Final version of the accepted paper. Cite as: *"Nesar Ahmad Wasi and Muhammad Abulaish, An Unseen Features Enhanced Text Classification Approach, In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Queensland, Australia, 18-23 June 2023, pp. 1-9."*

An Unseen Features Enhanced Text Classification Approach

Nesar Ahmad Wasi
Department of Computer Science
South Asian University
New Delhi, India
nesarahmadwasi17@gmail.com

Muhammad Abulaish
Department of Computer Science
South Asian University
New Delhi, India
abulaish@sau.ac.in

***Abstract*—In this paper, we discuss the issue of features that emerge during the prediction phase of a machine learning model, termed as *unseen features*. Because *unseen features* are absent from the vocabulary of the trained model, they are often rejected during the preprocessing stage of the learning model in standard machine learning approaches. We introduce the idea of *unseen features* and a method for identifying and using them for classification tasks. Because the dimension of feature vector required for trained machine learning model is going to differ upon incorporating *unseen features* of the testing data sample, it is not practical to directly incorporate *unseen features* since they only exist during the prediction phase of a machine learning model. As a result, the feature space for the training set is transformed to the embedding space which facilitates the use of *unseen features*. The proposed approach is empirically evaluated using standard metrics over three benchmark datasets in diverse circumstances (natural and balanced datasets) and on various text types – long-texts (*aka* structured texts) and short-texts (*aka* unstructured texts) considering five distinct classification algorithms. The experimental findings confirm the effectiveness of using *unseen features* during a machine learning model's deployment phase. The proposed *unseen features* enhanced technique outperforms the conventional approaches in both balanced class distribution and natural class distribution scenarios by a significant margin of at least 10%.**

***Index Terms*—Machine learning; Unseen features; Out-of-distribution; Text classification**

I. Introduction

In text classification problems, the words used to describe an entity do alter with time. The terms that are currently available and commonly used in the domain are input to the machine learning model when it is trained for a specific problem. When a machine learning model encounters new terms used in a domain over time, it is unable to incorporate them since the models have not previously encountered these words during machine training. These new terms are disregarded because they are ineffective for the classification process. In this paper, we present a novel method for identifying and managing features that emerge for the first time during a machine learning model's testing or deployment phase. When a machine learning model encounters an unknown feature, the model typically discards or ignores that feature. Since these features were not present during the machine learning model's training phase, we term them as *unseen features*. We are the first to identify and handle the issue of *unseen features*, to the best of our knowledge. Unseen features are ignored because they are currently thought to be of no consequence during classification. However, they seem to be be useful in machine learning, if we could put them to good use.

As the machine-learning model is trained, and it is deployed in an application area for prediction. The deployed machine-learning model have a fixed feature vector length; when a data sample is assigned for classification during the deployment phase, it needs to be converted to a feature vector of the same length. Unseen features do not carry specific knowledge associated with them; therefore, the deployed machine-learning model cannot utilize such features directly. We have used word embedding to utilize unseen features. The word embedding space retains the contextual semantics of words by representing them as dense vectors.

We have conducted in-depth experiments in two scenarios (balanced class distribution and natural class distribution) using both structured and unstructured types of texts on three benchmark datasets – the *IMDB movie review* dataset, the Chen *et al.* [1] dataset, and the *Twitter US airline sentiment* dataset. We have employed five classification methods, such as Support Vector Machines (SVM), Gradient Boosting (GB), Random Forest (RF), Logistic Regression (LR), and Decision Tree (DT) for classification tasks, and used precision, recall, F1-score, and accuracy to assess the efficacy of the proposed unseen features enhanced classification technique. The outcomes of the performance evaluation point to the need of applying unseen features during the deployment phase. The performance evaluation findings also indicate that the proposed approach is substantially more able than conventional approaches to identify the minority class – whether it be a positive or negative class.

The key contributions of this study can be summed up as follows: (i) A technique for finding unseen features that a machine learning model might encounter during testing or deployment phase, (ii) A method for enhancing the classification accuracy of machine learning models by incorporating unseen features.

The remainder of the paper is structured as follows. A comparative analysis of similar paradigms, including out-of-distribution, outlier detection, anomaly detection, open-set recognition, novelty detection, transfer learning, and online/incremental learning, is presented in section II. The functioning and algorithmic details of the proposed unseen features enhanced classification method is described in section III. Datasets, evaluation metrics, experimental findings, and analysis are presented in section IV. Finally, a conclusion along with future direction of research is presented in section V.

II. Related Works

In this section, we discuss closely related paradigms in the existing literature. The proposed approach can be misinterpreted with concepts such as open-world learning, out-of-distribution, outlier detection, anomaly detection, open-set recognition, novelty detection, transfer learning, and online/incremental learning. Each of the aforementioned paradigms and the differences with the proposed approach are discussed as follows.

