Decaffe: DHT Tree-Based Online Federated Fake News Detection

Cheng-Wei Ching
University of California Santa Cruz
Santa Cruz, CA 95064, USA
cching1@ucsc.edu

Liting Hu
University of California Santa Cruz
Santa Cruz, CA 95064, USA
liting@ucsc.edu

Abstract

The proliferation of mobile social networks (MSNs) has transformed information dissemination, leading to increased reliance on these platforms for news consumption. However, this shift has been accompanied by the widespread propagation of fake news, posing significant challenges in terms of public panic, political influence, and the obscuring of truth. Traditional data processing pipelines for fake news detection in MSNs suffer from lengthy response times and poor scalability, failing to address the unique characteristics of news in MSNs, such as prompt propagation, large-scale quantity, and rapid evolution. This paper introduces a novel system named *Decaffe* – a DHT Tree-Based Online Federated Fake News Detection system. *Decaffe* leverages distributed hash table (DHT)-based aggregation trees for scalability and real-time detection, and it employs two model fine-tuning methods for adapting to mobile network dynamics. The system's structure includes a root, branches, and leaves for effective dissemination of a pre-trained model and ensemble-based aggregation of predictive results. *Decaffe* uniquely combines centralized server-based and decentralized serverless model fine-tuning approaches with personalized model fine-tuning, addressing the challenges of real-time detection, scalability, and adaptability in the dynamic environment of MSNs.

CCS Concepts

• **Computer systems organization** → *Peer-to-peer architectures.*

ACM Reference Format:
Cheng-Wei Ching and Liting Hu. 2024. *Decaffe*: DHT Tree-Based Online Federated Fake News Detection. In *2024 8th International Conference on Control Engineering and Artificial Intelligence (CCEAI 2024), January 26–28, 2024, Shanghai, China.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3640824.3640840

1 Introduction

With the boom of the internet and the rapid popularity of smartphones, *mobile social network* (MSNs) have become an important platform for information dissemination. More and more people tend to seek out and consume news directly from social media in MSNs rather than the traditional news media. This is because (i) it is easier to consume news on social media than the traditional news media such as newspapers or television news; and (ii) users like to comment on, share, and discuss the news with friends or other users in MSNs. However, such high openness and autonomy of MSNs lead the widespread of fake news. For instance, the government banned several What's App groups for spreading fake news about distorting the information of Agnipath Scheme For instance, a survey conducted by Statista in 2022 indicates that two-third of US adults say they have come across false information on social media [3]; and MSNs are considered the least trusted news source worldwide in 2022 [1].

Fake news spread in MSNs poses a serious negative impact on individuals and society. First, fakes news is found easy to cause public panic, which results in mass panic behaviors. Second, fake news hold much sway over the trend of political events. Propagandists take advantage of the fact that people consider a lie truthful when the lie is repeated enough times over social media. Many studies show that during the significant political events, such as the US Presidential elections and the Brexit referendum, user interactions with false content rose steadily on well-known social media [8, 9, 18]. Third, fake news makes it harder for people to see the truth. It triggers people's distrust and makes them confused, impeding their abilities to differentiate the truth from the falsity.

To help mitigate the negative effects caused by fake news, there is an urgent need to quickly detect fake news circulated in MSNs early in its propagation before it reaches a broad audience. There are three critical characteristics that differ news in MSNs from the traditional news media. First, *the prompt propagation.* When news is posted in MSNs, it spreads to numerous people in a very short period of time and will soon reach its peak rate of comments, retweets or share [28]. Second, *the large-scale quantity of news.* People nowadays spend a large amount of time browsing, sharing, and comments on news in MSNs, which creates a large-scale quantity of data in a short period of time [20]. Lastly, *the rapid evolution of news.* News in MSNs is usually forwarded or commented by people, and then rapidly converted into different context containing other details or personal opinions [18].

