A Novel Feature Selection Based Text Classification Using Multi-layer ELM

Rajendra Kumar Roul[1(✉)] and Gaurav Satyanath[2]

[1] Department of Computer Science, Thapar Institute of Engineering and Technology, Patiala, Punjab, India
raj.roul@thapar.edu
[2] Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA
gsatyana@andrew.cmu.edu

Abstract. Deep learning architectures used for text classification are becoming increasingly prevalent. However, the existing deep architectures have flaws such as slow speed, long training times, and the local minimum problem. Multi-layer Extreme Learning Machine has overcome these problems by avoiding back-propagation and thus saves a significant amount of training time, ensures global optimal, and can handle a vast quantity of data. The most important characteristic of Multi-layer ELM is its *feature space (FS)*, which allows the input features to be linearly separated without using any kernel techniques. The architecture of Multi-layer ELM and its technique of feature mapping are examined in this research with the help of a novel feature selection technique termed as Correlation-based Feature Selection (*CORFS*). Empirical results of the proposed feature selection technique are compared with state-of-the-art techniques. Different classification algorithms are extensively tested on Multi-layer ELM feature space and on TFIDF vector space to demonstrate the efficiency of the feature mapping technique. Results of the experiment revealed that the proposed feature selection technique is better than the conventional feature selection techniques, and the feature space of Multi-layer ELM outperforms TFIDF.

Keywords: Classification · Deep network · Multi-layer ELM · TF-IDF · Vector space

1 Introduction

Text mining can be understood as data mining on textual documents. Typical text mining tasks are text classification, clustering, retrieval, etc. Most of the earlier works used traditional machine learning techniques for text classification, such as support vector machine, naive Bayes, logistic regression, maximum entropy, decision trees, etc. But they are not able to capture the discriminative features automatically from the training data. Their performances heavily depend on data representation, and it is labor-intensive. The literature on text classification has been dominated by deep learning techniques motivated by the outstanding results of deep neural networks in text mining, image processing, and natural language processing [1,2]. But they have limitations like they need large memory bandwidth, huge training time is required because

P. P. Roy et al. (Eds.): BDA 2022 India, LNCS 13773, pp. 33–52, 2022.
https://doi.org/10.1007/978-3-031-24094-2_3

of backpropagation, architecture is very complex, preserving interdependencies among the internal layers for a long time is quite difficult etc. Hence it is not easy to generalize the text classification models to a new domain. An efficient deep learning classifier called Multi-layer ELM was introduced in the year 2013 by Kasun et al. [3] to address the above problems.

1.1 Research Motivation

Overall these existing machine and deep classification techniques have the following limitations:

i. Displaying the data becomes more complex when a large storage space is required due to the growth of the dataset size.
ii. When the input data grows exponentially on the limited dimensional space, distinguishing input features on *TF-IDF* vector space becomes challenging.
iii. Machine and deep learning classifier performances heavily depend on data representation, which is labor-intensive.

Feature selection which selects an optimal subset from the massive volume of the dataset, can alleviate the dimensionality curse but cannot separate the features in lower dimensional space due to the dynamic growth of data items [4,5]. Kernel approaches [6,7] are commonly utilized by any classification process to deal with this challenge. Kernel methods have been used for classification techniques in the past, and better results have been obtained. A detailed survey of kernel and spectral methods for classification has been done by Filippone M et al. [8]. Though kernel methods can handle the data separation in a lower dimensional space by projecting them to a higher-dimensional space, they are expensive (i.e., time-consuming) because of using the dot product to compute the structural similarity among the input features. Using the feature mapping technique of ELM, Huang et al. [9] admitted that by mapping the input vector non-linearly to a high-dimensional feature space, the features become simple and separable linearly, and thus can outperform the kernel approaches [10]. But ELM is a single-layer architecture, thus requiring an extensive network, which is challenging to design to perfectly match the heavily changed input data.

In this vein, this research investigated the feature space of Multi-layer ELM (ML-ELM) [11], which extensively exploits the advantages of *ELM feature mapping* [12,13] and ELM autoencoder to address the constraints mentioned above. The goal of this study is to investigate the extended feature space of ML-ELM (*HDFS-MLELM*) and to thoroughly test this feature space for text classification in comparison to the *TF-IDF vector space* (*VS-TFIDF*).

1.2 Research Contribution

The major contributions of the paper can be summarized as follows:

- This work studies *HDFS-MLELM*, and uses text data to thoroughly investigate multiple classification algorithms on *HDFS-MLELM* and on *VS-TFIDF*.

- It is clear from the past literature that no research on classification using text data has been done on the Multi-layer ELM's enlarged feature space. As a result, in light of the benefits mentioned above, this study can be considered as a new direction in the text classification domain.
- A novel feature selection termed Correlation-based Feature Selection *CORFS* is proposed for selecting the essential features from a big corpus.
- To demonstrate its usefulness, the performance of Multi-layer ELM employing the suggested *CORFS* technique has been compared with several machines and deep learning classifiers.
- Text classification results of various traditional classifiers after running them on ELM feature space and on *HDFS-MLELM* are compared in order to show the effectiveness of *HDFS-MLELM*.
- The experimental results of the proposed approach are compared with the state-of-the-art approaches.

Rest of the paper is as follows: Sect. 2 introduces the preliminaries of Multi-layer ELM and its feature mapping technique. The proposed methodology is discussed in Sect. 3. Section 4 carried out the experimental work. The paper is concluded in Sect. 5.

2 Prelims

2.1 Multi-layer ELM

As demonstrated in Fig. 3, Multi-layer ELM (ML-ELM) is a hybrid of ELM (shown in Fig. 1) and ELM autoencoder (shown in Fig. 2) with more than one hidden layer and is discussed using the following steps.

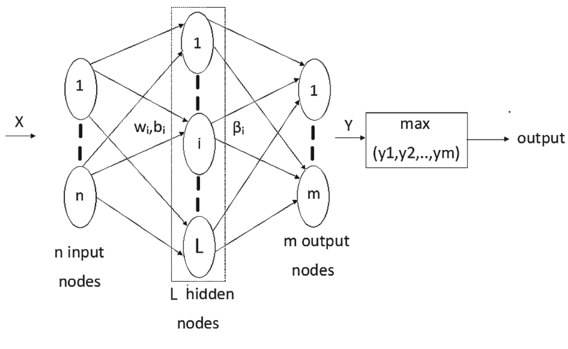

Fig. 1. Overview of ELM

- Unsupervised training occurs between the hidden layers using ELM Autoencoder [14]. Unlike other deep networks, ML-ELM does not require fine-tuning since ELM's autoencoder capacity is an excellent match for ML-ELM [15].
- Stacks are built on top of the ELM Autoencoder in a progressive way to create a multi-layer neural network architecture. The output of one trained ELM Autoencoder is fed into the next ELM Autoencoder, and so on.