In traditional machine learning scenarios we have data from a particular task/ domain $\mathcal{X}$ along with its label $\mathcal{Y}$, it is assumed that $\mathcal{Y}_{test} \subseteq \mathcal{Y}_{train}$, or $\mathcal{Y}_{test} - \mathcal{Y}_{train} = \phi$, such scenario is called as closed-world assumptions. This assumption indicates that testing classes must be seen during training phase of an approach. In approaches that follow close-world assumption, if an instance of a new class appears during the deployment phase, it will be arbitrarily assigned to any of the classes known to the machine learning model [2].

In open-world scenario, a machine learning model can deal with those instances that appear during deployment phase for the first time. In this scenario there is possibility of distribution shift as in this case $\mathcal{Y}_{test} - \mathcal{Y}_{train} \neq \phi$. The approach that handles such kind of situation is called open-world classification. Our approach is different from the open-world classification problem, because our approach is able to identify those features that appear for the first time during deployment phase i.e. it is a feature-level approach, whereas in open-world scenario, instances that belong to different tasks are identified and addressed, i.e., it is task-level approach. In conventional approaches, when features that appear for the first time during deployment phase, i.e., are not present in train vocabulary, are discarded or ignored.

Because of open-world scenario there is a possibility of distribution shift in either of the independent or dependent variables or both. If the distribution shift is in independent variable, i.e., there is distribution shift in $\mathcal{X}$, it is called as covariate shift. However, if the distribution shift is in dependent variable, i.e., there is distribution shift in $\mathcal{Y}$, it is called as semantic shift. Based on the classification of distribution shift presented by Yang *et al.* [3], there are five types of distribution shift – anomaly detection, outlier detection, open-set recognition, out-of-distribution, and novelty detection. Anomaly detection can be both in semantic shift and covariate shift, i.e., the test instances can be from the different distributions or belong to different classes. In outlier detection, the aim is to identify those instances that are significantly different from the rest of the instances present in data distribution. Both covariate shift and semantic shift can occur in outlier detection.

In conventional machine learning, when an instance that belongs to different class appears during deployment phase, it is assigned arbitrarily to one of the known classes; in order to address this issue, open-set recognition is proposed. In open-set recognition aim is to correctly assign a test instance to either the of ”known know classes” or ”unknown unknown classes”. No covariate shift occurs in open-set recognition. In out-of-distribution setting, the aim is to detect test instances that belong to different classes or from different domains. Both covariate shift and semantic shift can occurs in out-of-distribution setting. In novelty detection, an instance that does not belong to in-train-distribution classes is identified, i.e., in novelty detection, an instance of a new class is identified. Covariate shift does not occur in novelty detection.

The problem that we want to address in this paper is different from the one addressed by the sub-categories of distribution shift because we address problem at the feature-level.

In transfer learning, a machine learning model trained for one domain/task (source domain) is applied for classification in a different but related task (target domain). In literature, usually, the source domain has numerous instances of labeled data, and the target domain has fewer instances of labeled data. Some well-known transfer learning approaches are [4], [5], and [6]. The proposed unseen features enhanced approach is different from the transfer learning-based approaches because of the utilization of unseen features.

In online/incremental learning, a machine learning model is trained continuously on the incoming sequential data and produces an updated model. The proposed unseen features enhanced approach differs from the online/incremental learning-based approaches because once the proposed machine learning model is deployed, it is not trained/updated further.

III. Unseen Features Enhanced Text Classification Approach

In a supervised machine learning approach, training data along with its label, i.e., $D^t = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_N, y_N)\}$ is required to train machine learning approach and produce a predictive function. In training data D^t, the training instances $\{x_1, x_2, x_3, \ldots, x_N\}$ are denoted by $X \in \mathbb{R}^{N \times S}$ and class label instances $\{y_1, y_2, y_3, \ldots, y_N\}$ are denoted by $Y \in \mathbb{R}^{N \times C}$, S is the number of features, N is the number of training instances, and C is the number of classes. The predictive function $f(X)$ predicts an output Y for an input X. Here, X is an independent variable, and Y is called dependent variable. X is comprised of a number of features that appeared in the training phase of the machine learning model.

Conventional machine learning approaches, specifically text classification approaches, maintain a vocabulary of unique features that appears in the training phase, also called feature space. After a machine learning model is trained on a classification task, it is deployed in its corresponding application area. When the deployed model is faced with features that appeared

