proBE is evaluated on real world tweet disaster datasets named CrisisLexT26 [27] and CrisisLexT6. Dataset contains tweets for 26 and 6 disaster events, respectively. Each tweet is labelled with information source, information type, relevancy and Informativeness. Tweets that are related and informative are considered for evaluation. To avoid class imbalance, the minimum quantity of tweets for all events was evaluated, and around 400 tweets were chosen. Some of the events, such as the Venezuela refinery event 2013, have non-English tweets. When non-english tweets are eliminated, the amount of English tweets in such events is drastically reduced and thus removed. Following such filtration, the dataset contains 18 events, each with 400 tweets from CrisisLex26 and only two events are different in CrisisLex6 than the other. To this end, considering both the datasets, 20 events are filtered out for evaluation with 400 tweets each as shown in the table I. Data is divided into training, validation and testing with 60, 20, 20 ratio, respectively. Due to the Few-shot learning setting, there is no overlapping in training, validation and testing classes.

TABLE I: Dataset statistics

Datasets	#Events	#Events after filtration	Total #events	Total tweets
<b>Crisis LexT26</b>	26	18	20	8000 (approx)
<b>Crisis LexT6</b>	06	02		

1) *Pre-processing*: Tweets contain noise in the form of slang and casual language, making them unsuitable for direct feeding to the model. The data must be pre-processed before it can be fed into the model. Cleaning methods include the removal of punctuation, URLs, retweets, duplicate tweets, digits, white spaces, and other inappropriate data. Following that, non-english tweets were filtered out and the remaining tweets will be converted to lower case, and any residual stop-words will be removed with spaCy<sup>1</sup>.

The model described in the section IV is implemented in python 3.10 and is executed using open source library PyTorch<sup>2</sup> which is mainly used for experimental evaluation of the deep learning models. Few-shot model is evaluated using 3-way 5 or 10 shots and 4-way 5 or 10 shots. Embedding vectors are initialised with BERTweet with the dimensions of 768. BERTweet is trained on large Twitter corpus. Above model is trained using Stochastic Gradient Decent (SGD) optimiser. Several tests are carried out and the learning rate is set to 0.1, decaying after 1000 iterations. Model training is also verified for numerous sets of iterations before deciding on 4000 iterations for training and 1500 iterations for validation. A summary of hyper-parameters are presented in table II.

TABLE II: A summary of hyper-parameter settings

Hyperparameter	Values
Batchsize	5
Dropout	0.2
# filters (CNN)	64
Window size (CNN)	3
Weight decay	$10^{-5}$
Learning rate decay	0.1

### B. Evaluation Metrics

The proposed model is evaluated using four standard evaluation metrics: `-precision`, `recall`, `F-score` and `accuracy` where, precision can be calculated as number of true positives (TP) to the total number of true positive and false positives (FP). Recall is measured by comparing the total number of true positives across

all classes to the total number of true positives and false negatives (FN) across all the classes. Precision is appropriate when minimising false positives is the goal, while recall is appropriate when minimising false negatives is the goal. Further, F-score is the harmonic mean of the these above two factors. Finally, accuracy is defined as number of correct predictions to total number of predictions.

Results are measured in above mentioned metrics as described in equations 9, 10, 11 and 12, respectively.

$$Precision = \frac{TP}{TP+FP} \quad (9)$$

$$Recall = \frac{TP}{TP+FN} \quad (10)$$

$$F-Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (11)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (12)$$

Where, TP represents the total number correctly classified event query tweets. FP represents the total number of incorrectly classified event query tweets. TN represents the total number of correctly classified non-event query tweet. Finally, FN represents total number of incorrectly classified non-event query tweet.

### C. Evaluation Results and Comparative Analysis

All of the measures listed above are chosen to assess the model's performance. When dealing with new data in testing, only accuracy is not ideal since it just analyses the amount of correct predictions and does not consider the relative value of false positives and false negatives. To this end, it is crucial to take all measures into account in order to obtain a comprehensive understanding of the performance across different classes. Furthermore, state-of-the-art methods are also considered for comparison and proposed approach is compared with vanilla prototype [19] where, the authors introduced metric-based prototype networks for few-shot learning based on the notion that each class can be represented by the mean of its examples in a representation space learned by a neural network. Models are also compared to other encoder, acknowledging the significance of employing

<sup>1</sup><https://spacy.io/>

<sup>2</sup><https://pytorch.org/>



TABLE III: Evaluation results of proposed approach in comparison to the state-of-the-arts on 3-way 5 and 10 shot