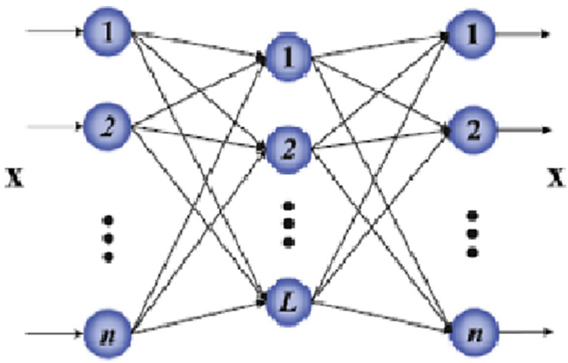

Fig. 2. Overview of ELM-autoencoder

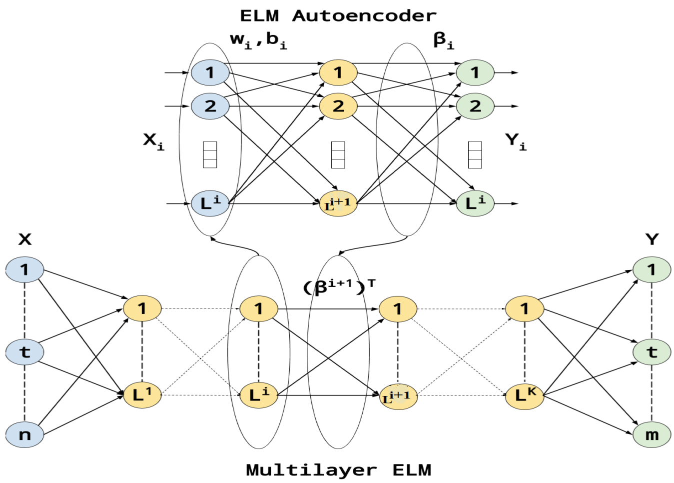

Fig. 3. Overview of Multi-layer ELM

– ELM Autoencoder's first level teaches the fundamental representation of input data. By integrating the previous level's output, the network learns a better representation in the next level, and so on. Equation 1 is used to calculate the numerical understanding between i^{th} and $(i-1)^{th}$ layers.

$$H_i = g((\beta_i)^T H_{i-1}) \tag{1}$$

where H_{i-1} and H_i are the input and output matrices of the i^{th} hidden layer, respectively. $g(.)$ is the activation function, and β is the learning parameter. The input layer is H_0, and the first hidden layer is H_1. Regularized least squares is used to get the output weight β [16].

– Finally, supervised learning is utilized to fine-tune the network (ELM is used for this purpose).

3 Methodology

1. *Documents Pre-processing*:
   Let corpus P consists of C classes. At the beginning of the feature engineering, all documents of each class are combined into a single set called D_{large}. Then lexical-analysis, stop-word deletion, HTML tag removal, and stemming [1] are done on D_{large}. Natural Language Toolkit[2] is used to extract index terms from d_{large}. After completing the basic data cleaning, the first set of features was derived from D_{large} and created a term-document matrix.
2. *Correlation Based Feature Selection (CORFS)*:
   Using k-means[3] clustering algorithm [17], the D_{large} is divided into n term-document clusters $td_i, i \in [1, n]$. The following steps discuss the methodology used to extract important features from each cluster td_i.
   i. Calculating Centroid:
   First the centroid of td_i is calculated using Eq. 2.

$$sc_i = \frac{\sum_{j=1}^{r} t_i}{r} \tag{2}$$

   Then cosine-similarity is computed between $t_j \in td_i$ and sc_i.
   ii. Generating correlation matrix:
   Equation 3 is used to find the correlation (cr)[4] between pair of terms t_i and t_j and is shown in Table 1.

$$cr_{t_i t_j} = \frac{C_{t_i t_j}}{\sqrt{(V_{t_i} * V_{t_j})}} \tag{3}$$

   where, $C_{t_i t_j}$ is the covariance (joint variability between two terms) between t_i and t_j. V_{t_i} and V_{t_j} are their variances respectively as defined below.

$$V_{t_i} = \frac{1}{b-1} \sum_{m=1}^{b} (X_{im} - \overline{X_i})^2$$

$$V_{t_j} = \frac{1}{b-1} \sum_{m=1}^{b} (X_{jm} - \overline{X_j})^2$$

   where $\overline{X_i}$ and $\overline{X_j}$ represents the mean of b documents having the terms t_i and t_j respectively. The covariance between t_i and t_j is computed using Eq. 4.

$$C_{t_i t_j} = \frac{1}{b-1} \sum_{m=1}^{b} (X_{im} - \overline{X_i})(X_{jm} - \overline{X_j}) \tag{4}$$

[1] https://pythonprogramming.net/lemmatizing-nltk-tutorial/.
[2] https://www.nltk.org/.
[3] k value is decided based on the experiment for which the best result is obtained.
[4] https://libguides.library.kent.edu/SPSS/PearsonCorr.

Table 1. Correlation matrix

	t_1	t_2	t_3	$\cdots$	t_r
t_1	cr_{11}	cr_{12}	cr_{13}	$\cdots$	cr_{1r}
t_2	cr_{21}	cr_{22}	cr_{23}	$\cdots$	cr_{2r}
t_3	cr_{31}	cr_{32}	cr_{33}	$\cdots$	cr_{3r}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
t_r	cr_{r1}	cr_{r2}	cr_{r3}	$\cdots$	cr_{rr}

iii. Rejection of high correlated terms from td_i:
Terms that are highly correlated in a cluster are generally considered as a sort of synonym, and hence they do not discriminate well in the cluster. Therefore, those terms should be removed from the cluster. To find those terms in td_i, initially, those terms that have the maximum cosine-similarity score in td_i get selected. Subsequently, a set of terms are identified which are highly correlated to t_i (≤ -0.87 or ≥ 0.89)[5] and that set of terms get removed from td_i. This step is repeated for the next highest cosine-similarity score term and so on till td_i gets exhausted. Finally, all highly correlated terms are removed from td_i.

iv. Computing Discriminating Power Measure (*DPM*):
(*DPM*) [18] is a technique that measures the relevance, i.e., the importance of a term in a cluster. If the *DPM* score of a term inside an unbiased cluster is very high, then that term is an important term for that cluster. It is because many documents of the cluster contain that term. The cohesion or tightness of that term is very close to the cluster's center.

- For each $t_i \in td_i$, the document frequency inside (DF_{in,t_i}) and outside (DF_{out,t_i}) of td_i are calculated using Eqs. 5 and 6 respectively.

$$DF_{in,t_i} = \frac{no.\ of\ documents\ \in td_i\ and\ have\ t_i}{no.\ of\ documents\ \in td_i} \tag{5}$$

$$DF_{out,t_i} = \frac{no.\ of\ documents\ have\ t_i\ and\ \notin td_i}{no.\ of\ documents\ \notin td_i} \tag{6}$$

- The difference between inside and outside document frequency of $t_i \in td_i$ is computed using Eq. 7.

$$DIFF_{td_i,t_i} = |DF_{in,t_i} - DF_{out,t_i}| \tag{7}$$

- Equation 8 computes the *DPM* score of each term.

$$DPM(td_i, t_i) = \sum_{i=1}^{P} DIFF_{td_i,t_i} \tag{8}$$

v. Selection of candidate terms having High *DPM* scores:
of term-document cluster are arranged as per the *DPM* scores, and higher $k\%$ terms are selected as the candidate terms. This step is repeated for each td_i so that every td_i has top $k\%$ candidate terms in them.

[5] decided experimentally so that we will not lose more terms.

3. Input feature vector generation:
   To build the input feature vector, all the top $k\%$ features of each td_i are merged into a list L_{list}.
4. Feature mapping of Multi-layer ELM:
   i. Multi-layer ELM heavily employs the universal classification [19,20] and approximation [21,22] capabilities of ELM.
   ii. ML-ELM cleverly leveraged the extended representation (i.e., $n < L$) technique of the ELM autoencoder [12,23], where n and L are the number of input and hidden layer nodes, respectively.
   iii. The features of ML-ELM are transferred from a low-dimensional feature space to a higher-dimensional feature space using Eq. 9. Mapping of the input vector to *HDFS-MLELM* is shown in Fig. 4 where, $h_i(\mathbf{x}) = g(w_i.\mathbf{x} + b_i)$.

$$h(\mathbf{x}) = \begin{bmatrix} h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \\ h_3(\mathbf{x}) \\ . \\ . \\ . \\ h_L(\mathbf{x}) \end{bmatrix}^T = \begin{bmatrix} g(w_1, b_1, \mathbf{x}) \\ g(w_2, b_2, \mathbf{x}) \\ g(w_3, b_3, \mathbf{x}) \\ . \\ . \\ . \\ g(w_L, b_L, \mathbf{x}) \end{bmatrix}^T \tag{9}$$

   $h(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \cdots, h_i(\mathbf{x}), \cdots, h_L(\mathbf{x})]^T$ transfer the input features to *HDFS-MLELM* [24,25].

