Continual Learning for Fake News Detection from Social Media

Yi Han(✉), Shanika Karunasekera, and Christopher Leckie

School of Computing and Information Systems, The University of Melbourne, Melbourne, Australia
{yi.han,karus,caleckie}@unimelb.edu.au

Abstract. The prevalence of fake news over social media has a profound impact on justice, public trust and society as a whole. Although significant effort has been applied to mitigate its negative impact, our study shows that existing fake news detection algorithms may perform poorly on new data. In other words, the performance of a model trained on one dataset degrades on another and potentially vastly different dataset. Considering that in practice a deployed fake news detection system is likely to observe unseen data, it is crucial to solve this problem without re-training the model on the entire data from scratch, which would become prohibitively expensive as the data volumes grow. An intuitive solution is to further train the model on the new dataset, but our results show that this direct incremental training approach does not work, as the model only performs well on the latest dataset it is trained on, which is similar to the problem of catastrophic forgetting in the field of continual learning. Instead, in this work, (1) we first demonstrate that with only minor computational overhead, balanced performance can be restored on both existing and new datasets, by utilising Gradient Episodic Memory (GEM) and Elastic Weight Consolidation (EWC)—two techniques from continual learning. (2) We improve the algorithm of GEM so that the drop in model performance on the previous task can be further minimised. Specifically, we investigate different techniques to optimise the sampling process for GEM, as an improvement over random selection as originally designed. (3) We conduct extensive experiments on two datasets with thousands of labelled news items to verify our results.

Keywords: Fake news detection · Continual learning · Social media

1 Introduction

A series of incidents over recent years have demonstrated the profound damage fake news can cause to society, and it has become an urgent challenge to study how to automatically and accurately identify fake news[1] before it is widespread.

[1] Here we use the definition in [38]: *fake news is intentionally and verifiably false news published by a news outlet.*

I. Farkaš et al. (Eds.): ICANN 2021, LNCS 12892, pp. 372–384, 2021.
https://doi.org/10.1007/978-3-030-86340-1_30

A variety of techniques have been proposed for fake news detection [17,38], including *content-based approaches* that use news headlines and body content to verify the validity of the news, *context-based approaches* that rely on the interactions between users, *e.g.*, tweet, retweet, reply, mention and follow, and *mixed approaches*. However, we find that **even though these methods may achieve satisfactory results on the dataset on which they are trained, their performance often degrades considerably on another and potentially vastly different dataset.** In practice, a deployed fake news detection system is likely to observe new, unseen data. Therefore, it is crucial to solve this problem without re-training the model from scratch every time a new dataset is obtained, which would become prohibitively expensive as the data volumes grow.

Specifically, we start with the most intuitive approach of direct incremental training—further train the model on the new dataset. However, our results suggest that using this approach the obtained model only performs well on the latest dataset it is trained on. This is similar to the problem of catastrophic forgetting [12] in the field of continual learning: when a deep neural network is trained to learn a sequence of tasks (in this case, a new dataset represents a different task), its performance degrades on the earlier tasks after it learns new tasks, as the new tasks override the weights. Therefore, in this work:

- We first demonstrate that with only minor computational overhead, balanced performance can be restored on both existing and new datasets, by utilising GEM [9] and EWC [7]—two popular techniques from continual learning, although GEM-trained models perform better in general.
- We improve GEM so that the drop in model performance on the previous task can be further minimised. GEM keeps a certain number of samples from the previous task when training a model on the new task. In contrast to existing approaches that use uniform random sampling, we investigate more sophisticated sampling techniques—maximum entropy sampling and support samples—so that the chosen instances are more informative.
- We conduct extensive experiments on two datasets with thousands of labelled news items. Specifically, our experimental results show that after the above sampling techniques are applied, the trained models can achieve better performance on the previous task, while maintaining their performance on the new task.

The remainder of this paper is organised as follows: Sect. 2 briefly reviews existing work on fake news detection; Sect. 3 describes the problem with current detection algorithms when facing new, unseen data; Sect. 4 investigates how to restore balanced performance on both existing and new data using GEM and EWC, as well as how to improve GEM; and finally Sect. 5 concludes the paper and offers directions for future work.

