Many Models Make Light Work: Using Multi-ML Models for Improved Rumor Detection

Anonymous Author(s)

ABSTRACT

There have been many efforts to detect rumors using various machine learning (ML) models, but there is still a lack of understanding their performance against different rumor topics and available features, resulting in a significant performance degrade against completely new and unseen (unknown) rumors. To address this issue, we investigate the relationship between ML models, features and rumor topics to select the best rumor detection model under specific conditions using 13 different ML models. Our experiment results demonstrate that there is no clear winner among the ML models in all different rumor topics with respect to the detection performance. Based on this observation, we propose an ensemble solution that consists of the top three ML models (Random Forest, XGBoost and Multilayer perceptron) based on our findings, as well as combining all the ML models we tested. Our experimental results showed significant improvements in detecting unknown rumors using the proposed ensemble solution, achieving a 0.79 F1 score, compared to using a single ML model which only achieved 0.58 F1 score (i.e., an increase of 0.21 F1 score) – making it a much more attractive solution to detect rumors in practice.

CCS CONCEPTS

• **Networks** → *Social media networks*; *Social media networks*; • **Theory of computation** → **Machine learning theory**.

KEYWORDS

Feature Analysis; Machine Learning; Rumor Detection; Social Media; Twitter

ACM Reference Format:

Anonymous Author(s). 2020. Many Models Make Light Work: Using Multi-ML Models for Improved Rumor Detection. In *WWW 2020, April 20 - 24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/xxxxxxx.xxxxxxx

1 INTRODUCTION

Millions of people share information on social networks everyday, which also allow false information to spread rapidly and mislead users. For instance, rumors affected political decisions (e.g., the US election in 2016 [8]), stock market [1], people's actions during natural disasters (e.g., 2010 Chile earthquake [23] and 2012 hurricane Sandy [10]), and the safety during accidents (e.g., 2013 Boston marathon bombing [27]). Although those rumors are difficult to detect, they exhibit certain features distinctive from non-rumors that could be used to detect them. For example, Horne and Adali [12] identified that fake news tends to emphasize the title using easy to understand words and phrases. Kwon *et al.* [17] also proposed a rumor detection model using the relative importance of user, structural, linguistic, and temporal features for rumor classification on Twitter. Nevertheless, the spread of rumors is still prevalent.[1] Hence, improving rumor detection systems still remains as an important challenge today.

Various machine learning (ML) models have been proposed to detect rumors automatically [16, 29]. Their work focused on improving the performance of their models for selected rumor events (e.g., crime-related tweets [36]), but mostly they were applied in the post-analysis phase (i.e., after the rumor has spread/collected). Consequently, many ML models were able to achieve competitive performance (in terms of accuracy, F1 scores, etc.) because they were trained specifically with the selected rumor events. Hence, using them to detect new rumor events significantly degrades the performance of existing ML models, which we demonstrate later in Section 4. Further, available features also significantly affect the performance of rumor detection [28]. Consequently, retraining those ML models with a new dataset every time to maintain the performance is impractical. Hence, it is important to develop a solution that maintains high performance under various environmental conditions such as detecting new rumor events with changing features.

To address the aforementioned problems, a better rumor detection solution is needed, which takes into account various environmental conditions. As a first step, we analyze the relationship between the rumor topics, features and ML models to understand the factors that affect their rumor detection performances, particularly for new rumor events (rumor event is an instance of the rumor topic, and we will denote them as *unknown rumors*). To achieve this goal, we conduct several experiments using *PHEME* dataset [35], which contains nine distinctive events across five rumor topics.[2] First, we collect the features used in existing studies such as [3], and then we categorize them into three *feature groups*. Next, we build 13 ML models to detect rumors with respect to feature groups and compare their performances. The results indicate that there is no clear winner, and the rumor detection performances differs significantly when rumor events and available features are changed. This indicates that using only a single ML model is not sufficient for effective rumor detection.

Considering our findings, we propose an ensemble solution combining multiple ML models as our second step. Our experimental analysis results show that the ensemble solution combining the top

WWW 2020, April 20 - 24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-5639-8/xx/xx.
https://doi.org/10.1145/xxxxxxx.xxxxxxx

[1] https://www.snopes.com/fact-check/

[2] The *PHEME* dataset served as a ground truth, which has been used previously for rumor detection, such as in [2]. Hence it is a suitable dataset to demonstrate our work and produce comparative results.

3 performing ML models achieved 0.79 F1 score, outperforming the performance of using only a single model that had the best F1 score of 0.58 (i.e., an increase of 0.21 F1 score). Hence, using the proposed ensemble solution can improve the overall rumor detection performance under various environmental conditions (i.e., different rumor events and different available features). The contributions of our paper are summarized as follows.

- To segregate ML models based on the performance with varying features and/or rumor topics;
- To rank ML models for detecting rumors with respect to rumor topics and available features;
- To propose an ensemble solution for improved unknown rumor detection using multi-ML models.

The rest of the paper is organized as follows. Section 2 presents the related work on rumor detection. Section 3 presents the overall process of evaluating ML models for rumor detection. Section 4 presents the experimental results with various ML models and discusses the result. Section 5 presents an ensemble solution for improved rumor detection using multi-ML models. Section 6 discusses our findings and limitations. Finally, Section 7 concludes this paper.

2 RELATED WORK

2.1 Rumor detection models

Various ML models have extensively been studied in the past decade to enhance their rumor detection capabilities [15, 34]. For instance, Yang *et al.* [32] identified rumors on Twitter using Logistic Regression (LR), Naive Bayes (NB) and Random Forest (RF) based on hot topics detection. Zhao *et al.* [33] presented enquiry-based rumor detection, but their result reported the performance using precision only, which achieved 0.52. Further, the solution relies on the existence of enquiries for the rumor event (i.e., if there are no sufficient queries about the rumor, it may not be detected).

Although a significant performance improvement was observed, the performance of existing ML techniques is significantly degraded for unseen and unexpected rumor events (see Section 4). To overcome this limitation, we analyze the relationships between rumor topics, features and detection models, and suggest an ensemble solution using multi-ML models to improve the detection performance (see Section 5).

2.2 Rumor detection features

There are numerous features that can characterize rumors, such as temporal [18], structural [31], message [7], network [7], user [19], content [19], etc. However, using more features does not necessarily improve the detection performance [7]. Castillo *et al.* [3] identified classes of features associated with rumors, and demonstrated that context and propagation features are specifically effective for rumor detection, indicating the importance of understanding the context of the rumor and the graph patterns in the social network. Kwon *et al.* [18] proposed a rumor detection method using temporal, linguistics, and structural features on Twitter. Using their proposed features, they were able to achieve up to 0.89 F1 score, indicating the importance of using appropriate features. Kwon *et al.* [17] analyzed the relative importance of user, structural, linguistic, and temporal features for rumor classification on Twitter. Based on the results, they suggested a new rumor classification algorithm that achieves competitive accuracy over both short and long time windows. Ma *et al.* [21] used a time series model to capture temporal characteristics of rumors, achieving up to 0.89 F1 score. However, other features are not compared in their work. Giasemidis *et al.* [7] extracted 87 features classified into three categories of message-based, user-based, and network-based in a dataset containing 72 rumors on Twitter, and conducted a study to identify rumors using various machine learning algorithms. The best F1 score achieved was using the Decision Tree (DT) with the value 0.97. Although various features are explored to improve the performance of rumor detection models, previous work did not take into account how different features performed when they are used for other rumor topics or events. In addition, the ability to collect some of those features is not feasible until a certain amount of rumor has spread, limiting their usage in rumor detection.