Unfortunately, the traditional data processing pipeline for fake news detection systems [7, 21] in MSNs mainly follow the classic big data analytics pipeline. Geo-distributed web servers first collect a large amount of users' microblog data and then transmit the data to a centralized storage tier (e.g., MongoDB). Upon the arrival of all the microblog data, big data analytics engines (e.g., Spark Streaming with MLlib [4]) begin processing and analyzing, and finally report the authenticity of the data. The pipeline above has two obvious disadvantages. One is the lengthy response time. A large amount of time is necessary to transmit microblog data from geo-distributed web servers to a central storage tier for further processing and analyzing, ending up returning the results. Such a disadvantage cannot tackle the prompt propagation of news in MSNs since the results may have expired. Another is the inferior

CCEAI 2024, January 26–28, 2024, Shanghai, China

ACM ISBN 979-8-4007-0797-1/24/01
https://doi.org/10.1145/3640824.3640840

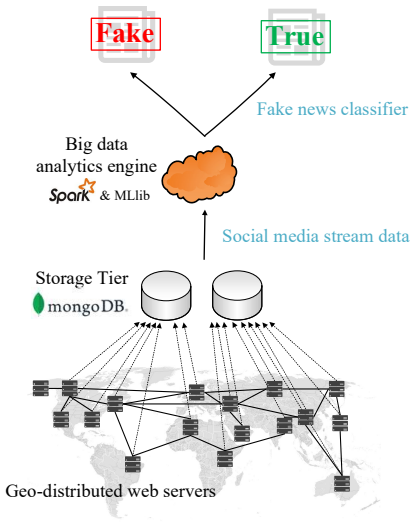


Figure 1: Traditional data processing pipeline.

scalability. The centralized data processing is ill-scalable to large-scale data, which fails to handle the large-scale quantity of news in MSNs. Two disadvantages above are the major motivation of this paper.

To this end, we propose a novel DHT Tree-Based Online Federated Fake News Detection (*Decaffe*). The key features of *Decaffe* are twofold. On the structural side, *Decaffe* establishes *distributed hash table* (DHT)-based aggregation trees to realize scalability and real-time fake news detection in MSNs. On the model fine-tuning side, *Decaffe* devises two model fine-tuning methods, *the centralized server-based model fine-tuning* and *the decentralized serverless model fine-tuning*, to fine tune *machine learning* (ML) model *Bidirectional Encoder Representations from Transformers* (BERT) for fake news detection and adapt to the mobile network dynamics.

The novelty of *Decaffe* is threefold. First, *Decaffe* uses DHT-based aggregation tree to realize the centralized server-based model fine-tuning, the decentralized serverless model fine-tuning, and ensemble-based model inference. Then, a simple yet effective differentiation method is designed to determine an appropriate model fine-tuning method. Lastly, *Decaffe* incorporates personalized model fine-tuning to take into account the heterogeneous data and detection requirements among different mobile devices.

2 Background and Preliminaries

2.1 Traditional data analytics pipeline

As shown in Fig. 1, existing studies [19, 22, 25] mostly reply on a centralized data analytics pipeline for fake news detection in MSNs. Specifically, geo-distributed web servers collects microblog data from the users or edge servers and upload the microblog data collected to a storage tier (e.g., MongoDB). Then, the big data analytics engines (e.g., Spark [4] with MLlin) process the microblog data and return the authenticity of the microblog data.

Such a centralized architecture may run well in tackling non-time-sensitive applications. However, when it comes to the fake

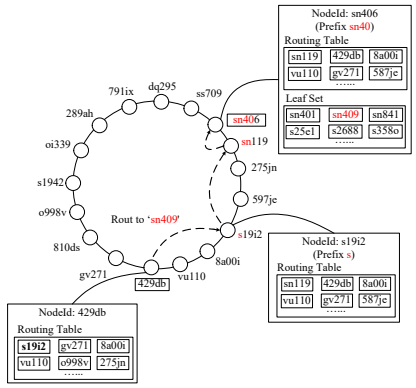


Figure 2: Message routing in Pastry.

news detection in MSNs, fake news evolves fast and usually has distinct topics and are posted and disseminated real-time, which makes the predictive results from the centralized architecture obsolete and inaccurate. This is because of (1) **high response time**. Not until the microblog data is uploaded do data analytics engines begin processing and analyzing, which cannot meet the requirement of real-time response; and (2) **Poor-scalability**.

If a new kind of topics of news emerges, the centralized architecture takes a large amount of time to adapt to it, which cannot offer accurate response when fake news evolves or changes its context.

2.2 Preliminaries

Decentralized Overlay in Pastry. Suppose there are N nodes participating in Pastry. On the initialization of Pastry, each node is assigned with an individual identifier (i.e., `nodeId`). The `nodeId` is used for node identification and message route. Given a message and a key, the message is guaranteed to reach the node with a `nodeId` numerically closet to the given key at most $O(\lceil \log_{2^b} N \rceil)$, where $b = 4$ by default. Moreover, each node maintains two data structures, *a routing table* and *a leaf set*, in support of message routing, self-organization, and fault recovery functionalities.