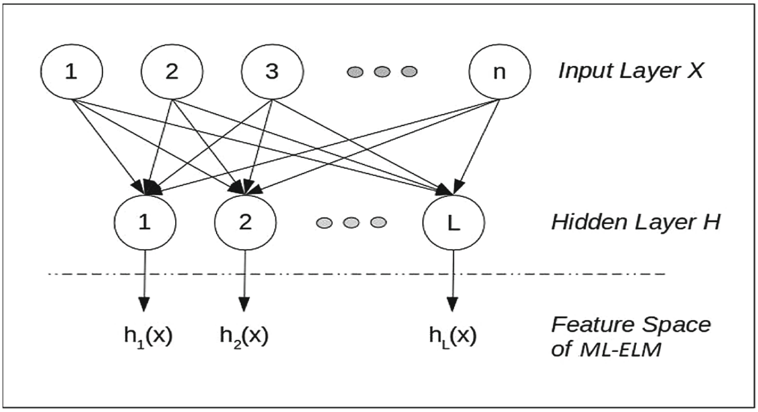

Fig. 4. Feature mapping technique of ML-ELM

   iv. L_{list} is mapped into *MLELM-HDFS* using Eq. 9. Before the transformation, L is set to a higher value than n. This makes all the features of L_{list} linearly separable.
5. Classification on *MLELM-HDFS*:
   Different supervised learning algorithms employing L_{list} as the input feature vector are run individually on *TFIDF-VS* and *MLELM-HDFS* respectively.

4 Analysis of Experimental Results

The setup for the experimental study is detailed in depth in this section. The performance evaluation of state-of-the-art classification algorithms in the feature space of ML-ELM is examined thoroughly. Experiments were done on the feature space of ML-ELM by altering the number of hidden layer nodes L of ML-ELM as per the three representations mentioned below, where n is the number of nodes in the input layer.

- for compress representation ($n > L$): $L = 0.4n$ and $L = 0.7n$
- for extended representation ($n < L$): $L = 1.4n$ and $L = 1.2n$
- for equal representation: ($n = L$): $L = 1.0n$

4.1 Experimental Setup

A Brief Description of the Datasets Utilized in the Experiment: To conduct the experiment, four benchmark datasets (WebKB[6], Classic4[7], 20-Newsgroups[8], and Reuters[9] are used and the details are shown in Table 2.

Table 2. Corpus statistics

Datasets	Training docs	Testing docs	Terms used for training	10% of terms
20-NG	11292	7527	32269	3239
DMOZ	38000	31067	39886	3989
Classic4	4256	2838	15970	1602
Reuters	5484	2188	13532	1351

Tuning Hyper-parameters: The proposed approach for the classification of text data is implemented using python 3.7.3 on Spyder IDE running on a system with Intel Core i11 processor, 32 GB RAM, and 24 GB GPU. GPU is used while running ANN, CNN, and RNN algorithms, and CPU while running Multi-layer ELM. For Multi-layer ELM, we have used 3 hidden layers with 150 nodes in each layer, activation function as Sigmoid (for hidden layer) and Softmax (for output layer). The model is trained using DGX workstation. Tables 3 and 4 show the parameter used for several machine and deep learning algorithms, respectively. Fixing all parameter values is done by repeating the experiment.

[6] http://www.cs.cmu.edu/afs/cs/project/theo-20/www/data/.
[7] http://www.dataminingresearch.com/index.php/2010/09/classic3-classic4-datasets/.
[8] http://qwone.com/~jason/20Newsgroups/.
[9] http://www.daviddlewis.com/resources/testcollections/reuters21578/.

Table 3. Setting different parameters (machine learning)

Classifier	Tuned parameter
SVM	kernel: {'linear'}, random_state: {0}, C: {0.025,1}, gamma: {1}, degree: {3}
K-NN	k: {5}, euclidean distance: {2}
Decision Trees	min_samples_leaf: {5}, criterion: {'entropy'}, random_state: {0}, max_depth: {10, 150, 500}
Random Forest	random_state: {0}, bootstrap:{'True'}, criterion: {'entropy'}, max_depth: {3, 150, 500, 1000}, n_estimators: {100}
Naive Bayes	alpha: {1}, fit_prior: {True}, class_prior: {None}, binarize: {0}
Extra Trees	max_depth: {3}, criterion: {'entropy'}, n_estimators: {100}, random_state: { 0}, min_samples_split: {5}, min_samples_leaf: {5}, max_features: {50}
ELM	no. of hidden layer: {1}, no. of nodes in the hidden layer: {150}, activation function: hidden layer({sigmoid}), ouptput layer ({softmax})
Adaboost	subsample: {0.5}, n_estimators: {10}, learning_rate: {1}, random_state: {0}, max_depth: {5},
Gradient Boosting	min_samples_split: {2}, min_samples_leaf: {5}, learning_rate:{ 0.1(shrinkage)}, subsample: {0.4}, random_state: {0}, n_estimators: {75 (no. of trees)}, max_depth: {3}

Table 4. Setting different parameters (deep learning)

Classifier	No. of Hidden layers	Activation-function	Dropout	Optimizer	Epoch	Batch size
CNN	3	ReLU(hidden layer), Softmax(output layer)	0.5	ADAM	190	80
ANN	4	Sigmoid (hidden layer), Sigmoid(output layer)	0.3	SGD	250	120
RNN	3	Tanh (hidden layer), Softmax(output layer)	0.3	GD	220	110

4.2 Discussion

Performance Evaluation of *CORFS* Technique: The proposed *CORFS* technique is compared with different traditional feature selection techniques (Bi-normal separation (BNS), Mutual Information (MI), Chi-square, and Information Gain (IG)), and the F-measures are shown in Tables 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 respectively for different datasets on top 1%, 5%, and 10% features, where bold indicates maximum. The F-measure of the proposed *CORFS* approach is compared with the state-of-the approaches, which is summarized in Table 17. The findings suggest that the proposed feature selection approach is equivalent to or better than the previous one and can be used to classify text documents using the ML-ELM feature space.

Performance Comparisons of Multi-layer ELM: It's worth noting that ML-ELM outperforms other machine learning classifiers in most feature selection strategies across various datasets, as shown in Table 18. Figures 5 and 6 show F-measure and accuracy comparisons of Multi-layer ELM with various deep learning techniques using the *CORFS* approach. Results indicate the effectiveness of ML-ELM over the machine and deep learning classifiers.

Table 5. 20-NG (Top 1%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.89121	0.87162	0.87361	0.88946	0.87514
SVM	0.89233	0.88966	0.89152	0.89443	0.89427
Decision Trees	0.88226	0.86306	0.86991	0.87513	0.87781
Gradient Boosting	0.85582	0.83783	0.84083	0.86062	0.85223
Adaboost	0.87321	0.88313	0.88382	0.88471	0.87334
NB(Multinomial)	0.86302	0.83861	0.83794	0.85793	**0.88661**
ML-ELM	0.91702	0.91238	0.90512	0.91669	**0.93803**
ELM	0.89241	0.87042	0.87544	0.89572	0.88421
Extra Trees	0.89671	0.87605	0.88642	0.88234	0.88761
RF	0.85992	0.85846	0.85901	0.84242	0.85584

Table 6. 20-NG (Top 5%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.94377	0.93461	0.93729	0.93375	**0.95509**
SVM	0.93459	0.92816	0.92814	0.93014	0.93457
Decision Trees	0.93516	0.88816	0.90252	0.92878	0.93314
Gradient Boosting	0.89954	0.88561	0.89489	0.89591	0.87908
Adaboost	0.89075	0.88366	0.86261	0.87112	0.87184
NB(Multinomial)	0.93122	0.90103	0.91607	0.92511	0.92191
ML-ELM	0.93873	0.94882	0.94664	0.95584	**0.96746**
ELM	0.93551	0.92673	0.93012	0.93682	0.92331
Extra Trees	0.90426	0.88001	0.90223	0.89423	0.89717
RF	0.85997	0.86254	0.85813	0.85763	0.85618