2 Background: Fake News Detection Algorithms and Datasets

Detecting fake news on social media has been a popular research problem over recent years. In this section, we briefly review the prior work on this topic, and introduce the datasets chosen in our experiments. Specifically, similar to [17, 24], we classify existing work into three categories: content-based approaches, context-based approaches and mixed approaches.

Content-Based Approaches. Content-based approaches use news headlines and body content to verify the validity of the news. It can be further classified into two categories [24,38]: (1) *knowledge-based detection.* In order for this type of method to work, a knowledge base or knowledge graph [15] has to be built first. Here, knowledge can be represented in the form of a triple: (Subject, Predicate, Object), *i.e.,* SPO triple. Then, to verify an item of news, knowledge extracted from its content is compared with the facts in the knowledge graph [3]. (2) *Style-based detection.* Since the purpose of fake news is to mislead the public, it often exhibits unique writing styles that are rarely seen in real news. Therefore, style-based methods aim to identify these characteristics [19,27,29].

In addition to textual information, images posted in social media have also been investigated to facilitate the detection of fake news [5,30,33,37].

Context-Based Approaches. Social context here refers to the interactions between users, including tweet, retweet, reply, mention and follow. These engagements provide valuable information for identifying fake news spread on social media. For example, Jin *et al.* [6] build a stance network where the weight of an edge represents how much each pair of posts support or contradict each other. Then fake news detection is based on estimating the credibility of all the posts related to the news item. Tacchini *et al.* [26] propose to detect fake news based on user interactions, *i.e.,* users who liked them on Facebook.

Unlike the above supervised methods, an unsupervised approach is proposed in Yang *et al.* [32]. It builds a Bayesian graphical model to capture the generative process among the validity of news, user opinions and user credibility.

Mixed Approaches. Mixed approaches use both news content and associated user interactions over social media to differentiate between fake news and real news. Ruchansky *et al.* [20] design a three-module architecture that combines the text of a news article, the received user response and the source of the news. Other methods that fall into this category include [25,36]

In addition to the above work, a few recent papers have started to work on explainability, *i.e.,* why their model labels certain news items as fake [10,18,21].

Datasets. A number of datasets covering different domains have been collected for fake news detection. In our work, we use the dataset of FakeNewsNet [22], which contains labelled news from two websites: politifact.com and gossipcop.com. The news content includes both linguistic and visual information, all the tweets and retweets for each item of news, and the information of the corresponding Twitter users (please refer to [22] for more details).

3 Problem Description

In previous work on fake news detection, most proposed methods were evaluated on multiple datasets separately. However, our experimental results on several detection algorithms suggest that models trained on one dataset, *e.g.*, PolitiFact, do not perform well on another dataset, *e.g.*, GossipCop. Note that these two datasets are chosen for demonstration purpose only. Similar findings can be made on other datasets as well, *e.g.*, recently collected COVID-19 datasets, or from two splits of the same dataset that are temporally far away from each other.

A natural thought is to re-train the model on both datasets, but this may not be feasible, or at least not ideal in practice: there will always be new data that our model has not seen before, and it does not make sense to re-train the model from scratch on the entire data every time a new dataset is obtained, especially since as the data size grows, this can become prohibitively expensive.

Therefore, we aim to **find an incremental training method to address the issue of dealing with new, unseen data in fake news detection.** Specifically, let one dataset, *e.g.*, PolitiFact, represent the existing data that our model has been trained on, and the other dataset, *e.g.*, GossipCop, represent the unknown data that our model will face in the future, we investigate how to train models incrementally so that balanced performance can be achieved on both datasets.

To answer the above question, we choose a widely-cited content-based approach HAN [34], and design a context-based method that applies graph neural networks (GNNs) to differentiate between the propagation patterns of fake and real news on social media. More details are given in the next subsection.

3.1 Propagation Patterns for Fake News Detection

Empirical evidence suggests that fake news and real news spread differently online [28], and the idea of using propagation patterns to detect fake news has been explored in a number of previous studies [1,8,10,11,14,23,31,39]. However, considering the capability of graph neural networks (GNNs) in dealing with non-Euclidean data, we use GNNs to differentiate between the propagation patterns of fake and real news on social media. In addition, given that machine learning models are vulnerable to adversarial attacks [4], we decide not to rely on any text information, *e.g.*, news content or tweet content, so that our model can be less susceptible to the manipulation of advanced fake news fabricators.