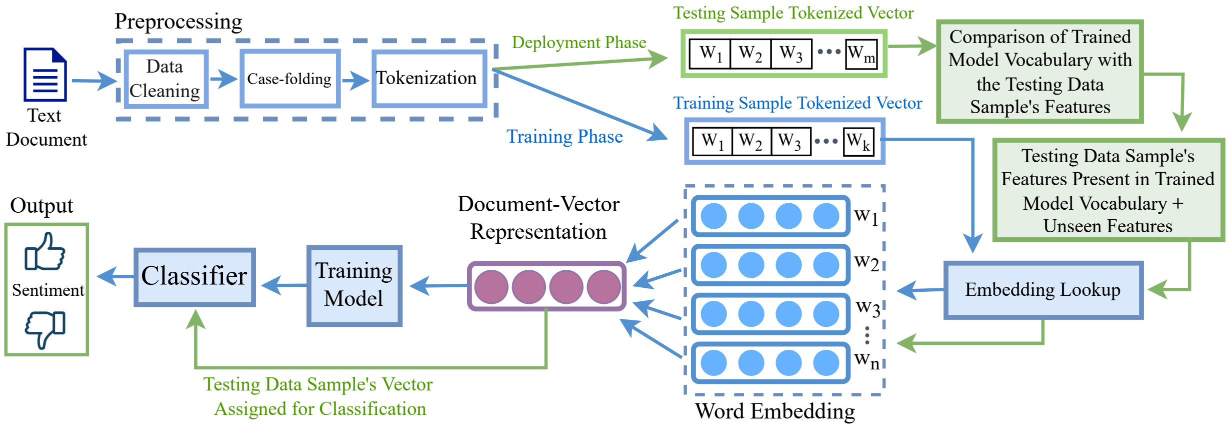

Preprocessing
Text Document
Data Cleaning
Case-folding
Tokenization
Deployment Phase
Testing Sample Tokenized Vector
W_1 W_2 W_3 ••• W_m
Comparison of Trained Model Vocabulary with the Testing Data Sample's Features
Training Phase
Training Sample Tokenized Vector
W_1 W_2 W_3 ••• W_k
Testing Data Sample's Features Present in Trained Model Vocabulary + Unseen Features
Embedding Lookup
w_1
w_2
w_3
w_n
Word Embedding
Document-Vector Representation
Training Model
Classifier
Output
Sentiment
Testing Data Sample's Vector Assigned for Classification

Fig. 1. Workflow of the unseen features enhanced text classification approach

for the first time or those that are absent in the model's feature space are neglected and will not be incorporated in the classification task. We call such features unseen features. The overall framework of the unseen features enhanced approach is presented in Figure 1. The functionality of each module is described in the following subsections.

A. *Preprocessing*

In preprocessing step, each data instance is cleaned from noise or unwanted chunks present in the data. The unwanted chunks can be in the form of special symbols or punctuations, extra spaces, digits, HTML tags, URLs, etc. The unwanted chunks are required to be removed from the data as they negatively affect the classification task and will deteriorate system accuracy. Further, each data sample may also contain contractions that are required to be expanded to their full form. Furthermore, the text in a data sample is converted to lower cases to avoid duplication, as the same word written in different cases will be counted as a separate feature. Next, the data instances go through a filter of stopwords. These are the most common words in the English language that are frequently used and do not provide valuable insights; therefore, they do not help the classification task and are removed. The Natural Language Toolkit (NLTK) is used to remove stopwords. There are 179 stopwords present in NLTK. Further, the data sample goes through the tokenization step. Each data sample is converted to a tokenized vector using the *NLTK word tokenizer*.

B. *Feature Representation*

Each tokenized instance of the dataset must be converted to a numerical representation that the machine understands. Various representation schemes can be used to convert textual data into a numerical representation or encoding. Bag of word representation is the widest-used representation scheme in which a vocabulary of all unique features is formed, and then each feature is counted in the corpus. In such a manner, for a single instance present in the dataset, a vector of the length of vocabulary is created, in which features that appear in this instance are encoded by its count. For features that do not appear in this instance are encoded by zero. One-hot encoding is another representation scheme that is very similar to the Bag of words representation scheme except that instead of the count of a feature, it uses the presence or absence of that feature; if a feature is present in the sample, encoded by 1 else it is encoded by zero.

In the TF-IDF representation scheme, the shortcoming of the bag of words approach is handled. In the bag of words, each feature present in a sample is treated equally, whereas, in the TF-IDF scheme, those features that are important in the sample are weighted according to their importance. Word Embedding is a vector space-based representation in which the context and semantics of words are preserved. In this scheme, those words that are closely related or frequently co-occur are represented with similar representations. There are different implementations of word embedding such as word2vec [7], GloVe [8], BERT [9], Fastext [10], ELMO [11], XLNet [12], GPT [13]. In the proposed approach, word embedding is used as the feature representation scheme for the tokenized data sample.

C. *Unseen Feature Detection*

In order to detect unseen features during the deployment phase, the features space associated with training data $\mathcal{F}_{tr} = \{\forall f_i \in V_{tr}\}$, V_{tr} is the vocabulary of training data, and feature space associated with the testing data sample $\mathcal{F}_{te}$ are used. Both $\mathcal{F}_{tr}$ and $\mathcal{F}_{te}$ are compared, features that appear for the first time in the deployment phase are $\mathcal{O}_t$, i.e.,

$$\mathcal{O}_t = \{\forall f_i \in \{\mathcal{F}_{te} - \mathcal{F}_{tr}\}, \, | f_i \in \mathcal{F}_{te}, f_i \notin \mathcal{F}_{tr}\} \qquad (1)$$

Ideally, features identified using equation (1) are discarded in standard machine learning. We incorporate unseen features during the deployment phase as described in section III-D.