Table 7. 20-NG (Top 10%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.94741	0.93731	0.93647	0.94376	**0.94922**
SVM	0.94284	0.94557	0.93646	0.94652	0.94537
Decision Trees	0.93995	0.91011	0.93352	0.93991	0.91132
Gradient Boosting	0.90146	0.89581	0.89862	0.90511	0.89581
Adaboost	0.88265	0.86262	0.86342	0.87252	0.87685
NB(Multinomial)	0.93821	0.92343	0.93271	0.93731	0.91936
ML-ELM	0.95635	0.96881	0.95566	0.95813	**0.96927**
ELM	0.93228	0.94052	0.93442	0.94671	0.92885
Extra Trees	0.89292	0.89241	0.89471	0.89272	**0.90571**
RF	0.86371	0.84922	0.86604	0.85912	0.85642

Table 8. Classic4 (Top 1%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.92632	0.90663	0.92185	0.92799	0.91670
SVM	0.91492	0.88287	0.89943	0.91785	0.90146
Decision Trees	0.83281	0.79881	0.87911	0.86601	0.83021
Gradient Boosting	0.90966	0.83336	0.88875	0.88188	0.88345
Adaboost	0.89162	0.88094	0.88437	0.88986	0.88393
NB(Multinomial)	0.84197	0.76852	0.80705	0.85317	**0.88163**
ML-ELM	0.94651	0.92222	0.91885	0.94563	**0.95707**
ELM	0.90285	0.88142	0.89945	0.92383	0.90181
Extra Trees	0.91773	0.89212	0.91534	0.91801	0.88714
RF	0.86046	0.84001	0.85312	0.86365	0.85874

Table 9. Classic4 (Top 5%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.94595	0.92332	0.94104	0.94301	**0.95654**
SVM	0.96512	0.93917	0.94051	0.94793	0.96022
Decision Trees	0.92631	0.90762	0.91491	0.90219	0.90988
Gradient Boosting	0.93411	0.92597	0.92951	0.93952	0.93277
Adaboost	0.87344	0.88183	0.84972	0.85483	0.84584
NB(Multinomial)	0.93813	0.92326	0.93096	0.94524	**0.94666**
ML-ELM	0.96667	0.94887	0.96254	0.96547	0.96542
ELM	0.94531	0.92523	0.94675	0.94171	0.94577
Extra Trees	0.91892	0.91892	0.91657	0.91922	0.89553
RF	0.84924	0.84847	0.84886	0.85558	0.84848

Table 10. Classic4 (Top 10%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.94542	0.92652	0.92758	0.97287	0.96768
SVM	0.94546	0.92124	0.92339	0.96868	0.96552
Decision Trees	0.92031	0.90688	0.91548	0.91054	0.89722
Gradient Boosting	0.94021	0.93594	0.93951	0.94238	0.93991
Adaboost	0.85482	0.85482	0.85482	0.84587	0.84587
NB(Multinomial)	0.95097	0.94561	0.94778	0.95352	**0.96918**
ML-ELM	0.97106	0.95877	0.96886	0.97944	**0.98577**
ELM	0.92333	0.94672	0.95634	0.96948	0.95633
Extra Trees	0.91525	0.92688	0.91851	0.91436	0.91675
RF	0.85072	0.84735	0.85205	0.84573	0.85087

Reasons for Better Performance of Multi-layer ELM over Other Classifiers: The following points highlighted the basic reasons behind the superiority of ML-ELM.

i. In ML-ELM, there is no need to fine-tune the hidden node settings and other parameters, and no back-propagations are required. This saves training time, and the learning speed becomes exceedingly rapid throughout the classification phase.
ii. ML-ELM is less expensive than other deep learning architectures because it does not require any GPU to run. When the dataset size grows, excellent performance is realized in ML-ELM.
iii. ML-ELM can map and linearly separate a huge volume of data in the extended space, thanks to its universal approximation and classification capabilities.
iv. The training in ML-ELM is mostly unsupervised except at the last level, where it is supervised.
v. Multiple hidden layers provide a high-level data abstraction, and each layer learns new input forms, making ML-ELM more efficient.

Table 11. Reuters (Top 1%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.93365	0.92335	0.92966	0.93972	0.91524
SVM	0.93242	0.93918	0.93146	0.94952	0.94924
Decision Trees	0.86441	0.85532	0.85344	0.85343	0.85147
Gradient Boosting	0.83336	0.83041	0.83153	0.83832	0.83452
Adaboost	0.64006	0.64004	0.64002	0.76296	**0.77929**
NB(Multinomial)	0.87202	0.84181	0.85837	0.86014	**0.87526**
ML-ELM	0.95413	0.95672	0.95679	0.96758	0.94602
ELM	0.94566	0.95022	0.93142	0.93375	0.93326
Extra Trees	0.92232	0.92944	0.92361	0.93203	0.90924
RF	0.89168	0.88852	0.89003	0.89517	0.88447

Table 12. Reuters (Top 5%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.95122	0.94781	0.94007	0.95424	0.94074
SVM	0.95228	0.94387	0.94643	0.96554	**0.96889**
Decision Trees	0.82963	0.87127	0.85762	0.85328	0.84876
Gradient Boosting	0.84274	0.83727	0.84063	0.84358	0.83985
Adaboost	0.63822	0.65845	0.62866	0.67425	**0.73312**
NB(Multinomial)	0.90244	0.89552	0.90328	0.91178	0.90667
ML-ELM	0.96395	0.96317	0.95839	0.96878	**0.96938**
ELM	0.94569	0.94897	0.95605	0.93452	0.94452
Extra Trees	0.92823	0.92323	0.93327	0.91607	0.91752
RF	0.90287	0.90607	0.90373	0.90356	0.89664

Table 13. Reuters (Top 10%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.95735	0.94174	0.94453	0.94695	0.95477
SVM	0.95487	0.94617	0.94684	0.94818	**0.96782**
Decision Trees	0.79521	0.84727	0.83482	0.81192	0.79144
Gradient Boosting	0.84552	0.84322	0.84262	0.84547	0.84518
Adaboost	0.63822	0.63841	0.63427	0.63416	**0.64808**
NB (Multinomial)	0.90078	0.90556	0.90812	0.90972	0.88692
ML-ELM	0.96723	0.95765	0.96772	0.96322	**0.96957**
ELM	0.92334	0.94666	0.94549	0.95504	0.94558
Extra Trees	0.91903	0.91693	0.91904	0.91982	0.91077
RF	0.90554	0.89852	0.90683	0.89948	0.90657

Table 14. DMOZ (Top 1%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.84016	0.81564	0.83202	0.77058	0.81838
SVM	0.85637	0.82417	0.84035	0.81685	**0.87784**
Decision Trees	0.65441	0.63747	0.67972	0.53565	0.63932
Gradient Boosting	0.76795	0.75208	0.76272	0.74438	0.76457
Adaboost	0.83837	0.81692	0.81552	0.79221	**0.84301**
NB(Multinomial)	0.68786	0.65052	0.66733	0.61532	0.63171
ML-ELM	0.87502	0.85369	0.86633	0.83656	0.84518
ELM	0.81261	0.80381	0.84786	0.78943	0.80537
Extra Trees	0.84671	0.82841	0.83191	0.81561	0.81501
RF	0.79123	0.77397	0.77501	0.74931	0.76596