Notation in GNNs. Consider a graph $G = (A, F)$ with n vertices/nodes and m edges, where $A \in \{0,1\}^{n \times n}$ is the adjacency matrix. $A_{i,j} = 1$ if there is an edge from node i to node j, and $A_{i,j} = 0$ otherwise; $F \in R^{n \times d}$ is the feature matrix, *i.e.,* each node has d features. Given A and F as inputs, the output of a GNN after the k^{th} step is: $H^{(k)} = f\left(A, H^{(k-1)}; \theta^{(k)}\right) \in R^{n \times d}$, where f is the propagation function parameterised by θ, and $H^0 = F$. $H^{(k)}$ can be used for node- or graph-level classification. There have been a number of implementations for the propagation function. In our work, since the goal is to label the propagation

pattern of each item of news, which is a graph, we choose the algorithm of DiffPool [35] that is specifically designed for graph classification.

Below we explain how we define the adjacency matrix and the feature matrix in our model, and then present a brief performance comparison.

Adjacency Matrix. Once an item of news is published, it may be tweeted by multiple users. We call these tweets that directly reference the news URL *root* tweets. Each of them and their retweets form a separate cascade [28], and all the cascades form the propagation pattern of an item of news.

Each propagation pattern is a graph, where a node refers to a tweet (including the corresponding user)—either the root tweet that references the news or its retweets. A special case is that an extra node representing the news is added to connect all cascades together. All the feature values for this node are set to zero. Edges here represent information flow, *i.e.,* how the news transfers from one person to another. However, since Twitter APIs do not provide the immediate source of a retweet, we first sort the tweets by their timestamps within each cascade, and then search for the potential source of a retweet from all the tweets published earlier. Specifically, there is an edge from node i to node j[2] if:

- The user of node i mentions the user of node j in the tweet, *e.g.*, user i retweets a news item and also recommends it to user j via mentioning;
- Tweet i is public and tweet j is posted within a certain period of time after tweet i. We set the time limit to ten hours in our experiments.

Note that edges only exist between nodes within the same cascade. We have also further considered the follower and following relations, but our results demonstrate that there is no significant improvement. In addition, since Twitter applies a much stricter rate limit on corresponding APIs, these types of information may not be available in real time, especially if a number of news items need to be validated at the same time and within a detection deadline.

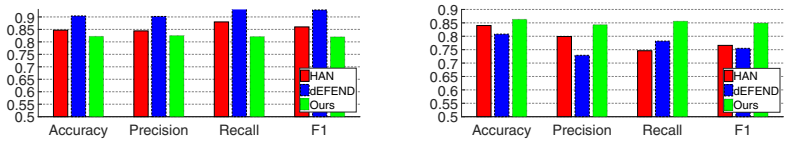
Feature Matrix. Since our method does not rely on any textual information, we only choose the following information from user profiles as the features for each node: (1) whether the user is verified, (2) the timestamp when the user was created, encoded as the number of months since March 2006—the time when Twitter was founded, (3) the number of followers, (4) the number of friends, (5) the number of lists, (6) the number of favourites; (7) the number of statuses, (8) the timestamp of the tweet, encoded as the number of seconds since the first tweet that references the news is posted. Another important reason why we choose the above features is that they are easily accessible—they are directly available within the tweet object, which is preferable for online detection.

Performance Comparison. We compare our method with the content-based approach HAN [34] and a state-of-the-art algorithm dEFEND [21]. To make our results comparable with those reported in [21] (as they also tested fake news detection algorithms on the same dataset), we follow the same procedure to train

[2] Node i is published before node j, and the information goes from user i to user j.

and test the GNNs: randomly choose 75% of the news as the training data while keeping the rest as the test data, and the final result is the average performance over five repeats. The model is evaluated with the following commonly used metrics: accuracy, precision, recall and F1 score.

For our method, the hyper-parameters for the DiffPool algorithm are set as follows: 2 pooling layers, 64 hidden dimensions and 64 embedding dimensions. In addition, **since it is more critical to detect fake news at an early stage before it becomes widespread, we train GNNs on a clipped dataset that only contains the first $K = 100$ tweets for each news item**[3]**.**



(a) PolitiFact (b) GossipCop

Fig. 1. Performance comparison on the datasets of PolitiFact and GossipCop.