We first go over the designs of routing tables and leaf sets. Routing tables in each node in Pastry are composed of node prefixes, such as IP address, latency, Pastry `nodeId`, arranged in rows by the common prefix length. On the other hand, leaf sets in each node in Pastry consists of a fixed number of nodes, 24 by default, whose `nodeIds` are numerically closet to the node. Leaf sets help nodes rebuild routing tables when nodes fail. Both routing tables and leaf sets are critical to the message routing in Pastry.

For the messaging routing in Pastry, messages are routed in a greedy fashion. A simple example is depicted in Figure 2. Suppose a message with a key of `sn409` is generated in the node `429db`, the node `429db` forwards the message to a node in its routing table whose common prefix shares at least one digit with the given key, where it is the `s19i2` in this example.

If such a node does not exist, the node `429db` forwards the message to a node whose common prefix shares with the given key at least as long as the local node, and is numerically closer to the given key than the local `nodeId`. Then, the node `s19i2` follows the same logic to forward the message. The message routing ends until

the node whose `NodeId` is numerically closest to the key receives the message. In this example, the node `sn409` receives the message and the message routing is complete.

Group Management by Scribe. Scribe is a communication system built upon Pastry for the management of application-level groups [10]. Each application-level group in Scribe is built upon a logical spanning tree that consists of multiple members (i.e., nodes in Pastry). When nodes leave or rejoin the overlay, Scribe manages group sizes accordingly, and supports rapid switch for the management of group membership [10, 27]. For the group generation in Scribe, Scribe inherits the pseudorandom Pastry key to name a group, called `groupId`, where the `groupId` is usually the hash of the group's textual name concatenated with its creator's name. In practice, a Scribe node uses the messaging routing in Pastry to route a `CREATE` message with the `groupId` as the key. The node whose `nodeId` is the numerically closet to the key becomes the root of the spanning tree for the application-level group.

3 *Decaffe* Design

In this section, we introduce the *Decaffe* system, discuss each two phases in *Decaffe*, and outline the details of workflows in the *Decaffe* system.

3.1 Overview

Decaffe operates in two phases. The first phase is the construction of DHT-based aggregation trees, and the second phase is the fake news detection application and online fine-tuning. In the first phase, *Decaffe* constructs DHT-based aggregation trees, and then uses the constructed trees to execute online model fine-tuning and ensemble-based inference in the second phase.

Each DHT-based aggregation tree consists of a root, multiple branches, and multiple leaf nodes. The root serves as the main control flows for the entire aggregation tree, such as the dissemination of initial model and intermediate fine-tuning model to the leaf nodes and the aggregation of final predictive results or intermediate fine-tuning models from the leaf nodes. The branches are responsible for two main works. One is the dissemination as what the root is supposed to do. Another is the intermediate aggregation of intermediate fine-tune models and predictive results such that the network bandwidth can be further conserved. The leaf nodes are responsible for performing local model fine-tuning for fake news detection. When new data comes to the leaf nodes, they first infer the new data with local fine-tuned model.

In the second phase, the leaf nodes execute two tasks concurrently. One task is the online mode fine-tuning and the other task is the ensemble-based model inference. The online model fine-tuning is launched when the current models in the leaf nodes are outdated. The ensemble-based model inference is for inferring the new coming data on the leaf nodes and the predictive results will be aggregated along the aggregation tree to the root. The final predictive results are attained in an ensemble manner.

3.2 Construction of DHT-based aggregation trees

Root. The root serves as the main control flows for the entire aggregation tree, including 1) the dissemination of initial pre-trained model and intermediate fine-tuning model to the leaf nodes, and 2) the aggregation of final predictive results or intermediate fine-tuning models from the leaf nodes.

The root uses a DHT-based aggregation tree as the main approach to communicate with the leaf nodes. The DHT-based aggregation trees are constructed as follows:

Step 1: The first step is the construction of a peer-to-peer overlay network leveraging Pastry [27]. Each node is first assigned a unique `nodeId` in a circular `nodeId` space with the range from 0 to $2^{128}-1$, where the `nodeId` of each node is uniformly distributed. Given a message and a key in a node, the message can be guaranteed to be routed to the node whose `nodeId` is numerically closest to the given key at most $O(\log_{2^b} N)$ hops, where $b = 4$ by default.

