2023
Volume 23, Number 1

ACM Transactions on

Internet Technology

Article 1 (22 pages) Real-time Pricing-based Resource Allocation in Open Market Environments
P. Mishra, A. Moustafa, and T. Ito

Article 2 (33 pages) Reaching for the Sky: Maximizing Deep Learning Inference Throughput on Edge Devices with AI Multi-Tenancy
J. Hao, P. Subedi, L. Ramaswamy, and I. K. Kim

Article 3 (18 pages) SAM: Multi-turn Response Selection Based on Semantic Awareness Matching
R. Zhang, T. Wu, S. Wen, S. Nepal, C. Paris, and Y. Xiang

Article 4 (30 pages) The Tip of the Buyer: Extracting Product Tips from Reviews
S. Hirsch, S. Novgorodov, I. Guy, and A. Nus

Article 5 (12 pages) Breaking CaptchaStar Using the BASECASS Methodology
C. Hernández-Castro, D. F. Barrero, and M. D. R-Moreno

Article 6 (31 pages) Knowledge Graph Construction with a *Façade*: A Unified Method to Access Heterogeneous Data Sources on the Web
L. Asprino, E. Daga, A. Gangemi, and P. Mulholland

Article 7 (21 pages) Joint Architecture Design and Workload Partitioning for DNN Inference on Industrial IoT Clusters
W. Fang, W. Xu, C. Yu, and N. N. Xiong

continued on back cover

Association for Computing Machinery

Advancing Computing as a Science & Profession

ACM Transactions on Internet Technology

Vol. 23 • No. 1 • 2023

Articles 1–23

Article 8 (29 pages) White Box: On the Prediction of Collaborative Filtering Recommendation Systems' Performance
I. Paun, Y. Moshfeghi, and N. Ntarmos

Article 9 (25 pages) Elastically Augmenting the Control-path Throughput in SDN to Deal with Internet DDoS Attacks
Y. Dai, A. Wang, Y. Guo, and S. Chen

Article 10 (26 pages) Facilitating Serverless Match-based Online Games with Novel Blockchain Technologies
F. Wu, H. Y. Yuen, H. Chan, V. C. M. Leung, and W. Cai

Article 11 (28 pages) The Doge of Wall Street: Analysis and Detection of Pump and Dump Cryptocurrency Manipulations
M. La Morgia, A. Mei, F. Sassi, and J. Stefa

Article 12 (22 pages) Federated Route Leak Detection in Inter-domain Routing with Privacy Guarantee
M. Zeng, D. Li, P. Zhang, K. Xie, and X. Huang

Article 13 (24 pages) Uncertainty-Aware Personal Assistant for Making Personalized Privacy Decisions
G. Ayci, M. Sensoy, A. Özgür, and P. Yolum

Article 14 (30 pages) L2DART: A Trust Management System Integrating Blockchain and Off-Chain Computation
A. De Salve, L. Franceschi, A. Lisi, P. Mori, and L. Ricci

Article 15 (22 pages) A Low-code Development Framework for Cloud-native Edge Systems
W. Zhang, Y. Zhang, H. Fan, Y. Gao, and W. Dong

Article 16 (24 pages) A Multi-type Classifier Ensemble for Detecting Fake Reviews Through Textual-based Feature Extraction
G. Budhi and R. Chiong

Article 17 (25 pages) Finding the Source in Networks: An Approach Based on Structural Entropy
C. Zhang, Q. Guo, L. Fu, J. Ding, X. Cao, F. Long, X. Wang, and C. Zhou

Article 18 (21 pages) SDN-enabled Resource Provisioning Framework for Geo-Distributed Streaming Analytics
H. Mostafaei and S. Afridi

Article 19 (22 pages) Attacking DoH and ECH: Does Server Name Encryption Protect Users' Privacy?
M. Trevisan, F. Soro, M. Mellia, I. Drago, and R. Morla

Article 20 (26 pages) Tuneman: Customizing Networks to Guarantee Application Bandwidth and Latency
S. Sharma, A. Kushwaha, M. Alizadeh, G. Varghese, and A. Gumaste

Article 21 (24 pages) Conco-ERNIE: Complex User Intent Detect Model for Smart Healthcare Cognitive Bot
B. Zhang, Z. Tu, S. Hang, D. Chu, and X. Xu

Article 22 (28 pages) BREAKING BAD: Quantifying the Addiction of Web Elements to JavaScript
R. Fouquet, P. Laperdrix, and R. Rouvoy

Article 23 (21 pages) Movie Account Recommendation on Instagram
Y.-J. Wang and A. J. T. Lee

ACM Transactions on **Internet Technology**

ACM
1601 Broadway, 10th Floor
New York, NY 10019-7434
Tel.: 212-869-7440
Fax: 212-869-0481
http://www.acm.org

Home Page: http://toit.acm.org/

The ACM Transactions on Internet Technology (ISSN 1533-5399) is published quarterly in Spring, Summer, Fall, and Winter by the Association for Computing Machinery (ACM), 1601 Broadway, 10th Floor, New York, NY 10019-7434. Printed in the U.S.A.

Association for Computing Machinery

Advancing Computing as a Science & Profession

ACM Transactions on Internet Technology

http://toit.acm.org/

Guide to Manuscript Submission
Submission to the *ACM Transactions on Internet Technology* is done electronically through http://acm.manuscriptcentral.com. Once you are at that site, you can create an account and password with which you can enter the ACM Manuscript Central manuscript review tracking system. From a drop-down list of journals, choose *ACM Transactions on Internet Technology* and proceed to the Author Center to submit your manuscript and your accompanying files.

You will be asked to create an abstract that will be used throughout the system as a synopsis of your paper. You will also be asked to classify your submission using the ACM Computing Classification System through a link provided at the Author Center. For completeness, please select at least one primary-level classification followed by two secondary-level classifications. To make the process easier, you may cut and paste from the list. Remember, you, the author, know best which area and sub-areas are covered by your paper; in addition to clarifying the area where your paper belongs, classification often helps in quickly identifying suitable reviewers for your paper. So it is important that you provide as thorough a classification of your paper as possible.

The ACM Production Department prefers that your manuscript be prepared in either LaTeX or Ms Word format. Style files for manuscript preparation can be obtained at the following location: http://www.acm.org/publications/submissions. For editorial review, the manuscript should be submitted as a PDF or Postscript file. Accompanying material can be in any number of text or image formats, as well as software/documentation bundles in zip or tar-gzipped formats.

Questions regarding editorial review process should be directed to the Editor-in-Chief. Questions regarding the post-acceptance production process should be addressed to the Associate Editor, Stacey Schick at schick@hq.acm.org.

Subscription, Single Copy, and Membership Information.

Send orders to:

ACM Member Services Dept.
General Post Office
PO Box 30777
New York, NY 10087-0777

For information, contact:

Mail: ACM Member Services Dept.
1601 Broadway, 10th Floor
New York, NY 10019-7434
Phone: +1-212-626-0500
Fax: +1-212-944-1318
Email: acmhelp@acm.org
Catalog: http://www.acm.org/catalog

Subscription rates for *ACM Transactions on Internet Technology* are $67 per year for ACM members, $62 for students, and $260 for nonmembers. Single copies are $18 each for ACM members and $40 for nonmembers. Your subscription expiration date is coded in four digits at the top of your mailing label; the first two digits show the year, the last two show the month of expiration.

About ACM. ACM is the world's largest educational and scientific computing society, uniting educators, researchers and professionals to inspire dialogue, share resources and address the field's challenges. ACM strengthens the computing profession's collective voice through strong leadership, promotion of the highest standards, and recognition of technical excellence. ACM supports the professional growth of its members by providing opportunities for life-long learning, career development, and professional networking.

Change of Address Notification. To notify ACM of a change of address, use the addresses above or send an email to coa@acm.org.

Please allow 6-8 weeks for new membership or change of name and address to become effective. Send your old label with your new address notification. To avoid interruption of service, notify your local post office before change of residence. For a fee, the post office will forward 2nd- and 3rd-class periodicals.

ACM Transactions on Internet Technology

COUNTRY

United States

Universities and research institutions in United States

Media Ranking in United States

SUBJECT AREA AND CATEGORY

Computer Science
Computer Networks and Communications

PUBLISHER

Association for Computing Machinery (ACM)

SJR 2024

1.000 Q1

H-INDEX

71

PUBLICATION TYPE

Journals

ISSN

15335399, 15576051

COVERAGE

2001-2024

INFORMATION

Homepage

How to publish in this journal

ling.liu@cc.gatech.edu

SCOPE

ACM Transactions on Internet Technology (TOIT) brings together many computing disciplines including computer software engineering, computer programming languages, middleware, database management, security, knowledge discovery and data mining, networking and distributed systems, communications, performance and scalability etc. TOIT will cover the results and roles of the individual disciplines and the relationshipsamong them.

Join the conversation about this journal

Quartiles

FIND SIMILAR JOURNALS

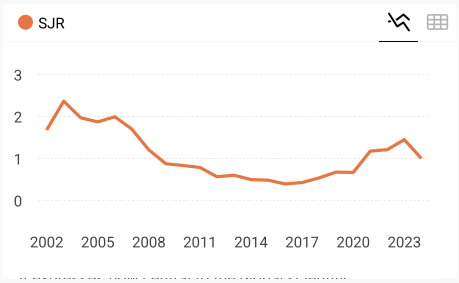
1 **Future Generation Computer Systems** NLD 81% similarity

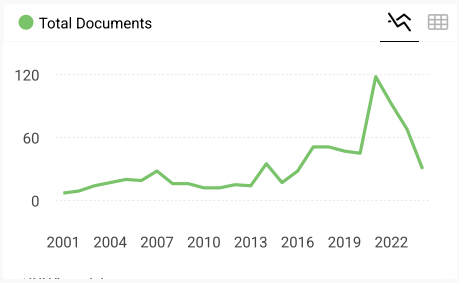
2 **Computer Science Review** IRL 80% similarity

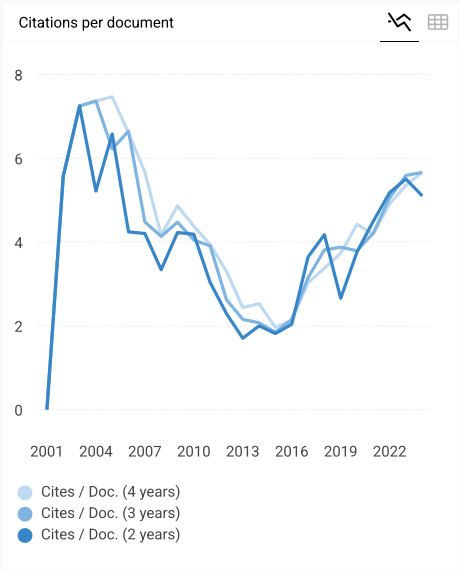
3 **Computing** AUT 76% similarity

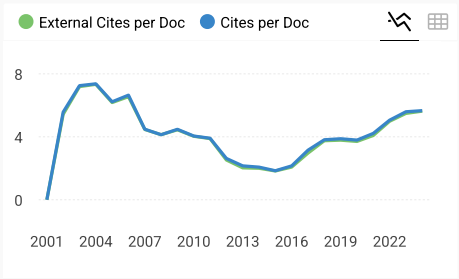
4 **Pervasive and Mobile Computing** NLD 76% similarity

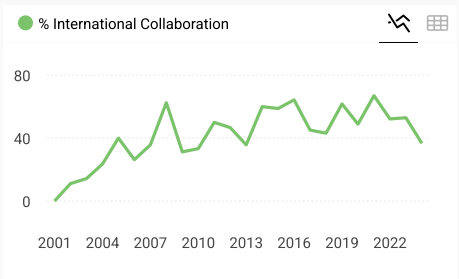
5 **Journa** USA

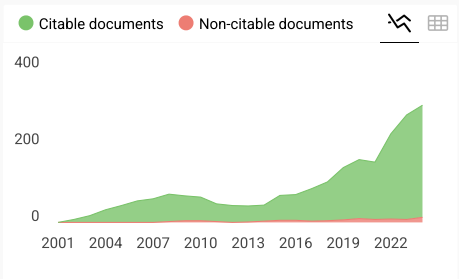


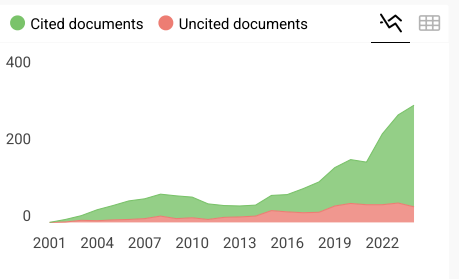


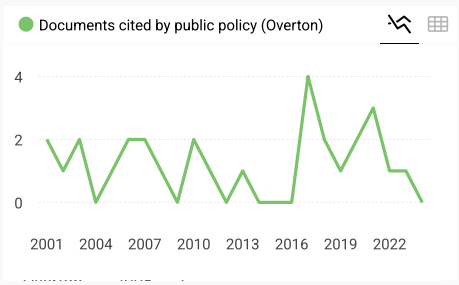


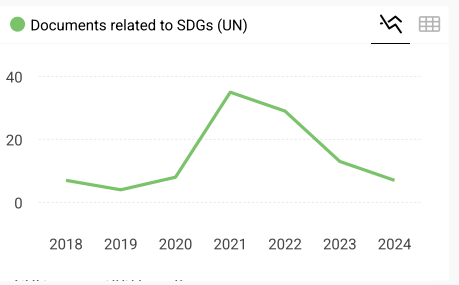


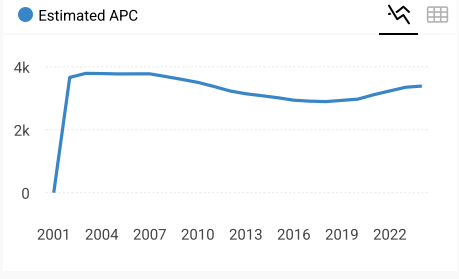


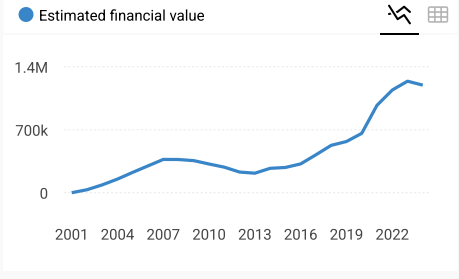












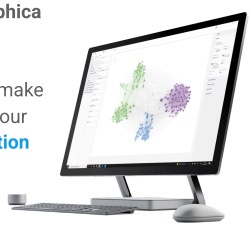
← Show this widget in your own website

Just copy the code below and paste within your html code:

<a href="https://www.scimag

SCImago Graphica

Explore, visually communicate and make sense of data with our **new data visualization tool**.