As can be seen from Fig. 1, by only relying on the limited set of non-textual features and the clipped dataset, our model can achieve comparable performance on PolitiFact, and the best result on GossipCop.

4 Dealing with Degraded Performance on New Data

As mentioned in the problem description, we have tested several fake news detection algorithms and find that models trained on PolitiFact perform poorly on GossipCop, and vice versa, where all four metrics drop to around 0.6 or below. An examination of the news content and the generated graphs reveals that (1) since PolitiFact is mainly about political news while GossipCop is more about entertainment news, the writing style, the commonly discussed subjects and topics are vastly different; (2) the graphs generated from PolitiFact and GossipCop are also distinct from each other, in terms of the numbers of nodes and edges.

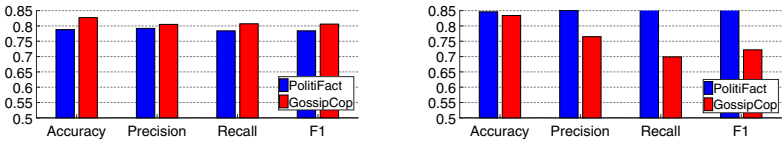
Similar observations can also be made between PolitiFact/GossipCop and other datasets, or from two splits of one dataset that are temporally far away from each other. In practice, no matter how much data a model has been trained on, it is likely that it will face unknown, different data in the future. This section investigates effective incremental training techniques so that balanced performance can be achieved on both existing and new data for fake news detection.

[3] We have also tested $K = 200, 500, 1000, \infty$ (not clipped). Those results are omitted due to space limits (the results are better under those settings).

4.1 Incremental Training Reverses the Model Performance

We first test incremental training, *i.e.,* further train the model obtained from PolitiFact (or GossipCop) on the other dataset of GossipCop (or PolitiFact). However, then the models only perform well on the latest dataset on which they are trained, while achieving degraded results on the former dataset. Note that during incremental training, we still randomly choose 75% of news as the training data and the rest as the test data.

This is similar to the problem of catastrophic forgetting which was first recognised in [12]: a neural network tends to forget the information learned in the previous tasks when training on new tasks. In our case, each new dataset can be considered as a new task. In the next subsection, we investigate how to solve the problem by proposing techniques based on continual learning.



(a) Our method (b) HAN

Fig. 2. Performance of models first trained on PolitiFact and then on GossipCop using GEM ($|\mathcal{M}| = 300$).

4.2 Continual Learning Restores Balanced Performance

In order to deal with catastrophic forgetting, a number of approaches have been proposed, which can be roughly classified into three types [16]: (1) regularisation-based approaches that add extra constraints to the loss function to prevent the loss of previous knowledge; (2) architecture-based approaches that selectively train a part of the network for each task, and expand the network when necessary for new tasks; (3) dual-memory-based approaches that build on top of complementary learning systems (CLS) theory, and replay samples for memory consolidation. In this paper, we consider the following two popular methods:

- Gradient Episodic Memory (GEM)—GEM uses episodic memory to store a number of samples from previous tasks, and when learning a new task t, it does not allow the loss over those samples held in memory to increase compared to when the learning of task $t-1$ is finished;
- Elastic Weight Consolidation (EWC)—its loss function consists of a quadratic penalty term on the change of the parameters, in order to prevent drastic updates to those parameters that are important to the old tasks.

In our case, the learning on the two datasets ($\mathcal{D}_1$ and $\mathcal{D}_2$) are considered as two tasks. When the model learns the first task, it is trained as usual; then during the learning of the second task, we incorporate GEM and EWC:

– Let C be the model, θ_1 be the parameters after the first task, and $\mathcal{M}$ be the set of instances sampled from the first dataset, then the optimisation problem under GEM becomes:

$$min_\theta \sum_{(x_i,y_i)\in\mathcal{D}_2} loss\left(C(x_i;\theta), y_i\right)$$
$$\text{subject to} \sum_{(x_j,y_j)\in\mathcal{M}} loss\left(C(x_j;\theta), y_j\right) \leq \sum_{(x_j,y_j)\in\mathcal{M}} loss\left(C(x_j;\theta_1), y_j\right)$$