Step 2: The second step builds a multicast tree leveraging Scribe [10]. Each node in the overlay is able to generate a group with a `groupId` that is usually the groups's textual name concatenated with its creator's name. Upon the generation of a group, each node can join the group by routing a `JOIN` message towards the `groupId`. Multicast messages can be sent from the root to any member node at most $O(\log N)$ hops.

Step 3: The root works with the branches on the aggregation of online model fine-tuning and ensemble-based model inference. For online model fine-tuning, the root and the branches aggregate the intermediate fine-tuned model, and the root disseminates the finally aggregated fine-tuned model back to the leaf node. For ensemble-based model, the root and the branches aggregate the predictive results from the leaf nodes, and the root provides the end users with final predictive results using the ensemble-based model inference.

Leaf nodes. The leaf nodes execute two major tasks concurrently: the online model fine-tuning and fake news detection. The root will disseminate the initial pre-trained model to the leaf nodes. Upon receiving the initial model, the leaf nodes perform local model fine-tuning with local data. When new data coming from the users, the leaf nodes infer the new data with the local fine-tuned model and send the predictive results to the root along the aggregation tree. Once the local fine-tuned model is outdated, the leaf nodes launch online model fine-tuning to update the local fine-tuned model.

Self-adjustable aggregation trees. The aggregation trees in *Decaffe* can self-tune the tree structure level by manipulating the fanout. Specifically, the fanout is the maximum degree of each node in the aggregation tree. For example, when the fanout is 2, then the aggregation tree is as much as a binary tree. When the fanout is N, where N is the number of nodes, then the aggregation tree changes to a star graph. The value of fanout can be adaptive to the characteristics of applications. Suppose an application is latency-intensive. The fanout of aggregation trees can be as small as possible such that the height of aggregation trees is small, which take less time on disseminating and aggregating information. On the other hand, higher aggregation trees are more fault-tolerant. Each node in *Decaffe* is constantly receiving a existence message from its parent. Then nodes will route the `JOIN` message again to rejoin the aggregation tree when their parents fail to transmit existence messages. When a aggregation tree has a smaller fan-out and a larger height, the failure of one node can only affect a small number of nodes.

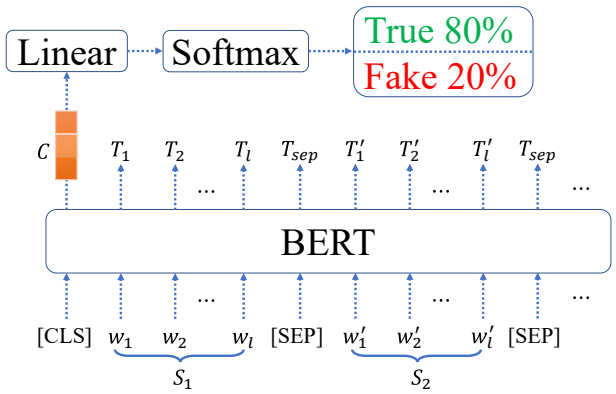


Figure 3: The BERT model structure

3.3 Online model fine-tuning and ensemble-based model inference

Online model fine-tuning. Inspired by the current success of BERT on*natural language processing* (NLP), we use BERT as the base model. BERT is a language representation model designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers [23]. In practice, we add one more output layer in the pre-trained BERT and the leaf nodes use local data to fine-tine the pre-trained BERT. The pre-trained BERT model is fine-tuned as follows:

Step 1: When receiving new data from the users, the leaf nodes first use the standard WordPiece embeddings to tokenize the new data into multiple tokens.

Step 2: Feed the tokens to the pre-trained BERT and obtain the predictive results.

Step 3: Calculate the loss between the ground truth and the predictive results and the corresponding gradients.

Step 4: Upload the intermediate fine-tuned models to the root or to the next leaf nodes along the aggregation tree, depending on the selected model fine-tuning rules, the centralized server-based model fine-tuning or the decentralized serverless model fine-tuning.

Model structure and loss function. The BERT model is shown in Fig. 3. In particular, the input of BERT includes `CLS` the classification token, `SEP` the separation token, $w_1, \cdots, w_l$ and $w_1', \cdots, w_l'$ that represents the word in sentence S_1 and S_2', respectively. The output of BERT consists of $T_1, \cdots T_l$ and $T_1', \cdots T_l'$ the hidden vector for input $w_1, \cdots, w_l$ and $w_1', \cdots, w_l'$, respectively, and C and T_{sep} represent the hidden vector of the input `[CLS]` and `[SEP]`.