Metrics based on Scopus® data as of March 2025

Leave a comment

Name

Source details

ACM Transactions on Internet Technology

Years currently covered by Scopus: from 2001 to 2025

Publisher: Association for Computing Machinery

ISSN: 1533-5399 E-ISSN: 1557-6051

Subject area: Computer Science: Computer Networks and Communications

Source type: Journal

View all documents > | Set document alert | Save to source list

CiteScore 2024
13.9

SJR 2024
1.000

SNIP 2024
1.345

CiteScore | CiteScore rank & trend | Scopus content coverage

CiteScore 2024

$$13.9 = \frac{4{,}089 \text{ Citations } 2021 - 2024}{295 \text{ Documents } 2021 - 2024}$$

Calculated on 05 May, 2025

CiteScoreTracker 2025

$$12.0 = \frac{2{,}297 \text{ Citations to date}}{192 \text{ Documents to date}}$$

Last updated on 05 October, 2025 • Updated monthly

CiteScore rank 2024

Category	Rank	Percentile
Computer Science Computer Networks and Communications	#25/507	95th

View CiteScore methodology > | CiteScore FAQ > | Add CiteScore to your site

A Multi-type Classifier Ensemble for Detecting Fake Reviews Through Textual-based Feature Extraction

GREGORIUS SATIA BUDHI, School of Information and Physical Sciences, The University of Newcastle, Callaghan, NSW, Australia, and Informatics Department, Petra Christian University, Surabaya, Indonesia
RAYMOND CHIONG, School of Information and Physical Sciences, The University of Newcastle, Callaghan, NSW, Australia

The financial impact of online reviews has prompted some fraudulent sellers to generate fake consumer reviews for either promoting their products or discrediting competing products. In this study, we propose a novel ensemble model—the **Multi-type Classifier Ensemble (MtCE)**—combined with a textual-based featuring method, which is relatively independent of the system, to detect fake online consumer reviews. Unlike other ensemble models that utilise only the same type of single classifier, our proposed ensemble utilises several customised machine learning classifiers (including deep learning models) as its base classifiers. The results of our experiments show that the MtCE can adequately detect fake reviews, and that it outperforms other single and ensemble methods in terms of accuracy and other measurements for all the relevant public datasets used in this study. Moreover, if set correctly, the parameters of MtCE, such as base-classifier types, the total number of base classifiers, bootstrap, and the method to vote on output (e.g., majority or priority), can further improve the performance of the proposed ensemble.

CCS Concepts: • **Computing methodologies** → **Machine learning**;

Additional Key Words and Phrases: Fake review detection, online commerce security, novel ensemble model, machine learning, deep learning

ACM Reference format:
Gregorius Budhi and Raymond Chiong. 2023. A Multi-type Classifier Ensemble for Detecting Fake Reviews Through Textual-based Feature Extraction. *ACM Trans. Internet Technol.* 23, 1, Article 16 (April 2023), 24 pages.
https://doi.org/10.1145/3568676

1 INTRODUCTION

Fake review detection is a hot research topic in natural language processing [55]. A fake review is a consumer review of a product with fictitious opinions written deliberately to sound authentic, for commercial motives; i.e., to promote or damage the reputation of the reviewed product [42, 50]. There is a significant possibility that fake reviews distort the actual evaluation of a product

Authors' addresses: G. S. Budhi, School of Information and Physical Sciences, The University of Newcastle, Callaghan, NSW 2308, Australia, and Informatics Department, Petra Christian University, Surabaya 60236, Indonesia; emails: Gregorius.Satiabudhi@uon.edu.au, Greg@petra.ac.id; R. Chiong (corresponding author), School of Information and Physical Sciences, The University of Newcastle, Callaghan, NSW 2308, Australia; email: Raymond.Chiong@newcastle.edu.au.

1533-5399/2023/04-ART16 $15.00
https://doi.org/10.1145/3568676

[23], create harmful effects, and eventually reduce trust in consumer reviews and undermine the effectiveness of the online market [43]. These fraudulent acts occur in response to the vital role of consumer reviews in shaping purchasing behaviour in online markets [68]; consumers trust opinions expressed in online reviews and depend on them to make decisions [4, 14, 39]. Without detection efforts, reviews may be replete with lies, fakes and deceptions, and thus completely useless—or worse [55].

The majority of studies in fake review detection follow three main approaches: those based on the content of the reviews, those based on the behaviour of the reviewers, or a combination of these [3, 29]. Content-based methods focus on the linguistic features of text such as words, **parts of speech (POS)**, n-gram, **term frequency (TF)** or other linguistic characteristics [21, 27, 55]. The behaviour-based approach focuses more on reviewers' behaviour, such as the user's identity, reviewed product, total number of reviews, ratings given, and duration of reviews [1, 3, 59]. This approach is straightforward, needs only a small number of features, and can deliver good results if properly designed. However, the behaviour-based approach is entirely dependent on additional information, such as metadata, provided by the system. In contrast, the content-based approach is independent of the system, requiring only the review text.

The content-based approach has two sub-categories. The first creates input features from the review text using **Bag of Words (BOW)**, Word2Vec, and skip-gram; thus, the input features for detection are words or terms created from the review text itself, and we call this approach textual-based featuring. The features extracted by this approach are more or less similar, mainly in the form of n-gram terms [14, 21, 27, 42, 63]. This textual-based approach is promising and is used in many studies to extract features from review texts. While independent of the system and needing only the text, this approach usually produces good results [11, 21, 27, 41, 42, 49, 55, 63, 72]. The second form of content-based method attempts to extract information and properties from the text, such as the length of the text, total words, and total sentences, in addition to linguistic characteristics of the text, such as POS, subjectivity, complexity, diversity and similarity, and sentiment analysis [12, 26, 28, 57, 65, 66, 71]. This type of content-based featuring is lightweight compared with textual-based featuring; for example, only 80 features in [12] compared with 5,000 features in [11]. This approach, too, is independent of the system since it requires only the text; however, its performance is usually relatively poor if not combined with other approaches [12].

In this study, we focus on the textual-based approach. For our experiments, we used Ott et al.'s [49] and Li et al.'s [41] datasets. These datasets are public fake review datasets used in many studies (e.g., see [12, 21, 27, 42, 55, 57, 72]). Before extracting the input features, we implemented text preprocessing such as tokenisation, stopword removal, detection of negation words, correction of elongation words, and POS lemmatisation. To extract the input features, we used the BOW method, n-gram words (from unigram to trigram words) and the count vectorised method. These methods convert a collection of text documents to a matrix of token counts. The token count features are in descending order by TF across the corpus (dataset). To detect fake reviews, we propose a novel **machine learning (ML)** ensemble model—the **Multi-type Classifier Ensemble (MtCE)**—that utilises several different types of standard (single) classifiers as its base classifier.

Ensemble classifiers are increasingly used in many different areas of study, such as text analysis [10, 39], computer security [30, 31], environmental science [62], medical research [69], electrical fault detection [47], and stock market prediction [34]. The main idea behind ensemble classifier models is to combine multiple single classifiers to produce a combination that outperforms any single classifier itself [8, 56]. In general, ensemble models can be categorised into four groups: bagging, boosting, stacked generalisation, and the mixture of experts [52]. Bagging, also called bootstrap aggregating, accumulates multiple predictors/classifiers and runs a plurality vote when predicting classes. These base classifiers are differentiated using a bootstrap technique to replicate

the learning set [5]. The **Random Forest (RF)** and Pasting Small Votes ensemble models are a variant of bagging [6, 7]. The second group of ensemble models is boosting. Similar to bagging, boosting is also an aggregation of multiple classifiers. However, instead of using bootstrap, it uses the boosting technique to train the base classifiers. This technique can convert weak learning algorithms to achieve arbitrarily high accuracy [60]. **Adaptive Boosting (AB)** [24] and **Gradient Boosting (GB)** [25] are two well-known algorithms that use this technique. In stacked generalisation [67], the classifiers are stacked to one another so that the output of some first-level base classifiers becomes the input of a second-level meta classifier. The combination of expert methods [32] combines the weighted output of several base classifiers in the consecutive combiner.

While promising, we see the potential for improvement from these ensemble classifiers; they typically use the same type of single classifier as the base classifier. To differentiate the output of each base classifier, they implement bootstrapping or boosting of the training sample inputs via bagging or boosting, stacking the classifiers, or applying different weights for their output. In contrast, our proposed MtCE combines several types of single classifiers working together as an ensemble model. As every single classifier has strengths and weaknesses in classification or detection, by combining different types of single classifiers in an ensemble, we use the strength of one classifier to 'cover' for the weakness of the other classifiers, and vice versa. The types of base classifiers ensembled in our MtCE can be customised by the user based on the problem and their experience and knowledge of this problem. To further increase the performance of our model, we implemented the bootstrap technique [5] and the method to vote on output (majority or priority) to produce the final output. For the base classifiers of our proposed ensemble model, we implemented several standard single classifiers that performed well in previous research on text mining [9–12, 16, 17], including the **Logistic Regression (LR)** [46], **Linear-kernel Support Vector Machine (LSVM)** [13, 15], **Multi-Layer Perceptron (MLP)** [58], and **Convolutional Neural Network (CNN)** [40]. In addition, two other classifiers that are usually applied as base classifiers in several ensemble models, namely the **Decision Tree (DT)** [53] and **Naïve Bayes (NB)** [44], were also used as base classifiers. The MtCE was compared with ensemble models widely used in the literature, including **Bagging Predictors (BP)** [5], RF, AB, and GB.

Theoretically, our work contributes to the artificial intelligence research domain, especially ML, by proposing a novel ensemble approach that can solve classification, detection and prediction problems. Most ensemble models in the literature either (1) have only one type of base classifier (such as RF, AB, BP and GB), or (2) have a small combination of fixed types and number of base classifiers that are designed to solve a particular problem [33, 55, 63, 64]. In contrast, our proposed model (the MtCE) provides the freedom to select the types and the number of base classifiers depending on one's requirements. Bootstrapping is included in the MtCE process to improve the model's stability and accuracy. Finally, a priority threshold approach is introduced, in addition to majority voting, to decide the final result; this priority threshold approach can improve the recall and overcome the imbalanced issue.

From a practical perspective, this work also contributes to addressing online commerce security problems—we have successfully implemented a model that can detect fake online consumer reviews with better results than previous research. Product reviews from buyers are commonly used as a guide by most consumers to make purchasing decisions on electronic commerce these days [9]. Reliable and effective detection of fake reviews will increase the trustworthiness of online commerce [10]. On the other hand, sellers use consumer reviews to evaluate the brand perception and consumers' satisfaction [22]. For this reason, maintaining the quality of product reviews is vital. Therefore, fake review detection is an urgent task and a high priority for online commerce portal providers.

Table 1. An Overview of Related Work (The Algorithms in Italics are Ensemble Models)

Author	Year	Featuring type	Algorithm
Sun et al. [63]	2016	Textual-based	*Bagging of 2 SVMs and PWCC*
Zhang et al. [71]	2016	Content- and behaviour-based	NB, SVM, DT, *RF*
Etaiwi and Naymat [21]	2017	Textual-based	NB, SVM, DT, *RF, GB*
Ren and Ji [55]	2017	Textual-based	SVM, GRNN, Recurrent Neural Network (RNN), CNN, *CNN-GRNN (Integrated)*
Dong et al. [19]	2018	Behaviour-based	*Neural Autoencoder Decision Forest*
Hazim et al.[26]	2018	Content and behaviour-based	*Generalised Boosted Regression Model (GBM) Gaussian, GBM Poisson, GBM Bernoulli, GB, AB*
Kumar et al. [39]	2018	Behaviour-based	LR, k-NN, NB, SVM, *RF, AB*
Zhang et al. [72]	2018	Textual-based	Recurrent CNN, SVM, CNN, *CNN-GRNN*
Barbado et al. [3]	2019	Behaviour-based	LR, DT, Gaussian NB (GNB), *RF, AB*
Martens & Maalej[45]	2019	Behaviour-based	DT, MLP, LSVM, RBF kernel SVM (RSVM), GNB, *RF*
Tang et al. [64]	2020	Behaviour-based	CNN, *CNN-bfGAN + SVM*
Alsubari et al. [2]	2020	Content-based	DT, *RF, AB*
Budhi et al. [11]	2021	Textual-based	LR, MLP, LSVM, *RF, AB, BP*
Budhi et al. [12]	2021	Content- and behaviour-based	DT, LR, LSVM, MLP, CNN, *RF, GB, AB, BP*
Elmogy et al. [20]	2021	Textual-based	LR, NB, k-NN, SVM, *RF*
Javed et al. [33]	2021	Textual-, content-, and behaviour-based	*Ensemble of 3 CNNs*
Shan et al. [61]	2021	Content- and behaviour-based	DT, SVM, NB, MLP, *RF*
Kumar et al. [38]	2022	Textual-, content-, and behaviour-based	Long short-term memory, Artificial Neural Network, RNN, RSVM, LR, k-NN, NB, *RF, GB, Light GB Method*

The rest of this paper is organised as follows. In the next section, we explain the design of our proposed model, the MtCE. After that, we discuss how we conducted experiments to test the performance of the MtCE, such as the datasets used, the testing framework, details of the textual-based approach, types of base classifiers, and the measurements. In Section 4, we discuss the results and analyses, and finally, we draw conclusions and outline the possibilities for future work.