2.3 Deep learning for rumor detection

Another rising technique for rumor detection is deep learning (DL). Guo *et al.* [9] presented a hierarchical LSTM network with social attention to detect rumors with an accuracy of 0.84 and F1 score of 0.83 for a Twitter dataset. Chen *et al.* [4] presented a deep attention model built on RNNs, which selectively learns temporal representations of sequential posts to identify rumors. Wang *et al.* [28] presented a Neural Model using Dynamic Propagation Structures. They also presented the performance variation with different features, which showed a significant difference with the highest F1 score of 0.83 with content and structure features combined. Ruchansky *et al.* [26] presented a hybrid DL model *CSI* using LSTM to capture the temporal pattern of user activity and RNN to characterize the user behavior to detect fake news using Twitter and Weibo datasets. They grouped the related features into the text of an article, the user response it receives, and the source users promoting it. Ma *et al.* [22] presented generative adversarial learning for rumor detection, which strengthened the discriminator to learn stronger rumor indicative representations.

Although DL-based models showed a performance improvement in rumor detection, previous work did not evaluate their model's performance when used for detecting new or unseen (i.e., unknown) rumors [4, 15]. Further, DL-based models require a large set of data for training [5], as well as LSTM-based models requiring sequential data. Hence, such methods cannot be used when rumor features do not exhibit sequential properties (e.g., structural and user features). That is, when the rumor topics and features are changed, it can have a significant impact on the performance of the models. To validate, we implemented an equivalent model[3] to CSI presented in [26], and the best result we found was accuracy of 0.56 and F1 score of 0.50 when used for detecting unknown rumor events. Therefore, we conclude that the performance of DL-based models is useful for known rumor events, but they are significantly worse when used for detecting unknown rumors with different available features, which is a huge constraint in practice.

[3]The implementation of the CSI-equivalent model can be found at *hidden.*

3 EVALUATION FRAMEWORK OF ML-BASED RUMOR DETECTION MODELS

Rumor detection using ML models normally has four main steps: (1) data collection, (2) feature extraction, (3) ML model training, and (4) performance evaluation, which are described as follows.

Step 1. Data collection: Raw data can be collected from social media, such as Twitter, using various APIs, which are then preprocessed to extract features. In this paper, the data collection step is simplified using a publicly available dataset named *PHEME* [35], which contains nine individual rumor events (further grouped into five rumor topics) that are already labelled (i.e., rumor or non-rumor). Due to the size of the dataset being small for "Ebola" (Medical) and "Gurlitt" (Political) events, we combined them to create a new dataset *Mixed* with multiple rumor topics. Table 1 shows the statistics of the labelled tweets for each event in the PHEME dataset[4]. Existing rumor detection techniques can also be used to collect and classify rumor data. However, as we find out later in Section 4, using existing techniques, especially ML model-based ones, may not be accurate to produce correctly labelled data due to their lack of performance against unknown rumor events.

Table 1: Classifying PHEME dataset rumor topics [35] (R: Rumor, NR: Non-Rumor).

Rumor Topic	Event	R	NR	Total
Crime	Sydney Siege	522	699	1,221
	Ottawa Shooting	470	420	890
	Ferguson	284	859	1,143
	Charlie Hebdo	458	1,621	2,079
Politic	Putin Missing	126	112	238
Entertainment	Prince	229	4	233
Impact	Germanwings Crash	238	231	469
Mixed	Ebola & Gurlitt	75	77	152
Total		2,402	4,023	6,425

Step 2. Feature extraction: First, we categorize rumor features into three groups similar to the work in [7]: Content (C), Propagation (P), and User (U). There are many other features that can be collected, but not all of them are useful for detecting rumors. For example, Castillo *et al.* [3] showed useful features for detecting rumors in Twitter. Based on their findings, we selected the top 14 features to be used in this paper. Table 2 summarizes the rumor features into three feature groups.

Step 3. ML models and training: There are many ML models available today, and they can be categorized into five classes [14]: (1) logic-based (e.g., Decision Tree), (2) perceptron-based (e.g., Multilayer Perceptron), (3) Statistical learning (e.g., Naive Bayes), (4) instance-based (e.g., K-Nearest Neighbor), and (5) support vector machines (e.g., Linear and RBF SVM). We selected 13 most popularly used ML models covering all ML categories, and implemented them for comparisons[5], which are (of no particular order): Logistic Regression (LR), Decision Tree (DT), K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Gaussian Process (GP), Random Forest (RF), Naive Bayes (NB), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Gradient Boosting (GBM), Adaptive Boost (AdaBoost), XGBoost, and Multilayer perceptron (MLP).

[4]Note that rumors and non-rumors are shown as **R** and **NR**, respectively. Also, rumor topics are populated from https://www.kaggle.com/sfarooqi/news-classification.
[5]scikit-learn library was used to implement them in Python [25]

Table 2: Rumor feature groups with selected features.

Group	Feature	Description
Content (C)	has question mark	tweet contains question mark(s)
	has exclamation mark	tweet contains exclamation mark(s)
	has URL	tweet contains URL
	has hashtag	tweet contains hashtags
	unigram bow vector	the number of unique words (i.e., token) in the tweet
	POS tagging	vector representation of grammatical categories of each token
Propagation (P)	time span	timespan between account registration and posting time
	retweet count	the number of retweeted tweets
	favorite count	the number of likes on the tweet
User (U)	is verified	user is verified by Twitter
	has description	User has personal description
	followers count	the number of followers
	friends count	the number of friends
	statuses count	the total number of tweets posted

To measure their performance in rumor detection, we compute Accuracy, Precision, Recall, and F1 score [13] as follows.

- **Accuracy**: the proportion of correctly classified tweets;
- **Precision**: the proportion of tweets classified as rumors that actually are rumors;
- **Recall**: the proportion of rumors that were accurately classified;
- **F-measure**: the harmonic mean of *precision* and *recall*.

In our experiment, we divide the dataset differently for the single ML models and the ensemble solution.

- **Single ML model:** The dataset is split into *train*:*test* ratio for training and testing purposes.
- **Ensemble solution:** The dataset is split into *train*:*validate*:*test* for the train set, validation set and test set, respectively.

The training set is used to optimize the ML models' configurations. The validation set is used for the ensemble solution to assign appropriate weight values to different ML models in the set (details are given later in Section 5). Lastly, the test set is used to measure the performance for all ML models and the ensemble solution.

For testing the detection of known rumor events, we train the models using the rumor event data. This approach is the same as previous work. To test the detection of unknown rumor events, we train the models using *other* rumor event data, which in effect treats the testing rumor dataset as previously unseen.

Step 4. Performance evaluation: As described in Section 2, previous work achieved high performance, but based on knowing

the rumor event(s) and using it to train their models, which limits their practical usage (i.e., need to know what is rumor beforehand, as well as collecting such data for training). To evaluate their performance under real-world conditions, we also consider using them to detect rumor events that have not been used previously (i.e., the rumor data is not used for training). Hence, two types of tests are conducted: (1) cross validation using known rumors, and (2) testing for unknown rumor events. The performance of them for the two types of tests are presented in the following sections.

4 SINGLE ML MODEL RUMOR DETECTION

This section evaluates the performance of existing ML models for cross validation (i.e., detecting known rumor events) and unknown rumor events. For the training purpose, we set $train = 8$ and $test = 2$ (i.e., 80% training set and 20% test set).[6] Results are shown in Figure 1a for known rumor events, and Figure 1b for unknown rumor events. Here, a darker shade means high performance (i.e., tending towards 1), and light shade means low performance (i.e., tending towards 0). We analyze the results from the point of rumor events and features in sections 4.1 and 4.2, respectively.