We use the hidden vector C as sequence representation for fake news detection. The major parameters for federated-based fine-tuning is the weights in `Linear` $w_{cla} \in \mathbb{R}^{L \times H}$, where $L \in \{0, 1\}$ is the number of labels and in fake news detection, H, the dimension of the hidden vector C.

For the loss function, we leverage the loss function in *personalized federated optimization* (PFO) [11, 16] in order to retain the data characteristics in the different leaf nodes and use federated-based method to perform online model fine-tuning. Therefore, the loss function for the leaf node n to perform online model fine-tuning is as follows:

$$l_n(w_{glo}) := \mathbb{E}_{(x,y)\sim\mathcal{D}_n}\Big[\sum_{i\in\{0,1\}} \mathbf{1}_{y=i} \log f_i(x; w_{cla})\Big] + \frac{\lambda}{2}\|w_{per}^n - w_{cla}\|, \quad (1)$$

where $(x, y) \sim \mathcal{D}_n$ denotes the data pair in local data $\mathcal{D}_n$, w_{per}^n, the personalized fine-tuned model in the leaf node n, w_{glo}, the global fine-tuned model, $f_i(x; w_{cla})$, the loss function for class i parameterized by w_{cla} to the sample x. The leaf nodes will follow the above loss function to obtain intermediate fine-tuned models.

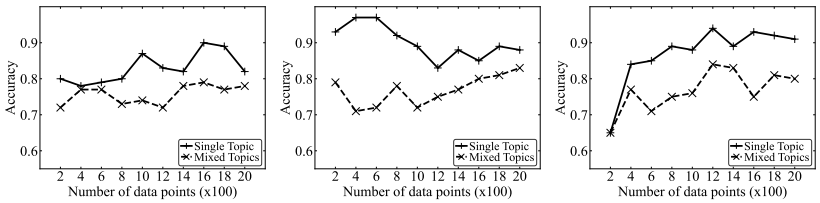
Centralized server-based model fine-tuning. We see that the leaf nodes only optimize with respect to their local data. However, due to the characteristic of the rapid evolution of news, applying only one fine-tuned model for detection is insufficient. Therefore, the leaf nodes upload the intermediate fine-tuned models with respect to their loss to the root along the aggregation tree so that the leaf nodes can obtain a fine-tuned model that have better generalized performance for varying new data from MSNs.

As shown in Fig. 1, the centralized server-based model fine-tuning relies on the communication between the root and the leaf nodes. Specifically, in communication round t, each leaf node $n \in N$ use the local new data $\mathcal{D}_n$ to perform local model fine-tuning and obtain the intermediate fine-tuned models $w_{cla}^{n,t}$. Then, the intermediate fine-tuned models are uploaded to the parent nodes of the corresponding leaf nodes. The parent nodes perform intermediate aggregation as follows: $w_{cla}^{n,t} = \sum_{j\in C_n} \frac{w_{cla}^{j,t}}{|C_n|}$, where C_n represents the child nodes of the node n, and $|\cdot|$, the cardinality operator. The intermediate aggregation can reduce the communication overhead by a factor of $O(\log N)$ since it maintains at most $O(\log N)$ instead of N point-to-point connections for N leaf nodes.

When receiving the intermediate fine-tuned models from its child nodes, the root r performs final model fine-tuning to yield a new global fine-tuned model as follows: $w_{cla}^{t+1} = w_{cla}^{t} - \eta \sum_{j\in C_r} \frac{w_{cla}^{j,t}}{|C_r|}$. Lastly, the root uses the aggregation tree to multicast the updated fine-tuned model to each leaf node. Once the leaf nodes require to perform online model fine-tuning again, the process above is repeated again.

