Contents lists available at GrowingScience

International Journal of Data and Network Science

homepage: www.GrowingScience.com/ijds

An improved multi-stage framework for large-scale hierarchical text classification problems using a modified feature hashing and bi-filtering strategy

Abubakar Ado[a], Abdulkadir Abubakar Bichi[a], Usman Haruna[a], Mohammed Almaiah[b*], Yahaya Garba Shawai[c], Rommel AlAli[d*], Tayseer Alkhdour[e], Theyazn H.H Aldhyani[f], Mahmoad Alrawad[g] and Rami Shehab[e]

[a]*Department of Computer Science, Yusuf Maitama Sule University, Kano, Nigeria*
[b]*King Abdullah the II IT School, The University of Jordan, Amman 11942, Jordan*
[c]*Directorate of examination and assessment, National Open University of Nigeria*
[d]*Associate Professor, The National Research Center for Giftedness and Creativity, King Faisal University, Saudi Arabia*
[e]*College of Computer Science and Information Technology, King Faisal University, Al-Ahsa 31982, Saudi Arabia*
[f]*Applied College in Abqaiq, King Faisal University, Al-Ahsa 31982, Saudi Arabia*
[g]*College of Business, King Faisal University, Al-Ahsa, 31982, Saudi Arabia*

C H R O N I C L E

Article history:
Received: January 30, 2024
Received in revised format: May 18, 2024
Accepted: June 12, 2024
Available online: June 12, 2024

Keywords:
Hierarchical Classification
Dimensional Reduction
Feature Hashing
Large-scale
Bi-filtering

A B S T R A C T

The classification of large-scale textual dataset is associated with a huge number of instances and millions of features which must be discriminated between large numbers of categories. The task requires the utilization of a defined hierarchy structure and tools that automatically classify instances within the hierarchy known as Large Scale Hierarchical Text Classification (LSHTC). Predicting the labels of instances by the employed classifiers is challenging due to the high number of features. Furthermore, the existing Dimensional Reduction (DR) approaches in cooperation with the LSHTC framework are still quite inefficient. In such a problem, an effective Hierarchical Dimensional Reduction approach can be advantageous in improving the performance of the LSHTC. Therefore, in this paper, we enhance the performance of LSHTC by proposing a Multi-stage Hierarchical Dimensional Reduction (MHDR) approach based on Modified Feature Hashing (MFH) and Hierarchical Bi-Filtering (HBF) method. In addition to alleviating bad collision and result discrepancy, experimental results show that the proposed approach has achieve the best performance in terms of micro-f1 and macro-f1 by recording average scores of 58.47% and 54.77% using TD-SVM, and average scores of 51.14% and 48.70% using TD-LR, respectively. The method also achieved 11% speed-up than the approaches compared.

1. Introduction

Textual data has increased dramatically because of the development of digital technology. This data was produced by a variety of sources, such as social networks and online directories (Ado et al., 2021). High-volume is a well-known and common problem related to such kind of generated data. In addition, the currently generated text data, known as a large-scale dataset, typically faces issues about a high set of feature dimensions and large number of labelled classes. For example, several large-scale text datasets, such as Mozilla Directory, Yahoo! Directory, and International Patent Record, consist of millions of instances and billions of features spread across thousands of classes (Naik & Rangwala, 2016a). Even though classifying instances with such properties into many predefined classes has grown substantial interest in the fields of NLP and Data Science Babbar et al., (2016). But the task has become more difficult due to the continuous exponential growth of textual data and its complexity (Pilnenskiy & Smetannikov, 2020; Naik & Rangwala, 2018), while at the same time giving rise to new issues that

* Corresponding author.
E-mail address m.almaiah@ju.edu.jo (M. Almaiah) ralali@kfu.edu.sa (R. AlAli)

ISSN 2561-8156 (Online) - ISSN 2561-8148 (Print)

doi: 10.5267/j.ijdns.2024.6.012

are extremely challenging to solve using traditional methods. Furthermore, to efficiently evaluate and extract useful information, several new applications require a classification with an extraordinarily high number of instances, features, and classes (Pavia et al., 2022). This can simply be accomplished by first defining a structure taxonomy on the data (Naik & Rangwala, 2016). Typically, hierarchical structures provide an appropriate way for organizing data.

Taxonomy or hierarchy as the name implies is a well-known approach for structuring large-scale datasets in numerous real world application domains (Naik & Rangwala, 2016b; Roul & Sahoo, 2018; Guo et al., 2018), including web page classification, image classification, music genre classification, gene sequences classification, and more importantly document classification (Charuvaka & Rangwala, 2015). Classification of all these large-scale data is spine around HC problem, also known as Large-Scale Hierarchical Classification (LSHC). Currently, a considerable number of LSHC approaches have been proposed to deal with the challenges and problems that arose such as high processing time, poor label prediction, and high memory utilization (Ado et al., 2021; Naik & Rangwala, 2016a). One popular approach for lowering processing times and memory footprint is to incorporate an appropriate technique for Dimensional Reduction (DR) before training a classification model (Pilnenskiy & Smetannikov, 2020).