– Let λ be the regularisation weight, F be the Fisher information matrix, and $\theta^*_{\mathcal{D}_1}$ be the parameters of the Gaussian distribution used by EWC to approximate the posterior of $p(\theta|\mathcal{D}_1)$, then the loss function under EWC is:

$$\sum_{(x_i,y_i)\in\mathcal{D}_2} loss\left(C(x_i;\theta), y_i\right) + \frac{\lambda}{2}F(\theta - \theta^*_{\mathcal{D}_1})^2$$

Table 1. Performance of models first trained on GossipCop and then on PolitiFact using EWC ($|\mathcal{M}| = 300, \lambda = 10^3 \sim 10^5$, the other results are omitted).

λ	Our method								HAN							
	PolitiFact				GossipCop				PolitiFact				GossipCop			
	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
10^3	0.71	0.71	0.71	0.71	0.76	0.74	0.68	0.69	0.72	0.72	0.72	0.72	0.69	0.65	0.71	0.64
3×10^3	0.72	0.72	0.72	0.72	0.73	0.70	0.66	0.67	0.73	0.73	0.73	0.72	0.68	0.65	0.71	0.64
10^4	**0.72**	**0.72**	**0.71**	**0.71**	**0.79**	**0.77**	**0.73**	**0.74**	0.73	0.72	0.72	0.72	0.71	0.66	0.73	0.66
3×10^4	0.72	0.72	0.72	0.72	0.76	0.74	0.71	0.72	**0.72**	**0.73**	**0.72**	**0.72**	**0.73**	**0.68**	**0.74**	**0.68**
10^5	0.71	0.71	0.71	0.71	0.77	0.75	0.71	0.72	0.73	0.73	0.72	0.72	0.72	0.67	0.74	0.67

Note that when estimating the Fisher information matrix F, we sample a set of instances ($\mathcal{M}$) and compare the model performance under different sample sizes.

In terms of parameters, we test sample size $|\mathcal{M}| = 100, 200, 300$ (all the samples are chosen randomly), and $\lambda = 1, 3, 10, 30, 10^2, 3\times10^2, 10^3, 3\times10^3, 10^4, 3\times10^4, 10^5$ (for EWC only). Figure 2 shows the results of models first trained on PolitiFact and then on GossipCop using GEM when $|\mathcal{M}| = 300$, and Table 1 presents the performance of models first trained on GossipCop and then on PolitiFact using EWC when $|\mathcal{M}| = 300, \lambda = 10^3, 3\times10^3, 10^4, 3\times10^4, 10^5$ (the other results are omitted due to space limits). The results demonstrate that both methods can achieve relatively balanced performance over the two datasets, although GEM trained models work better in general. Comparing Figs. 1 and 2, we can see that the GEM-trained models almost restore their performance on the previous task, where the drop in all four metrics is below 3% in most cases.

Efficiency. In terms of efficiency, we observe that: (1) compared with the normal training process, training with GEM and EWC requires slightly more time: 5% to 10%—this is a significant improvement over re-training from scratch, the time of which grows linearly with the number of nodes and edges in our case; (2) there is no significant difference in training time between GEM and EWC; and (3) the impact of the parameters on the training time is also not significant.

4.3 Optimise the Sampling Process to Further Minimise Performance Drop

In the above experiments, the set of instances $\mathcal{M}$ from the previous task is chosen randomly. In this section, we explore other techniques so that the selected samples are more informative about the data of the previous task. Note that all the experiments below are conducted using the propagation-based approach introduced in Sect. 3.1 with GEM (which outperforms EWC) and $|\mathcal{M}| = 300$, while we leave the improvement of content-based approaches for future work.

Technique I: Maximum Entropy Sampling (MES). We first consider maximum entropy sampling, which aims to select a subset $\mathcal{S}$ from the entire dataset $\mathcal{N}$ such that the obtained information of $\mathcal{N}$ is maximised. According to the principle of MES, the entropy of the remaining data points $\mathcal{N}\backslash\mathcal{S}$ must be minimised, while the entropy of $\mathcal{S}$ must be maximised, *i.e.,*

$$\mathcal{M} = \underset{\mathcal{S}}{\operatorname{argmax}}\, \mathcal{H}(\mathcal{S}) = \underset{\mathcal{S}}{\operatorname{argmax}} - \sum_{x_i} p(x_i) log_2 p(x_i), x_i \in \mathcal{S}$$

In our case, considering that the graphs generated from the two datasets have quite different numbers of nodes, we **calculate the entropy over the graph size**.