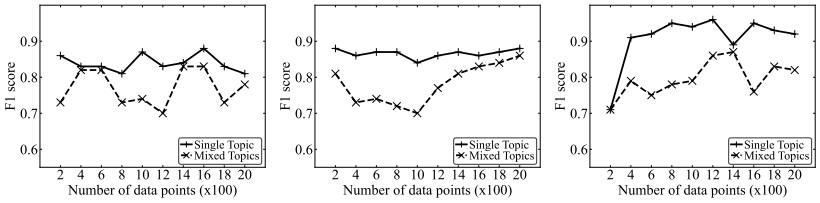
We can observe that an one-trip communication from leaf nodes to the root and back to leaf nodes is necessary for the centralized server-based model fine-tuning. In spite of the simplicity of the centralized server-based model fine-tuning, the entire aggregation tree easily suffer from communication and computational bottlenecks that happen to each member node [24], especially in a mobile social network, where the large-scale quantity of news and the rapid quantity of data are two critical characteristics of MSNs. Moreover, to secure the user data from the model inversion attacks that reproduce the data for model fine-tuning [12, 14, 15], some security-intensive applications incorporate random noises into model fine-tuning [6, 12]. Such a random noise is proved to pose negative impact on predictive performance.

The decentralized serverless model fine-tuning. To mitigate the possible communication and computational bottleneck, we device the decentralized serverless model fine-tuning. Specifically, the leaf nodes first perform local model fine-tuning and then upload their gradients to one of leaf nodes that *have social links one another* [13, 17].



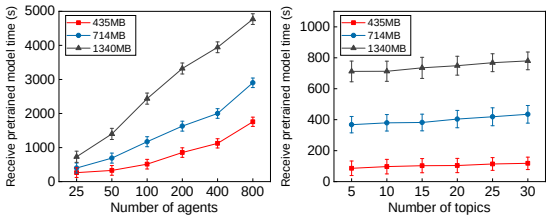
(a) COVID19 News (b) Vaccination News (c) Election News

Figure 4: Accuracy on three news datasets: COVID19 news, vaccination news, and election news.



(a) COVID19 News (b) Vaccination News (c) Election News

Figure 5: F1 score on three news datasets: COVID19 news, vaccination news, and election news.



(a) Numbers of agents in an aggregation tree. (b) Numbers of aggregation trees for multiple topics.

Figure 6: Received pre-trained model time

After receiving the intermediate fine-tuned models from their trust leaf nodes, the leaf nodes aggregate the intermediate fine-tuned models and continue uploading gradients until the root receives the intermediate fine-tuned models.

As shown in Fig. 2, the decentralized serverless model fine-tuning relies on the peer-to-peer communication between leaf nodes and ends in the root. Specifically, in communication round t, each leaf node $n \in N$ use the local new data $\mathcal{D}_n$ to perform local model fine-tuning and obtain the intermediate fine-tuned models w_{cla}^{n}. Then, the leaf nodes send the intermediate fine-tuned models to the friend leaf nodes that has social links with them. The friend leaf nodes perform intermediate aggregation as follows: $w_{cla}^{i,t} = \sum_{j \in \mathcal{F}_i} \frac{w_{cla}^{j,t}}{|\mathcal{F}_i|}$, where $\mathcal{F}_i$ represents the leaf nodes that have social links with the leaf node i.

Upon receiving the intermediate fine-tuned models from all of the friend leaf nodes, the root r performs final model fine-tuning to yield new global fine-tuned model as follows: $w_{cla}^{t+1} = w_{cla}^{t} - \eta \sum_{j \in \mathcal{F}_r} \frac{w_{cla}^{j,t}}{|\mathcal{F}_r|}$.

Compared to the centralized server-based model fine-tuning, the decentralized serverless model fine-tuning has the following two advantages. One is that the communication and computational bottlenecks can be largely distributed since only a set of leaf nodes are uploading and another set of leaf nodes are computing. Another is that the random noises are unnecessary sine the leaf nodes only share the intermediate fine-tuned models with their friend leaf nodes. Even though the intermediate fine-tuned models will be sent to the leaf nodes two hops away, they cannot know exactly what it is. This is because the intermediate fine-tuned models are first aggregated then sent. It is nearly impossible to the values before addition.

Ensemble-based model inference. The leaf nodes perform online mode fine-tuning and model inference concurrently. When receiving new data and needing to infer the predictive results for it, the leaf nodes use the fine-tuned model to obtain the predictive results, and then upload the predictive results to the root along the aggregation trees. The root finally aggregates the predictive results in an ensemble-based manner.

4 Performance Evaluation

4.1 Setup

Emulation Deployment. We run all experiments on up to 4 servers, each with 16 Intel Xeon Gold 6130@2.10GHz cores and 256GB of RAM, running GNU/Linux 3.10.0. On top of these servers, we boot up 100 virtual machines to host 1000 DHT nodes in total, each with 4 cores and 8GB of memory, running Linux Ubuntu 16.04.4.