DR phase plays a vital role in improving the performance of the LSHTC framework by only utilizing the optimal features (Silla & Freitas, 2011). These features are found to be useful in discriminating among the classes at every internal node within the hierarchy. Many DR approaches have been proposed to tackle the problem of high-dimensionality. However, the existing DR based on FH and multi-filtering approaches integrated into the framework are unreliable due to bad collisions and result discrepancy (Pavia et al., 2022; Krishnan et al., 2019; Naik & Rangwala, 2016; Roul & Sahoo (2018). Furthermore, bad collisions are inherent problems present in current FH methods, and these collisions occur in the process of hashing features into a lower hash space. This could lead to substantial information loss, mainly when collisions occur between features with different class distributions. Multi-filtering approaches suffer from result discrepancy issues, the problem occurs due to the different rankings assigned to a single feature by the integrated filter methods (Roul & Sahoo, 2018). This issue miss-leads the multi-strategy approaches to filter out highly contributory features in the filtering process. Therefore, integrating inappropriate DR approaches into the LSHTC framework may lead to lower prediction accuracy and computational expense (Gopal & Yang, 2013; Stein et al., 2019).

To address this issue and build on advanced research in LSHTC, this paper proposes an improved framework termed Multi-stage Hierarchical Dimensional (MHDR). This aims to scale-up HC models performance by hierarchically reducing the dimensionality of the input features set.

The main contributions of this paper are as follows:

1. The paper identifies the problems of Large-scale hierarchical classification and proposes a Multi-stage Dimensional Reduction Method (MDRM) to improve LSHTC framework, enhancing its performance.
2. A Modified Feature Hashing (MFH) approach is proposed that can eliminate the collision rates between dissimilar features, thereby improving HC models performance.
3. A new bi-filtering strategy for hierarchical classification is proposed to mitigate the issue of losing most important features, thus improving filtering the process.

After the introduction, the subsequent sections of this paper are organized as follows: A literature review is placed in Section 2. The methodology and the proposed approaches are presented in Section 3. The details of the adapted datasets and experiments are explained in Section 4, followed by a results discussion, which is placed in Section 5. Lastly, the paper ends with Section 5 by giving a brief conclusion and possible future work.

2. Literature review

This section of the paper gives a comprehensive literature review of LSHC. Firstly, hierarchical classification is briefly explained, followed by a discussion on Dimensional Reduction (DR) techniques. And lastly, the section ends with a summary of some related works.

2.1 Hierarchical Classification (HC)

In response to the widespread use of hierarchies in numerous application domains, particularly text categorization, a team of researchers from the Institute of Informatics, Greece, and Laboratoire d'Informatique de Grenoble, France, have organized the Large-Scale Hierarchical Text Classification (LSHTC) challenge to increase the motivation of HC (Ramírez-Corona et al., 2016). This effort eventually resulted in a series of contests (2009, 2011, 2012, and 2014) that made it possible to establish benchmarks for the problem (Naik & Rangwala, 2016; Roul & Sahoo, 2018; Guo et al., 2018). This is accomplished by analyzing large-scale datasets with a high number of instances and classes to evaluate the methods' performance. Due to their efforts, presently, hierarchical datasets are now extensively employed across diverse application domains, including ImageNet (a system designed to index hundreds of millions of images), audio hierarchy (a system designed to organize and classify music signals), international patent classification (a system for perusing patent documents), DMOZ hierarchy (a system for categorizing and organizing web pages), and gene hierarchy (a system for classifying and organizing gene sequences).

HC, often referred to as structured classification, is a type of classification task where the classes of the instances include pre-defined hierarchical information, typically represented as a directed acyclic graph (multi-label) or a tree (single-label) (Naik & Rangwala, 2018). HC leverages the structure of the taxonomy to divide a large-scale classification task into a subset of smaller problems, one per each node within the tree taxonomy.

Two primary approaches are typically used for designing hierarchical classification models (HC model). The first strategy, known as *flat classification*, is simpler and does not take interdependence relationships between classes into account when training the model Naik & Rangwala (2018). The second approach is more complicated, and it is popularly known as *Hierarchical Classification*. It considers the interdependencies between the classes when training the model. The first approach is further divided into Local Classifier (LC) and Global Classifier (GC) (Naik & Rangwala, 2018). The LC approach considers parent-child relationships locally when training the model Ado et al., (2021). In contrast, GC trains a single global model while taking the hierarchy information into account and handles the classification problem (Stein et al., 2019). Compared to the local approach, this strategy is more computationally expensive and complex. Both approaches adopt a similar strategy when predicting the class of new instances. Furthermore, LC can be divided into three approaches based on top-down exploration of the hierarchies Stein et al., (2019): Local Classifier per Node (LCN), Local Classifier per Parent Node (LCPN), and Local Classifier per Level (LCL) (Serrano-Pérez & Sucar, 2021). More importantly, both the approaches followed a similar approach in predicting new examples.

2.2 Feature Selection

*Feature Selection (FS***)** is one of the well-known DR techniques that is usually applied to high-dimensional datasets (Cunningham & Ghahramani, 2015). The technique selects highly discriminated features from the original set of features that could improve the performance of learning algorithms. The technique is usually applied in the pre-processing stage, avoiding the curse of dimensionality. This technique tries to select and construct a suitable subset of informative features which will serve as the representative of the original features set so that it can be fed into machine learning algorithms for processing without degrading their performance (Vora & Yang, 2017; Azeez et al., 2022; Rong, Gong & Gao, 2019). FS technique essentially removes irrelevant, noisy, and redundant features from an input feature set extracted from a given dataset. There are two main approaches generally used in selecting features or subsets of features from a set of input features: the ranking approach and the subset evaluation approach. The earlier approach ranks features according to their importance (scores) using some criterion evaluation (metrics) and selects the top *k*-features. While the latter approach selects a minimum subset of informative features using subset evaluation criteria with the involvement of machine learning models. Generally, feature selection methods are broadly grouped into three main categories, namely, Filter methods, Wrapper methods, and embedded methods (Vora & Yang, 2017; Azeez et al., 2022; Rong, Gong & Gao, 2019).