Table 15. DMOZ (Top 5%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.87807	0.86862	0.87753	0.88152	0.86491
SVM	0.87913	0.88137	0.88866	0.88722	0.85949
Decision Trees	0.70749	0.74333	0.73445	0.70754	0.71887
Gradient Boosting	0.79423	0.78232	0.78506	0.78552	0.77723
Adaboost	0.79832	0.81162	0.81848	0.80972	**0.82283**
NB(Multinomial)	0.77866	0.75184	0.76398	0.75469	0.76444
ML-ELM	0.89678	0.88565	0.90039	0.90672	**0.90918**
ELM	0.86724	0.83253	0.84516	0.87521	0.84679
Extra Trees	0.83864	0.82967	0.84824	0.84483	0.83079
RF	0.78484	0.78763	0.79159	0.78826	0.77707

Table 16. DMOZ (Top 10%)

Classifier	BNS	Chi-square	IG	MI	CORFS
Linear SVC	0.88881	0.88006	0.86353	0.88186	0.87572
SVM	0.87053	0.88834	0.86881	0.88044	0.87601
Decision Trees	0.70036	0.73239	0.71893	0.70537	0.70294
Gradient Boosting	0.80386	0.79181	0.79742	0.79183	0.78614
Adaboost	0.81145	0.79832	0.79835	0.80098	0.80889
NB(Multinomial)	0.78502	0.77861	0.78698	0.77904	**0.79622**
ML-ELM	0.90231	0.90438	0.90533	0.90988	**0.91279**
ELM	0.86626	0.88769	0.88013	0.87566	0.84248
Extra Trees	0.83462	0.83173	0.83147	0.82881	0.81986
RF	0.78082	0.79575	0.79054	0.78477	0.77781

Performance Evaluation of Classification Algorithms: For practical reasons, six distinct classification approaches are performed on the *HDFS-MLELM* and the *VS-TFIDF*, employing four datasets individually. The obtained accuracies and F-measures are shown in Figs. 7, 8, 9 and 10 and Figs. 11, 12, 13 and 14 respectively.

The following conclusions are drawn from the findings:

i. Compared to the *VS-TFIDF*, the empirical findings in all three feature spaces of Multi-layer ELM are superior.
ii. Linear SVM outperforms other supervised learning algorithms, owing to its convex optimization property [35] and generalization property [36], both of which are independent of feature space dimension.
iii. F-measure and accuracy are better in *HDFC-MLELM* whereas it is close on equal dimensional space.

The performance of the proposed approach is compared with the state-of-the-art classification approaches, and the results are shown in Table 19, where bold indicates the maximum accuracy.

Table 17. Performance of Feature selection algorithms (bold indicates maximum)

Authors	Classifier used	Dataset	F-measure (%)
Wang *et al.* [26]	KNN, Naive-Bayes, Decision Trees, SVM, Random Forest, Booster **Best performance: Booster**	New Popularity	67.31
Khoder *et al.* [27]	KNN, SVM, LDA, ICS_DLSR, EM_ICS_FS **Best performance: EM_ICS_FS**	Extended Yale B Face, Outdoor Scene	95.88
Stefano *et al.* [28]	Naive-Bayes, KNN, SVM, Multi-layer perceptron (MLP), **Best performance: MLP**	OPTODIGIT, MFEAT, MNIST, NIST	95.07
Tubishat *et al.* [29]	Association Rule Mining (ARM), Meta classifier, **Best performance: Meta classifier**	Opinion Corpus for Arabic (OCA)	90.80
Jiang *et al.* [30]	MDEFS, MDEFS(NE), BBPSO-FS, **Best performance: BBPSO-FS**	LSVT, Waveform1	91.78
Got *et al.* [31]	FW, BDE, BPSO, BGWO, WOA, jDE **Best performance:FW**	Musk, Sobar, Spectf	88.97
Miao *et al.* [32]	LLfea, Lapscore, SPEC, MCFS, UDFS, EUFS, RSR, NOVRSR, **Best performance:NOVRSR**	BA, JAFFE, ORL	52.18
Ezenkwu *et al.* [33]	Random Forest, SVM, **Best performance: Random Forest**	Ionosphere, Glass Identification, Dermatology, Isolet, Statlog Heart, Landsat satellite, Semeion, Soybean	70.43
Adamu *et al.* [34]	CSA, CCSA, PSO, BPSO, ECCSPSOA, **Best performance: ECCSPSOA**	Wine, Dermatology, Heart, Ionosphere, Lung Cancer, Hepatitis, Parkinson, Divorce, Thoracic Surgery, Phishing Website, Absenteeism at Work	89.76
Proposed (*CORFS*)	Linear SVC, SVM, Decision Trees, Gradient Boosting, Adaboost, Multinomial NB, ML-ELM, ELM, Extra trees, Random Forest, **Best performance: ML-ELM**	20-NG, Classic4, Reuters, DMOZ	**98.57**

Table 18. Comparing ML-ELM with machine learning classifiers using *CORFS*

Classifier	20- Newsgroups			Classic4			Reuters			DMOZ		
	1%	5%	10%	1%	5%	10%	1%	5%	10%	1%	5%	10%
SVC (linear)	87.514	93.454	94.921	90.146	95.655	96.551	91.524	94.074	95.477	81.838	86.491	87.572
SVM (linear)	89.428	95.509	94.536	91.670	96.021	96.768	**94.924**	96.889	96.782	**87.784**	85.949	87.601
Gradient Boosting	85.224	87.908	89.582	88.345	93.278	93.991	83.452	83.985	84.518	76.457	77.723	78.614
Decision Trees	87.782	93.314	91.133	83.021	90.987	89.722	85.147	84.876	79.144	63.932	71.887	70.294
NB (Multinomial)	88.662	92.191	91.937	88.163	94.667	96.918	87.526	90.667	88.692	63.171	76.444	79.622
Adaboost	87.335	87.184	87.686	88.393	84.585	84.587	77.929	73.312	64.808	84.301	82.283	80.889
Random Forest	85.584	85.618	85.641	85.874	84.849	85.087	88.447	89.664	90.657	76.596	77.707	77.781
Extra Trees	88.762	89.717	90.572	88.714	89.554	91.675	90.924	91.752	91.077	81.501	83.079	81.986
ELM	88.420	92.331	92.886	90.182	94.578	95.633	93.326	94.453	94.558	80.537	84.679	84.248
MLELM	**93.802**	**96.746**	**96.928**	**95.707**	**96.541**	**98.577**	94.602	**96.938**	**96.957**	84.518	**90.918**	**91.279**

4.3 Comparisons of ELM and ML-ELM Feature Space

Traditional classifiers are run on *HDFS-MLELM* and ELM feature space. Figures 15, 16, 17 and 18 compare the performances of different classifiers on the higher dimensional feature space($L = 1.4n$) ML-ELM and ELM. The results indicate that the performances of classifiers are better in ML-ELM feature space compared to ELM feature space. The reason is due to the multilayer processing of ML-ELM compared to a single layer in ELM. SVM shows a better performance compared to other classifiers on both feature spaces.