Datasets and models. We evaluate *Decaffe* using three real-world datasets: COVID-19 Fake News Dataset [26], COVID-19 World Vaccination Progress [2], and US General Election [5]. Each dataset is regarded as an individual topic. We train a transformer model (BERT [23]) to classify the news into true or false classes.

4.2 Prediction Results

Figure 4 and Figure 5 show *Decaffe*'s prediction accuracy and F1 score, respectively. Single topic means that the training data over the nodes in an aggregation tree belongs to the same dataset, whereas mixed topics mean that each node in an aggregation tree has training data from different datasets. To rule out the effect of the amount of training data on classification accuracy and F1 score, we increase the number of data points in each node from 200 to 2000. We can see that single topic supported by *Decaffe* achieves higher accuracy and F1 score on three datasets. This is because *Decaffe* enables each topic to build an individual aggregation tree so that the nodes in different aggregation trees can contribute their model updates to correct global models.

4.3 Scalability Analysis

Figure 6(a) shows the received pre-trained model time for the whole nodes in an aggregation tree to receive different sizes of pre-trained models. Figure 6(b) shows the received pre-trained model time for the whole nodes in multiple aggregation trees to receive different sizes of pre-trained models. We can see that for a single aggregation tree, the received time only increases linearly, not exponentially. This is because the time is limited by the aggregation tree depth $O(\log N)$ by using the DHT-based aggregation tree. On the other hand, we can see that for multiple aggregation trees, the increase of the received time is negligible. This is because the communication overhead for model dissemination is amortized over the whole nodes in the decentralized overlay.

5 Conclusion

Decaffe represents a significant advancement in the field of fake news detection within mobile social networks. By addressing the critical challenges of real-time detection, scalability, and adaptability to mobile network dynamics, *Decaffe* provides an effective solution to the pervasive issue of fake news in MSNs. The system's innovative use of DHT-based aggregation trees and its dual approach to model fine-tuning enable it to handle the large-scale and rapidly evolving nature of news on these platforms. The ensemble-based model inference and personalized model fine-tuning methods further enhance its efficacy and adaptability to diverse data and detection requirements. Our comprehensive evaluation using three real-world fake news datasets demonstrates *Decaffe*'s superior performance and functionality, marking it as a pivotal contribution to online fake news detection systems. This system not only defends against the ever-changing landscape of fake news but also sets a precedent for future research and development in this crucial domain.

Acknowledgments

This work is supported by the National Science Foundation under Grants NSF-OAC-23313738, NSF-CAREER-23313737, NSF-SPX-2202859, and NSF-CNS-2322919.