4.1 Effects of topics for rumor detection

4.1.1 Known rumor detection. First, we evaluate the performance of rumor detection models with varying rumor topics. To better understand each of the rumor events, Table 3 shows some key statistics of the dataset.[7] In general, we notice that the non-rumor (NR) tweets have higher favorite, retweet and friends counts (average 1.84, 1.09 and 1.12 times higher, respectively, without "Ebola & Gurlitt" event), as well as the number of keywords used and the amount of unique keywords (labeled *Unique* in the table) across all tweets. However, looking at the individual event, the ratio differs significantly. For example, events "Prince" and "Ebola & Gurlitt" has a significantly lower ratio than other events.

Figure 1a shows that the best performing model changes with events, where each event is associated with a specific rumor topic: darker the color, higher F1 score achieved. Our evaluation results are summarized in Table 4, which shows the best ML model for detecting known rumors based on their F1 score achieved for each rumor event–the best model and its F1 score are highlighted in bold font style. From this, we observe that MLP achieved the highest F1 score for the "Charlie Hebdo" event, but LR achieved the highest F1 score for the "Ebola & Gurlitt" event, and GBM for "Ottawa Shooting" event. This indicates that even if the rumor event falls within the same category, such as "Crime" for the above examples, the best performing model varies. However, there is a group of models that appear more than others, which are NB, GBM and RF for "Crime", RF for "Entertainment" and "Impact", and MLP for "Politic" and "Mixed" topics.

Looking from the statistics point of view (i.e., in relation to Table 3, there is no indication of knowing whether the rumor detection performance will be good or not. That is, basic statistics of events cannot be used directly to determine whether detecting rumors using ML models will be effective or not. On the other hand, looking from the rumor to non-rumor (R-to-NR) ratio of the dataset also provides a better performance estimate, where high performance was achieved when the R-to-NR ratio is close to being even (i.e., 50:50). For example, the rumor events that are easily detected are (e.g., performance greater than 0.9); (1) German Wings (0.968 F1 score with R-to-NR ratio 50.7:49.3), (2) Ebola & Gurlitt (0.935 F1 score with R-to-NR ratio 49.3:50.7), and (3) Ottawa Shooting (0.920 F1 score with R-to-NR ratio 52.8:47.2).

Table 3: Statistics on rumor events.

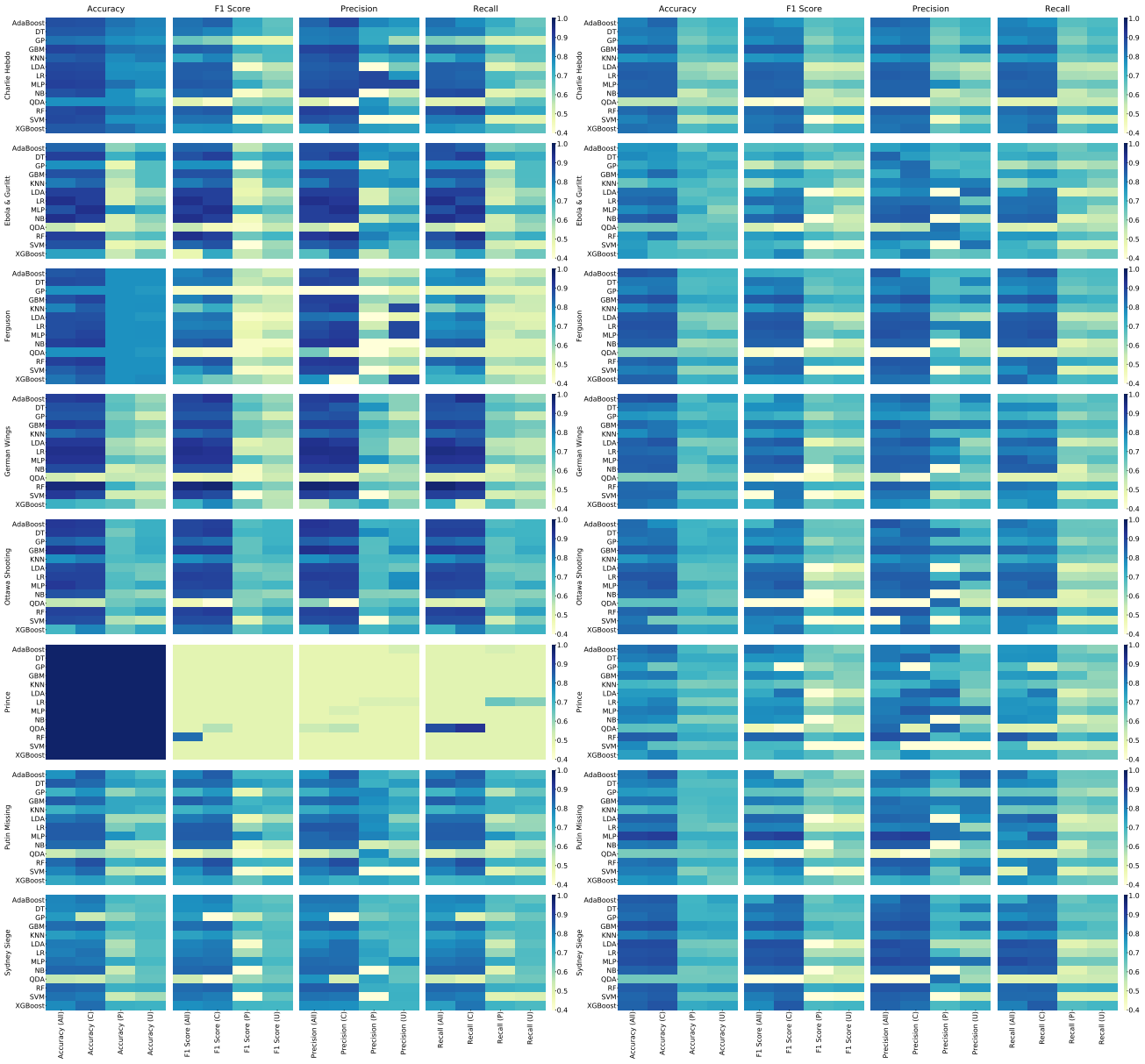
Event	type	Favorite	Retweet	Friends	Keywords	Unique
Sydney Siege	NR	526	490	3820	2509	1855
	R	225	341	4912	1566	912
	Ratio	2.34	1.44	0.78	1.60	2.03
Ottawa Shooting	NR	243	353	4273	1580	1133
	R	105	325	2041	1286	839
	Ratio	2.32	1.09	2.09	1.23	1.35
Ferguson	NR	193	453	5176	3093	2498
	R	113	294	4513	1173	578
	Ratio	1.71	1.54	1.15	2.64	4.32
Charlie Hebdo	NR	233	455	4362	5183	4539
	R	103	355	1626	1347	703
	Ratio	2.26	1.28	2.68	3.85	6.46
Putin Missing	NR	9	16	1384	706	575
	R	6	15	4022	632	501
	Ratio	1.56	1.04	0.34	1.12	1.15
Prince	NR	3	2	746	24	15
	R	6	11	1598	794	785
	Ratio	0.53	0.21	0.47	0.03	0.02
German Wings	NR	109	238	2172	996	764
	R	50	226	6730	784	552
	Ratio	2.17	1.06	0.32	1.27	1.38
Ebola & Gurlitt	NR	1	2	1766	376	309
	R	93	262	5523	305	238
	Ratio	0.01	0.01	0.32	1.23	1.30

Table 4: Best ML model for detecting known rumors.