Approach		3-way 5 shot				3-way 10 shot			
Model	Encoder	Accuracy	F-score	Precision	Recall	Accuracy	F-score	Precision	Recall
proBE	BERTweet	70.16	79.00	80.00	80.00	76.18	81.00	80.00	80.00
Prototypical [19]	BERTweet	61.39	60.00	67.00	60.00	63.26	69.00	70.00	69.00
proBE	CNN	33.34	17.00	78.00	33.00	33.35	17.00	33.00	78.00
Prototypical [19]	CNN	33.30	26.00	56.00	33.00	33.40	26.00	56.00	33.00

TABLE IV: Evaluation results of proposed approach in comparison to the state-of-the-arts on 4-way 5 and 10 shot

Approach		4-way 5 shot				4-way 10 shot			
Model	Encoder	Accuracy	F-score	Precision	Recall	Accuracy	F-score	Precision	Recall
proBE	BERTweet	61.05	65.00	67.00	65.00	69.26	77.00	77.00	77.00
Prototypical [19]	BERTweet	50.05	57.00	54.00	55.00	51.33	51.00	53.00	51.00
proBE	CNN	24.99	20.00	44.00	25.00	25.01	23.00	24.00	26.00
Prototypical [19]	CNN	25.00	10.00	81.00	25.00	25.11	10.00	81.00	25.00

an encoder to represent the text as an early step that can effect performance. CNN is chosen for comparison as mentioned by the authors of [26] where, text is initially embedded with GloVe and then CNN is applied to get the aggregated hidden annotation of each word and its position by a convolution kernel with the window size 3. Efficacy of proposed approach with mentioned baseline methods is represented in tabular form in tables III and IV. It can be observed that our approach significantly outperforms the baselines, and the addition of context with the help of BERTweet makes a substantial difference because contextual representation becomes more effective when dealing with limited information. In addition, training and validation accuracy comparison is also depicted in figure 4 and clearly shows that our model clearly outperforms the training and validation accuracy as well.

## VI. CONCLUSION AND FUTURE WORK

This study sought to propose a solution to the problem of event detection in online social networks following the failure of labelling the enormous data due to high cost of annotation and inability of detecting new events. The objective is to employ less labelled data (few samples) and generalise more to identify new events in the amorphous data of online social networks. To this end, a model is proposed where, the encoding of tweets is highlighted first as it is the integral aspect of data representation. BERTweet is employed for tweet encoding which is specifically trained on Twitter data to capture context utilising Twitter’s built-in and differentiating features. Key characteristics of social network data such as hashtags and mentions are also incorporated in data representation due to their indispensable role in Twitter data. The attentive prototype module is subsequently employed in the following step, where, tweet and feature attention are applied with the assumption that some tweets and features have more information and context and hence deserve more attention. The strategy of drawing data for training and testing using different distributions in few-shot learning aids to help detecting new events and also address the covariant-shift issue that can be seen in dynamic data of online social networks. Proposed model has shown significantly better results as compared to baselines in unstructured and informal social network data. Exploration of more informative feature extraction, such as using a graph-like structure to include other aspect of online social networks and extract information at a deeper level, as well as application of the work in the real world, could be a potential future work.

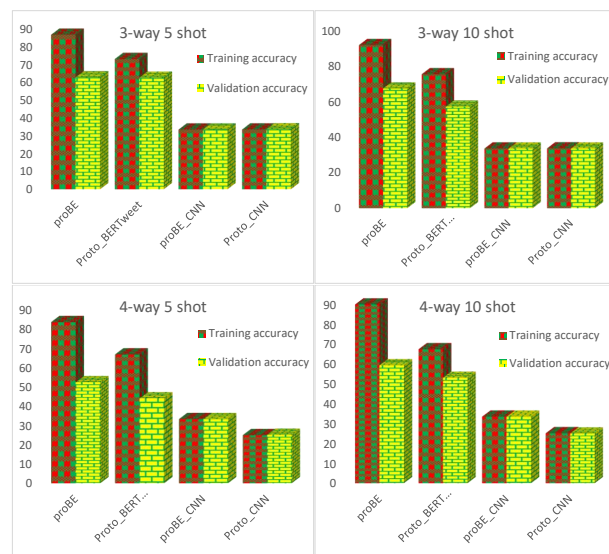


Fig. 4: Training vs validation accuracy comparison with state-of-the-arts on 3 and 4-way(5 and 10 shot)

## REFERENCES

- [1] C. Li, A. Sun, and A. Datta, “Twevent: segment-based event detection from tweets,” in *Proceedings of the 21st ACM international conference on Information and knowledge management*, 2012, pp. 155–164.