D. *Incorporating Unseen Features*

Unseen features do not carry specific knowledge associated with them; therefore, the deployed machine-learning model cannot utilize such features directly. The deployed machine learning model have a fixed feature vector length; when a data

TABLE I
STATISTICS OF THE DS2.

Domain	# Positive Instances	# Negative Instances	# Neutral Instances	Proportion of Negative Instances
Alarm Clock	624	274	102	30.51
Baby	762	150	88	16.45
Bag	809	110	81	11.97
Cable Modem	845	121	34	12.53
Dumbbell	764	146	90	16.04
Flashlight	816	108	76	11.69
Gloves	796	127	77	19.50
GPS	739	179	82	13.76
Graphics Card	797	136	67	14.58
Headphone	704	187	109	20.99
Home Theater System	644	261	95	28.84
Jewelry	791	110	99	12.21
Keyboard	693	203	104	22.66
Magazine Subscriptions	672	247	81	26.88
Movies TV	829	101	70	10.86
Projector	733	186	81	20.24
Rice Cooker	764	175	61	18.64
Sandal	835	115	50	12.11
Vacuum	717	203	80	22.07
Video Games	718	190	92	20.93

sample is assigned for classification during the deployment phase, it needs to be converted to a feature vector of the same length. In the proposed approach, word embedding associated with $\mathcal{O}_t$ is used to utilize unseen features during the classification task. Prior to assigning the testing data sample to the deployed model for the classification task, it is converted to its document-vector representation as per section III-E.

E. Document Representation using Embedding

In order to encode a tokenized document during the training phase, the word embedding vector associated with each token is extracted from word embedding $\mathbb{W}_{|\mathcal{V}_{\mathbb{W}}|\times l}$, here, $\mathcal{V}_{\mathbb{W}}$ is the vocabulary of word embedding, and l is the dimension of latent space. To encode document d, average of the word embedding $\mathbb{W}_{|\mathcal{V}_{\mathbb{W}}|\times l}$ associated with words that appear in the document d is computed and assigned to the encoded feature vector E_d.

$$E_d = \frac{\sum_{j=1}^{|d|} \mathbb{W}_{|\mathcal{V}_{\mathbb{W}}|\times l}[w_j]}{|d|} \tag{2}$$

In equation (2), w_j denotes feature/word present in document d and $|d|$ denotes the size of the document or number of words that appeared in document d. Equation (2) is used to convert each document in training data i.e., $d_n \in D^t$ to a numerical vector form during the training phase. The dimension of vector E_d is l. Further, the encoded vector E_d is used to train the machine learning model.

During the deployment phase of the machine learning model, to encode tokenized testing data sample d^t, equation (2) is used. In the deployment phase, unseen features $\mathcal{O}_t$ are also incorporated into its document vector representation. Further, the document-vector representation of the testing data sample is assigned to the deployed classifier for classification.

IV. Experimental Setup and Results

In this section, we present the experimental setup of the proposed approach. We also discuss datasets used in experiments. Further, we present the experimental results and analysis.

A. Datasets and Experimental Setup

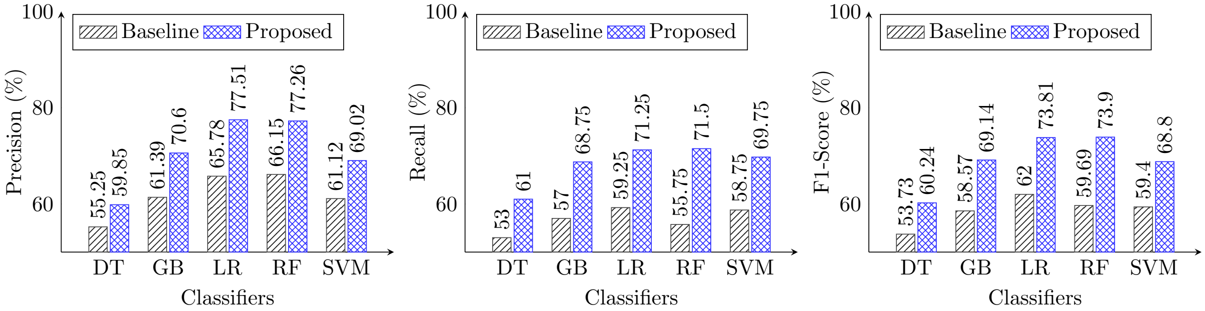
We have conducted extensive experiments on three benchmark datasets used in text classification specifically for the task of sentiment classification – IMDB movie review dataset (DS1) [14], Chen *et al.*'s [1] dataset (DS2), and Twitter US Airline Sentiment dataset (DS3). DS1 contains 50000 reviews which are collected from IMDB. Reviews present in DS1 are labeled based on the movie rating. The lowest rating is 0, whereas the highest rating of a movie is 10. Those movie reviews that have a movie rating of less than 5 are labeled as negative. Those movie reviews that have a rating greater than 6 are labeled as positive reviews. Those movie reviews that have ratings of 5 and 6 are treated as neutral reviews. Neutral reviews are discarded. 25000 movie reviews are present in each positive and negative class. DS1 is a balanced dataset.