2 RELATED WORK ON ENSEMBLE MODELS

The previous section provided a concise introduction to both fake review detection and ensemble models. This section discusses recent fake review detection studies from the literature (2016 to the present) that have proposed new ensemble models or implemented existing ensemble models for detection of fake reviews (see Table 1).

The objective of most previous studies has been to propose feature extraction approaches for the input of standard ML and **deep learning (DL)** methods, including their ensemble models. As discussed in Section 1, these feature extraction approaches can either be textual-based, which creates features from the words/terms of the review text itself [11, 20, 21, 72]; or content-based, which creates features from the text and extracts features from the text's information, properties, sentiment polarities, and characteristics [2]; or behaviour-based, which emphasises the behaviour of the reviewers rather than their reviews [3, 19, 39, 45].

The above approaches have also been combined to improve the detection results [12, 26, 38, 61, 71]. These studies investigated existing ML, DL, and ensemble models (i.e., AB, BP, RF, and GB) to detect fake reviews using features proposed via the combined approaches. The base classifiers of these ensemble models are of one type; for example, the RF utilises decision trees as its base classifiers, and GB utilises regression trees—both of them utilise and modify the trees in their

processes. Therefore, it is almost impossible to replace their tree-type base classifiers with others. While the type of the base classifier in some ensemble methods can be changed by design, the replacement must still be one type, e.g., it is impossible to mix more than one type of base classifier for AB and BP.

Some studies in the literature have also proposed novel ensemble models. For example, Sun et al. [63] proposed a bagging style of two SVMs and a CNN to detect fake reviews. They utilised special SVMs, namely $\text{BIGRAMS}_{\text{SVM}}$ and $\text{TRIGRAMS}_{\text{SVM}}$, to detect term features from the review text input, with a unique CNN model (**Product Word Composition Classifier (PWCC))** to capture product-related review features. The output of these three classifiers was processed in a bagging style method to produce the final detection result. Similarly, a forest of decision trees, called the Neural Autoencoder Decision Forest, was proposed by Dong et al. [19] to detect fake reviews. Their ensemble was designed by combining the Deep Neural Decision Tree (proposed by Kontschieder et al. [37]) with the Autoencoder method. An integration of several CNNs and **Gated Recurrent Neural Networks (GRNNs)** was introduced by Ren and Ji [55] in 2017. The CNNs were implemented to produce continuous sentence vectors from words, then handled by several forward and backward GRNNs to produce document representations of the reviews. A unique model of CNN, namely bfGAN, was proposed by Tang et al. [64] in 2020, and combined with the SVM for fake review detection. They showed that their model can effectively detect cold-start spam/fake reviews; the cold start problem is when a new reviewer posts a review. Javed et al. [33] proposed an ensemble of three classifiers (CNNs) for fake review detection. Each classifier detects different feature groups: a textual-based, a content-based (non-textual), and a behaviour-based group of features; a voting mechanism is then used to produce the final detection. The above-discussed approaches are similar in terms of one important aspect: they have proposed a specific ensemble model with specific types and numbers of base classifiers. In these ensemble models, the types and numbers of their base classifiers cannot be modified, since each base classifier has a specific task and purpose.

Our proposed ensemble model, the MtCE, is different. In the MtCE, diverse base classifiers can be mixed together so that the strength of one type of base classifier can overcome the weaknesses of other types of base classifiers and vice versa. The total number of base classifiers and the number of each type of base classifier can also be configured freely based on specific requirements. Our model also applies the bootstrapping technique to ensure stability and improve accuracy. For the final classification result, we introduce a priority threshold technique that prioritises a particular class in conjunction with common majority voting that is usually applied by other ensembles. This priority threshold technique can overcome the imbalanced problem when applied to the scarce/rare class as well as improve the recall in binary classification if applied to the main (positive) class.

3 THE MTCE

The MtCE is a novel ensemble of classifiers developed in light of the fact that every single classifier has its strengths and weaknesses (see Figure 1). This ensemble is proposed based on the following idea:

> *Every single classifier has been created with a different method, formulation and purpose, and thus, has strengths and weaknesses in different spots. For example, (1) for the same dataset, each classifier may correctly classify the different set of records; (2) a classifier is usually created to solve one or two problems, and not tested for the case of other problems; and (3) some classifiers perform well for smaller datasets but not for bigger ones, or vice versa. Therefore, by combining different single classifiers in one ensemble, the strengths of one classifier may 'cover' for the weaknesses of another.*

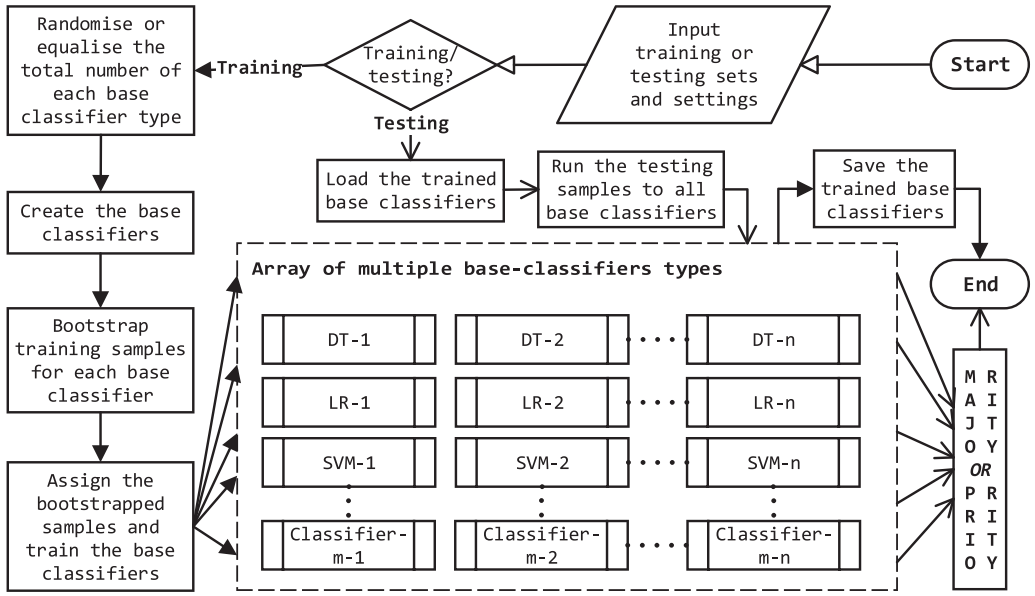


Fig. 1. Design of the MtCE.

In Figure 1, we depict two processes: the training process (the full head arrows ➔) and the testing process (the line head arrows →). All processes begin with inputting the training or testing sets and the parameter settings of the MtCE (see hollow head arrows -▷).

3.1 Base Classifiers of the MtCE

Base classifiers here include several classifier objects that are utilised within an ensemble model. These classifiers will be run jointly in a specific way (depending on the ensemble design) to produce results. These results are then processed to produce the final result of the ensemble model. For our MtCE, first the user will decide on the types of base classifiers and their total numbers. After that, for the training process, the user determines if an equal number of base classifiers will be created for each setting of each type, or if the numbers will be chosen randomly. For example, suppose the total base classifiers are 10 of three types (LR, DT, NB); an equally split setting will create 4 LR, 3 DT, and 3 NB, while for a random setting, each classifier type will be randomised from 1 to 8 (total classifiers − (total types − 1)) classifiers. While splitting the number of base classifier types equally is more sensible if the user does not know the exact nature of the problem at hand, randomising them sometimes could give better results. Moreover, with a simple modification in the implementation phase, the user can also set the exact total number of each base classifier type, if required. In this study, we did not test the effect of setting exact numbers of each type of base classifier since, for our problem, it will become similar to the randomised setting.

3.2 Bootstrap Sampling

After creating the base classifiers, each base classifier is assigned a subset of training samples based on the bootstrap sampling process. Bootstrapping the samples could improve the stability and accuracy of the model [5]. The bootstrap sampling method draws sample data repeatedly with replacement from the source (data), based on the percentage setting [5, 35]. After the training

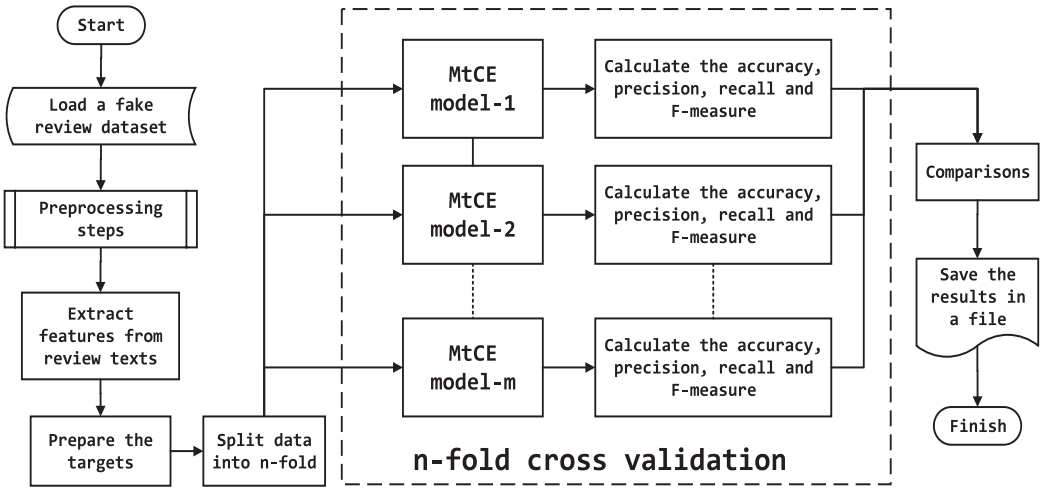


Fig. 2. Framework to test and compare the MtCE.

process for each base classifier finishes, the weights and parameters of all base classifiers are saved in a file.

3.3 Determining the Final Output

The testing process is straightforward. After inputting the testing samples and loading the previously trained base classifiers, all samples are run with the base classifiers. The final output will be determined by one of two processes: the majority vote or the priority class threshold. The majority vote chooses the majority class outputs from the results of the base classifiers. If the majority votes of all classes are equal, the final result is the smallest class in the corpus. The priority class threshold provides a final result based on the priority class chosen in the setting; that is, if the positive class is chosen as the priority, then, if the positive class outputs from base classifiers are the same as or more than the threshold, the final output is a positive class. The priority class threshold can range from absolute, which needs only one base classifier to pick the chosen class, to the maximum threshold before it becomes a majority vote (i.e., more than 50% + 1 in a binary classification problem). This priority class threshold mechanism will focus on the class that is prioritised and will increase the detection performance of this class.

4 TESTING THE PERFORMANCE OF THE MTCE

4.1 Framework for Testing

To evaluate our proposed ensemble, the MtCE, we implemented the comparison framework that we used previously to investigate classifiers for sentiment polarity detection of consumer reviews [10]. In this study, we modified the framework so that it could be applied to test the MtCE for fake review detection (see Figure 2). We used this framework to test different settings of the MtCE using similar datasets and comparing their performances. We ran the same test using several single classifiers as base classifiers of the MtCE and several well-known ensembles for comparison.

As can be seen in Figure 2, after a review text is loaded to a string, it is processed in the preprocessing subroutine to remove unnecessary words and corrections. Features are then extracted from the texts using a textual-based approach, as discussed in Section 1. After combining with the designated targets, these feature-target vectors are split into 10 folds for the **cross-validation (CV)** process. This same 10-fold setting is then trained and tested on several different

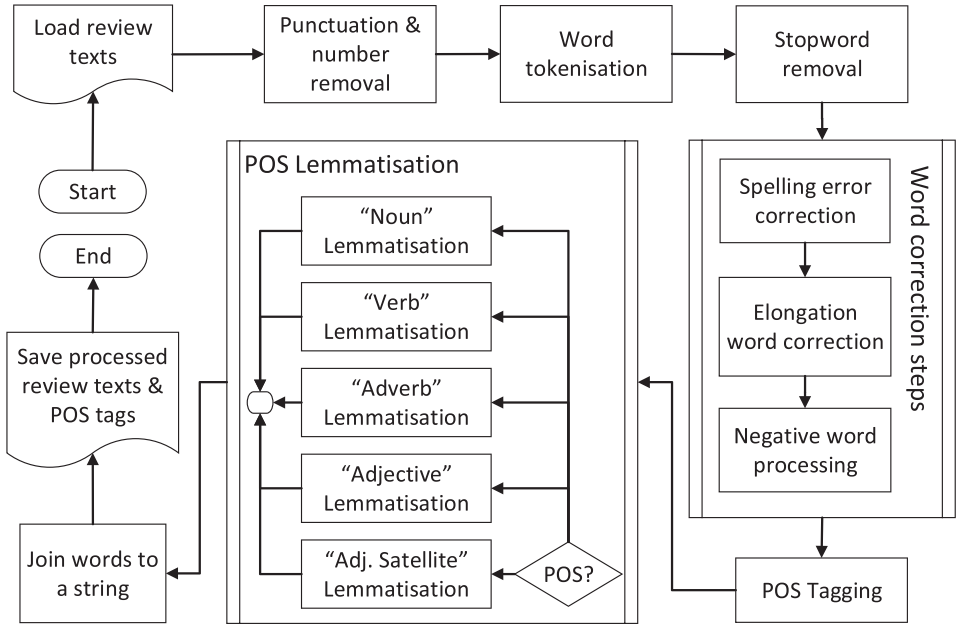


Fig. 3. Preprocessing steps [11].

combinations of MtCE models. Afterwards, the measurement results of these different MtCE models are compared to determine the best setting of the MtCE for fake review detection.