2.3 Feature Hashing

Feature hashing (FH), popularly known as hashing trick or hash kernel proposed by Weinberger et al., (2009), is one of the most recent DR techniques used in scaling-up machine learning algorithms for large-scale classification Vora & Yang (2017). The technique hashed original high-dimensional input space vectors R^d into lower-dimensional space vectors $R^{k,}$ preserving Euclidean distance Azeez et al., (2022). FH is formally defined as: *"Given an input features in a d-dimension space $x \in R^d$, a mapping function f: $R^d \rightarrow R^k$ is learned to hash it into k-dimension R^k where $k << d$, R^d denotes the original vectors space while R^k is the hashed vectors space"* (Rong, Gong & Gao, 2019). The sparse high-dimensional features *x* is projected to a low-dimensional space *f(x)* via a defined map function Rong, Gong & Gao, (2019). Therefore, FH is the most effective and powerful tool in terms of enormous memory, time savings, and model size reduction (Rong, Gong & Gao, 2019). The main downside of the technique is a hash collision, whereby multiple features could be hashed together into a single index, which as a result, seriously affects the learning model performance (Rong, Gong & Gao, 2019). Even a single collision could deteriorate the performance of a classifier.

2.3 Related works

However, few research papers have proposed DR approaches for LSHTC that have been shown to achieve significant improvement. Ristoski and Paulheim (2014) proposed a method that leverages hierarchies for feature selection. Firstly, the proposed method filters out features that are redundant along hierarchy paths, and further performs pruning on the obtained features subset by considering relevance features. Wibowo and Williams (2011) use simple feature selection methods at each node in the hierarchy to adaptively select a fixed size of features subset from the upper part of every document. Zhou et al. (2011) modified the objective function of the HC-model; this forces the model to select the same set of features at each internal node rooted to the leaf node in the hierarchy. Mladenić and Grobelnik (2003) investigate six different filter-based FS methods with Hierarchical model using large-scale HC datasets. Their findings show that the best performance was recorded with *Odd* ratio among the six methods compared. In view of minimizing space requirement and processing time. Zhao et al. (2015) proposed hierarchical feature hashing termed "HESHING" to effectively reduce dimensionality of LSHC problem. They uniquely defined multiple hash functions for each node within the structured categories taxonomy, findings shown reduced in information loss that may occur because of collisions. Naik and Rangwala (2016) proposed a method by embedding feature selection in the LSHC framework. The study investigates four different filter-based methods in both adaptive and global approaches. Experiment results show that by utilizing FS methods, HC model achieved around three times speed-up and about 45% memory improvement without sacrificing classification accuracy.

3. Methods

In this section, we present the proposed approach to overcome the stated problem of dimensionality affecting the LSHC problem. Firstly, we present two distinct approaches to handle bad collisions and results discrepancy, and then we present an integrated approach that combines these approaches in a single framework. Table 1 presents the notations used in this paper and their meaning.

Table 1
Notations and meaning

$\mathcal{H}$	Original Tree Hierarchy
$\mathcal{H}_m$	Modified tree Hierarchy
$\aleph$	Set of all nodes in $\mathcal{H}$
ℓ (T)	Set of leaf nodes (categories) in $\mathcal{H}$; $\ell \subseteq \aleph$
$\aleph - \ell$	Set of internal nodes ; $\aleph - \ell \subseteq \aleph$
$\mathcal{Q}$	Root node in $\mathcal{H}$
$\mathcal{P}(n)$	Parent of node n
$\mathcal{C}(n)$	Set of all children of node n
$sib(n)$	Siblings of node n
$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{m}$	Dataset of m training instances, where $x_i \subseteq \mathcal{X}$ and $y_i \subseteq \ell$
$T_n(x)$	Total number of training instances at node n; $T_n(x) \subseteq m$
R^d	Original feature space
$\phi(x,u)$	Hash feature space ; $\phi(x,u) \in R^d$ and $u \in \aleph$
f_i	i^{th} feature
SL	Subset of relevant features selected using filter method; $SL \subseteq \phi(x,u)$

3.1 Hierarchical Classification (HC) method

The HC algorithm divides the classification problem into smaller task problems, one for each internal node of the tree hierarchy, using the hierarchical structure of the dataset that is provided. A different classifier is deployed at each internal node of the tree structure to enable HC to be performed using the considered classification learning model. The classification method, also known as the top-down strategy, proceeds greedily down the tree structure until it reaches the target leaf node yi to forecast category label yi for unknown input instance xi. The procedure begins at the root node, $\mathcal{Q}$, and proceeds recursively to select the best child nodes, (n), until it reaches the terminal node, $yi \in \ell$. The entire HC process is illustrated in Algorithm 1.