DS2 is crawled by Chen *et al.* [1] from amazon.com. In DS2, there are product reviews from 20 domains. In each domain, there are 1000 reviews. Each domain have positive, negative, and neutral reviews. Each product review have a rating from 1 to 5, rating 1 indicates the lowest, and 5 indicates the highest rating. Product reviews with a rating greater than 3 are assigned positive labels. Product reviews with a rating of less than 3 are assigned negative labels. Further, in our experiments, those reviews with a rating of exactly 3 are treated as neutral reviews. We have discarded neutral reviews in our experiments. The detailed statistics of DS2 are presented in Table I. We have conducted experiments on two variants of the DS2 – balanced class distribution and natural class distribution. As per Table I, DS2 is skewed towards positive class in natural class distribution. For performing experiments over DS2 in natural class distribution, we consider the number of positive instances and negative instances presented in Table I. For balanced class distribution, we have randomly created a corpus of 200 reviews in each domain. We sampled 100 positive and 100 negative instances in each domain through a random process. In order to perform experiments, data from every domain is considered as an independent dataset.

DS3 is crawled from Twitter, released by *CrowdFlower*. DS3 contains 14640 customer's tweets about 6 United States airlines – *American*, *Delta*, *Southwest*, *Virgin America*, *United*, and *US Airways*. Each tweet present in this dataset is assigned one of the three labels – *positive*, *negative*, and *neutral*. There are 2363 tweets that are labeled as positive tweets, 9178 tweets labeled as negative tweets, and 3099 tweets labeled as neutral tweets. In our experiments, we have discarded those tweets that are labeled as neutral tweets. We have conducted experiments on two variants of DS3 – balanced class distribution and natural class distribution. DS3 is skewed towards negative class in natural class distribution. For performing experiments over DS3 in natural class distribution, we consider the number of positive instances and negative instances present in both

TABLE II
RESULTS OF THE BASELINE AND UNSEEN FEATURES ENHANCED APPROACHES OVER DS1

Classifier	**Conventional Approaches**				**Unseen Features Enhanced Approaches**			
	Precision	**Recall**	**F1-Score**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Accuracy**
SVM	67.31	69.34	68.31	67.83	80.93	81.00	80.96	80.96
GB	64.33	64.56	64.45	64.38	77.30	76.21	76.75	76.91
LR	67.01	67.98	67.50	67.26	80.13	79.22	79.67	79.79
RF	62.98	62.47	62.72	62.87	75.76	76.19	75.98	75.91
DT	54.53	53.78	54.15	54.46	65.58	63.76	64.66	65.15
Macro Average	63.23	63.63	63.43	63.36	75.94	75.28	75.60	75.74

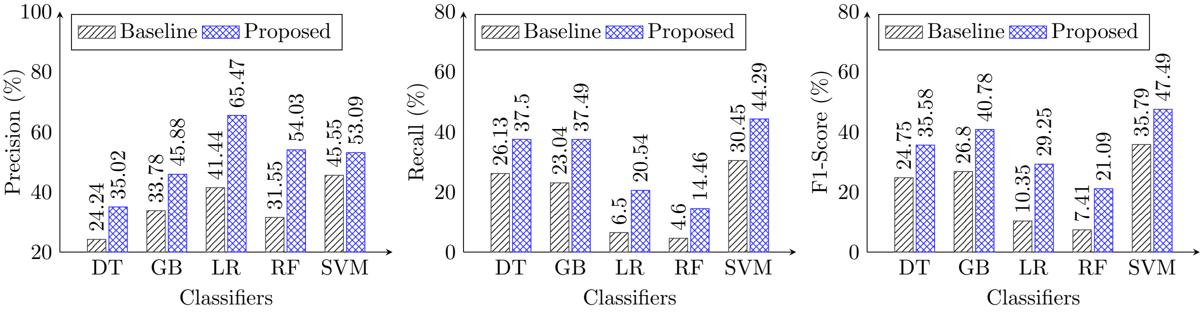


(a) Precision

(b) Recall

(c) F1-Score

Fig. 2. Average of classifiers over all domains of balanced class distribution of DS2



(a) Precision

(b) Recall

(c) F1-Score

Fig. 3. Average of classifiers over all domains of the negative class (in natural class distribution) of DS2 (minority class is negative class)

classes. For balanced class distribution, we randomly sampled a corpus of 4726 tweets, of which there are 2363 tweets from each of the classes. We have kept the train and test ratio to 80% : 20% for all datasets. We have used the hold-out method to evaluate the performance of proposed approach with baselines. We have used the 100 dimensional pre-trained GloVe word embedding.