In the preprocessing steps (see Figure 3), we implemented several methods as follows:

(a) Removal of punctuation, numbers and stopwords.
(b) Correction of spelling errors, elongation words and negative words.
(c) POS tagging and lemmatisation.

Punctuation and number removal is a necessary step for a textual-based approach, since the input features of this approach are the words. Therefore, we needed to remove the non-word parts of the text reviews. Stopword is a term for words commonly used in English sentences such as 'the', 'is', 'at', 'which' and 'on'. These words can be removed from the text before it is used as a training record. These kinds of words may mislead detectors in recognising the trait words of a class, because they are often the most common words in a corpus.

Misspelled words create unnecessary issues for detection since the detector will recognise them as different words from their correct forms. Like misspelled words, elongation words such as 'yesss', 'fiiine' and 'yoouu' can also increase the diversity of words in the training example and make the classifiers harder to train. We utilised Peter Norvig's code for spelling correction [48] to correct misspelled and elongation words. Negative words have many forms, depending on the grammar, but their purpose is similar—to change a positive sentence to be negative. Therefore, we shifted all negative words to their primary form to reduce the diversity of words.

POS tagging categorises words by their syntactic function, such as noun, pronoun, adjective, verb, and adverb. This step is essential because it puts the words in their context [21] so that in the lemmatisation process, we can lemmatise the word to the correct context. Lemmatisation is a method for changing the word back to its basic form; POS lemmatisation returns the chosen word to its basic syntactic function. This step reduces the diversity of words inside the dataset and makes it easier to recognise.

Table 2. Statistics for Several Public Datasets

Dataset Author	Domain	Total Records	Fake		Genuine	
			Total	%	Total	%
Ott et al. [49]	Hotel (Various)	1600	800	50.00	800	50.00
Li et al. [41]	Doctor (Various)	558	357	63.98	201	36.02
	Hotel (Various)	1880	1080	57.45	800	42.55
	Restaurant (Various)	402	202	50.25	200	49.75

Because we extracted the input features directly from social media messages, we used the BOW feature extraction method commonly used for textual-based features [18]. This method works by decomposing the entire text into a group of singular words. Similar to previous work [11], in this study, we used it in combination with the n-gram technique to capture singular words and terms that convey one meaning but are composed of multiple words. For terms, the BOW checks for the existence of a contiguous sequence of n words from the given text sample. This study limited this checking to trigrams since sequences of more than three words are rare in real-world texts. After decomposing the text, the terms were sorted based on their frequency across the corpus.

4.2 Base Classifiers and Ensemble Classifiers for Experimental Purposes

As mentioned in Section 2, our proposed ensemble can apply multiple types of single classifiers. While in theory, any single classifier could be used as the base classifier of the MtCE, to test its performance, we investigated several combinations of ML and DL that performed well in previous research on text mining [9–12, 16, 17], which include the LR, LSVM, MLP, and CNN models. We also investigated two other classifiers, DT and NB, since these two models are commonly used as default base classifiers in some ensemble models, such as the RF [7], BP [5], and AB [24]. For comparison purposes, we tested several ensemble models, including the BP, RF, AB, and GB [25].

We built all ML classifiers and ensembles, except the CNN, using scikit-learn components [51]; the CNN was built with Keras [36] wrapped with scikit-learn components (scikit-learn does not provide a CNN component but a way to wrap DL components of Keras to be used with or within scikit-learn). To ensure the results could only be affected by our model implementation and not by modifying classifier parameters, we used the default parameters for all single classifiers used as base classifiers and the ensemble models.

4.3 Datasets

Intuitively, our proposed ensemble classifier can be applied to a wide range of problems, similar to other ML ensemble classifiers. However, to test the performance of the MtCE, we conducted experiments using four public fake review datasets (see Table 2 for additional details). The public datasets from Ott et al. [49] and Li et al. [41] were created using domain experts, such as Amazon's Mechanical Turk service (Turkers) and hotel employees, to provide false/fake reviews for some online services. Li et al.'s hotel dataset is an extension of Ott et al.'s dataset.

4.4 Cross-validation and Measurements

We evaluated the performance of the MtCE using n-fold CV, which is widely applied to evaluate the generalisation performance of an algorithm [30], to reduce bias between the dataset and the training or testing set [54] and avoid overfitting [70]. We implemented scikit-learn [51] for performance measurements of our experiments. These measurements and their respective formulas can be found in Table 3.

Table 3. Measurement Functions and Formulas

No	Name	Sklearn Function	Equation
1	Accuracy	accuracy_score()	$A(y,\hat{y}) = \frac{1}{n_{samples}} \sum_{k=0}^{n_{samples}-1} 1(\widehat{y_k} = y_k)$ where y is the set of predicted pairs, $\hat{y}$ is the set of true pairs and $n_{samples}$ is the total number of samples.
2	Precision	precision_score()	$P(y_k, \widehat{y_k}) = \frac{tp}{tp+fp}$ where k is the set of classes, y_k is the subset of y with class k, tp is true positive, and fp is false positive.
3	Recall	recall_score()	$R(y_k, \widehat{y_k}) = \frac{tp}{tp+fn}$ where fn is false negative.
4	F-measure/F1	f1_score()	$F_1(y_k, \widehat{y_k}) = 2*\frac{P(y_k,\widehat{y_k})*R(y_k,\widehat{y_k})}{P(y_k,\widehat{y_k})+R(y_k,\widehat{y_k})}$

5 RESULTS AND DISCUSSIONS

5.1 Experimenting with the MtCE for Fake Review Detection

The first group of experiments aimed to test the performance of the MtCE for fake review detection. For its base classifiers, we implemented the CNN, LR, LSVM, and MLP, which performed well in our previous studies [9–12]. We also implemented the DT and NB, which are used as default base classifiers in many ensembles (e.g., RF, BP, and AB). The parameters of these base classifiers are the defaults of the scikit-learn components used to implement them [51]. As shown in the Appendix, we tested all possibilities from 2-, 3-, 4-, to 5-combination and presented 11 selected combinations in Table 4. Besides combining base classifiers, the other settings were fixed: total base classifiers = 15, shared equally for each type; bootstrap = 0.5; and final output = majority vote. The datasets we used in the experiments were Ott et al.'s dataset [49] and all three of Li et al.'s datasets [41]. All experiments were conducted using the 10-fold CV method. For the detailed comparison, we also tested the datasets using single classifiers as above and several well-known ensemble classifiers (RF, BP, AB, and GB). Results of the single and ensemble classifiers are provided in Table 5.

Comparing results in Table 4 with Table 5, some combinations of base classifiers in the MtCE exceed the performance of all of the base classifiers (see the bold-green scores in Table 4). For example, the accuracy of MtCE(LR-MLP) for Ott et al.'s = 88%; in comparison, in Table 5, the accuracy values of LR and MLP are 87.13% and 87.56%, respectively. However, not all combinations improved and supported each other as we hoped. A few weak classifiers degraded the performance of stronger classifiers when combined with them. This gave rise to the result that the performance of the MtCE was the average of the performance of its base classifiers; for example, the LSVM accuracy for Ott et al.'s = 85.88%, while MLP = 87.56%, and the combination of both in MtCE(LSVM-MLP) = 87.50%. This implies that the LSVM degraded the performance of MLP, and given this, rather than implementing our proposed ensemble, it would be better to use the MLP directly. However, in the same case, MtCE(LSVM-MLP) for Ott et al.'s improved the recall, from 86.21% (LSVM) and 88.37% (MLP) to 90.02%. In such cases, using our ensemble would be acceptable since the recall in binary classification/detection is the same as the accuracy of the positive class or the main class (in our problem, the fake review class). Thus, high recall means the ability of the ensemble to detect positive classes is also high.

Another finding from this set of experiments is that different datasets might need different base classifier combinations. For example, MtCE(MLP-NB) that performed best for Ott et al.'s dataset performed worst for Li et al.'s doctor dataset (increasing the recall but decreasing all other

Table 4. Results of the Combination of Base Classifiers for the MtCE Model (Selected)

Num[1]	MtCE Base Classifier Combination	Ott et al.'s Dataset[2]				Li et al.'s Dataset (Doctor)[2]			
		Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
2	LR-MLP	**88.00**	*86.85*	**89.44**	**88.08**	84.41	*84.23*	**92.97**	88.33
5	LSVM-MLP	87.50	85.72	**90.02**	**87.74**	**86.56**	**87.05**	**93.12**	**89.88**
7	LSVM-NB	**89.19**	**88.32**	**90.43**	**89.31**	*84.75*	86.85	**90.32**	88.19
9	MLP-NB	**89.63**	**88.52**	**91.22**	**89.80**	*85.31*	*84.00*	**95.30**	*89.12*
18	LR-LSVM-NB	**89.00**	**88.68**	**89.59**	**89.07**	84.58	85.70	**91.44**	88.34
22	LSVM-MLP-NB	**89.13**	87.86	**91.00**	**89.31**	85.13	86.39	91.88	88.79
26	CNN-LR-MLP	**88.19**	**87.32**	**89.33**	**88.27**	84.40	83.75	**93.51**	88.29
35	LR-LSVM-MLP-NB	**89.19**	**88.77**	**89.95**	**89.28**	85.66	87.01	91.38	88.94
37	LR-MLP-DT-NB	**88.75**	**88.41**	89.25	**88.79**	84.06	83.85	**93.06**	88.04
47	LR-LSVM-MLP-DT-NB	**88.94**	**88.38**	**89.56**	**88.94**	84.24	84.89	92.00	88.12
50	CNN-LR-LSVM-MLP-NB	**88.88**	**88.21**	**89.83**	**88.95**	86.21	86.45	**93.18**	89.54
		Li et al.'s Dataset (Hotel)[2]				Li et al.'s Dataset (Restaurant)[2]			
2	LR-MLP	**87.34**	85.09	**94.30**	**89.45**	85.11	83.99	**88.36**	85.69
5	LSVM-MLP	86.12	84.71	**92.67**	**88.44**	**85.83**	83.98	**88.55**	**86.02**
7	LSVM-NB	**87.45**	87.09	**91.75**	**89.34**	85.35	84.51	86.33	85.21
9	MLP-NB	87.23	86.69	**91.92**	**89.17**	86.04	85.36	87.18	**86.15**
18	LR-LSVM-NB	85.96	85.20	**91.34**	88.14	**86.58**	85.71	**88.08**	**86.41**
22	LSVM-MLP-NB	86.97	86.09	**92.34**	**89.05**	85.32	83.68	**88.65**	85.75
26	CNN-LR-MLP	85.85	84.79	**91.96**	88.17	83.34	82.42	84.09	82.88
35	LR-LSVM-MLP-NB	86.70	85.12	**92.92**	88.78	82.32	81.51	84.77	82.80
37	LR-MLP-DT-NB	**87.82**	86.67	**93.26**	**89.81**	84.38	83.61	85.67	84.57
47	LR-LSVM-MLP-DT-NB	**87.55**	86.19	**93.28**	**89.58**	84.63	83.29	87.36	85.06
50	CNN-LR-LSVM-MLP-NB	86.76	85.98	**91.98**	**88.81**	83.80	82.59	86.17	83.98

[1]The number here is associated with the number in the Appendix.

[2]**Bold-green** = the MtCE result is higher than that of all base classifiers in Table 5; *Italic-red* = the MtCE result is higher than all base classifiers; Normal-black = at least one base classifier result is higher than that of the MtCE.

measurements). The best combination for Li et al.'s doctor dataset was MtCE(LSVM-MLP); for Li et al.'s hotel dataset, it was MtCE(LR-MLP-DT-NB); and for Li et al.'s restaurant dataset, it was MtCE(LR-LSVM-NB). A closer inspection of Table 5 shows that the base classifiers of MtCE's best combination for a particular dataset mostly performed well on the same dataset too, when they were used in standalone modes. For example, on Ott's MtCE(MLP-NB), the MLP and NB also performed well on Ott's dataset, although they were not as good as when combined in the MTCE. This observation is valid on other combinations for other datasets, except the DT on Li et al.'s hotel dataset, which was not performing well by itself but performed best in combination with other base classifiers in the MtCE. This indicates that, while combining good classifiers could produce better results, integrating a weak classifier sometimes could also boost the results of MtCE.