Algorithm1: Top-Down Hierarchical Classification Algorithm

```
Input: Input: instance $x_i$ , parameters $w_c^{\mathcal{Q}}$
Output: $\ell_s$
1   Set $\mathcal{Q}$ = 0    \\ Start from the root node
2   Repeat
3       $\mathcal{Q} = argmax_{\{c:(\mathcal{Q},c)\in E\}} (w_c^{\mathcal{Q}})^T \hat{x_i}$    \\ Transverse through the most weighted child
4   Until $|c(n)| = 1$        \\ Until a node with single category label is reach.
5   Return $\ell_s$           \\ Return the predicted label
```

3.2 Multi-class Top-Down LR and SVM model

For a given dataset with a known taxonomy (hierarchy) structure $\mathcal{H}$, to categorized between the children's nodes $\mathcal{C}(n)$ of any parent nodes $\mathcal{P}(n)$, a multi-class learning model is train for each of the non-leaf nodes (including root node) $n \in \aleph - \ell$ in the taxonomy. The LR and SVM objectives utilize log loss and hinge loss, respectively, to reduce empirical risk and *l_2-norm* squared term to monitor complexity and avoid overfitting the model. For training multi-class LR and SVM model in respect of i^{th} child (C_i) of the corresponding internal node n, their respective objective functions are formulated in Eqs. (1-2).

$$LR\_f_n^c = \min_{w_n^c} \left[\lambda \sum_{i=1}^{T_n} \log(1 + \exp(-(y_i)_n^c (w_n^c)^T x_i)) + \|w_n^c\|_2^2 \right] \quad (1)$$

$$SVM\_f_n^c = \min_{w_n^c} \left[\lambda \sum_{i=1}^{T_n} |1 - \ (-(y_i)_n^c (w_n^c)^T x_i)|_+ + \|w_n^c\|_2^2 \right] \quad (2)$$

where $\lambda > 0$ denotes the penalty parameter value for misclassification, $\|\cdot\|_2^2$ denotes regularization term, and T_n is the total number of instances. For each child Ci that belongs to the associated node n inside the taxonomy $\mathscr{H}$, one of the equations is solved (depending on the model employed) to obtain the optimal weight vector $W_n = [W_n^c]_{c \in C}(n)$. The multi-class learnt model for that specific node is constructed using the whole set of parameters for (n) or $W = [W_n]_{n \in N}$, whereas the TD-learned model is constructed using the total parameters for all non-leaf nodes, $W = [W_n]_{n \in N}$.

3.3 Proposed Multi-stage Hierarchical Dimensional Reduction (MHDR) Method

We first proposed two main important algorithms for dimensional reduction that are based on FH and FS, namely, MFH and HBFA. Firstly, a MFH method (present as Algorithm1) is proposed to hierarchically hash original features set into smaller hashed index. Secondly, a new HBFA (present as Algorithm2) is proposed to select the highly discrepant features from the resultant hashed space at each internal node within the hierarchy. The proposed algorithms are detail below:

3.3.1 Modified Feature Hashing (MFH) Method

We preassume that R^d is a set of variable-length features. Let say the maximum length in R^d is denoted by L_n, which should be treated as a scalable parameter. The total set of bins is given by B such that $b_z \in B$, where $z = (1,2,\ldots, m)$. All the bins size are uniform, and each bin can reach the maximum size of T, where $L_n \leq T$, and is also consider as a scalable parameter. A local index x_i of each feature in each dataset is computed using Eq. (3). The computed local index is utilized to get the hash weight w_i of each feature as can be seen in Eq. (4). We then finally apply the hash function to get the global index of each feature as shown in Eqs. (5-7).

$$x_i = ((\sum_{t_i \in f_a} t_i) - 1)\, mod(T);\; t_f \leq L_n, t_f \leq T \tag{3}$$

$$w_i = x_i \times B_n, \qquad i = 1,2,\ldots n \tag{4}$$

$$j = w_i + h(\cdot);\; w_i \to f_a \in R^d \times W \tag{5}$$

$$i = j mod(B_n); j \in R^k \tag{6}$$

$$f_{(i,j)} = \sum_{j:h(j)=i} f_a \xi(i) \tag{7}$$

where $L_i \leq T$, and B_n denotes number of bins, l_i represents each token's length in the corpus, x_i and w_i denote the local index and the term weight, respectively, and $f_{(i,j)}$ denotes the global feature index. It should be noted that those collisions which are likely to happen between not similar features are significantly prevented by the local index. However, unavoidable collisions could occur with local indexing but nevertheless, most of the bad collisions are prevented. The steps of the proposed approach are presented in Algorithm 2.