The MES problem is NP-hard [13]. A quasi-optimal solution adopts a greedy strategy: it starts with an empty set $\mathcal{S} = \emptyset$, and in each step, a new sample is chosen that maximises the marginal gain, *i.e.,* $x = \operatorname{argmax}_{x_i \notin \mathcal{S}} \mathcal{H}(\mathcal{S} \cup \{x_i\}) - \mathcal{H}(\mathcal{S})$. However, this greedy algorithm is computationally expensive, and is not suitable for large datasets. We explain our approaches later in this section.

Technique II: Support Samples. A similar idea has been explored in [2], which is inspired by margins in SVMs. For SVMs the support vectors determine the decision boundary, and in our case, we can define the margin as $Margin(x) = C(x, y) - C(x, 1 - y)$, where C is the classifier, x is the input, and $y \in \{0(real), 1(fake)\}$ is the label. A negative margin means that x is misclassified, while a larger margin suggests that the classifier is more confident of the prediction. Since the purpose of sampling instances from the previous task is to ensure that the model performance does not degrade, it does not make sense to choose misclassified instances, nor would it be efficient to select samples with large margins.

Proposed Sampling Approaches. Our sampling approaches combine the above two techniques—(1) first we calculate the margin for each graph in the

previous task, and initialise $\mathcal{S}$ with the graphs whose margin is within the range of $(0, \delta)$, $\delta \in [0, 1]$. Three values, $0.05, 0.1, 0.2$, are tested and we finally set $\delta = 0.1$. Note that the size of this initialised set is normally much smaller than the sample size of 300. (2) Then we propose the following two strategies (Algorithm 1):

- **Strategy I** goes through the graphs in $\mathcal{N}\backslash\mathcal{S}$ ordered by their margin values, and add one graph x_i if the entropy increases, *i.e.,* $\mathcal{H}(\mathcal{S}\cup\{x_i\}) > \mathcal{H}(\mathcal{S})$;
- **Strategy II** adopts a stochastic greedy method [13], where in each step we randomly sample a set of graphs ($\mathcal{R}$) from $\mathcal{N}\backslash\mathcal{S}$, and find $x_i \in \mathcal{R}$ that maximises the marginal gain, *i.e.,* $x = \text{argmax}_{x_i \in \mathcal{R}} \mathcal{H}(\mathcal{S} \cup \{x_i\}) - \mathcal{H}(\mathcal{S})$. Please refer to [13] for how to choose the size of $\mathcal{R}$. In our experiments, we set $|\mathcal{R}| = max(\frac{|\mathcal{N}|}{|\mathcal{M}|}, 20)$.

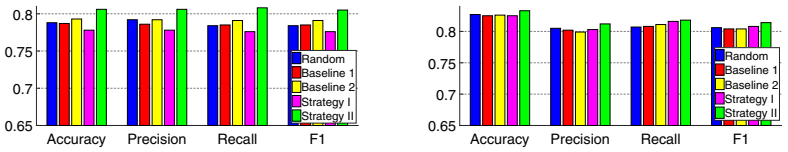
In addition, we design another two strategies as baselines: (1) choose the graphs with the top $|\mathcal{M}| = 300$ smallest margin values, and (2) initialise $\mathcal{S} = \{x | 0 < Margin(x) \leq \delta = 0.1\}$, sort the remaining graphs $\mathcal{N}\backslash\mathcal{S}$ by size, and sample uniformly at random.

Figure 3 compares the five sampling strategies—(1) random as originally designed, (2) Baseline 1, (3) Baseline 2, (4) Strategy I, (5) Strategy II—for models first trained on PolitiFact and then on GossipCop using GEM with $|\mathcal{M}| = 300$ (results for models first trained on GossipCop and then on PolitiFact are omitted due to space limits). We can see that while all models perform similarly on GossipCop (*i.e.,* the new task), Strategy II can improve the results on PolitiFact (*i.e.,* the previous task), which indicates the effectiveness of this sampling method. However, Strategy I does not work well—a comparison reveals that the selected samples differ significantly from those under Strategy II.