Rumors	Features			
	All	C	P	U
Crime				
Sydney Siege	**NB 0.835**	NB **0.839**	MLP 0.745	RF 0.715
Ottawa Shooting	**GBM 0.920**	**GBM 0.920**	DT 0.699	RF 0.733
Ferguson	NB 0.850	**RF 0.868**	XGB 0.592	RF 0.591
Charlie Hebdo	**MLP 0.859**	RF 0.855	XGB 0.731	RF 0.698
Politic				
Putin Missing	RF 0.854	**MLP 0.874**	DT 0.782	LR 0.726
Entertainment				
Prince	**RF 0.828**	QDA 0.570	All 0.495	All 0.495
Impact				
German Wings	RF 0.957	**RF 0.968**	MLP 0.685	GBM 0.622
Mixed				
Ebola & Gurlitt	**LR 0.935**	**MLP 0.935**	MLP 0.710	DT 0.735

To better understand their differences, the performance measure for each event is presented in Figure 2, showing the average performance of all ML models across the selected rumor events. This result shows that the performance to detection different rumor

[6]The preliminary study using the combined dataset for training, as well as evaluating the variations to the training set size is carried out in our preliminary work in *hidden*.
[7]Keywords are found after removing stop words using the natural language processing library NLTK: https://www.nltk.org/.



(a) Known rumor topics. **(b) Unknown rumor topics.**

Figure 1: Performance evaluation of ML models.

events differ even if they belong to the same rumor topic. Also, the proportion of rumors to non-rumors data can degrade the performance significantly (e.g., poor detection performance for the "Prince" event, which has 1:57 non-rumor to rumor ratio). Based on these results, we cannot conclude that there is a single ML model that can be used to achieve the best rumor detection performance (further statistical analysis is shown later in Section 5.3).

4.1.2 Unknown rumor detection. Second, we evaluate the results from detecting unknown rumors. To do this, we trained our selected ML models using some rumor events only, and then used the remaining dataset for testing. For each event dataset E, we used all other events to train ML models and evaluate the performance of the models with E as the testing samples. Figure 1b shows the performance of detecting unknown rumors, and Table 5 shows the best ML models for detecting unknown rumors based on their F1 scores. Similar to detecting known rumors, there is a group of models that appear more than others, which are MLP, LR, GBM for "Crime" topic, MLP for "Politic" and "Impact" topics, RF for "Entertainment" topic, and NB for "Mixed" topic. There are some

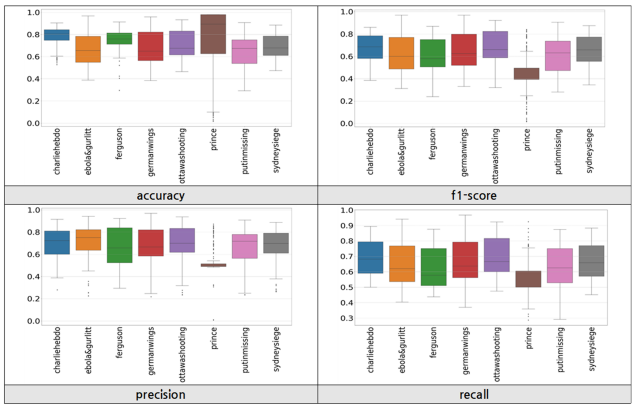


Figure 2: Performance w.r.t. topics (known rumor).

overlaps, but largely the best performing models have changed across the rumor topics.

The best F1 scores achieved are all ranging between 0.84 and 0.90 (i.e., smaller variance compared with known rumor detection with the F1 score range of 0.82 to 0.97). Again, it is difficult to deduce any meaningful detection performance based on the statistical measures from Table 3. Also, the R-to-NR ratio is no longer a good performance estimate. For example, "Putin Missing" event had the highest F1 score of 0.906, followed by "Sydney Siege" event with F1 score of 0.875, which had less even R-to-NR ratio than "German Wings" event. Further, "Ferguson" and "Charlie Hebdo" events, which have around 25:75 R-to-NR ratio, achieved 0.868 and 0.858 F1 scores respectively, a comparable F1 score to "German Wings" and "Ebola & Gurlitt" events, which only achieved 0.867 and 0.849 F1 scores respectively. Hence, the R-to-NR ratio of the dataset cannot be used to determine the performance of rumor detection when dealing with unknown rumors.

In addition, detecting known events overall achieved higher performance in comparison to detecting unknown rumors. Based on our experimental results, three main observations are made that stood out in comparison to detecting known rumor events:

Table 5: Best ML model for detecting unknown rumors.

Rumors	Features			
	All	C	P	U
Crime				
Sydney Siege	LDA 0.874	**MLP 0.875**	RF 0.717	RF 0.729
Ottawa Shooting	GBM 0.849	**MLP 0.861**	RF 0.699	RF 0.739
Ferguson	LR 0.864	**LR 0.868**	GBM 0.725	RF 0.713
Charlie Hebdo	**GBM 0.858**	GBM 0.854	GBM 0.676	RF 0.722
Politic				
Putin Missing	MLP 0.889	**MLP 0.906**	RF 0.678	MLP 0.680
Entertainment				
Prince	RF 0.837	**RF 0.839**	RF 0.682	RF 0.717
Impact				
German Wings	MLP 0.850	**MLP 0.867**	RF 0.685	GBM 0.622
Mixed				
Ebola & Gurlitt	NB 0.834	**NB 0.849**	RF 0.695	AdaB 0.690

(1) **Degraded performance in general**: Treating the rumor event as unknown resulted in poorer F1 score achieved. For example with the "German Wings" event, the best F1 score is 0.867 using MLP treating it as an unknown rumor, while the best F1 score is 0.968 using RF treating it as a known rumor. [8]

(2) **Best performing ML model has changed**: The best performance achieved between known and unknown rumor detection has changed significantly. In fact, only the "Politic" topic achieved the best performance using the same model (i.e., MLP), and all other topics used a different ML model to achieve the best performance (see Tables 4 and 5).

(3) **The R-to-NR ratio does not affect the performance of ML models**: We can no longer estimate the performance of unknown rumor detection based on the R-to-NR ratio, which was possible when the rumor event was known.

Although the detectors do not know anything about the new rumor events, they were able to characterize some of the rumor features from other rumor events used for training. As mentioned by Horne *et al.* [12], rumors can have a distinctive structure compared with non-rumors, so such structures can be identified as a feature for the ML models to detect other rumor topics and events. However, as pointed out in (1), the overall performance has degraded because some features may not well represent the new rumor event (e.g., content-based features may not be very accurate due to keywords being significantly different). Also, the best ML model to use for different events has changed as pointed out in (2), and only the rumor topic *Politic* had the same ML model to achieve the best performance. Both observations (1) and (2) (as well as observation (3)) indicate that when detecting unknown rumors, we cannot rely on currently best performing models that were trained using known rumor events.