Algorithm 2: Modified Feature Hashing

```
Input:
    Original Feature Space: $R^d$
    Hash Functions: $h(\cdot)$; $\xi(\cdot)$
    Number of Bins: $B_n$
    Maximum Bin Size: $T$
Output:
    Hashed Feature Space: $R^k$
1   Begin
2   For all $f_a \in R^d$ do
3       $t_f = \sum_{t_i \in f_a} t_i$
4       $Local\_index = (t_f - 1) mod(T)$
5       $Term_{weigth} = Local\_index * B_n$
6       $Global_{index}(i) = Term_{weight} + hash(f_a) mod(B_n)$
7       $R_i^k = R_i^k + \xi(f_a)$
8       End for
9   Return $R^k$
10  End
```

3.3.2 Hierarchical Bi-strategy Filtering Approach (HBFA)

The approach first uses IG and t-test methods to compute the scores of each feature, and then normalizes the scores to make them comparatively equivalent. The IG and T-test score is computed as:

$$IG(t,c) = \sum_{c \in \{c_i, \bar{c}_i\}} \sum_{t \in \{t_k, \bar{t}_k\}} P(t,c) \times \log\left(\frac{P(t,c)}{P(t) \times P(c)}\right) \tag{8}$$

where *P(t, c)* is the conditional probability of category *c* and existence of feature *t*, *P(t)* is the probability of category containing feature *t*, *P(c)* is the probability of category *c*. $\overline{t_k}$ *and* $\overline{c_k}$ denote not presence of feature, and not presence of category, respectively.

$$t-test(t_i, c_k) = \frac{\left|\overline{tf_{ki}} - \overline{tf_i}\right|}{m_k \times s_i} \tag{9}$$

where S_i is the standard deviation within a category, C_k is the k^{th} category, k is the total number of categories, tf_{ki} is the average TF of term t_i in category k, tf_i is average TF of term t_i in the corpus. Normalization is computed by dividing each score by the maximum score obtained for each method independently. Then the V-score of each feature is computed by using its normalized scores as can be seen in Eq. (10). The normalized scores generated by each method will be used to sort and select top ranked features based on expert defined threshold τ_1. Unlike other approaches, our approach automatically set a new threshold τ_2 by considering feature similarities between the optimal features subsets obtained. We defined the new threshold τ_2 as the maximum of the minimum normalised scores, which is formulated as given in Eq. (11). Furthermore, the new defined threshold τ_2 is used to form the optimal features set by further selecting the most discrepant features from the initial selected features sets as seen in Eq. (12).

$$V_{score} = \sqrt{(IG_{score}^{N})^2 + (T_{score}^{N})^2} \tag{10}$$

$$\tau_2 = Min\left(max\begin{pmatrix} A_{IG} \leftrightarrow (A_{IG} \cap B_{TT}) \\ B_{TT} \leftrightarrow (A_{IG} \cap B_{TT}) \end{pmatrix}^{\tau_1}\right) \tag{11}$$

$$f_i = \begin{cases} 1, & V_{score}^{i} \geq \tau_2 \\ 0, & Otherwise \end{cases}, \qquad f_i \in A \cup B \tag{12}$$

where IG_{score}^{N} and T_{score}^{N} denote normalized scores of each feature based on IG and T-test, respectively, A and B are the sets of original features together with their scores computed by IG and T-test, A_{IG} and B_{TT} are the subsets of the initial features selected with their scores by IG and T-test, V_{score}^{i} is the score magnitude of each feature f_i.). Algorithm 3 presents the pseudocode of the entire process

Algorithm 3: The Proposed Bi-Strategy Filtering Approach

```
INPUT:
    $D$: Set of documents with $m$ Input-output pair $(x_i, y_i)$
    $y$: Set of label categories
    $\tau_1$: initial threshold
OUTPUT:
    $SL$: Subset of selected features
FEATURE_SCORING(D, y)        \\compute features score and ranked them
1.      L1  ←  [ ], L2  ←  [ ]
2.      R^d  ←  EXTRACT TREMS IN DOCUMENTS (D)
3.      For each $f_i$ in R^d do:
4.          A(fi, y)  ←  COMPUTE(SCORE(fi, y)) using IG algorithm
5.          APPEND(L1(A(fi, y), fi))
6.          A(fi, y)  ←  COMPUTE(SCORE(fi, y)) using t-test algorithm
7.          APPEND(L2(A(fi, y), fi))
8.      End for
9.      SORT(L1), SORT (L2)
10.     Return L1, L2
Begin
1       L1, L2  ←  FEATURE_SCORING(D, y)
2.      FS1  ←  $\tau_1$% {L1} = {f1, ...fq}    \\ Initial sequence of features selected using IG
3       FS2  ←  $\tau_1$% {L2} = {f1, ...fn}    \\ Initial sequence of features selected using t-test
4.      j  ←  q
5.      for fi in [SORT.descend(FS1)]    \\ Compute V-score of each feature in FS1
6.          Normalize (fi)     \\ by dividing it with the max IG score
7.          If fi ∈ {FS1} ∩{FS2}
8.              V_score(1)  ←  COMPUTE(V_score_of fi using Eq. (10))
9.          Break
10.     for fj in [SORT.descend(FS2)]    \\ Compute V-score of each feature in FS2
11.         Normalize (fj)     \\ by dividing it with the max t-score
12.         If fi ∈ {FS1} ∩{FS2}
13.             V_score(2)  ←  COMPUTE(V_score_of fj using Eq.(10))
14.         Break
15.     $\tau_2$  ←  MIN(V_score(1), V_score(2))    \\ Set new threshold
16.     APPEND [SL, {FS1∩FS2}]
17.     for fi in [{L1 ∪ L2} – {SL}] do    \\ Reselect most optima features based on new threshold
18.         If V_score(fi) >= $\tau_2$
19.             APPEND[SL, (fi)]
20.     End for
21      Return SL
End
```