We found that the MtCE performed better than other ensembles of homogenous classifiers, as indicated by comparing the results in Table 5 (numbers 7–10) to the MtCE results in Table 4. These facts suggest that, when implemented correctly, the combination of multi-type classifiers could compensate each other's weaknesses and outperform the combination of a homogenous type of ensemble classifiers such as the RF, BP, AB, or GB. However, for the cases of RF, BP, and AB, we suspect it is also because these ensembles use the DT as their base classifiers/predictors. DT

Table 5. Results of Single and Ensemble Classifiers

Num.	Classifiers	Ott et al.'s Dataset				Li et al.'s Dataset (Doctor)			
		Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
1	CNN	79.63	81.48	77.36	79.13	67.76	70.64	88.40	77.02
2	LR	87.13	87.05	87.25	87.11	83.35	85.48	89.71	87.26
3	LSVM	85.88	85.67	86.21	85.89	83.13	86.28	87.80	86.91
4	MLP	87.56	87.05	88.37	87.66	85.67	85.89	92.94	89.16
5	DT	71.50	72.21	69.71	70.83	65.79	72.98	74.01	73.32
6	NB	88.50	87.97	89.45	88.63	87.12	90.68	88.98	89.68
7	RF	87.00	87.44	86.62	86.92	76.35	74.05	97.24	83.98
8	BP	75.00	74.23	76.79	75.41	74.19	74.50	90.21	81.45
9	AB	80.06	79.64	80.82	80.09	74.93	79.56	81.28	80.16
10	GB	82.81	83.67	81.63	82.56	76.52	77.08	90.63	82.99
		Li et al.'s Dataset (Hotel)				Li et al.'s Dataset (Restaurant)			
1	CNN	78.83	82.48	80.63	81.29	71.63	71.66	73.20	71.75
2	LR	85.59	85.50	90.25	87.77	81.59	81.03	84.52	81.74
3	LSVM	84.20	84.59	88.26	86.33	82.11	79.93	84.40	81.53
4	MLP	86.12	85.89	90.82	88.24	85.56	84.85	87.69	85.73
5	DT	68.88	73.81	71.15	72.38	60.24	60.34	63.27	61.11
6	NB	87.29	89.73	87.92	88.78	86.08	86.36	86.63	86.10
7	RF	81.97	80.78	90.28	85.17	81.34	78.31	87.86	81.88
8	BP	75.96	76.35	84.14	80.03	71.65	71.08	74.51	71.95
9	AB	79.89	80.77	84.87	82.67	70.89	70.64	70.35	70.38
10	GB	80.48	78.83	89.94	84.00	76.12	77.43	75.46	75.78

performance was not good on all datasets, which affected the performance of its ensemble variants.

While the deep learning-based CNN as a standalone classifier did not perform as well as other single classifiers, somewhat surprisingly, it performed exceptionally well when combined with other classifiers in the MtCE. This increase in accuracy was significant; for example, MtCE(CNN-LR-LSVM-MLP-NB) it is 7.93% for Li et al.'s hotel dataset and 18.45% for Li et al.'s doctor dataset. The improvement in recall was also good, with a minimum of 4.78% for Li et al.'s doctor dataset to 12.97% for Li et al.'s restaurant dataset. Therefore, the deep learning-based CNN could be combined perfectly with other single ML classifiers as base classifiers for the MtCE; it gave good results and improved all other base classifiers' performances as well. This is another strong indication that combining different classifiers with their strengths and weaknesses in our proposed MtCE can improve the overall performance. It is because the weaknesses of one base classifier can be covered by the strengths of other base classifiers (i.e., ensemble diversity).

5.2 Effects of MtCE Parameters

In this section, we discuss several input parameters of the MtCE that may affect the performance of this proposed model. All of the settings are the same as in Section 5.1 except the parameter we are testing. For these experiments, we chose five base classifier combinations of the MtCE in Table 4—that is, MtCE(LSVM-MLP), MtCE(LSVM-NB), MtCE(LR-LSVM-NB), MtCE(LR-MLP-DT-NB), and MtCE(LR-LSVM-MLP-DT-NB)—as these are the five combinations of base classifier types that have performed the best across all four datasets. We ran all experiments for parameter testing using Ott

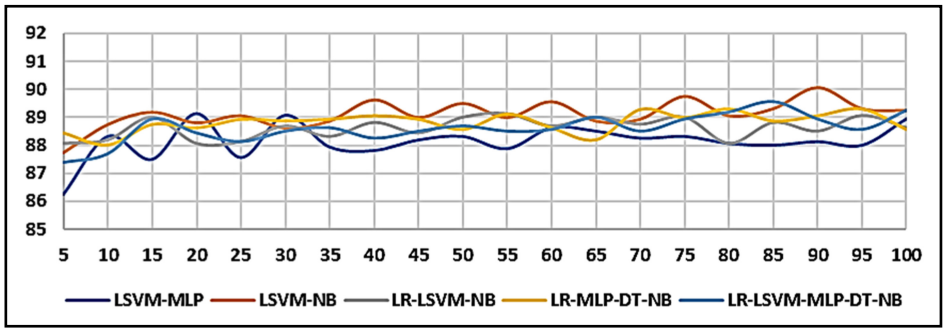


Fig. 4. The accuracy (y-axis, in %) of the MtCE with 5–100 total base classifiers (x-axis).

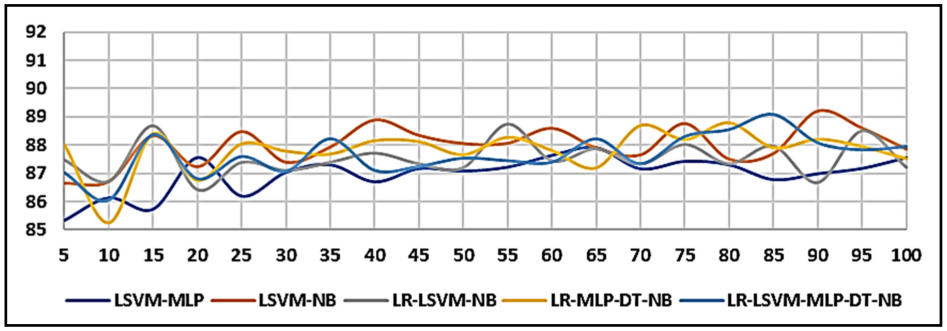


Fig. 5. The precision (y-axis, in %) of the MtCE with 5–100 total base classifiers (x-axis).

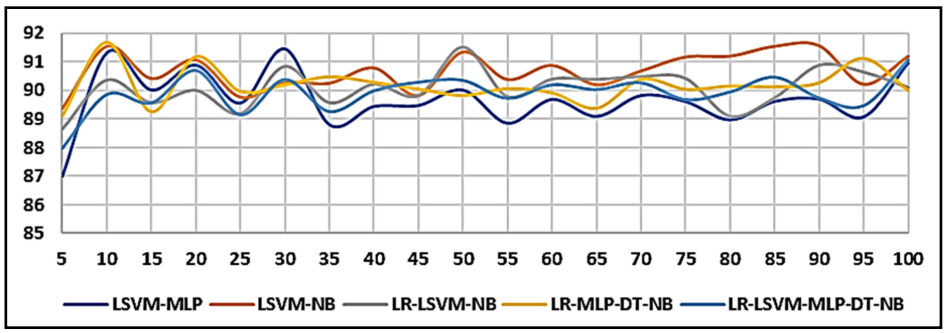


Fig. 6. The recall (y-axis, in %) of the MtCE with 5–100 total base classifiers (x-axis).

et al.'s dataset only, since these five MtCE combination settings performed well in this dataset (see the bold-green scores in Table 4). However, after finding the ideal set of parameters, we ran them with the other datasets and compared the results with our previous approach implemented on the same datasets.

5.2.1 Total Number of Base Classifiers. First, we investigated the effect of base classifiers' total numbers. Here, we ran experiments on all five combinations with the total number of base classifiers varying from 5 to 100. Results can be seen in Figure 4 for accuracy, Figure 5 for precision, Figure 6 for recall, and Figure 7 for F-measure. From these figures, we can see that accuracy, precision, recall, and F-measure increased at first, but after 20 classifiers, all scores fluctuated around a number ± 1.5%.

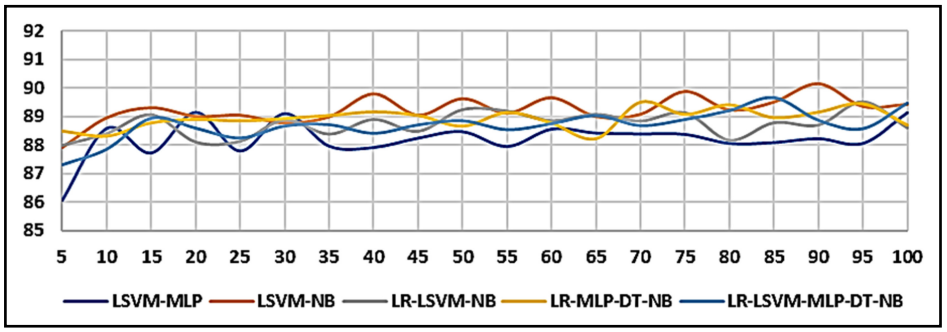


Fig. 7. The F-measure (y-axis, in %) of MtCE with 5–100 total base classifiers (x-axis).

Table 6. Results of Equal Split vs Randomly Assigned-Type of MtCE Base Classifiers on 10 Runs of the 10-fold CV Process

Base Classifier Combination	Equally Split or Randomly Assigned	Accuracy		Precision		Recall		F-measure	
		Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev
LSVM-MLP	Equal	88.21	0.38	86.76	0.35	90.23	0.56	88.39	0.41
	Random	87.82	0.60	86.41	0.59	89.81	0.69	88.00	0.60
LSVM- NB	Equal	89.44	0.36	88.20	0.50	91.12	0.39	89.57	0.37
	Random	89.37	0.19	88.16	0.38	90.93	0.35	89.47	0.22
LR-LSVM-NB	Equal	88.97	0.35	88.14	0.48	90.06	0.42	89.04	0.35
	Random	88.72	0.37	87.79	0.41	89.89	0.49	88.78	0.38
LR-MLP-DT-NB	Equal	89.18	0.32	88.34	0.44	90.24	0.39	89.21	0.29
	Random	89.03	0.35	88.19	0.46	90.12	0.38	89.09	0.35
LR-LSVM-MLP-DT-NB	Equal	88.28	0.66	87.37	0.64	89.59	0.86	88.41	0.67
	Random	88.12	0.37	87.10	0.39	89.49	0.49	88.21	0.39

Accuracy fluctuated around 88.5%, precision around 88%, and recall around 90%, except for MtCE(LSVM-MLP), which was 1% lower.

Intuitively, these fluctuations after 20 classifiers mean that adding more classifiers would not improve the measurements a lot. However, since the case is only for fake review detection on Ott et al.'s dataset, we will leave it to the user decide on the number of base classifiers needed for their problem. Next, since we think a 1.5% difference is quite significant, we used the best accuracy results for each MtCE combination for the next batch of experiments. These combinations are 20 classifiers for MtCE(LSVM-MLP), 55 classifiers for MtCE(LR-LSVM-NB), 80 classifiers for MtCE(LR-MLP-DT-NB), and 85 classifiers for MtCE(LR-LSVM-MLP-DT-NB).

5.2.2 Equal Split or Randomly Assigned Base Classifier Type. To investigate the equally split vs randomly assigned base classifier type, we ran the experiments 10 times for each equally split and randomly assigned setting for each MtCE combination. The average results and their standard deviations can be seen in Table 6. This table shows that the equally split option gave slightly better average results for all measurements relative to the random option. This implies that when the total of each base classifier is equal, they support each other, and the strength of one type of classifier covers the weakness of the other type when combined in the MtCE. On the other hand, the random option could make one or two base classifiers more numerous than the others, and

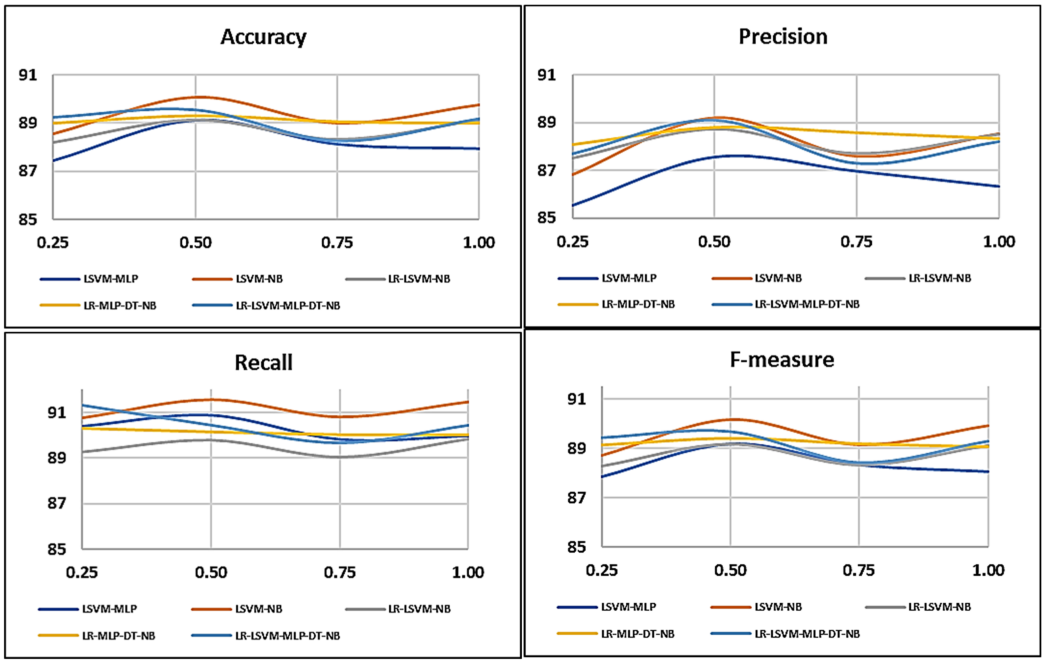


Fig. 8. The accuracy, precision, recall, and F-measure (y-axis, in %) of the MtCE with bootstrap settings from 0.25 to 1.00 (x-axis).

they dominated the detection process. The standard deviation of all measurement scores below 1 means the variation of the results from 10 runs was low and close to each other. Furthermore, we can see that the standard deviation of 2-combination, the MtCE(LSVM-MLP), on the equally split setting was lower than the random assignment. This is expected since, in an equally split setting, each base classifier total is the same for all 10 runs, while in the random option, this varies. However, the exact opposite happened in 5-combination, the MtCE(LR-LSVM-MLP-DT-NB). Here, the standard deviation of the equally split setting was higher than the random option, implying that the randomness might make the base classifiers more homogenous than when split equally between its five base classifier types. For the 3- and 4-combinations, the standard deviation scores are similar.