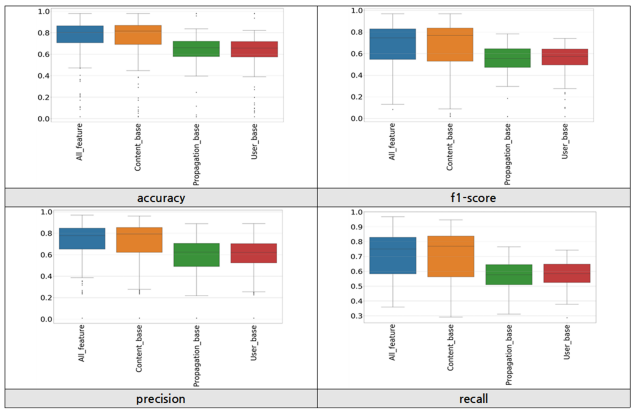


Figure 3: Performance w.r.t. features (known rumor).

4.2 Effects of features for rumor detection

If we look at different features, for example, the "Charlie Hebdo" event, MLP had the highest F1 score when all features are available,

[8] There are some exceptions, but the largest improvement is with "Sydney Siege" event with the improvement of 0.036 F1 score, which is only a small improvement. We suspect the model benefited from training using other events from the same topic.

but RF was the best with content-only features, and XGBoost with propagation-only features. Rather surprisingly, using content-only features for some rumor events achieved higher F1 score than using all features (i.e., in the cases of *Ferguson, German Wings, Putin Missing and Sydney Siege*). Hence, our results show that regardless of detecting known or unknown rumors, using content-based features influenced the performance most significantly (unless the dataset used is skewed for known rumor detection, such as "Prince" event). Figure 3 shows the variation in the performance with respect to different features for known rumors. Because we aggregated all rumor events together, the fine-grained differences between rumor events are not captured. However, it still shows the overall difference between features. This result is also similar to detecting unknown rumors. It clearly shows that using all features or content-only features result in significantly higher performances.

Comparing *all features* v.s. *content-only feature*, the content-based features had a higher performance overall even for unknown rumor detection (i.e., even if we do not know what rumor we are looking for, the emerging rumor event may have similar characteristics [12]). On the other hand, using the user-based feature performed the worst. Looking at within each feature group, there is no one particular ML model that could be selected as the best performing one due to a large variation (similarly with rumor events above). Hence, no one particular ML model is more sensitive to different feature groups than the others.

4.3 Verdict

The results show that if the rumor is known, we can significantly improve rumor detection performance. Also, using content-based feature or combinations of all features provided the best rumor detection performance regardless of detecting known or unknown rumor events. However, we have a problem of (1) selecting the most appropriate ML model to optimize the rumor detection performance, (2) the best ML model can differ even under the same rumor topic category, and (3) the rumor detection performance would drop when detecting new unknown rumor events.

5 ENSEMBLE SOLUTION: MULTI-ML MODELS

Our results shown in Section 4 indicate that it would NOT be a good strategy to use a single ML model to detect various types of rumors in social media, because the best performing model changes based on the rumor topic and features. To grasp the best properties of different ML models, we propose an ensemble solution (ES) using multi-ML models in order to enhance rumor detection performance. The ensemble method is a well established approach for producing a highly accurate classifier by combining less accurate ones [6]. The effectiveness of ES has been studied in various applications such as medical imaging [30] and energy [20].

First, the ensemble solution setup is described and then the performance against single ML models is compared in Section 5.1 and 5.2, respectively. Then, we conducted a statistical analysis using the Kruskal-Wallis (KW) test to determine the significant difference in detection performance between ML models including the proposed ES in Section 5.3.

5.1 Ensemble solution setup

The aim of the ES is to collect useful properties of different ML models which are not captured when used individually. However, not all models provide the same benefits, so a differentiation mechanism is used to distinguish their usefulness. Equation (1) shows the ES as a function f to determine whether the given data, d, (e.g., a tweet) is rumor or benign (i.e., non-rumor). Also, a weight value w ($0 \leq w \leq 1$) is assigned to each model in f, which controls the proportion of solution affected by those models. Because the final result is not always 0 or 1, but in between, a threshold value of 0.5 is used (i.e., $f(d) \leq 0.5$ is "benign"; otherwise "rumor").

$$f(d) = \arg\max(w_1 \times M_1 + w_2 \times M_2 + \ldots w_n \times M_n) \quad (1)$$

To determine the best strategy of selecting ML models to include in the ES, and also to evaluate how to appropriately assign weight values, we compare four strategies: (ES1) top N models, equal weights, (ES2) top N models, optimized weights, (ES3) using all available models, equal weights, and (ES4) using all available models, optimized weights. Using those four strategies, we can examine the performance improvements against single model approaches, as well as to determine the model and weight selections. The selection of an appropriate strategy is described as follows.[9]

- **Model Selection**
  - **All**: No prior knowledge of ML models and their rumor detection performance.
  - **Top** N: Has each ML model performance measured and be able to rank them.
- **Weight Values**
  - **Equal**: User does not have access to evaluate and configure ML models for rumor detection.
  - **Optimized**: User can pre-train and evaluate ML models using different rumor events.

Top N: We can select the top N models to form the ES, because some models perform poorly for all rumor events and features, as seen in Section 4 (i.e., there is always another model that performs better). Hence, incorporating those under-performing ones may not necessarily improve rumor detection, and possibly negatively impact the performance. The top N models are selected based on the performance observed in the past (i.e., we must evaluate their performance prior to assembling them into the ES). As such, this approach will add overhead for pre-evaluating the ML models for selection.

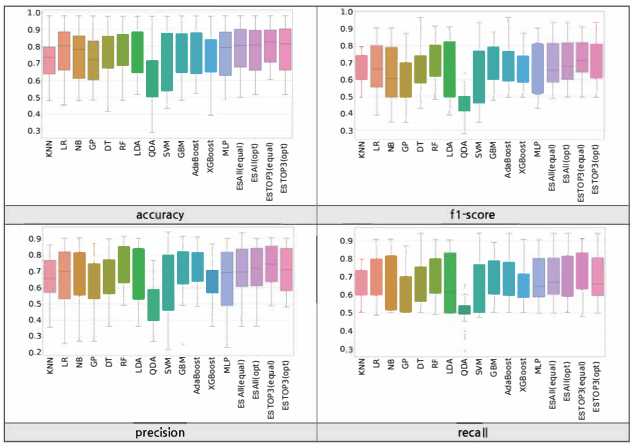
Weight optimization: The weight values can be optimized to improve the performance as demonstrated in [11]. This is done through hyper-parameter optimization using 25% of the dataset (as mentioned in Section 3). Because some models may perform better than others (as shown in Section 4), they can be given higher weight values proportional to their relative rumor detection performance. However, this will require further training using the existing dataset[10], and also if the training set is biased, the end result may not enhance the performance.

[9] We denote using all models as *ALL* and top 3 models as *TOP3*, and assigning equal weights as *equal* and optimized weights as *opt*. For example, using all models with equal weights are denoted as ES ALL(equal).

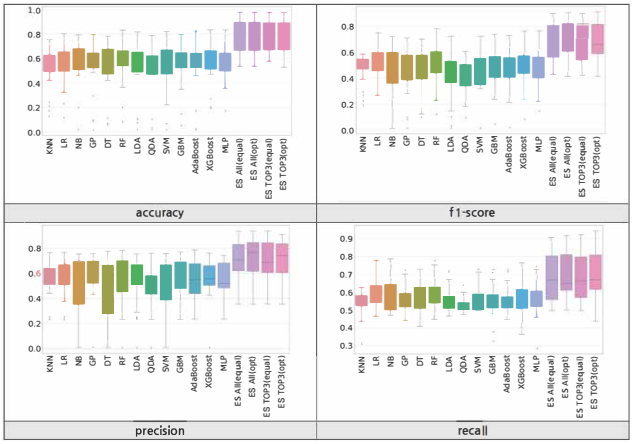
[10] That is, single models may be trained and optimized already (e.g., off the shelf), but if there is no available dataset, we cannot optimize weights.