Algorithm 4 presents the proposed MHDR that integrates multi-stage dimensional reduction (FH and FS) into the LSHTC framework. The order in which the stages (presented below) are executed is imperative to obtain the maximum hierarchical classification performance improvement. The algorithm consists of three different main stages. The first stage utilizes MFH method (presented as Algorithm2) for hierarchically hashing original features set into smaller hashed index. The second stage utilizes HBFA (presented as Algorithm3) for selecting the most informative features from the resultant hashed space at each non-leaf within the hierarchy. Let $\mathcal{H}_m$ be the hierarchy:

Step 1: After the hierarchy $\mathcal{H}_m$ is fed into the proposed method, the proposed method employs the concept of Hierarchical Feature Hashing (HFH) $\psi(f, y)$ to hierarchically map the features into hashed space dimensions. The approach applies Algorithm 2 (MFH algorithm) on each internal node $\aleph - \ell$ in $\mathcal{H}_m$.

Step 2: Next is the FS step, which aims to select a subset of highly relevant features between siblings' categories. This step reduces the hierarchical error rate and improves the learning model performance. The proposed approach utilizes Algorithm 3 (HBFA algorithm) on the produced hashed space $\phi(f, u)$ of each internal node, $\aleph - \ell$ to effectively select discriminant features. Unlike other approaches that select the exact features set at each node; the proposed approach selects different variable features set at each node.

Step 3: Is known as the learning and prediction stage. The stage hierarchically classifies Large-scale datasets into pre-defined labels. For a given taxonomy, the reduced features set *SL* (produced in the previous stage) is used to train multi-class learning models for each non-leaf nodes $n \subseteq \aleph - \ell$ in $\mathcal{H}_m$. Let *n* be an internal node containing only a small set of discrepant features, a multi-class learning model is trained on the node to categorize between its children's nodes $\mathcal{C}(n)$.

Algorithm 4: Proposed Multi-Stage Hierarchical Dimensional Reduction Approach

```
INPUT:    Original Hierarchy $\mathcal{H}$; Input-output pair $(x_i, y_i)$
OUTPUT:   Weight Vectors of the Learned Model $w = [w_i], where(\ i = 1,..n),\ n \in \aleph - \ell$
1.    $w = [\ ]$        \\ Initialisation
2.    \\ stage 1: Feature Hashing
3.    For every $(f_i, y_i)$ with $y_i \in \aleph$, $f_i \in$ F, do
4.        hash $f_i$ using Algorithm1
5.    End for
6.    Stage 2: Feature Selection
7.    For every $f_i \in \phi(x, u)$ do
8.        Select top k most discriminate features "$SL$" using Algorithm2
9.    End for
10.   \\Stage 3: model learning on the resultant reduced features space SL
11.   For every $n \in \aleph - \ell$ do        \\ to learn models for classifying $\mathcal{C}(n)$ at $\mathcal{P}(n)$
12.       If $n \notin \ell$ then
13.           If multi-class learning model = TD-LR          \\ Top-down Logistic Regression
14.               Train multi-class LR model on the reduced feature space "$SL$" using eq. (1)
15.               Update.[w, $w_n$]             \\ updating model weight vectors
16.           End if
17.           If multi-class learning model = TD-SVM       \\ Top-down Support Vector Machine
18.               Train multi-class SVM on the reduced feature space "$SL$" using equ. (2)
19.               Update.[w, $w_n$]             \\ updating model weight vectors
20.           End if
21.       End if
22.   End for
23.   Return w
```

Unlike other approaches that select the same feature at each internal node, our approach selects different features at each internal node which is refer to as adaptive FS.

4. Dataset and Experiment Evaluation

Summary of the datasets and the implantation settings used are briefly explained in this section.

4.1 Dataset

The experimental evaluation is conducted using three benchmark textual datasets including International Patient Classification (IPC), 20Newsgroup (20NG), and Directly Mozilla (DMOZ-small). 20NG is formed from Usenet Newsgroup's documents collection Dhillon (2003). The newsgroups names represent the content forum, and we hierarchically organize them by manually imposing parent-child relationships. IPC offers a hierarchy structure of independent language symbols for patents classification and patent applications based on various technology areas to which they relate Fall et al., (2003). DMOZ is generally a multi-lingual World Wide Web links for open content directory, it is a multiple web document structured into numerous classes using the tree structure. DMOZ-small is a variant of DMOZ-dataset and has also been released as part of 2010 and

2012 large scale challenge known as LSHTC 9 Partalas et al., (2015). Table 2 summarizes the important characteristics of the adapted datasets. All the datasets are shown to be high-dimensional.