B. Evaluation Results and Analysis

In order to evaluate unseen features enhanced approach with conventional approaches, we conducted experiments using 5 different classifiers – Support Vector Machine (SVM), Gradient Boosting (GB), Logistic Regression (LR), Random Forest (RF), and Decision Tree (DT). The training phase of both types of conventional and unseen features enhanced approaches are the same. However, unseen features are not incorporated during the conventional approaches' deployment phase. Implementation of all classifiers is based on the Scikit-learn library of python [15]. For SVM, we used linear kernel, which works best in text classification [16], [17]. We used L2-regularization with the regularization parameter C set to 1 for the SVM classifier. We set the default hyper-parameters in Scikit-learn library for GB, LR, RF, and DT classifiers. For performance evaluation on balanced class datasets, we use standard evaluation metrics – *precision*, *recall*, *F1-score*, and *accuracy*. Due to class imbalance, we have considered *precision*, *recall*, and *F1-score* for performance evaluation in natural class distribution. The performance evaluation results of conventional approaches and unseen features enhanced approaches over DS1 are presented in Table II. In Table V and Table VI, we present performance evaluation results of DS2

TABLE III
NATURAL CLASS DISTRIBUTION: PERFORMANCE EVALUATION RESULTS OF THE BASELINE AND UNSEEN FEATURES ENHANCED APPROACHES OVER DS3 TO IDENTIFY MINORITY CLASS (MINORITY CLASS IS POSITIVE CLASS)

Classifier	**Conventional**			**Unseen Features Enhanced**		
	Precision	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
SVM	82.73	24.78	38.14	78.12	62.67	69.54
GB	61.94	32.97	43.04	70.60	59.78	64.74
LR	71.10	33.41	45.45	79.06	63.78	70.60
RF	80.34	20.26	32.36	81.99	47.56	60.20
DT	37.18	40.30	38.68	50.69	56.89	53.61
Macro Average	66.66	30.34	39.53	72.09	58.14	63.74

TABLE IV
BALANCED CLASS DISTRIBUTION: PERFORMANCE EVALUATION RESULTS OF THE BASELINE AND UNSEEN FEATURES ENHANCED APPROACHES OVER DS3

Classifier	**Conventional Approaches**				**Unseen Features Enhanced Approaches**			
	Precision	**Recall**	**F1-Score**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Accuracy**
SVM	72.55	76.92	74.67	73.47	84.14	85.16	84.65	83.93
GB	69.23	71.1	70.15	69.24	82.23	80.89	81.56	80.97
LR	72.69	76.92	74.75	73.57	84.04	84.55	84.3	83.62
RF	70.81	71.1	70.95	70.4	80.39	84.15	82.22	81.08
DT	62.64	59.25	60.9	61.31	72.48	70.12	71.28	70.61
Macro Average	69.58	71.06	70.28	69.60	80.66	80.97	80.80	80.04

TABLE V
NATURAL CLASS DISTRIBUTION: AVERAGE PERFORMANCE EVALUATION RESULTS OF BASELINE AND UNSEEN FEATURES ENHANCED APPROACHES OVER DS2 TO IDENTIFY MINORITY CLASS (MINORITY CLASS IS NEGATIVE CLASS)

Domain	**Conventional Approaches**			**Unseen Features Enhanced Approaches**		
	Precision	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
AlarmClock	55.09	29.53	34.28	66.28	44.13	51.33
Baby	14.22	10.32	11.74	53.36	21.94	25.45
Bag	38.61	14.62	17.19	53.71	21.54	25.19
CableModem	16.51	15.46	15.58	24.23	24.54	23.99
Dumbbell	23.32	12.26	14.89	55.48	31.61	37.21
Flashlight	4.93	3.75	3.96	9.91	8.75	9.19
Gloves	14.88	8.00	9.11	38.76	13.00	16.18
GPS	36.67	20.59	23.32	52.16	34.71	37.34
GraphicsCard	15.39	8.75	10.87	45.40	24.37	29.69
Headphone	34.75	18.79	20.82	56.43	30.30	35.35
Home Theater System	72.03	45.72	54.00	74.62	60.71	66.57
Jewelry	13.93	18.46	15.75	15.13	24.62	18.24
Keyboard	49.29	19.57	23.72	61.87	27.83	36.69
Magazine Subscriptions	58.81	40.45	43.76	70.43	61.34	63.89
Movies TV	31.88	12.00	13.01	23.81	20.00	19.48
Projector	58.09	20.47	27.37	70.14	49.77	55.75
RiceCooker	58.75	22.73	28.95	68.77	38.18	46.81
Sandal	14.07	5.22	6.91	43.53	11.30	14.47
Vacuum	66.29	20.00	26.02	71.60	34.34	44.15
Video Games	28.70	16.32	19.18	58.34	34.21	39.78
Macro Average	35.31	18.15	21.02	50.70	30.86	34.84

in balanced class distribution and natural class distribution, respectively.