5.2.3 Bootstrap Effect. The subsequent experiments aimed to investigate the bootstrap parameter. We used fixed parameters as above, except the bootstrap setting ranged from 0.25 to 1 (no bootstrapping). As can be seen in Figure 8, the best accuracy, precision, and recall were often achieved when the bootstrap setting was 0.5, where every base classifier was trained using 50% of training samples randomly. This indicates that bootstrap setting 0.5 is ideal for the tested MtCE settings, i.e., the five best base-classifier combinations. Therefore, intuitively, it can be used as the default bootstrap setting of MtCE. Based on these findings, for the next set of experiments, we set the bootstrap to be 0.5.

5.2.4 Final Output: Majority Vote or Class Priority Threshold. The last parameter to test is how the output of base classifiers should be processed. There are two options in terms of transferring the output of base classifiers to the final output: majority vote and priority class. To test the effect of priority, we set the priority to the positive class (fake review class), in the range between absolute priority and 45% priority. Here, absolute priority means the final result will be recorded as

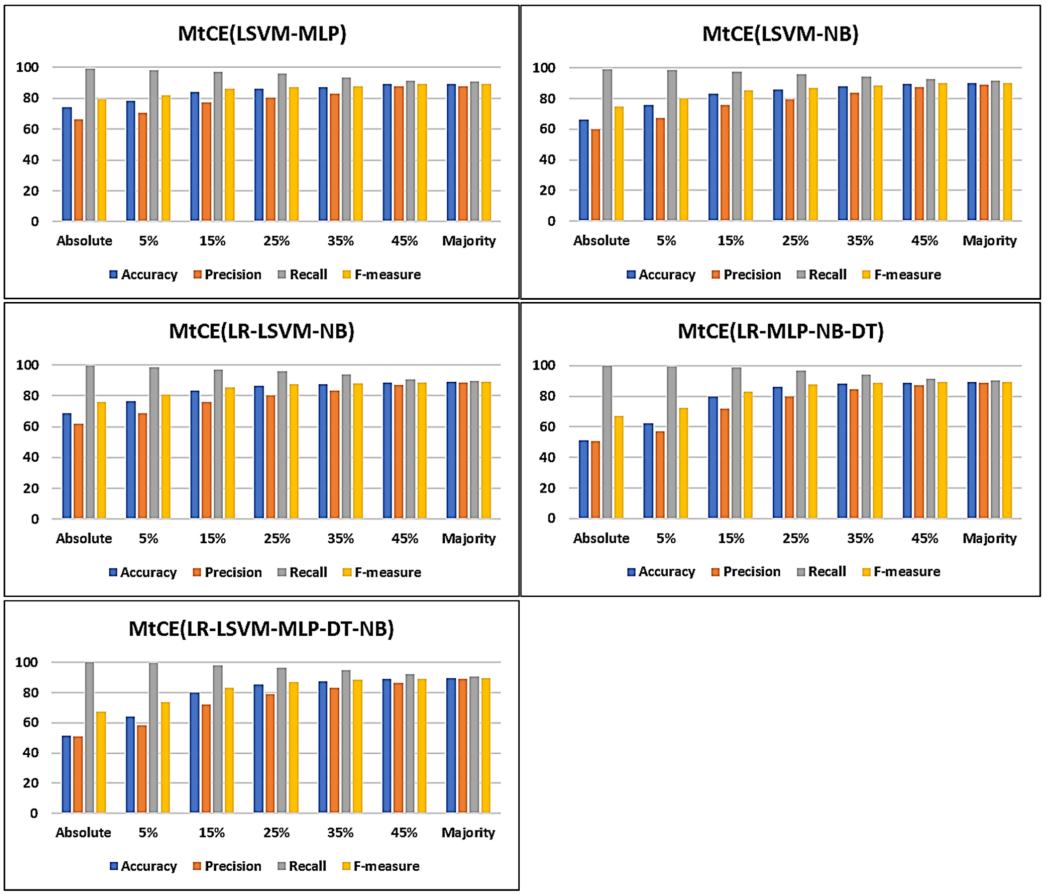


Fig. 9. Results of final target experiments, in terms of accuracy, precision, recall, and F-measure (y-axis, in %), from absolute priority to 45% priority, and majority (x-axis).

positive/fake if one or more of the base classifiers records this. Percentage priority means the final result will be recorded as positive/fake if the number of base classifiers with this result is the same or more than the percentage threshold. The results of the experiments can be seen in Figure 9.

As per Figure 9, the priority setting increases the recall but decreases other measurements. Moreover, as noted, the higher the recall, the more accurate the positive class (fake review class) detection. Thus, we can conclude that the priority setting is helpful when samples of positive class are scarce or when the focus is on anomaly detection. The key is how high we choose the percentage of priority so that the increase in recall does not greatly decrease the other measurements. For example, in Figure 9, for 25% priority of MtCE(LSVM-MLP), where recall increased by 5.05%, the accuracy, precision, and F-measure decreased by 2.75%, 7.15%, and 1.70%, respectively.

5.3 Comparison with other Studies on the Same Datasets

As commonly presented in related studies [42, 55, 57, 72], in this section, we present the MtCE's and other models' best results in light of various considerations. Many factors can influence results, such as the measurement formulas or tools, datasets used, number of records involved in experiments, and how the experiments were conducted (e.g., CV vs traditional train-test-validation

Table 7. Best Performance of the MtCE and other Studies for the Same Datasets

No.	Dataset	Description	Acc (%)	Pre (%)	Rec (%)	F1 (%)
1	Hotel (various) [49]	Ott et al. [49], SVM on positive sentiment subset	88.4	89.1	87.5	88.3
		Fusilier et al. [27], PU-L modified	-	85.2	72.8	78
		Rout et al. [57], DT	92.11	-	-	-
		Etaiwi and Naymat [21], SVM, all preproc. steps	85	51	86	-
		Zhang et al. [72], DRI-RCNN, 0.9 T.P.	-	-	-	86.59
		Budhi et al. [12], GB, over-sampling	70.62	67.58	81.29	73.8
		*MtCE(LSVM-NB, 90, Equal, 0.5, Majority)**	*90.06*	***89.19***	***91.56***	***90.16***
2	Hotel (various) [41]	Li et al. [41], OvR SAGE-Unigram	81.8	81.2	84	-
		Ren and Ji [55], Integrated, Employee/Turker subset	92.6	-	-	90.1
		Li et al. [42], SWNN	-	84.1	87	85
		Budhi et al. [12], GB, under-sampling	71.29	73.61	82.79	77.93
		*MtCE(LR-MLP-DT-NB, 15, Equal, 0.5, Majority)**	*87.82*	***86.67***	***93.26***	*89.81*
3	Restaurant (various) [41]	Li et al. [41], OvR SAGE-Unigram	81.7	84.2	81.6	-
		Ren and Ji [55], Integrated, Customer/Turker subset	86.9	-	-	86.8
		Li et al. [42], SWNN		83.3	88.2	81
		Budhi et al. [12], GB, over-sampling	72.44	71.23	75.16	73.14
		*MtCE(LR-MLP-DT-NB, 80, Equal, 0.5, Majority)**	*86.07*	***84.47***	***88.89***	*86.37*
4	Doctor (various) [41]	Li et al. [41], OvR SAGE-Unigram	74.5	77.2	70.1	-
		Ren and Ji [55], Integrated, Customer/Turker subset	76	-	-	74.1
		Li et al. [42], SWNN	-	83.7	87.6	82.9
		Budhi et al. [12], GB, over-sampling	73.35	79.44	86.94	83.02
		*MtCE(LSVM-MLP, 15, Equal, 0.5, Majority)**	***86.56***	***87.05***	***93.12***	***89.88***

*MtCE(<type of base classifiers>, <total base classifiers>, <equally split or random assign of base classifiers>, <bootstrap percentage>, <majority or priority output>).

split). Therefore, the results presented in Table 7 should be read with the following considerations in mind:

(1) We present the MtCE's and other studies' results based on the same datasets (see Table 2). We used all samples that could be processed from each dataset. Note that some studies used only a subset of the dataset or split the dataset into subsets and measured each subset differently. All of our experiments were conducted using 10-fold CV.
(2) It is impossible to ensure that all the studies used the same formulas to measure accuracy, precision, recall, and F-measure. Therefore, MtCE results were measured using widely used formulas for binary target prediction (see Table 3). All measurements were in the 'macro' average setting using scikit-learn measurement components [51].
(3) The original results from the dataset creators were used as the baseline (see the underlined description and scores in Table 7. We present only the best result for each dataset from each study.

We do not claim that the MtCE is better or worse than methods in other studies, since several factors can affect results. However, we can confidently compare the MtCE to our previous study in fake review detection [12] because the measurement formulas are similar. As is evident in Table 7, compared with our previous approach, the MtCE is superior. Accuracy improvement ranges from a minimum of 13.2% in Li et al.'s doctor dataset to a maximum of 19.4% in Ott et al.'s dataset; precision improves from 7.6% to 21.6% and recall from 6.2% to 13.7%.

6 CONCLUSION AND FUTURE WORK

From the experiments above, we can conclude that our proposed ensemble, the MtCE, was able to adequately detect fake or fraudulent reviews, which have become an acute problem on e-commerce platforms. Compared with our previous approach, the MtCE improved accuracy, precision and recall by up to 19.4%, 21.6% and 13.7%, respectively.

Not all combinations of base classifier types produced the expected result of an improvement in the performance of all base classifiers of the MtCE. In some combinations, weak classifiers dragged down the performance of stronger classifiers, especially in terms of accuracy measurements. However, when the combinations were correct, the MtCE produced better results than its base classifiers in standalone modes. It is important to note that while accuracy was not the highest, recall was the highest in many cases. This is a positive result since, in binary classification, recall is the same as accuracy of a positive case. In this case, the MtCE was able to better detect the fake review class. Comparison with well-known ensembles, such as the RF, AB, BP, and GB, showed that a correct combination of the MtCE could outperform other ensembles, proving that combining multi-type classifiers in one ensemble process performed better than combining the same classifiers, as is usually done in other ensemble methods.

We proved that DL methods such as the CNN model could be combined with ML counterparts as base classifiers to give better results than if used alone. The number of base classifiers affected performance detection; however, accuracy and other measurements oscillated after 20 base classifiers. While reducing the amount of data for the training process, the bootstrap setting could also affect the performance of the MtCE. From the experiments, we found that 50% bootstrapping gives better results than other values, including the non-bootstrap setting (bootstrap setting 100%). While the majority vote setting for the output offers a fair chance for each class to be detected, the priority threshold boosts detection of the prioritised class. This would be useful if the dataset is imbalanced or the MtCE is used to detect/classify anomalies.

Despite the extensive experimental studies presented in this paper, there remain opportunities/possibilities for further improvement. First, we are aware that multiple types of base classifiers will increase the complexity of the ensemble and increase the cost and processing time for parameter tuning. To overcome this problem, in the near future, we plan to expand on our previously proposed algorithms for ensemble parameter optimisation, namely the Multi-level **Particle Swarm Optimisation (PSO)** and its parallel version [8]. These nature-inspired algorithms are based on a low-cost PSO algorithm. They were designed to optimise the parameters of an ensemble model, such as BP and AB, choose its base classifier type, and optimise the parameters of these base classifier candidates. Other avenues for future work include investigating the implementation of the MtCE for multi-class problems, heavily imbalanced datasets, and anomaly detection.