Table 2
Characteristics of the datasets

Dataset	#training Doc	# Testing Doc	#Features	#Classes	#Nodes	Height
20NG	11269	7505	61188	20	28	4
IPC	46324	28926	1123497	451	553	4
DMOZ-small	6323	1858	51033	1139	2388	6

4.2 Experiment settings

The standard *train-test* partitions available in the large-scale challenging website (http://lshtc.iit.demokritos.gr/LSHTC3_oracleUpload) is utilized for all the experiments, see Table 2. We further split the training set into 90% as the main training set (to train Top-Down models) and 10% as the small validation set (to fine tune the regularization parameter). We further split the training set into 90% as the main training set for training the employed TD-models (TD-SVM and TD-LR) and 10% as the small validation set for fine-tuning the regularization parameter. We train each model by choosing a misclassification parameter value (λ) in the set that ranges from 0.01 to 1000 in increments of "*10 (that is multiple of ten)", and then utilize the validation dataset to select the best parameter value. For training of the HC, we set the learning models to compulsory leaf node prediction. For FH, we use the extended version (MurmurHash3) as the hash function. As normal convention, we used hash bit size ranging from 5-bit to 28-bit hash in 1 step increment to investigate the performance of the proposed MFH. The initial experiments demonstrated no noticeable performance improvement beyond 2^{14}. Thus, we set the hash size to this value for the subsequent experiments. The percentage of collisions is obtained by considering only the number of features that have non-zero weights. For FS, we select the best initial threshold using the validation dataset by varying the size of features ranging from 1% to 80% of the entire features set in steps of 20. From the preliminary results, we notice no significant performance improvement beyond the 60% threshold; thus, this value is selected as the initial threshold. Finally, to validate the proposed approaches, the standard twins' metrics (micro-f1 and macro-f1) are used. These twins' metrics are usually used for evaluating the performance of HC. Micro-f1 gives equal importance to every document, while macro-f1 gives equal importance to every category. They are calculated using the following equations:

$$Micro-F1=\frac{2PR}{P+R} \tag{13}$$

$$Macro-F1=\frac{1}{|\ell|}\sum_{n=1}^{|\ell|}\frac{2P_nR_n}{P_n+R_n} \tag{14}$$

P and *R* can be obtained using the following equations:

$$P=\frac{\sum_{n=1}^{|\ell|}TP_n}{\sum_{n=1}^{|\ell|}TP_n+FP_n} \tag{15}$$

$$R=\frac{\sum_{n=1}^{|\ell|}TP_n}{\sum_{n=1}^{|\ell|}TP_n+FN_n} \tag{16}$$

where $TP_{n,}$ (True Positives) are positive documents correctly classified as positive. $FP_{n,}$ (False Positives) are negative documents classified as positive, and FN_n (False Negatives) are positive documents classified as negative, for distinct classes in ℓ, P denotes precision, R denotes recall, P_n and R_n are the precision and recall for leaf node *n*, and$|\ell|$ denotes a total number of labels or classes.

5. Results Discussion

We first investigate the performance of our proposed MFH and HBFA by comparing it with their counterparts, and then we present the performance of the proposed MHDR with the existing approaches. Fig. 1 shows the performance of our proposed MFH and that of Conventional Feature Hashing (CFH) with varying length of bit size (in 1 step interval for hash range values), starting from 5 to 28 hash bits. To generalize our findings, we compared both the methods on three different text datasets which comprise of 20NG, IPC and DMOZ dataset. We record the performance based on Micro-F1 and Macro-F1 scores. From the sub-figures in Fig. 1, we can see that our proposed MFH outperformed CFH in all the detailed comparisons with minor performance improvement of 3% averagely. This is because our approach mitigates bad collisions that do occur between dissimilar features. MFH can reduce bad collisions to a certain degree by utilizing the concept of term weight. Moreover, both the methods return almost the same performance from 2^{10} downward and 2^{20} upward hash bits, this is due to presence of unavoidable bad collisions between 0 to 2^{10} hash bits (almost 100%) and presence of very few bad collisions at or beyond 2^{20} (almost 0%). Specifically, from sub-figures (C and D), the classification model's performance starts sacrificing from 2^{18} hash bits while the other sub-figures (A, B, E and F) depict that the performance starts sacrificing from 2^{14} hash bits. This variation is due to differences in dimensionality of the datasets, IPC has a very high number of features compared to 20NG and DMOZ. Therefore, dimensional reduction using feature hashing approaches need approximately 2^{14} to 2^{18} hash bits (depending on the dimensionality of the dataset) to achieve competitive performance.