It can be observed from Table II, Table III, Table IV, Table V, Table VI, Figure 2, Figure 3, Figure 4, Figure 5, and Figure 6, that incorporating unseen features in a classification approach results in improving the performance of the system. In Table III, Table V, and Figure 5, we present the performance evaluation results of the baseline and unseen features enhanced approaches to identify the minority class. In Table V, the minority class is the negative class. In Table III, the minority class is the positive class. The classification of minority class is hard to identify as there are fewer instances of that class. From Table III, Table V, and Figure 5, it can also be observed that the unseen features enhanced approach is able to identify the minority class significantly better, and it shows the superiority of the unseen features enhanced approaches over baseline approaches. Unseen features enhanced approach performs better on both types of short and long texts. Reviews present in both DS1 and DS2 are of a long-text type or more structural in nature. However, tweets present in DS3

TABLE VI

BALANCED CLASS DISTRIBUTION: AVERAGE PERFORMANCE EVALUATION RESULTS OF THE BASELINE AND UNSEEN FEATURES ENHANCED APPROACHES OVER DS2

Domain	**Conventional Approaches**				**Unseen Features Enhanced Approaches**			
	Precision	**Recall**	**F1-Score**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Accuracy**
AlarmClock	65.85	61.00	63.09	63.59	65.56	54.00	58.66	61.03
Baby	56.96	55.00	55.57	55.90	70.22	69.00	69.40	68.20
Bag	55.46	55.00	55.13	54.36	71.35	69.00	70.00	69.74
CableModem	77.38	61.00	67.97	70.25	72.37	67.00	69.28	69.74
Dumbbell	63.89	60.00	61.61	62.05	71.00	56.00	62.37	65.13
Flashlight	68.33	56.00	61.27	63.59	66.48	66.00	65.88	65.13
Gloves	53.12	43.00	47.46	51.28	62.74	56.00	59.11	60.51
GPS	57.85	50.00	53.23	54.87	70.02	72.00	70.52	69.74
GraphicsCard	66.07	52.00	57.93	61.03	65.95	56.00	60.14	61.54
Headphone	60.24	64.00	61.84	59.49	67.44	71.00	69.00	67.18
Home Theater System	64.25	65.00	64.27	62.05	78.64	87.00	82.53	81.03
Jewelry	70.28	64.00	66.95	67.69	80.81	83.00	81.50	80.51
Keyboard	59.26	58.00	58.06	58.46	73.77	75.00	73.93	72.82
Magazine Subscriptions	62.20	77.00	68.67	63.59	75.12	75.00	75.04	74.36
Movies TV	61.04	58.00	59.02	59.49	70.07	76.00	72.77	71.28
Projector	62.41	45.00	51.29	56.41	61.09	67.00	63.85	61.54
RiceCooker	61.80	52.00	56.05	58.46	81.33	78.00	79.482	80.00
Sandal	56.05	49.00	51.80	53.85	65.42	71.00	68.01	65.64
Vacuum	62.81	61.00	61.56	61.54	67.47	55.00	59.89	62.57
Video Games	53.47	49.00	50.74	52.31	80.14	66.00	72.18	73.84
Macro Average	61.94	56.75	58.68	59.51	70.85	68.45	69.18	69.08

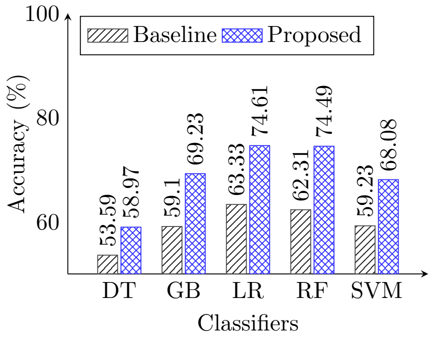
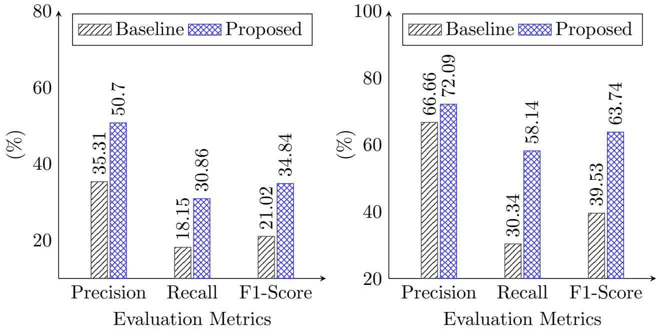


Fig. 4. Average of accuracies of classifiers over all domains of balanced class DS2



(a) Macro average over DS2

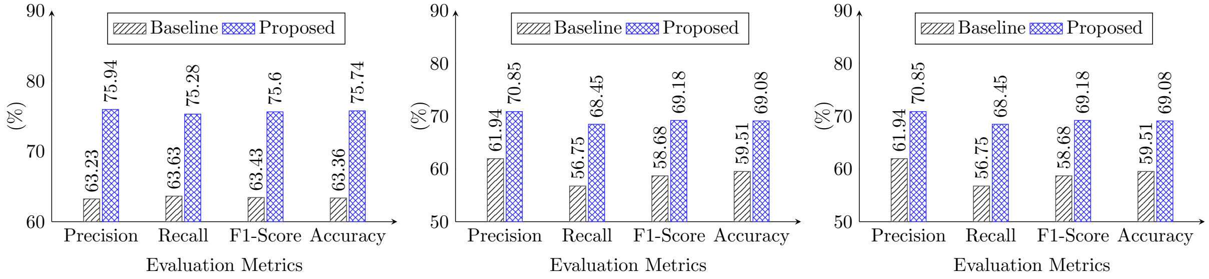
(b) Macro average over DS3

Fig. 5. Macro average of classifiers of identifying minority class in natural class distribution of DS2 and DS3 (in DS2 the minority class is negative, and positive class in DS3)

are more like short-text or unstructured type text, which is more common in online social media platforms. Short-text or unstructured text is difficult to classify.