References

[1] . *2022 EDELMAN TRUST BAROMETER.* https://www.edelman.com/sites/g/files/aatuss191/files/2022-01/2022EdelmanTrustBarometerFINAL_Jan25.pdf.
[2] . *COVID-19 World Vaccination Progress* . https://www.kaggle.com/datasets/gpreda/covid-world-vaccination-progress/code.
[3] . *Fake news in the U.S. - statistics & facts.* https://www.statista.com/topics/3251/fake-news/#dossierKeyfigures.
[4] . *Spark MLlib.* https://spark.apache.org/mllib/.
[5] . *US Genernal Election Dataset.* https://github.com/sydney-machine-learning/sentimentanalysis-USelections.
[6] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 308–318.
[7] Divy Agrawal, Sanjay Chawla, Bertty Contreras-Rojas, Ahmed Elmagarmid, Yasser Idris, Zoi Kaoudi, Sebastian Kruse, Ji Lucas, Essam Mansour, Mourad Ouzzani, et al. 2018. RHEEM: enabling cross-platform data processing: may the big data be with you! *Proceedings of the VLDB Endowment* 11, 11 (2018), 1414–1427.
[8] Hunt Allcott, Matthew Gentzkow, and Chuan Yu. 2019. Trends in the diffusion of misinformation on social media. *Research & Politics* 6, 2 (2019), 2053168019848554.
[9] Marco T Bastos and Dan Mercea. 2019. The Brexit botnet and user-generated hyperpartisan news. *Social science computer review* 37, 1 (2019), 38–54.
[10] Miguel Castro, Peter Druschel, A-M Kermarrec, and Antony IT Rowstron. 2002. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in communications* 20, 8 (2002), 1489–1499.
[11] Cheng-Wei Ching, Jia-Ming Chang, Jian-Jhih Kuo, and Chih-Yu Wang. 2023. Dual-Objective Personalized Federated Service System with Partially-labeled Data over Wireless Networks. *IEEE Transactions on Services Computing* (2023).
[12] Cheng-Wei Ching, Xin Chen, Taehwan Kim, Bo Ji, Qingyang Wang, Dilma Da Silva, and Liting Hu. 2024. Totoro: A Scalable Federated Learning Engine for the Edge. In *Proceedings of the Nineteenth European Conference on Computer Systems*. 1–18.
[13] Cheng-Wei Ching, Hung-Sheng Huang, Chun-An Yang, Jian-Jhih Kuo, and Ren-Hung Hwang. 2021. Efficient Online Decentralized Learning Framework for Social Internet of Things. In *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–6.
[14] Cheng-Wei Ching, Hung-Sheng Huang, Chun-An Yang, Yu-Chun Liu, and Jian-Jhih Kuo. 2021. Efficient communication topology via partially differential privacy for decentralized learning. In *2021 International Conference on Computer Communications and Networks (ICCCN)*. IEEE, 1–9.
[15] Cheng-Wei Ching, Tzu-Cheng Lin, Kung-Hao Chang, Chih-Chiung Yao, and Jian-Jhih Kuo. 2020. Model partition defense against gan attacks on collaborative learning via mobile edge computing. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 1–6.
[16] Cheng-Wei Ching, Yu-Chun Liu, Chung-Kai Yang, Jian-Jhih Kuo, and Feng-Ting Su. 2020. Optimal device selection for federated learning over mobile edge networks. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 1298–1303.
[17] Cheng-Wei Ching, Chung-Kai Yang, Yu-Chun Liu, Chia-Wei Hsu, Jian-Jhih Kuo, Hung-Sheng Huang, and Jen-Feng Lee. 2020. Energy-efficient link selection for decentralized learning via smart devices with edge computing. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 1–6.
[18] Giandomenico Di Domenico, Jason Sit, Alessio Ishizaka, and Daniel Nunan. 2021. Fake news, social media and marketing: A systematic review. *Journal of Business Research* 124 (2021), 329–341.
[19] Zhiming Hu, James Tu, and Baochun Li. 2019. Spear: Optimized dependency-aware task scheduling with deep reinforcement learning. In *2019 IEEE 39th*

International Conference on Distributed Computing Systems (ICDCS). IEEE, 2037–2046.

[20] Chiungjung Huang. 2017. Time spent on social network sites and psychological well-being: A meta-analysis. *Cyberpsychology, Behavior, and Social Networking* 20, 6 (2017), 346–354.

[21] Chien-Chun Hung, Ganesh Ananthanarayanan, Leana Golubchik, Minlan Yu, and Mingyang Zhang. 2018. Wide-area analytics with multiple resources. In *Proceedings of the Thirteenth EuroSys Conference*. 1–16.

[22] Albert Jonathan, Abhishek Chandra, and Jon Weissman. 2018. Multi-query optimization in wide-area streaming analytics. In *Proceedings of the ACM symposium on cloud computing*. 412–425.

[23] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[24] Jian-Jhih Kuo, Cheng-Wei Ching, Hung-Sheng Huang, and Yu-Chun Liu. 2021. Energy-efficient topology construction via power allocation for decentralized learning via smart devices with edge computing. *IEEE Transactions on Green Communications and Networking* 5, 4 (2021), 1806–1819.

[25] Kwangsung Oh, Minmin Zhang, Abhishek Chandra, and Jon Weissman. 2021. Network Cost-Aware Geo-Distributed Data Analytics System. *IEEE Transactions on Parallel and Distributed Systems* 33, 6 (2021), 1407–1420.

[26] Parth Patwa, Shivam Sharma, Srinivas Pykl, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. 2021. Fighting an infodemic: Covid-19 fake news dataset. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation: First International Workshop, CONSTRAINT 2021, Collocated with AAAI 2021, Virtual Event, February 8, 2021, Revised Selected Papers 1*. Springer, 21–29.

[27] Antony Rowstron and Peter Druschel. 2001. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*. Springer, 329–350.

[28] Liang Wu and Huan Liu. 2018. Tracing fake-news footprints: Characterizing social media messages by how they propagate. In *Proceedings of the eleventh ACM international conference on Web Search and Data Mining*. 637–645.