Table 19. Performance of text classification algorithms (bold indicates maximum)

Authors	Classifier used	Dataset	Accuracy (%)
Guangquan *et al.* [37]	CNN, LSTM, **Best performance: CNN**	Amazon review dataset	97.33
Jeow *et al.* [38]	Random Forest, LR, LSTM, **Best performance: LSTM**	Notes dataset of Cincinnati Hospital Medical Centre	85.80
Hamouda *et al.* [39]	Naive-Bayes, Random Forest, SVM, LightGBM, Decision Trees, k-NN, **Best performance: SVM**	Arabic dataset	90.47
Bichitrananda *et al.* [40]	DNN, k-NN, SVM, RNN, CNN, FRS - RNN+ CNN, **Best performance: FRS-RNN + CNN**	20-Newsgroup	98.50
Janani *et al.* [41]	Naïve Bayes, k-NN, SVM, PNN, Adaboost, Random Forest, **Best performance: PNN**	20-Newsgroup, Reuters	93.70
Yan *et al.* [42]	k-NN, Decision Trees, Adaboost, FNN, SVM, HSAN-Capsule model, **Best performance: HSAN-Capsule**	Movie reviews dataset(IMDB)	90.12
Xiang *et al.* [43]	ABLSTM	online consultation data of medical healthcare	98.34
Shiyao *et al.* [44]	Frog-GNN	Amazon dataset, HuffPost dataset, FewRel dataset	94.28
Zhong *et al.* [45]	Multinomial Naive-bayes, SVM, k-NN, Decision Trees, Random Forest, Extra Trees, **Best performance: SVM**	Reuters-21578, 20 Newsgroups dataset	97.20
Shenghong *et al.* [46]	SVM, Neural network, Decision trees, Random Forest, Adaboost, **Best performance: SVM**	Chinese text dataset	79.50
Proposed work	Multinomial Naive-Bayes, Random Forest, k-NN, Linear SVM, Decision Trees, Extra Trees, **Best performance: Linear SVM on ML-ELM feature space**	20-NG, Classic4, Reuters, DMOZ	**98.80**

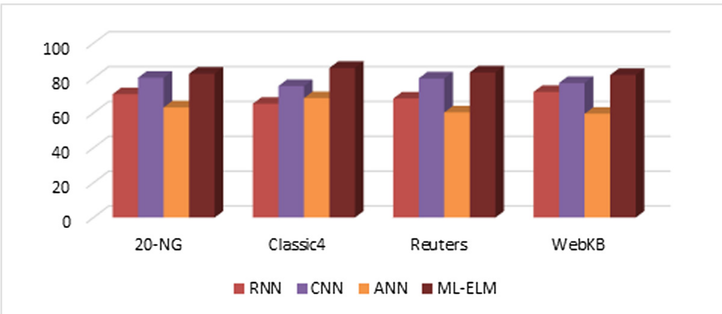

Fig. 5. F1-measure

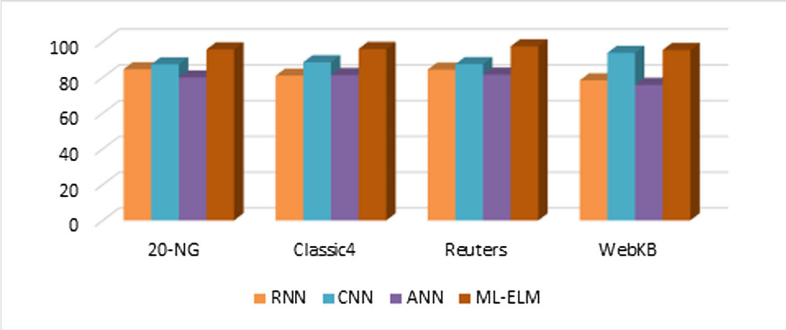

Fig. 6. Accuracy

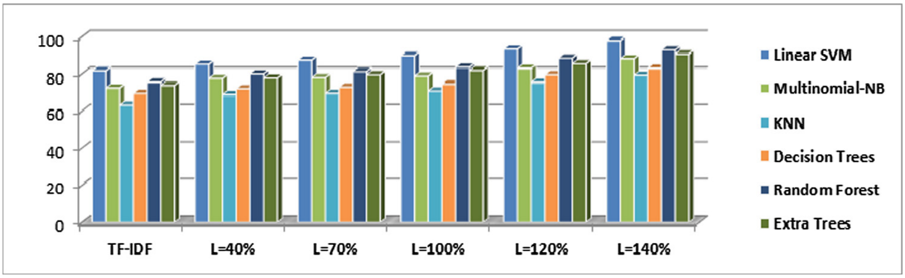

Fig. 7. 20-NG (Accuracy)

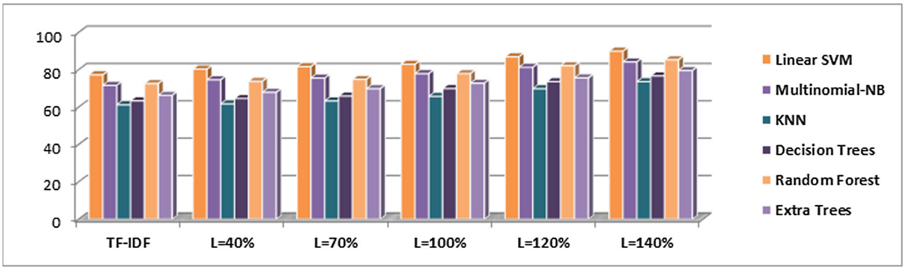

Fig. 8. Classic4 (Accuracy)

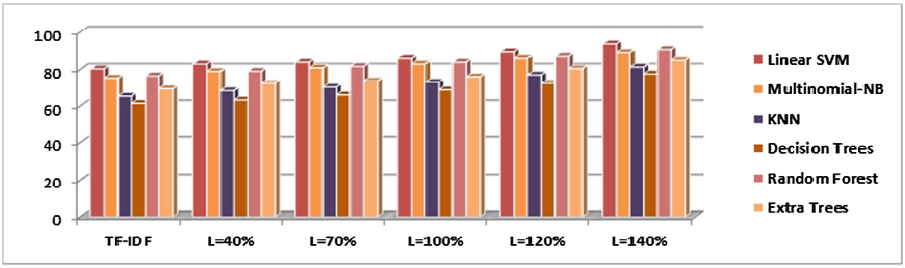

Fig. 9. Reuters (Accuracy)

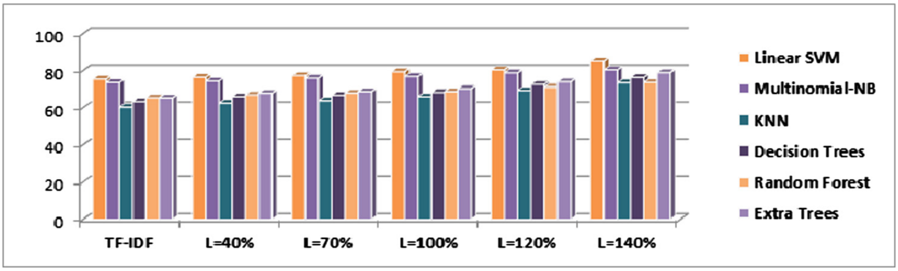

Fig. 10. DMOZ (Accuracy)

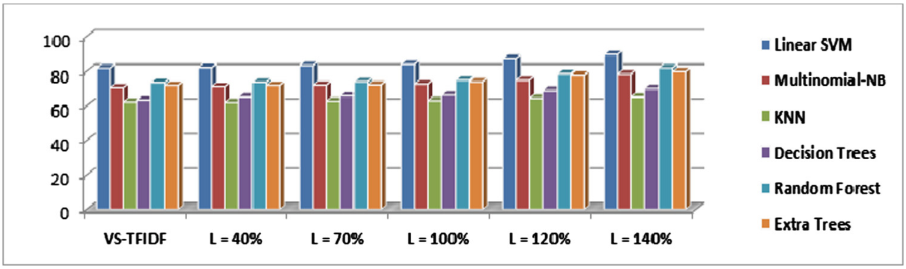

Fig. 11. 20NG (F1-measure)

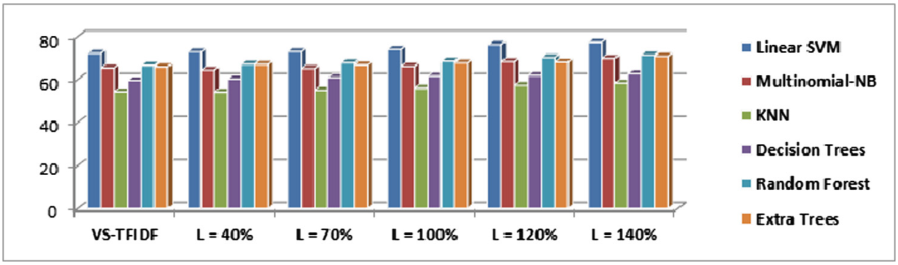

Fig. 12. Classic-4 (F1-measure)