APPENDIX

A RESULTS OF THE COMBINATION OF BASE CLASSIFIERS FOR THE MTCE MODEL

Num.	MtCE Base Classifier Combination	Ott et al.'s Dataset				Li et al.'s Dataset (Doctor)			
		Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
1	LR-LSVM	87.31	86.70	88.16	87.39	83.33	84.42	90.61	87.16
2	LR-MLP	88.00	86.85	89.44	88.08	84.41	84.23	92.97	88.33
3	LR-DT	86.69	86.96	86.48	86.67	80.82	80.96	91.62	85.91
4	LR-NB	88.56	87.47	90.07	88.72	85.29	86.51	92.21	89.00
5	LSVM-MLP	87.50	85.72	90.02	87.74	86.56	87.05	93.12	89.88
6	LSVM-DT	86.06	86.19	86.01	86.03	82.64	83.84	91.12	87.07
7	LSVM-NB	89.19	88.32	90.43	89.31	84.75	86.85	90.32	88.19
8	MLP-DT	87.81	87.80	87.28	87.50	85.13	83.71	95.45	89.01
9	MLP-NB	89.63	88.52	91.22	89.80	85.31	84.00	95.30	89.12
10	DT-NB	88.50	88.08	89.09	88.53	85.68	85.99	92.73	89.01
11	CNN-LR	85.13	85.54	84.33	84.89	82.59	83.88	90.07	86.75
12	CNN-LSVM	84.88	83.70	86.80	85.08	83.15	85.03	89.20	86.95
13	CNN-MLP	86.19	86.33	86.29	86.22	83.52	83.56	92.74	87.69
14	CNN-DT	81.31	81.96	80.53	81.15	79.59	80.82	89.31	84.75
15	CNN-NB	86.88	86.45	87.60	86.91	85.31	86.04	92.13	88.78
16	LR-LSVM-MLP	87.56	86.22	89.36	87.73	83.13	83.63	92.08	87.46
17	LR-LSVM-DT	86.50	85.61	87.68	86.56	82.61	83.56	91.25	86.89
18	LR-LSVM-NB	89.00	88.68	89.59	89.07	84.58	85.70	91.44	88.34
19	LR-MLP-DT	88.06	87.64	88.66	88.09	81.72	81.82	91.57	86.29
20	LR-MLP-NB	88.50	87.68	89.65	88.59	85.13	85.40	92.81	88.85
21	LSVM-MLP-DT	87.88	87.26	88.90	88.01	84.42	84.66	92.54	88.21
22	LSVM-MLP-NB	89.13	87.86	91.00	89.31	85.13	86.39	91.88	88.79
23	MLP-DT-NB	88.00	87.00	89.30	88.09	84.95	84.25	94.29	88.71
24	CNN-LR-DT	85.25	84.72	86.04	85.30	81.55	81.68	91.92	86.19
25	CNN-LR-LSVM	86.63	86.41	86.62	86.42	80.65	81.80	89.33	85.24
26	CNN-LR-MLP	88.19	87.32	89.33	88.27	84.40	83.75	93.51	88.29
27	CNN-LR-NB	87.50	86.60	88.59	87.52	84.58	85.68	91.33	88.23
28	CNN-LSVM-DT	84.88	85.12	84.72	84.87	83.21	84.92	90.47	87.49
29	CNN-LSVM-MLP	87.75	87.16	88.35	87.71	83.87	84.51	91.67	87.76
30	CNN-LSVM-NB	87.69	87.23	88.02	87.59	85.89	86.61	92.38	89.15
31	CNN-MLP-DT	86.50	86.55	86.25	86.29	83.70	83.51	92.99	87.82
32	CNN-MLP-NB	87.75	86.65	89.17	87.87	84.05	83.73	93.60	88.21
33	CNN-NB-DT	86.81	86.71	87.12	86.81	82.79	82.55	92.70	87.29
34	LR-LSVM-MLP-DT	87.94	87.55	88.16	87.79	82.99	82.63	92.49	87.02
35	LR-LSVM-MLP-NB	89.19	88.77	89.95	89.28	85.66	87.01	91.38	88.94
36	LR-LSVM-DT-NB	88.25	87.83	89.11	88.39	84.06	85.49	90.49	87.75
37	LR-MLP-DT-NB	88.75	88.41	89.25	88.79	84.06	83.85	93.06	88.04
38	LSVM-MLP-DT-NB	88.63	87.70	89.74	88.61	86.38	87.52	92.38	89.67
39	CNN-LR-LSVM-DT	85.94	85.82	86.07	85.87	82.44	83.18	90.71	86.65

(Continued)

Continued

Num.	MtCE Base Classifier Combination	Ott et al.'s Dataset				Li et al.'s Dataset (Doctor)			
		Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
40	CNN-LR-LSVM-MLP	86.88	86.69	87.33	86.90	83.17	84.29	90.92	87.29
41	CNN-LR-LSVM-NB	87.25	85.55	89.77	87.52	84.77	86.34	91.00	88.39
42	CNN-LR-MLP-DT	87.38	86.90	88.17	87.49	82.80	82.37	93.17	87.38
43	CNN-LR-MLP-NB	87.75	87.22	88.63	87.84	84.76	84.26	94.15	88.82
44	CNN-LSVM-MLP-DT	87.13	86.76	87.66	87.12	86.21	86.60	92.66	89.48
45	CNN-LSVM-MLP-NB	87.63	87.04	88.48	87.69	85.66	85.59	92.99	89.00
46	CNN-MLP-DT-NB	88.00	87.88	88.21	87.97	84.59	84.50	93.04	88.47
47	LR-LSVM-MLP-DT-NB	88.94	88.38	89.56	88.94	84.24	84.89	92.00	88.12
48	CNN-LR-LSVM-DT-NB	87.88	88.05	87.52	87.69	84.42	85.27	91.48	88.09
49	CNN-LR-LSVM-MLP-DT	87.50	87.10	88.08	87.52	82.07	82.71	91.21	86.61
50	CNN-LR-LSVM-MLP-NB	88.88	88.21	89.83	88.95	86.21	86.45	93.18	89.54
51	CNN-LR-MLP-DT-NB	87.44	86.75	87.98	87.35	85.66	85.57	93.53	89.20
52	CNN-LSVM-MLP-DT-NB	88.25	88.54	87.90	88.16	85.48	85.22	93.55	89.05
53	CNN-LR-LSVM-MLP-DT-NB	87.31	87.02	87.71	87.31	83.17	83.50	92.15	87.40
		Li et al.'s Dataset (Hotel)				Li et al.'s Dataset (Restaurant)			
1	LR-LSVM	85.37	84.23	91.73	87.75	81.13	80.49	82.52	80.59
2	LR-MLP	87.34	85.09	94.30	89.45	85.11	83.99	88.36	85.69
3	LR-DT	84.52	83.82	90.57	87.00	81.34	79.91	83.92	81.53
4	LR-NB	87.39	87.06	91.74	89.30	84.33	82.56	87.35	84.76
5	LSVM-MLP	86.12	84.71	92.67	88.44	85.83	83.98	88.55	86.02
6	LSVM-DT	83.72	83.43	89.47	86.29	81.87	81.45	83.01	81.99
7	LSVM-NB	87.45	87.09	91.75	89.34	85.35	84.51	86.33	85.21
8	MLP-DT	86.01	84.23	92.95	88.31	85.55	86.74	84.42	85.23
9	MLP-NB	87.23	86.69	91.92	89.17	86.04	85.36	87.18	86.15
10	DT-NB	85.74	86.89	88.14	87.50	85.07	85.05	84.76	84.76
11	CNN-LR	83.51	84.05	87.95	85.92	79.84	79.35	80.62	79.56
12	CNN-LSVM	82.98	83.63	87.61	85.56	81.34	83.23	80.11	81.07
13	CNN-MLP	84.68	84.71	89.66	87.03	81.82	81.39	84.25	82.20
14	CNN-DT	81.38	82.87	85.28	84.00	75.63	74.72	77.99	75.96
15	CNN-NB	84.68	86.66	86.73	86.63	83.59	82.63	86.58	84.31
16	LR-LSVM-MLP	86.54	84.99	92.96	88.73	83.33	82.29	84.50	83.27
17	LR-LSVM-DT	85.48	84.70	91.30	87.83	80.83	78.96	83.45	80.75
18	LR-LSVM-NB	85.96	85.20	91.34	88.14	86.58	85.71	88.08	86.41
19	LR-MLP-DT	87.18	85.75	93.40	89.33	84.79	84.37	85.77	84.63
20	LR-MLP-NB	87.18	86.31	92.34	89.19	84.80	83.32	86.30	84.49
21	LSVM-MLP-DT	86.33	84.69	93.17	88.69	82.35	82.10	83.87	82.35
22	LSVM-MLP-NB	86.97	86.09	92.34	89.05	85.32	83.68	88.65	85.75
23	MLP-DT-NB	86.86	86.12	92.02	88.93	84.80	84.00	85.89	84.67
24	CNN-LR-DT	84.52	83.48	91.13	87.10	79.12	77.82	80.86	79.11
25	CNN-LR-LSVM	84.57	83.82	90.67	87.04	79.35	78.24	82.94	79.95
26	CNN-LR-MLP	85.85	84.79	91.96	88.17	83.34	82.42	84.09	82.88
27	CNN-LR-NB	86.54	86.95	90.12	88.47	83.34	82.92	84.77	83.00

(Continued)

Continued

Num	MtCE Base Classifier Combination	Ott et al.'s Dataset				Li et al.'s Dataset (Doctor)			
		Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
28	CNN-LSVM-DT	84.57	84.83	89.24	86.88	79.10	78.69	80.41	79.14
29	CNN-LSVM-MLP	85.11	84.11	91.33	87.50	83.10	82.97	83.82	83.05
30	CNN-LSVM-NB	86.33	86.77	89.95	88.26	78.50	75.48	80.49	77.64
31	CNN-MLP-DT	85.05	84.02	91.18	87.42	82.40	81.95	84.67	82.49
32	CNN-MLP-NB	87.13	86.78	91.49	89.04	84.86	87.11	85.31	86.01
33	CNN-DT-NB	85.16	85.70	88.99	87.29	82.57	84.78	80.46	82.08
34	LR-LSVM-MLP-DT	85.90	84.13	92.97	88.29	83.10	80.86	85.57	82.80
35	LR-LSVM-MLP-NB	86.70	85.12	92.92	88.78	82.32	81.51	84.77	82.80
36	LR-LSVM-DT-NB	86.28	85.79	91.29	88.41	84.35	83.62	86.91	84.73
37	LR-MLP-DT-NB	87.82	86.67	93.26	89.81	84.38	83.61	85.67	84.57
38	LSVM-MLP-DT-NB	87.18	86.13	92.67	89.20	84.57	82.77	87.86	84.89
39	CNN-LR-LSVM-DT	85.11	84.17	91.10	87.44	82.08	81.18	82.93	81.78
40	CNN-LR-LSVM-MLP	85.00	84.51	90.96	87.44	82.10	81.55	84.17	82.46
41	CNN-LR-LSVM-NB	85.85	85.19	91.30	88.09	82.88	82.45	83.32	82.62
42	CNN-LR-MLP-DT	85.43	84.25	91.93	87.87	83.54	82.08	83.21	82.24
43	CNN-LR-MLP-NB	86.54	85.87	91.66	88.62	83.62	82.77	84.95	83.54
44	CNN-LSVM-MLP-DT	85.11	84.34	91.26	87.55	80.37	80.19	82.41	80.66
45	CNN-LSVM-MLP-NB	86.76	85.74	92.15	88.82	83.35	83.36	83.37	83.02
46	CNN-MLP-DT-NB	86.91	86.48	91.43	88.87	84.32	83.74	85.90	84.59
47	LR-LSVM-MLP-NB-DT	87.55	86.19	93.28	89.58	84.63	83.29	87.36	85.06
48	CNN-LR-LSVM-DT-NB	86.28	85.77	91.39	88.44	84.03	82.07	86.59	84.07
49	CNN-LR-LSVM-MLP-DT	85.96	84.83	92.06	88.28	82.13	83.09	81.83	81.64
50	CNN-LR-LSVM-MLP-NB	86.76	85.98	91.98	88.81	83.80	82.59	86.17	83.98
51	CNN-LR-MLP-DT-NB	86.06	85.09	91.85	88.31	84.30	84.98	83.43	84.04
52	CNN-LSVM-MLP-DT-NB	85.74	85.20	91.13	88.01	83.59	82.36	85.31	83.56
53	CNN-LR-LSVM-MLP-DT-NB	85.32	85.38	89.80	87.50	83.09	82.68	83.40	82.87

REFERENCES

[1] A. U. Akram, H. U. Khan, S. Iqbal, T. Iqbal, E. U. Munir, and M. Shafi. 2018. Finding rotten eggs: A review spam detection model using diverse feature sets. *KSII Transactions on Internet and Information Systems* 12, 10 (2018). DOI: http://dx.doi.org/10.3837/tiis.2018.10.026

[2] S. N. Alsubari, M. B. Shelke, and S. N. Deshmukh. 2020. Fake reviews identification based on deep computational linguistic features. *International Journal of Advanced Science and Technology* 29, 8s (2020), 3846–3856.

[3] R. Barbado, O. Araque, and C. A. Iglesias. 2019. A framework for fake review detection in online consumer electronics retailers. *Information Processing & Management* 56, 4 (2019), 1234–1244. DOI: http://dx.doi.org/10.1016/j.ipm.2019.03.002

[4] L. Bing, T.-L. Wong, and W. Lam. 2016. Unsupervised extraction of popular product attributes from e-commerce web sites by considering customer reviews. *ACM Transactions on Internet Technology* 16, 2 (2016), 1–17. DOI: http://dx.doi.org/10.1145/2857054

[5] L. Breiman. 1996. Bagging predictors. *Machine Learning* 24, 2 (1996), 123–140. DOI: https://doi.org/10.1007/bf00058655

[6] L. Breiman. 1999. Pasting small votes for classification in large databases and on-line. *Machine Learning* 36, 1 (1999), 85–103. DOI: http://dx.doi.org/10.1023/A:1007563306331

[7] L. Breiman. 2001. Random forests. *Machine Learning* 45, 1 (2001), 5–32. DOI: https://doi.org/10.1023/a:1010933404324

[8] G. S. Budhi, R. Chiong, and S. Dhakal. 2020. Multi-level particle swarm optimisation and its parallel version for parameter optimisation of ensemble models: A case of sentiment polarity prediction. *Cluster Computing* 23 (2020), 3371–3386. DOI: https://doi.org/10.1007/s10586-020-03093-3

[9] G. S. Budhi, R. Chiong, I. Pranata, and Z. Hu. 2017. Predicting rating polarity through automatic classification of review texts. In *Proceedings of the 2017 IEEE Conference on Big Data and Analytics (ICBDA)*. Kuching, Malaysia, 19–24. DOI: https://doi.org/10.1109/ICBDAA.2017.8284101

[10] G. S. Budhi, R. Chiong, I. Pranata, and Z. Hu. 2021. Using machine learning to predict the sentiment of online reviews: A new framework for comparative analysis. *Archives of Computational Methods in Engineering* 28 (2021), 2543–2566. DOI: https://doi.org/10.1007/s11831-020-09464-8

[11] G. S. Budhi, R. Chiong, and Z. Wang. 2021. Resampling imbalanced data to detect fake reviews using machine learning classifiers and textual-based features. *Multimedia Tools and Applications* 80 (2021), 13079–13097. DOI: https://doi.org/10.1007/s11042-020-10299-5

[12] G. S. Budhi, R. Chiong, Z. Wang, and S. Dhakal. 2021. Using a hybrid content-based and behaviour-based featuring approach in a parallel environment to detect fake reviews. *Electronic Commerce Research and Applications* 47 (2021), 101048. DOI: https://doi.org/10.1016/j.elerap.2021.101048

[13] C. Campbell and Y. Ying. 2011. *Learning with Support Vector Machines*. Morgan & Claypool.