5.2 ES performance evaluation

To evaluate the performance of the ES using the four strategies (i.e., All(equal), All(opt), TOP3(equal) and TOP3(opt)), the same experimental analysis is carried out using them as in Section 4 (i.e., using the 8 rumor events). For training, we set $train = 6$, $validate = 2$ and $test = 2$ (i.e., training, validation and testing are set at 60%, 20% and 20%, respectively). The results are shown in Figure 4.



(a) Known rumors



(b) Unknown rumors

Figure 4: Ensemble solution v.s. single ML models.

Known rumor detection: Similar to the single ML model detectors, the ES (regardless of the strategies) performs relatively equivalent (as shown in Figure 4a). That is, although the ES generally has higher mean scores, the variance largely overlaps with single model detectors. This is an expected result given that we are already able to achieve high performance even using single ML models when the rumor event is known (i.e., there is not much room for improvements by combining them). QDA is the only model that performed significantly worse than the rest. We also show later in Section 5.3 that there is no statistically significant difference among the models with respect to the rumor detection performance when the event is known.

Unknown rumor detection: A more notable difference is observed when testing with unknown rumor events. Figure 4b shows clearly that the ES achieves significantly higher performance than the single model detectors. Single model detectors have significantly dropped in their performances in our experiment (i.e., the average accuracy for known and unknown rumors has dropped from 0.7 to 0.6), a notable drop compared with the ES (which stayed relatively the same with the mean F1 score around 0.7). Another observation is that the performance across different strategies of the ES are relatively the same. This indicates that it is feasible to combine already trained ML models into an ES without requiring further tuning for weights or ranking them. One of the key reasons that the ES performs better than single models when detecting unknown rumors is that the rumor data is picked up by different groups of ML models. Consequently, a subset of those models will put more weight towards classifying it as a rumor in comparison to the single model detectors which do not have support decision methods. As shown in Table 5, different models capture different characteristics of unknown rumors. Hence, relying on a single detector fails when the rumor events vary (which is similar to the real-world environment with various unseen and unexpected rumors).

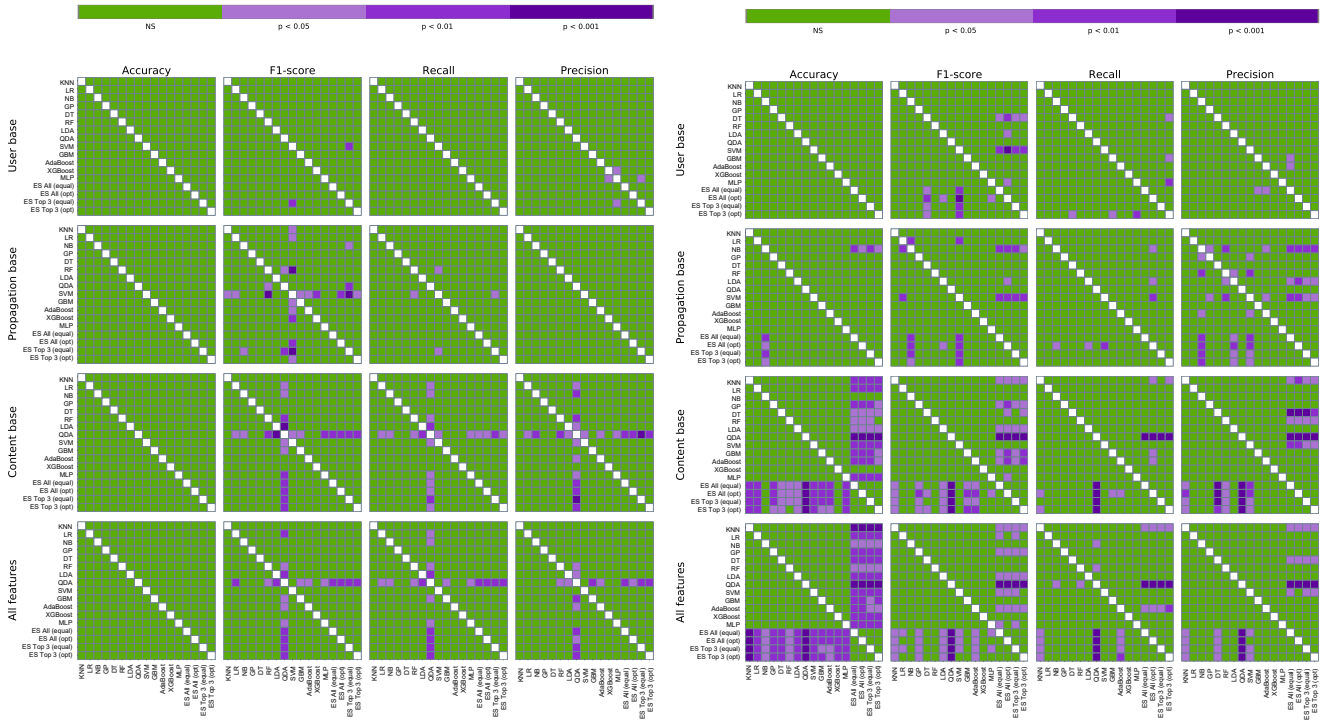
To validate the performance improvement using the ES to detect unknown rumors, a statistical analysis is carried out in the next section.

5.3 Statistical analysis

To distinguish the performance among the tested models, we conduct statistical analysis using the KW test to compare the performance between the ML models and the ES. The KW test allows us to compare the distributions of multiple groups (i.e., ML models and the ES) on a dependent variable that is measured (i.e., rumor detection performance). That is, we evaluate whether the performance differs across the ML models and the ES using this test. To adjust p-values in multiple comparison procedures, we use the Bonferroni correction method [24].

Figure 5 shows the KW-test results (with Bonferroni correction) between the model pairs we evaluated with respect to feature groups. We can differentiate the sensitivity of different ML models to each feature group, which leads to identifying alternative models to use instead of all ML models. As shown in Figure 5a, there is no statistically significant difference among the rumor detection models for known rumors. Hence, it is feasible to use any one of those models for known rumor detection as their performance would be similar. Of course, the amount of data needed to train them can vary. Existing work tackles this problem by discussing early detection, but this requires collecting the potential rumor data and training the model. We further discuss this in Section 6.

On the other hand, we observed significant performance differences between the ES and single ML models, as shown in Figure 5b. These test results confirmed our observations in Figure 4b. Specifically, for the accuracy measure using all features and content-only features, the ES outperformed **ALL** single ML models with a significant difference. The same was also observed for F1 scores using all features, but because single ML models had a wide variance in their results, the null hypothesis could not be rejected for them. Mainly, NB, DT, RF, GBM and XGBoost are the models with no



(a) Known rumors **(b) Unknown rumors**

Figure 5: Statistical difference evaluation between rumor detection models.

significant difference to the ES, but the mean performance scores are still lower than any of the ES strategies.

Our performance evaluation and statistical test results have shown that the proposed ES can improve the detection performance for unknown rumor events. Our experiments also demonstrate that there is no significant difference in different strategies of the ES.