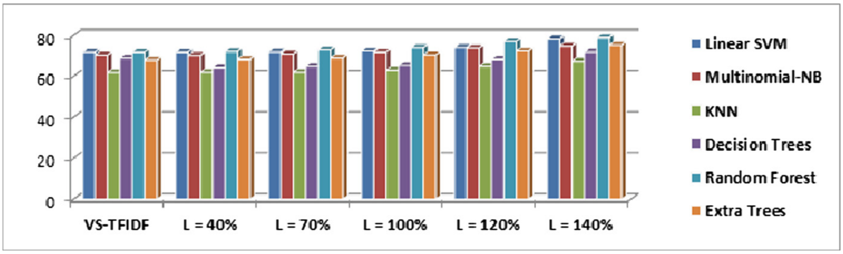

Fig. 13. Reuter (F1-measure)

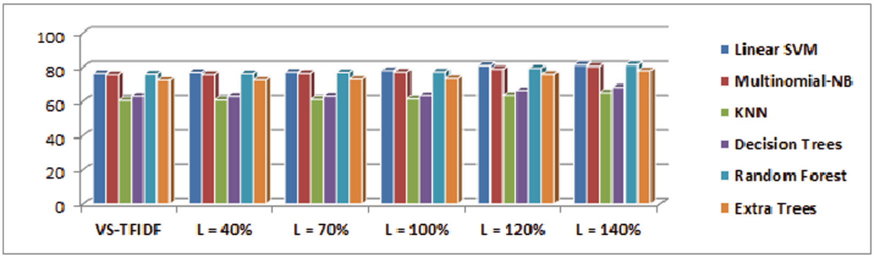

Fig. 14. DMOZ (F1-measure)

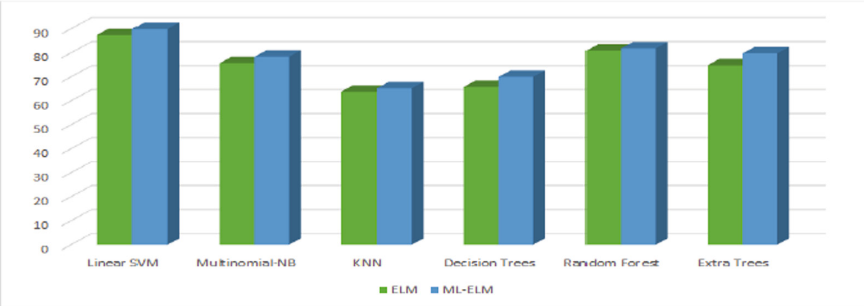

Fig. 15. F1-measure comparisons on ML-ELM and ELM Feature space (20-NG)

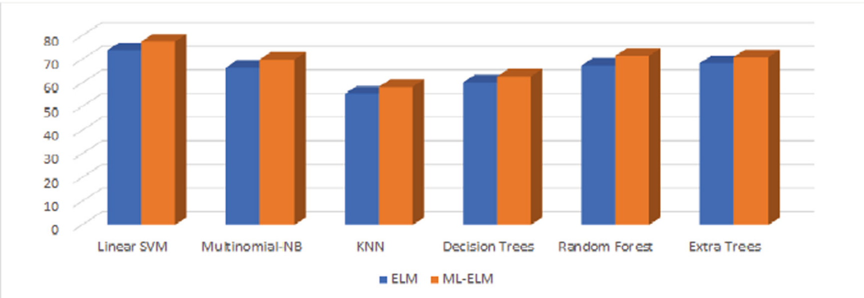

Fig. 16. F1-measure comparisons on ML-ELM and ELM Feature space (Classic4)

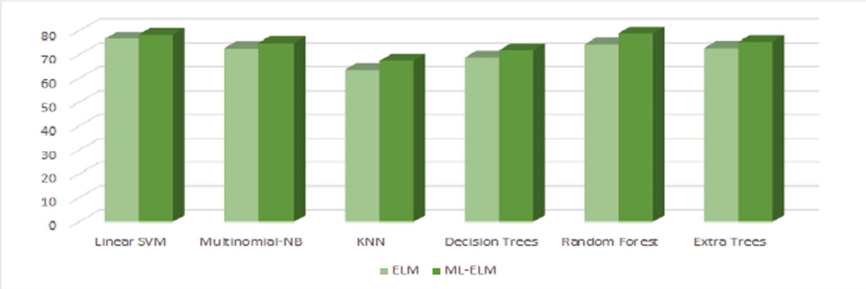

Fig. 17. F1-measure comparisons on ML-ELM and ELM Feature space (Reuters)

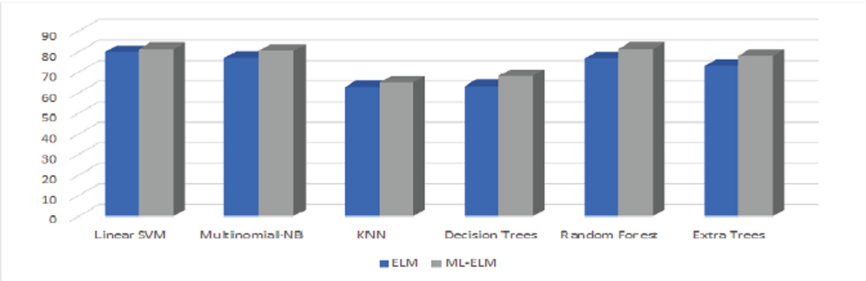

Fig. 18. F1-measure comparisons on ML-ELM and ELM Feature space (DMOZ)

5 Conclusion

The suggested approach investigates the significance of the Multi-layer ELM feature space in-depth. Initially, the corpus is subjected to a novel feature selection technique (*CORFS*), which removes superfluous features from the corpus and improves the classification performance. An extensive empirical study on several benchmark datasets has demonstrated the efficiency of the suggested technique on *HDFS-MLELM* compared to the *VS-TFIDF*. According to empirical investigations, SVM outperforms other classifiers on both feature spaces for all the datasets. After a thorough examination of the experimental results, it has been determined that the Multi-layer ELM feature space

- is able to solve the three major problems faced by the current machine/deep learning techniques as highlighted in Sect. 1.
- can replace the costly kernel techniques.
- is more suitable and much useful for text classification in comparison with *TF-IDF* vector space.

This work can be extended on the following lines:

i. Deep learning methods such as CNN, RNN, and ANN need a vast amount of data and many tuned parameters to train the network. As part of future work, combining these deep learning architectures with ML-ELM can reduce the requirement of tuned parameters without compromising their performances.

ii. More applications of ML-ELM can be studied to verify its generalization capability on huge datasets having noise.
iii. The variance of hidden layer weights is still under investigation to fully comprehend ML-ELM's operation.

Acknowledgement. : We thank Thapar Institute of Engineering and Technology for providing the seed money grant for this research work.