Algorithm 1: Sampling Strategies

Input : Sample size $|\mathcal{M}|$; The number of instances from the previous task $|\mathcal{N}|$
Output : Samples, $\mathcal{S}$

Initialise $\mathcal{S} = \{x | 0 < Margin(x) \leq \delta = 0.1\}$
Strategy I:
Sort $\mathcal{N}\backslash\mathcal{S}$ by their margin values from smallest to largest
while $|\mathcal{S}| < |\mathcal{M}|$ **do**
    for $x_i \in \mathcal{N}\backslash\mathcal{S}$ **do**
        if $\mathcal{H}(\mathcal{S}\cup\{x_i\}) > \mathcal{H}(\mathcal{S})$ **then**
            $\mathcal{S} = \mathcal{S} \cup \{x_i\}$
Strategy II:
while $|\mathcal{S}| < |\mathcal{M}|$ **do**
    $\mathcal{R}$ = randomly sample $max(\frac{|\mathcal{N}|}{|\mathcal{M}|}, 20)$ instances from $\mathcal{N}\backslash\mathcal{S}$
    $x = \text{argmax}_{x_i \in \mathcal{R}} \mathcal{H}(\mathcal{S} \cup \{x_i\}) - \mathcal{H}(\mathcal{S})$
    $\mathcal{S} = \mathcal{S} \cup \{x\}$
return $\mathcal{S}$



(a) Performance on PolitiFact (b) Performance on GossipCop

Fig. 3. Comparison of different sampling strategies for models first trained on PolitiFact and then on GossipCop using GEM ($|\mathcal{M}| = 300$).

5 Conclusions and Future Work

The prevalence of fake news over social media has become a serious social problem. Although a number of detection methods have been proposed, we identify the problem that models trained on a given dataset may not perform well on new data, and direct incremental training cannot solve the issue. Since this is similar to catastrophic forgetting in continual learning, we propose to apply two popular approaches, GEM and EWC, during the incremental training, so that balanced performance can be achieved on both existing and new data. This avoids retraining on the entire data, which becomes prohibitively expensive as data size grows. In addition, we further improve the results by optimising the sampling process with maximum entropy sampling and support samples.

For future work, we will investigate whether Algorithm 1 also improves the performance of content-based approaches. Specifically, entropy needs to be redefined, and one possibility is to calculate it over the topic of each news item.

References

1. Bian, T., et al.: Rumor detection on social media with bi-directional graph convolutional networks. arXiv:2001.06362 (2020)
2. Chen, Z., Lin, T.: Revisiting gradient episodic memory for continual learning (2019). https://openreview.net/pdf?id=H1g79ySYvB
3. Cui, L., Seo, H., Tabar, M., Ma, F., Wang, S., Lee, D.: DETERRENT: knowledge guided graph attention network for detecting healthcare misinformation. In: 26th ACM SIGKDD, KDD 2020, pp. 492–502 (2020)
4. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. eprint arXiv:1412.6572 (2014)
5. Jin, Z., Cao, J., Zhang, Y., Zhou, J., Tian, Q.: Novel visual and statistical image features for microblogs news verification. IEEE Trans. Multimedia **19**(3), 598–608 (2017)
6. Jin, Z., Cao, J., Zhang, Y., Luo, J.: News verification by exploiting conflicting social viewpoints in microblogs. In: 30th AAAI, pp. 2972–2978 (2016)
7. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. NAS **114**(13), 3521 (2017)
8. Liu, Y., Wu, Y.F.B.: Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In: 32nd AAAI, pp. 354–361 (2018)

9. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. In: 31st NeurIPS, pp. 6467–6476. Curran Associates, Inc. (2017)
10. Lu, Y.J., Li, C.T.: GCAN: graph-aware co-attention networks for explainable fake news detection on social media. arXiv:2004.11648 (2020)
11. Ma, J., Gao, W., Wong, K.F.: Detect rumors in microblog posts using propagation structure via kernel learning. In: 55th ACL, pp. 708–717 (2017)
12. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: The sequential learning problem. In: Psychology of Learning and Motivation, vol. 24, pp. 109–165. Academic Press (1989)
13. Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrák, J., Krause, A.: Lazier than lazy greedy. In: 29th AAAI, pp. 1812–1818 (2015)
14. Monti, F., Frasca, F., Eynard, D., Mannion, D., Bronstein, M.M.: Fake news detection on social media using geometric deep learning. arXiv:1902.06673
15. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. IEEE **104**(1), 11–33 (2016)
16. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: a review. arXiv:1802.07569 (2018)
17. Pierri, F., Ceri, S.: False news on social media: a data-driven survey. SIGMOD Rec. **48**(2), 18–27 (2019)
18. Popat, K., Mukherjee, S., Yates, A., Weikum, G.: Debunking fake news and false claims using evidence-aware deep learning. arXiv:1809.06416 (2018)
19. Pérez-Rosas, V., Kleinberg, B., Lefevre, A., Mihalcea, R.: Automatic detection of fake news. In: 27th COLING, pp. 3391–3401 (2018)
20. Ruchansky, N., Seo, S., Liu, Y.: CSI: a hybrid deep model for fake news detection. In: 26th CIKM, pp. 797–806 (2017)
21. Shu, K., Cui, L., Wang, S., Lee, D., Liu, H.: DEFEND: explainable fake news detection. In: 25th KDD, pp. 395–405 (2019)
22. Shu, K., Mahudeswaran, D., Wang, S., Lee, D., Liu, H.: FakeNewsNet: a data repository with news content, social context and spatialtemporal information for studying fake news on social media. arXiv:1809.01286 (2018)
23. Shu, K., Mahudeswaran, D., Wang, S., Liu, H.: Hierarchical propagation networks for fake news detection: investigation and exploitation. arXiv e-prints arXiv:1903.09196 (2019)
24. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: a data mining perspective. SIGKDD Explor. **19**(1), 22–36 (2017)
25. Shu, K., Wang, S., Liu, H.: Beyond news contents: the role of social context for fake news detection. In: 12th WSDM, pp. 312–320 (2019)
26. Tacchini, E., Ballarin, G., Della Vedova, M.L., Moret, S., de Alfaro, L.: Some like it hoax: Automated fake news detection in social networks. arXiv e-prints arXiv:1704.07506 (2017)
27. Volkova, S., Shaffer, K., Jang, J.Y., Hodas, N.: Separating facts from fiction: linguistic models to classify suspicious and trusted news posts on twitter. In: 55th ACL, pp. 647–653 (2017)
28. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. Science **359**(6380), 1146–1151 (2018)
29. Wang, W.Y.: "Liar, liar pants on fire": a new benchmark dataset for fake news detection. In: 55th ACL, pp. 422–426 (2017)
30. Wang, Y., et al.: EANN: event adversarial neural networks for multi-modal fake news detection. In: 24th KDD, pp. 849–857 (2018)
31. Wu, K., Yang, S., Zhu, K.Q.: False rumors detection on Sina Weibo by propagation structures. In: 31st ICDE, pp. 651–662 (2015)

32. Yang, S., Shu, K., Wang, S., Gu, R., Wu, F., Liu, H.: Unsupervised fake news detection on social media: a generative approach. In: 33rd AAAI, vol. 33, pp. 5644–5651 (2019)
33. Yang, Y., Zheng, L., Zhang, J., Cui, Q., Li, Z., Yu, P.S.: TI-CNN: convolutional neural networks for fake news detection. arXiv:1806.00749 (2018)
34. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: 2016 NAACL, pp. 1480–1489 (2016)
35. Ying, R., You, J., Morris, C., Ren, X., Hamilton, W.L., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. In: 32nd NeurIPS, pp. 4805–4815 (2018)
36. Zhang, J., Dong, B., Yu, P.S.: FAKEDETECTOR: effective fake news detection with deep diffusive neural network. arXiv:1805.08751 (2018)
37. Zhou, X., Wu, J., Zafarani, R.: SAFE: similarity-aware multi-modal fake news detection. In: 24th PAKDD, pp. 354–367 (2020)
38. Zhou, X., Zafarani, R.: Fake news: a survey of research, detection methods, and opportunities. arXiv:1812.00315 [cs] (2018)
39. Zhou, X., Zafarani, R.: Network-based fake news detection: a pattern-driven approach. arXiv e-prints arXiv:1906.04210 (2019)