6 DISCUSSION

1. Early detection: One of the key aspects of rumor detectors is to identify rumors as early as possible. Initially, rumor data had to be fed to the ML model manually for training. To improve this, others have proposed techniques such as enquiry-based [33], selecting appropriate features [17, 18], social attention [9], etc. But as shown in Section 4, rumor detection performance varies not only with features, but also with rumor events. Enquiry-based and social attention also has limitations that such data must be generated, so rumors without enquiries or mass attention can bypass these methods. On the other hand, our proposed approach using the ES is trained using past rumor events. Hence, we are not bound to specific properties of the rumor, and also the detection performance could be increased as more data is used to train the ES. However, this evaluation method does affect the performance when detecting known rumors, as the timeline will be affected (i.e., the order of rumors for detection is not chronological). Nevertheless, the experimental results in Figure 4 showed that if the rumor event is known, then the performance variation for any models is not significantly different as shown in Figure 5a (i.e., using any model will have relatively the same result, as long as the dataset is not heavily skewed). Moreover, we expect that the ES would potentially be further enhanced using techniques such as GAN for rumor generation [22], especially for unseen and unexpected rumor events. Therefore, the proposed ES solution is a viable method for early rumor detection.

2. Rumor dataset: Gathering the dataset for rumors is a challenging task due to the lack of automated processes to annotate them. Because existing rumor detectors are not reliable to detect unknown rumor events, they cannot be used to annotate rumors. Although the proposed ES is able to improve the detection of rumors for unknown rumor events, it still requires training using known rumor data. To address this problem, techniques such as rumor generation using GAN [22] can be used to further enhance the classification of unknown rumor events. This limitation affects the use of certain models and techniques as well. For example, neural network algorithms (e.g., RNN, LSTM, GRU, etc.) require parsing the rumor dataset into a sequence, which is not present in some datasets. Hence, rumor generation and collection are still challenging tasks, which will be investigated in our future work.

Although we used some correlated dataset based on rumor topics (as shown in Table 1), rumor detection performance varied significantly across the events we examined, as shown in Table 5. This implies that rumor events themselves do not necessarily share features that are correlated within the same rumor topic. Hence, other correlation amongst rumor topics needs to be investigated.

3. Rumor feature: Our experimental results showed that content-based feature enhanced the performance of all models significantly compared to other features, and in many cases combining all features did not necessarily improve the performance over the content-based feature (they are not statistically different in performance). We suspect that because using other features, P and U, did not improve the detection performance, by combining them into all features may added noise in detection rather than improving the performance. This indicates the importance of understanding the context of the rumor. To further enhance the detection performance, the content-based feature can be enriched by incorporating contextual information (e.g., context-aware rumor detection using sentimental information). Zubiaga *et al.* [34] incorporated natural language processing (NLP) to predict the veracity of the rumor, which performed better when compared to without using the NLP. Similarly, we can enhance content-based features by incorporating context-aware analysis of the rumor.

Propagation and user-based features were not effective in rumor detection in our experiments. As seen in Table 3, rumor events had quite different base metrics which contribute towards the propagation and user-based features, but the rumor detection performance by ML models was not affected by them. It could be due to rumor spread having similar properties as non-rumor spread (for propagation), and although rumor spreading users had much less metric values (e.g., less number of friends, retweets and favorite counts), the overall characteristics may not be significant for the tested models to identify them. A further look into the usefulness of features will be examined in our future work.

4. Best performing ML models: Using the ES (regardless of any strategies to combine them) is the suggested solution when detecting unknown rumors, as it outperforms all other single model-based detectors as shown in Section 5. If limited features are given (e.g., User only, or Propagation only), then many of the tested models will perform relatively the same. If only user-based features are given, a few models to avoid using are DT, LDA, QDA and MLP. If only propagation-based features are given, then NB, LDA and SVM are not recommended.

Although we did not find any strong relationship between rumor topics and single ML models in rumor detection performance, the following models could be suggested based on our statistical test shown in Figure 5: NB, RF, GBM and XGBoost. Although those model's accuracy is significantly less than using the ES, their F1 score achieved relatively the same with the ES. Other models not listed above are not recommended as their performance is significantly less than the ES.

5. Configuring the ensemble solution: In this paper, we discussed the four strategies (All(equal), All(opt), TOP3(equal) and TOP3(opt)) of the ES, but found that using any of them generated similar performance. Hence, the weight assignment and choosing N best ML models do not necessarily improve the performance. That is, even if N is a small number, it can be helpful to reduce training and testing times.[11] However, we assumed that the selected models are already optimized to detect rumors. If we are given off-the-shelf ML models only, then hyper-parameter configuration could impact the performance significantly. Moreover, because the detection performance varies when rumor topics change, it may also be possible to dynamically adjust the weights between ML models in the ES based on empirical studies. To investigate further to formulate the most effective ES, we will investigate the use of other ML models with and without optimization, as well as in conjunction with other rumor topic-related features in our future work.

[11] Of course, the training and testing times increase with larger N.

7 CONCLUSION

Many different ML models have been proposed to detect rumors. However, they still lack the capabilities to grasp different rumor topics and features, leading to degraded rumor detection performance. Further, using those models to detect new and emerging rumor topics severely affect their performance as they are not necessarily configured to do so. We have validated this in our experimental analysis, which showed that there is no one ML model that can outperform others, especially when various rumor topics are considered. We also discovered that using content-based features only might be a better strategy than using all features for using an ML model to detect unseen rumors specifically.

To address the aforementioned problems, we proposed an ensemble solution (ES) combining multi-ML models together to work cooperatively to detect rumors of various topics with different features given. The experimental results showed that there is a significant improvement in the performance when using the ES to detect rumors that are previously unknown, achieving up to 0.79 average F1 score, compared to 0.58 using a single ML model (i.e., an increase of 0.21 average F1 score). With the enhanced rumor detection performance, the ES is suitable and practical for early rumor detection without needing any of the unknown rumor data, which can take a significant time to identify and collect, while the caused damage increases.

REFERENCES

[1] Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter Mood Predicts the Stock Market. *Journal of Computational Science* 2, 1 (2011), 1–8.

[2] Cody Buntain and Jennifer Golbeck. 2017. Automatically Identifying Fake News in Popular Twitter Threads. In *Proc. of 2017 IEEE International Conference on Smart Cloud (SmartCloud 2017)*. 208–215. https://doi.org/10.1109/SmartCloud.2017.40

[3] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information Credibility on Twitter. In *Proc. of the 20th International Conference on World Wide Web (WWW 2011)*. 675–684. https://doi.org/10.1145/1963405.1963500

[4] Tong Chen, Xue Li, Hongzhi Yin, and Jun Zhang. 2018. Call Attention to Rumors: Deep Attention Based Recurrent Neural Networks for Early Rumor Detection. In *Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2018)*, Mohadeseh Ganji, Lida Rashidi, Benjamin C. M. Fung, and Can Wang (Eds.). Springer International Publishing, 40–52.

[5] Chen, Tong and Chen, Hongxu and Li, Xue. 2018. Rumor Detection via Recurrent Neural Networks: A Case Study on Adaptivity with Varied Data Compositions. In *Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2018)*, Mohadeseh Ganji, Lida Rashidi, Benjamin C. M. Fung, and Can Wang (Eds.). Springer International Publishing, 121–127.

[6] Thomas G. Dietterich. 2000. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1–15.

[7] Georgios Giasemidis, Colin Singleton, Ioannis Agrafiotis, Jason RC Nurse, Alan Pilgrim, Chris Willis, and Danica Vukadinovic Greetham. 2016. Determining the Veracity of Rumours on Twitter. In *Proc. of the International Conference on Social Informatics (SocInfo 2016)*. Springer, 185–205.