C. Discussion

Incorporating unseen features during the deployment phase of an approach is a complex problem to address because unseen features appear for the first time in the deployment phase and do not carry knowledge that other features seen during the training phase; therefore, these features can not be incorporated directly in the task of classification. The idea of transforming training and testing data to an embedding space is one solution to address the issue of leveraging unseen features in the classification task. However, there could be other ways to address the problem of unseen features. The approach we present addresses the issue of unseen features. In our approach, the unseen features are looked at in the vocabulary of the word embedding. If the feature is present in the vocabulary of word embedding, that unseen feature is incorporated into the classification task. However, there is a possibility of words that are absent in the vocabulary of word embedding. To address those words, we assign them a zero-valued embedding. However, we believe that these words can be addressed more efficiently. One possible solution to address these words that do not appear in word embedding vocabulary is to split them into a character-level representation and then aggregate the word embedding associated with each character to form a new embedding that represents the unknown word. We believe addressing the aforementioned issue is beyond the scope of this paper because it touches on the concept of the out-of-vocabulary problem in word embedding.



(a) Precision

(b) Recall

(c) F1-Score

Fig. 6. Macro average of classifiers over balanced class distribution of all datasets

V. Conclusion

In this research, we have presented an improved text classification method using unseen features. When a conventional machine learning model is deployed, if it is fed a data sample with new features, the model does not take these features into account since the dimensions of the feature vector used for training and of the testing data sample are different. Such features are referred to as unseen features. The dimensions of the training data space and the testing data sample must match in order to incorporate unseen features during the deployment phase. In order to achieve this, the tokenized feature vector of the training data during the training phase and the testing data sample during the deployment phase is transformed into word embedding space which can evenly represent both sets of features. The results of the performance evaluation over three benchmark datasets point to the importance of the unseen features enhanced approach in a variety of situations (natural and balanced class distributions), different types of text – long-texts (*aka* structured texts) and short-texts (*aka* unstructured texts), and different classifiers (decision tree, gradient boosting, logistic regression, random forest, and support vector machine). The suggested unseen features improved strategy outperforms the traditional approaches in both balanced class distribution and natural class distribution cases by a sizeable margin of at least 10%.

References

[1] Z. Chen, N. Ma, and B. Liu, "Lifelong learning for sentiment classification," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: ACL, Jul. 2015, pp. 750–756.

[2] N. A. Wasi and M. Abulaish, "Document-level sentiment analysis through incorporating prior domain knowledge into logistic regression," in *Proceedings of the 2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 2020, pp. 969–974.

[3] J. Yang, K. Zhou, Y. Li, and Z. Liu, "Generalized out-of-distribution detection: A survey," *CoRR*, vol. abs/2110.11334, 2021.

[4] S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *Proceedings of the 23rd National Conference on Artificial Intelligence*. AAAI Press, 2008, p. 677–682.

[5] F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He, "Supervised representation learning: Transfer learning with deep autoencoders," in *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press, 2015, p. 4119–4125.

[6] Y. Yao and G. Doretto, "Boosting for transfer learning with multiple sources," in *Proceedings of the 23rd IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE Computer Society, 2010, pp. 1855–1862.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the 1st International Conference on Learning Representations, ICLR, Scottsdale, Arizona, USA, May 2-4, Workshop Track Proceedings*, 2013.

[8] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP, October 25-29, 2014, Doha, Qatar*. ACL, 2014, pp. 1532–1543.

[9] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, Minneapolis, MN, USA, June 2-7*. ACL, 2019, pp. 4171–4186.

[10] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext.zip: Compressing text classification models," *CoRR*, vol. abs/1612.03651, 2016.

[11] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: ACL, Jun. 2018, pp. 2227–2237.

[12] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Proceedings of the 32th Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[13] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[14] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: ACL, June 2011, pp. 142–150.

[15] 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 E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[16] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proceddings of the 10th European Conference on Machine Learning: ECML*, C. Nédellec and C. Rouveirol, Eds. Berlin, Heidelberg: Springer, 1998, pp. 137–142.

[17] F. Colas and P. Brazdil, "Comparison of SVM and some older classification algorithms in text classification tasks," in *Proceddings of the 4th IFIP International Conference on Theoretical Computer Science (TCS 2006), IFIP 19th World Computer Congress, TC-1 Foundations of Computer Science, August 23-24, 2006, Santiago, Chile*, ser. IFIP, vol. 217. Springer, 2006, pp. 169–178.