- [2] K. Morabia, N. L. B. Murthy, A. Malapati, and S. Samant, "Sedtwik: segmentation-based event detection from tweets using wikipedia," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, 2019, pp. 77–85.
- [3] H. Hettiarachchi, M. Adedoyin-Olowe, J. Bhogal, and M. M. Gaber, "Embed2detect: temporally clustered embedded words for event detection in social media," *Machine Learning*, vol. 111, no. 1, pp. 49–87, 2022.
- [4] S. Sharma, M. Abulaish, and T. Ahmad, "Kevent—a semantic-enriched graph-based approach capitalizing bursty keyphrases for event detection in osn," in *2022 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE, 2022, pp. 588–595.
- [5] M. Abulaish, S. Sharma, and M. Fazil, "A multi-attributed graph-based approach for text data modeling and event detection in twitter," in *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*. IEEE, 2019, pp. 703–708.
- [6] H. Ji and R. Grishman, "Refining event extraction through cross-document inference," in *Proceedings of ACL-08: Hlt*, 2008, pp. 254–262.
- [7] V. D. Lai, "Event extraction: A survey," *arXiv preprint arXiv:2210.03419*, 2022.
- [8] M. Hasan, M. A. Orgun, and R. Schwitter, "A survey on real-time event detection from the twitter data stream," *Journal of Information Science*, vol. 44, no. 4, pp. 443–463, 2018.
- [9] F. Atefeh and W. Khreich, "A survey of techniques for event detection in twitter," *Computational Intelligence*, vol. 31, no. 1, pp. 132–164, 2015.
- [10] D. Ahn, "The stages of event extraction," in *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, 2006, pp. 1–8.
- [11] Y. Hong, J. Zhang, B. Ma, J. Yao, G. Zhou, and Q. Zhu, "Using cross-entity inference to improve event extraction," in *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, 2011, pp. 1127–1136.
- [12] T. Nguyen and R. Grishman, "Graph convolutional networks with argument-aware pooling for event detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.
- [13] S. Liu, Y. Chen, K. Liu, and J. Zhao, "Exploiting argument information to improve event detection via supervised attention mechanisms," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1789–1798.
- [14] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015, pp. 167–176.
- [15] V. D. Lai and T. H. Nguyen, "Extending event detection to new types with learning from keywords," *arXiv preprint arXiv:1910.11368*, 2019.
- [16] S. Deng, N. Zhang, J. Kang, Y. Zhang, W. Zhang, and H. Chen, "Meta-learning with dynamic-memory-based prototypical network for few-shot event detection," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020, pp. 151–159.
- [17] S. Thrun, "Is learning the n-th thing any easier than learning the first?" *Advances in neural information processing systems*, vol. 8, 1995.
- [18] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," *Advances in neural information processing systems*, vol. 29, 2016.
- [19] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.
- [20] G. Koch, R. Zemel, R. Salakhutdinov *et al.*, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2. Lille, 2015.
- [21] M. Yu, X. Guo, J. Yi, S. Chang, S. Potdar, Y. Cheng, G. Tesauero, H. Wang, and B. Zhou, "Diverse few-shot text classification with multiple metrics," *arXiv preprint arXiv:1805.07513*, 2018.
- [22] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *International conference on machine learning*. PMLR, 2016, pp. 1842–1850.
- [23] W. Liu, J. Pang, N. Li, F. Yue, and G. Liu, "Few-shot short-text classification with language representations and centroid similarity," *Applied Intelligence*, pp. 1–12, 2022.
- [24] M. M. A. Qudar and V. Mago, "Tweetbert: a pretrained language representation model for twitter text analysis," *arXiv preprint arXiv:2010.11091*, 2020.
- [25] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, 2016, pp. 1480–1489.
- [26] T. Gao, X. Han, Z. Liu, and M. Sun, "Hybrid attention-based prototypical networks for noisy few-shot relation classification," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 6407–6414.
- [27] A. Olteanu, C. Castillo, F. Diaz, and S. Vieweg, "Crisislex: A lexicon for collecting and filtering microblogged communications in crises," in *Proceedings of the international AAAI conference on web and social media*, vol. 8, no. 1, 2014, pp. 376–385.