[14] E. F. Cardoso, R. M. Silva, and T. A. Almeida. 2018. Towards automatic filtering of fake reviews. *Neurocomputing* 309 (2018), 106–116. DOI: http://dx.doi.org/10.1016/j.neucom.2018.04.074

[15] C. C. Chang and C. J. Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2, 3 (2011), 1–27. DOI: https://doi.org/10.1145/1961189.1961199

[16] R. Chiong, G. S. Budhi, and S. Dhakal. 2021. Combining sentiment lexicons and content-based features for depression detection. *IEEE Intelligent Systems* 36, 6 (2021), 99–105. DOI: https://doi.org/10.1109/MIS.2021.3093660

[17] R. Chiong, G. S. Budhi, S. Dhakal, and F. Chiong. 2021. A textual-based featuring approach for depression detection using machine learning classifiers and social media texts. *Computers in Biology and Medicine* 135 (2021), 104499. DOI: https://doi.org/10.1016/j.compbiomed.2021.104499

[18] X. Deng, Y. Li, J. Weng, and J. Zhang. 2019. Feature selection for text classification: A review. *Multimedia Tools and Applications* 78, 3 (2019), 3797–3816. DOI: https://doi.org/10.1007/s11042-018-6083-5

[19] M. Dong, L. Yao, X. Wang, B. Benatallah, C. Huang, and X. Ning. 2018. Opinion fraud detection via neural autoencoder decision forest. *Pattern Recognition Letters* 132 (2018), 21–29. DOI: http://dx.doi.org/10.1016/j.patrec.2018.07.013

[20] A. M. Elmogy, U. Tariq, and A. I. A. Mohammed. 2021. Fake reviews detection using supervised machine learning. *International Journal of Advanced Computer Science and Applications* 12, 1 (2021), 601. DOI: https://dx.doi.org/10.14569/IJACSA.2021.0120169

[21] W. Etaiwi and G. Naymat. 2017. The impact of applying different preprocessing steps on review spam detection. *Procedia Computer Science* 113 (2017), 273–279.

[22] A. Felbermayr and A. Nanopoulos. 2016. The role of emotions for the perceived usefulness in online customer reviews. *Journal of Interactive Marketing* 36 (2016), 60–76. DOI: http://dx.doi.org/10.1016/j.intmar.2016.05.004

[23] V. W. Feng and G. Hirst. 2013. Detecting deceptive opinions with profile compatibility. In *Proceedings of International Joint Conference on Natural Language Processing*. Nagoya, Japan, 338–346.

[24] Y. Freund and R. E. Schapire. 1997. A decision-theoretic generalisation of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55, 1 (1997), 119–139. DOI: https://doi.org/10.1006/jcss.1997.1504

[25] J. H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics* 29, 5 (2001), 1189–1232

[26] M. Hazim, N. B. Anuar, M. F. A. B. Razak, and N. A. Abdullah. 2018. Detecting opinion spams through supervised boosting approach. *PLoS ONE* 13, 6 (2018), e0198884. DOI: http://dx.doi.org/10.1371/journal.pone.0198884

[27] D. Hernández Fusilier, M. Montes-Y-Gómez, P. Rosso, and R. Guzmán Cabrera. 2015. Detecting positive and negative deceptive opinions using PU-learning. *Information Processing & Management* 51, 4 (2015), 433–443. DOI: http://dx.doi.org/10.1016/j.ipm.2014.11.001

[28] A. Heydari, M. Tavakoli, and N. Salim. 2016. Detection of fake opinions using time series. *Expert Systems with Applications* 58 (2016), 83–92. DOI: http://dx.doi.org/10.1016/j.eswa.2016.03.020

[29] A. Heydari, M. A. Tavakoli, N. Salim, and Z. Heydari. 2015. Detection of review spam: A survey. *Expert Systems with Applications* 42, 7 (2015), 3634–3642. DOI: http://dx.doi.org/10.1016/j.eswa.2014.12.029

[30] Z. Hu, R. Chiong, I. Pranata, Y. Bao, and Y. Lin. 2019. Malicious web domain identification using online credibility and performance data by considering the class imbalance issue. *Industrial Management & Data Systems* 119, 3 (2019), 676–696. DOI: https://doi.org/10.1108/IMDS-02-2018-0072

[31] R. Chiong, Z. Wang, Z. Fan, and S. Dhakal. 2022. A fuzzy-based ensemble model for improving malicious web domain identification. *Expert Systems with Applications* 204 (2022), 117243. DOI: https://doi.org/10.1016/j.eswa.2022.117243

[32] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation* 3, 1 (1991), 79–87. DOI: https://doi.org/10.1162/neco.1991.3.1.79

[33] M. S. Javed, H. Majeed, H. Mujtaba, and M. O. Beg. 2021. Fake reviews classification using deep learning ensemble of shallow convolutions. *Journal of Computational Social Science* 4, 2 (2021), 883–902. DOI: http://dx.doi.org/10.1007/s42001-021-00114-y

[34] R. Chiong, Z. Fan, Z. Hu, and S. Dhakal. 2022. A novel ensemble learning approach for stock market prediction based on sentiment analysis and the sliding window method. *IEEE Transactions on Computational Social Systems*. DOI: http://dx.doi.org/10.1109/TCSS.2022.3182375

[35] R. W. Johnson. 2001. An introduction to the bootstrap. *Teaching Statistics* 23, 2 (2001), 49–54. DOI: https://doi.org/10.1111/1467-9639.00050

[36] Keras. 2019. Keras: The Python Deep Learning library.

[37] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bul'O. 2015. Deep neural decision forests. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, Santiago, Chile, 1467–1475.

[38] A. Kumar, R. D. Gopal, R. Shankar, and K. H. Tan. 2022. Fraudulent review detection model focusing on emotional expressions and explicit aspects: Investigating the potential of feature engineering. *Decision Support Systems* 155 (2022), 113728. DOI: http://dx.doi.org/10.1016/j.dss.2021.113728

[39] N. Kumar, D. Venugopal, L. Qiu, and S. Kumar. 2018. Detecting review manipulation on online platforms with hierarchical supervised learning. *Journal of Management Information Systems* 35, 1 (2018), 350–380. DOI: http://dx.doi.org/10.1080/07421222.2018.1440758

[40] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86, 11 (1998), 2278–2324. DOI: https://doi.org/10.1109/5.726791

[41] J. Li, M. Ott, C. Cardie, and E. Hovy. 2014. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Baltimore, MD, USA, 1566–1576.

[42] L. Li, B. Qin, W. Ren, and T. Liu. 2017. Document representation and feature combination for deceptive spam review detection. *Neurocomputing* 254 (2017), 33–41. DOI: http://dx.doi.org/10.1016/j.neucom.2016.10.080

[43] J. Malbon. 2013. Taking fake online consumer reviews seriously. *Journal of Consumer Policy* 36, 2 (2013), 139–157. DOI: http://dx.doi.org/10.1007/s10603-012-9216-7

[44] C. D. Manning, P. Raghavan, and H. Schuetze. 2008. Naïve Bayes text classification. In *Introduction to Information Retrieval* Cambridge University Press, 234–265.

[45] D. Martens and W. Maalej. 2019. Towards understanding and detecting fake reviews in app stores. *Empirical Software Engineering* 24, 6 (2019), 3316–3355. DOI: http://dx.doi.org/10.1007/s10664-019-09706-9

[46] S. Menard. 2010. *Logistic Regression: From Introductory to Advanced Concepts and Applications*. SAGE, Los Angeles, CA.

[47] Y. Mohammadi, A. Salarpour, and R. Chouhy Leborgne. 2021. Comprehensive strategy for classification of voltage sags source location using optimal feature selection applied to support vector machine and ensemble techniques. *International Journal of Electrical Power & Energy Systems* 124 (2021), 106363. DOI: http://dx.doi.org/10.1016/j.ijepes.2020.106363

[48] P. Norvig. 2016. How to write a spelling corrector.

[49] M. Ott, C. Cardie, and J. T. Hancock. 2013. Negative deceptive opinion spam. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia, USA, 497–501.

[50] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*. Portland, Oregon, 309–319.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, 85 (2011), 2825–2830.

[52] R. Polikar. 2006. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine* 6, 3 (2006), 21–45. DOI: http://dx.doi.org/10.1109/MCAS.2006.1688199

[53] J. R. Quinlan. 1986. Induction of decision trees. *Machine Learning* 1, 1 (1986), 81–106. DOI: https://doi.org/10.1007/bf00116251

[54] Q. Ren, M. Li, and S. Han. 2019. Tectonic discrimination of olivine in basalt using data mining techniques based on major elements: A comparative study from multiple perspectives. *Big Earth Data* 3, 1 (2019), 8–25. DOI: https://doi.org/10.1080/20964471.2019.1572452

[55] Y. Ren and D. Ji. 2017. Neural networks for deceptive opinion spam detection: An empirical study. *Information Sciences* 385–386, 213–224. DOI: http://dx.doi.org/10.1016/j.ins.2017.01.015

[56] Y. Ren, L. Zhang, and P. N. Suganthan. 2016. Ensemble classification and regression-recent developments, applications and future directions. *IEEE Computational Intelligence Magazine* 11, 1 (2016), 41–53. DOI: http://dx.doi.org/10.1109/mci.2015.2471235

[57] J. K. Rout, S. Singh, S. K. Jena, and S. Bakshi. 2016. Deceptive review detection using labeled and unlabeled data. *Multimedia Tools and Applications* 76, 3 (2016), 3187–3211. DOI: http://dx.doi.org/10.1007/s11042-016-3819-y

[58] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. MIT Press, 318–362.

[59] D. Savage, X. Zhang, X. Yu, P. Chou, and Q. Wang. 2015. Detection of opinion spam based on anomalous rating deviation. *Expert Systems with Applications* 42, 22 (2015), 8650–8657. DOI: http://dx.doi.org/10.1016/j.eswa.2015.07.019

[60] R. E. Schapire. 1990. The strength of weak learnability. *Machine Learning* 5, 2 (1990), 197–227. DOI: http://dx.doi.org/10.1007/BF00116037

[61] G. Shan, L. Zhou, and D. Zhang. 2021. From conflicts and confusion to doubts: Examining review inconsistency for fake review detection. *Decision Support Systems* 144 (2021), 113513. DOI: http://dx.doi.org/10.1016/j.dss.2021.113513

[62] J. Shin, S. Yoon, Y. Kim, T. Kim, B. Go, and Y. Cha. 2021. Effects of class imbalance on resampling and ensemble learning for improved prediction of cyanobacteria blooms. *Ecological Informatics* 61 (2021), 101202. DOI: http://dx.doi.org/10.1016/j.ecoinf.2020.101202

[63] C. Sun, Q. Du, and G. Tian. 2016. Exploiting product related review features for fake review detection. *Mathematical Problems in Engineering 2016*. 1–7. DOI: http://dx.doi.org/10.1155/2016/4935792

[64] X. Tang, T. Qian, and Z. You. 2020. Generating behavior features for cold-start spam review detection with adversarial learning. *Information Sciences* 526 (2020), 274–288. DOI: http://dx.doi.org/10.1016/j.ins.2020.03.063

[65] E. D. Wahyuni and A. Djunaidy. 2016. Fake review detection from a product review using modified method of iterative computation framework. In *Proceedings of MATEC Web of Conferences* 58, 03003. DOI: http://dx.doi.org/10.1051/matec

[66] X. Wang, K. L. S. He, and J. Zhao. 2016. Learning to represent review with tensor decomposition for spam detection. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Texas, USA, 866–875.

[67] D. H. Wolpert. 1992. Stacked generalisation. *Neural Networks* 5, 2 (1992), 241–259. DOI: https://doi.org/10.1016/S0893-6080(05)80023-1

[68] Y. Wu, E. W. T. Ngai, P. Wu, and C. Wu. 2020. Fake online reviews: Literature review, synthesis, and directions for future research. *Decision Support Systems* 132 (2020), 113280. DOI: http://dx.doi.org/10.1016/j.dss.2020.113280

[69] F. Xie, H. Fan, Y. Li, Z. Jiang, R. Meng, and A. Bovik. 2017. Melanoma classification on dermoscopy images using a neural network ensemble model. *IEEE Transactions on Medical Imaging* 36, 3 (2017), 849–858. DOI: http://dx.doi.org/10.1109/TMI.2016.2633551

[70] T. Xiong, Y. Bao, Z. Hu, and R. Chiong. 2015. Forecasting interval time series using a fully complex-valued RBF neural network with DPSO and PSO algorithms. *Information Sciences* 305 (2015), 77–92. DOI: https://doi.org/10.1016/j.ins.2015.01.029

[71] D. Zhang, L. Zhou, J. L. Kehoe, and I. Y. Kilic. 2016. What online reviewer behaviors really matter? Effects of verbal and nonverbal behaviors on detection of fake online reviews. *Journal of Management Information Systems* 33, 2 (2016), 456–481. DOI: https://doi.org/10.1080/07421222.2016.1205907

[72] W. Zhang, Y. Du, T. Yoshida, and Q. Wang. 2018. DRI-RCNN: An approach to deceptive review identification using recurrent convolutional neural network. *Information Processing & Management* 54, 4 (2018), 576–592. DOI: http://dx.doi.org/10.1016/j.ipm.2018.03.007

Received 6 November 2021; revised 12 July 2022; accepted 29 August 2022