References

1. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep Learning, vol. 1. MIT press, Cambridge (2016)
2. Roul, R.K., Gugnani, S., Kalpeshbhai, S.M.: Clustering based feature selection using extreme learning machines for text classification. In: 2015 Annual IEEE India Conference (INDICON), pp. 1–6. IEEE (2015)
3. Kasun, L.L.C., Zhou, H., Huang, G.-B., Vong, C.M.: Representational learning with extreme learning machine for big data. IEEE Intell. Syst. **28**(6), 31–34 (2013)
4. Roul, R.K., Bhalla, A., Srivastava, A.: Commonality-rarity score computation: a novel feature selection technique using extended feature space of elm for text classification. In: Proceedings of the 8th Annual Meeting of the Forum on Information Retrieval Evaluation, pp. 37–41 (2016)
5. Qian, W., Long, X., Wang, Y., Xie, Y.: Multi-label feature selection based on label distribution and feature complementarity. Appl. Soft Comput. **90**, 106167 (2020)
6. Zhang, L., Zhou, W.D., Jiao, L.C.: Kernel clustering algorithm. Chin. J. Comput. **6**, 004 (2002)
7. Roul, R.K., Arora, K.: A modified cosine-similarity based log kernel for support vector machines in the domain of text classification. In: Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017), pp. 338–347 (2017)
8. Filippone, M., Camastra, F., Masulli, F., Rovetta, S.: A survey of kernel and spectral methods for clustering. Pattern Recogn. **41**(1), 176–190 (2008)
9. Huang, G.-B., Ding, X., Zhou, H.: Optimization method based extreme learning machine for classification. Neurocomputing **74**(1), 155–163 (2010)
10. Huang, G.-B., Chen, L.: Enhanced random search based incremental extreme learning machine. Neurocomputing **71**(16), 3460–3468 (2008)
11. Roul, R.K.: Impact of multilayer elm feature mapping technique on supervised and semi-supervised learning algorithms. Soft Comput. **26**(1), 423–437 (2022)
12. Huang, G.-B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. Part B (Cybern.) **42**(2), 513–529 (2012)
13. Roul, R.K., Asthana, S.R., Kumar, G.: Study on suitability and importance of multilayer extreme learning machine for classification of text data. Soft Comput. **21**(15), 4239–4256 (2017)
14. Roul, R.K.: Detecting spam web pages using multilayer extreme learning machine. Int. J. Big Data Intell. **5**(1/2), 49–61 (2018)
15. Roul, R.K.: Suitability and importance of deep learning feature space in the domain of text categorisation. Int. J. Comput. Intell. Stud. **8**(1–2), 73–102 (2019)
16. Rifkin, R., Yeo, G., Poggio, T.: Regularized least-squares classification. Nato Sci. Ser. Sub Ser. III Comput. Syst. Sci. **190**, 131–154 (2003)

17. Hartigan, J.A., Wong, M.A.: Algorithm as 136: a k-means clustering algorithm. J. Roy. Stat. Soc. Ser. C (Appl. Stat.) **28**(1), 100–108 (1979)
18. Dreiseitl, S., Ohno-Machado, L.: Logistic regression and artificial neural network classification models: a methodology review. J. Biomed. Inf. **35**(5–6), 352–359 (2002)
19. Huang, G.-B., Chen, Y.-Q., Babri, H.A.: Classification ability of single hidden layer feedforward neural networks. IEEE Trans. Neural Netw. **11**(3), 799–801 (2000)
20. Roul, R.K., Rai, P.: A new feature selection technique combined with ELM feature space for text classification. In: Proceedings of the 13th International Conference on Natural Language Processing, pp. 285–292. ACL (2016)
21. Huang, G.-B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. Part B (Cybern.) **42**(2), 513–529 (2011)
22. Roul, R.K., Agarwal, A.: Feature space of deep learning and its importance: comparison of clustering techniques on the extended space of ml-elm. In: Proceedings of the 9th Annual Meeting of the Forum for Information Retrieval Evaluation, pp. 25–28 (2017)
23. Roul, R.K.: Deep learning in the domain of near-duplicate document detection. In: Madria, S., Fournier-Viger, P., Chaudhary, S., Reddy, P.K. (eds.) BDA 2019. LNCS, vol. 11932, pp. 439–459. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-37188-3_25
24. Roul, R.K.: Study and understanding the significance of multilayer-ELM feature space. In: Bellatreche, L., Goyal, V., Fujita, H., Mondal, A., Reddy, P.K. (eds.) BDA 2020. LNCS, vol. 12581, pp. 28–48. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-66665-1_3
25. Huang, G.-B., Chen, L.: Convex incremental extreme learning machine. Neurocomputing **70**(16), 3056–3062 (2007)
26. Wang, H., He, C., Li, Z.: A new ensemble feature selection approach based on genetic algorithm. Soft Comput. **24**(20), 15811–15820 (2020)
27. Khoder, A., Dornaika, F.: Ensemble learning via feature selection and multiple transformed subsets: application to image classification. Appl. Soft Comput. **113**, 108006 (2021)
28. De Stefano, C., Fontanella, F., Marrocco, C., Di Freca, A.S.: A ga-based feature selection approach with an application to handwritten character recognition. Pattern Recogn. Lett. **35**, 130–141 (2014)
29. Tubishat, M., Abushariah, M.A., Idris, N., Aljarah, I.: Improved whale optimization algorithm for feature selection in arabic sentiment analysis. Appl. Intell. **49**(5), 1688–1707 (2019)
30. Jiang, Z., Zhang, Y., Wang, J.: A multi-surrogate-assisted dual-layer ensemble feature selection algorithm. Appl. Soft Comput. **110**, 107625 (2021)
31. Got, A., Moussaoui, A., Zouache, D.: Hybrid filter-wrapper feature selection using whale optimization algorithm: a multi-objective approach. Expert Syst. Appl. **183**, 115312 (2021)
32. Miao, J., Ping, Y., Chen, Z., Jin, X.-B., Li, P., Niu, L.: Unsupervised feature selection by non-convex regularized self-representation. Expert Syst. Appl. **173**, 114643 (2021)
33. Ezenkwu, C.P., Akpan, U.I., Stephen, B.U.-A.: A class-specific metaheuristic technique for explainable relevant feature selection. Mach. Learn. Appl. **6**, 100142 (2021)
34. Adamu, A., Abdullahi, M., Junaidu, S.B., Hassan, I.H.: An hybrid particle swarm optimization with crow search algorithm for feature selection. Mach. Learn. Appl. **6**, 100108 (2021)
35. Bengio, Y., LeCun, Y., et al.: Scaling learning algorithms towards AI. Large-Scale Kernel Mach. **34**(5), 1–41 (2007)
36. Vapnik, V.N.: An overview of statistical learning theory. IEEE Trans. Neural Netw. **10**(5), 988–999 (1999)
37. Lu, G., Gan, J., Yin, J., Luo, Z., Li, B., Zhao, X.: Multi-task learning using a hybrid representation for text classification. Neural Comput. Appl. **32**(11), 6467–6480 (2020)
38. Huan, J., Sk, A.A., Quek, C., Prasad, D.: Emotionally charged text classification with deep learning and sentiment semantic. Neural Comput. Appl. (2021)

39. Chantar, H., Mafarja, M., Alsawalqah, H., Heidari, A.A., Aljarah, I., Faris, H.: Feature selection using binary grey wolf optimizer with elite-based crossover for arabic text classification. Neural Comput. Appl. **32**(16), 12201–12220 (2020)
40. Behera, B., Kumaravelan, G.: Text document classification using fuzzy rough set based on robust nearest neighbor (frs-rnn). Soft Comput. **25**(15), 9915–9923 (2021)
41. Janani, R., Vijayarani, S.: Automatic text classification using machine learning and optimization algorithms. Soft Comput. **25**(2), 1129–1145 (2021)
42. Cheng, Y., et al.: Hsan-capsule: a novel text classification model. Neurocomputing (2021)
43. Li, X., Cui, M., Li, J., Bai, R., Lu, Z., Aickelin, U.: A hybrid medical text classification framework: integrating attentive rule construction and neural network. Neurocomputing **443**, 345–355 (2021)
44. Xu, S., Xiang, Y.: Frog-gnn: multi-perspective aggregation based graph neural network for few-shot text classification. Expert Syst. Appl. **176**, 114795 (2021)
45. Tang, Z., Li, W., Li, Y.: An improved supervised term weighting scheme for text representation and classification. Expert Syst. Appl. **189**, 115985 (2022)
46. Mou, S., Du, P., Cheng, Z.: A brain-inspired information processing algorithm and its application in text classification. Expert Syst. Appl. **177**, 114828 (2021)