[8] Nir Grinberg, Kenneth Joseph, Lisa Friedland, Briony Swire-Thompson, and David Lazer. 2019. Fake News on Twitter During the 2016 U.S. Presidential Election. *Science* 363, 6425 (2019), 374–378. https://doi.org/10.1126/science.aau2706

[9] Han Guo, Juan Cao, Yazi Zhang, Junbo Guo, and Jintao Li. 2018. Rumor Detection with Hierarchical Social Attention Network. In *Proc. of the 27th ACM International Conference on Information and Knowledge Management (CIKM 2018)*. 943–951. https://doi.org/10.1145/3269206.3271709

[10] Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking Sandy: Characterizing and Identifying Fake Images on Twitter During Hurricane Sandy. In *Proc. of the 22nd international conference on World Wide Web (WWW 2013)*. ACM, 729–736.

[11] Justin Heinermann and Oliver Kramer. 2016. Machine Learning Ensembles for Wind Power Prediction. *Renewable Energy* 89 (2016), 671 – 679. https://doi.org/10.1016/j.renene.2015.11.073

[12] Benjamin D Horne and Sibel Adali. 2017. This Just In: Fake News Packs A Lot in Title, Uses Simpler, Repetitive Content in Text Body, More Similar to Satire Than Real News. In *Proc. of the 11th International AAAI Conference on Web and Social Media (AAAI ICWSM 2017)*.

[13] Matthew Kay, Shwetak N. Patel, and Julie A. Kientz. 2015. How Good is 85%?: A Survey Tool to Connect Classifier Evaluation to Acceptability of Accuracy. In *Proc. of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI 2015)*. 347–356. https://doi.org/10.1145/2702123.2702603

[14] S. B. Kotsiantis. 2007. Supervised Machine Learning: A Review of Classification Techniques. *Emerging Artificial Intelligence Applications in Computer Engineering* (2007), 3–24.

[15] Akshi Kumar and Saurabh Raj Sangwan. 2019. Rumor Detection Using Machine Learning Techniques on Social Media. In *Proc. of the International Conference on Innovative Computing and Communications (ICICC 2019)*, Siddhartha Bhattacharyya, Aboul Ella Hassanien, Deepak Gupta, Ashish Khanna, and Indrajit Pan (Eds.). Springer Singapore, Singapore, 213–221.

[16] Srijan Kumar, Robert West, and Jure Leskovec. 2016. Disinformation on the Web: Impact, Characteristics, and Detection of Wikipedia Hoaxes. In *Proc. of the 25th International Conference on World Wide Web (WWW 2016)*. 591–602. https://doi.org/10.1145/2872427.2883085

[17] Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. 2017. Rumor Detection Over Varying Time Windows. *PloS One* 12, 1 (2017), e0168344.

[18] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent Features of Rumor Propagation in Online Social Media. In *Proc. of the 13th IEEE International Conference on Data Mining (ICDM 2013)*. IEEE, 1103–1108.

[19] Yahui Liu, Xiaolong Jin, Huawei Shen, and Xueqi Cheng. 2017. Do Rumors Diffuse Differently from Non-rumors? A Systematically Empirical Analysis in Sina Weibo for Rumor Identification. In *Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2017)*. Springer, 407–420.

[20] S. Lu, Youngdeok Hwang, I. Khabibrakhmanov, F. J. Marianno, Xiaoyan Shao, J. Zhang, B. Hodge, and H. F. Hamann. 2015. Machine Learning Based Multi-physical-model Blending for Enhancing Renewable Energy Forecast - Improvement Via Situation Dependent Error Correction. In *2015 European Control Conference (ECC)*. 283–290. https://doi.org/10.1109/ECC.2015.7330558

[21] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect Rumors Using Time Series of Social Context Information on Microblogging Websites. In *Proc. of the 24th ACM International on Conference on Information and Knowledge Management (CIKM 2015)*. 1751–1754. https://doi.org/10.1145/2806416.2806607

[22] Jing Ma, Wei Gao, and Kam-Fai Wong. 2019. Detect Rumors on Twitter by Promoting Information Campaigns with Generative Adversarial Learning. In *Proc. of the 28th international conference on World Wide Web Conference (WWW 2019)*. 3049–3055. https://doi.org/10.1145/3308558.3313741

[23] Marcelo Mendoza, Barbara Poblete, and Carlos Castillo. 2010. Twitter Under Crisis: Can We Trust What We RT?. In *Proc. of the first workshop on social media analytics*. ACM, 71–79.

[24] Matthew A Napierala. 2012. What is the Bonferroni Correction. *AAOS Now* 6, 4 (2012), 40.

[25] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[26] Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. CSI: A Hybrid Deep Model for Fake News Detection. In *Proc. of the ACM Conference on Information and Knowledge Management (CIKM 2017)*. 797–806. https://doi.org/10.1145/3132847.3132877

[27] Kate Starbird, Jim Maddock, Mania Orand, Peg Achterman, and Robert M Mason. 2014. Rumors, False Flags, and Digital Vigilantes: Misinformation on Twitter After the 2013 Boston Marathon Bombing. *Proc. of the iConference (iConference 2014)*.

[28] S. Wang, Q. Kong, Y. Wang, and L. Wang. 2019. Enhancing Rumor Detection in Social Media Using Dynamic Propagation Structures. In *Proc. of the IEEE International Conference on Intelligence and Security Informatics (ISI 2019)*. 41–46. https://doi.org/10.1109/ISI.2019.8823266

[29] Helena Webb, Pete Burnap, Rob Procter, Omer Rana, Bernd Carsten Stahl, Matthew Williams, William Housley, Adam Edwards, and Marina Jirotka. 2016. Digital Wildfires: Propagation, Verification, Regulation, and Responsible Innovation. *ACM Transactions on Information Systems* 34, 3 (2016), 15:1–15:23. https://doi.org/10.1145/2893478

[30] Liyang Wei, Yongyi Yang, R. M. Nishikawa, and Yulei Jiang. 2005. A Study on Several Machine-learning Methods for Classification of Malignant and Benign Clustered Microcalcifications. *IEEE Transactions on Medical Imaging* 24, 3 (March 2005), 371–380. https://doi.org/10.1109/TMI.2004.842457

[31] YeKang Yang, Kai Niu, and ZhiQiang He. 2015. Exploiting the Topology Property of Social Network for Rumor Detection. In *Proc. of the 12th International Joint Conference on Computer Science and Software Engineering (JCSSE 2015)*. IEEE, 41–46.

[32] Zhifan Yang, Chao Wang, Fan Zhang, Ying Zhang, and Haiwei Zhang. 2015. Emerging Rumor Identification for Social Media with Hot Topic Detection. In *Proc. of the 12th Web Information System and Application Conference (WISA 2015)*. IEEE, 53–58.

[33] Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts. In *Proc. of the 24th International Conference on World Wide Web (WWW 2015)*. 1395–1405. https://doi.org/10.1145/2736277.2741637

[34] Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Computing Survey* 51, 2, Article 32 (Feb. 2018), 36 pages. https://doi.org/10.1145/3161603

[35] Arkaitz Zubiaga, Maria Liakata, and Rob Procter. 2017. Exploiting Context for Rumour Detection in Social Media. In *Proc. of International Conference on Social Informatics (SocInfo 2017)*. Springer, 109–123.

[36] Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing How People Orient to and Spread Rumours in Social Media by Looking at Conversational Threads. *PloS One* 11, 3 (2016), e0150989.