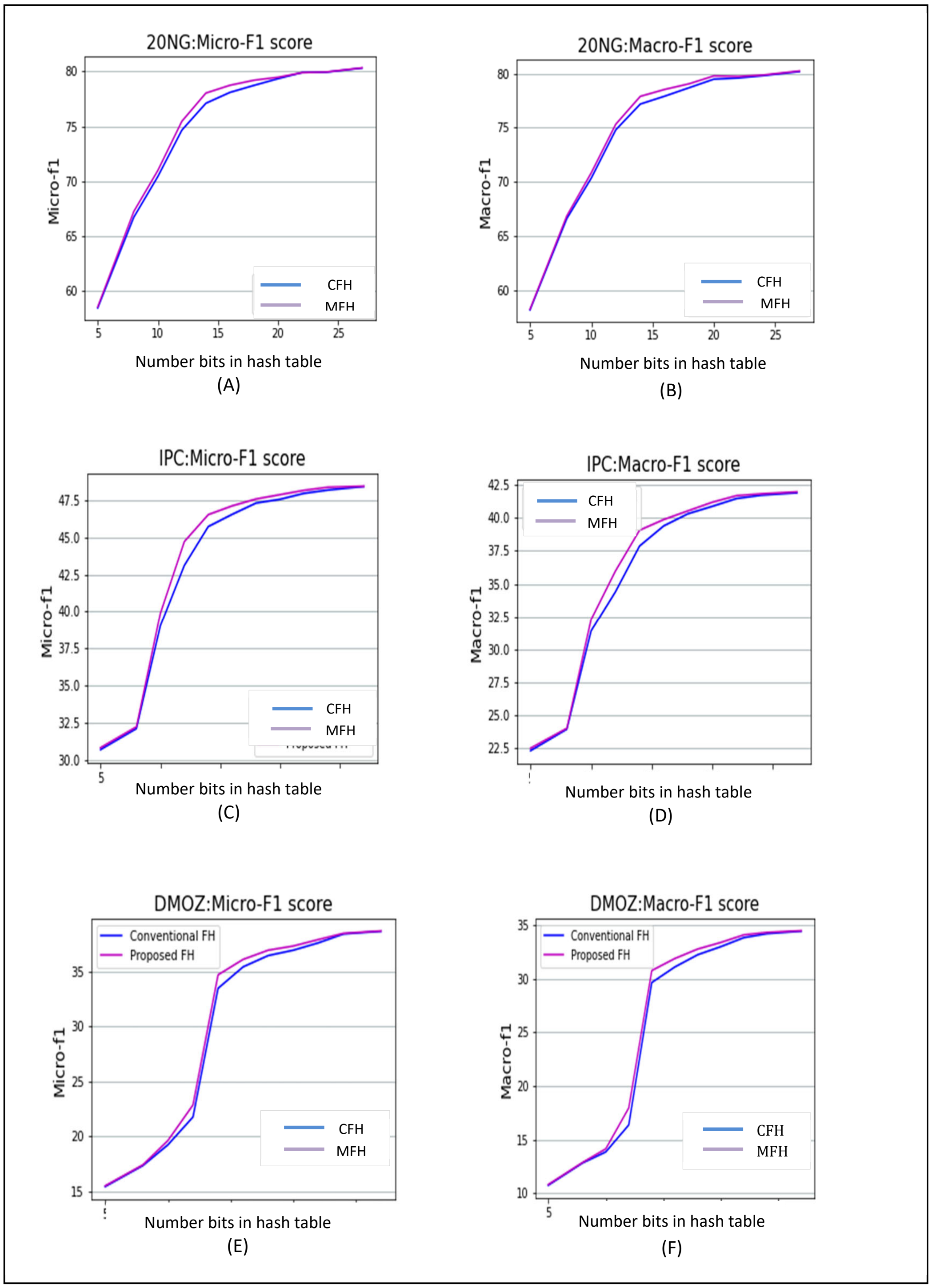

Fig. 1. Performance comparison of MFH and CFH with varying hash bit size using TD-LR on three different LHTC datasets.

To validate the proposed HBFA, we compared it with single strategy filter methods (*TD-LR+IG* and *TD-LR+t-test)* using 20NG, IPC and DMOZ datasets. The performance is recorded using a varying percentage of features starting from 1% to 80%

(in 19% interval for the first step and 20% interval for the subsequent steps). Table 2 shows the micro-f1 and macro-f1 scores based on the TD-LR model by integrating with the considered FS based filter methods (IG and t-test), and the proposed HBFA. The best performances are face bolded. We can notice that all the methods showed acceptable performance results compared to utilizing a higher percentage of features for all the datasets. Overall, the proposed HBFA has slightly better performance over the single strategy methods. HBFA tends to utilize both the advantages of IG and t-test to remove noisy and irrelevant features based on the reduction objective acquired from a sparse data leaf node, which might negatively affect the performance of a model. The *TD-LR+t-test* method recorded poor performance because of the discrepancy between the theoretical strategy of the method and statistical properties obtained from the sparse data node. Both the methods achieve worse performance when the percentage of features selected is less than 20%, and achieve higher performance when the selected features are fixed at 70%. It has been noticed that at less than 40% of features, the filter methods eliminate not only the irrelevant and noisy features but also a high number of informative features that positively contribute to discriminating the categories.

Table 2
Performance comparison based on macro-f1 and macro-f1 of the proposed method with varying percentages of features using TD-LR. The percentage on the top of the table is the size of the features. The best performance is face bolded

dataset	Methods	Micro-F1					Macro-F1				
20NG		1%	20%	40%	60%	80%	1%	20%	40%	60%	80%
	TD-LR+IG	**59.78**	77.36	78.54	78.92	80.19	**59.23**	77.17	78.00	78.52	79.63
	TD-LR+t-test	22.45	68.62	76.94	78.26	80.14	21.45	67.57	76.08	77.97	79.62
	HBFA	59.13	**78.51**	**79.00**	**80.56**	**80.24**	58.46	**77.89**	**78.32**	**79.76**	**79.78**
IPC	**TD-LR+IG**	38.78	45.40	46.00	47.30	47.52	25.78	38.73	39.68	40.15	**41.70**
	TD-LR+t-test	29.74	38.61	42.67	44.82	48.93	17.74	34.64	36.92	38.75	39.24
	HBFA	**40.13**	**50.04**	**51.94**	**51.66**	**52.00**	**33.87**	**40.34**	**40.20**	**40.20**	41.34
DMOZ-small	**TD-LR+IG**	**17.65**	**31.24**	32.07	33.40	34.85	12.22	18.87	17.03	19.35	19.80
	TD-LR+t-test	12.71	28.19	31.00	32.80	**38.97**	4.80	16.75	17.83	20.14	23.20
	HBFA	17.33	28.52	**33.66**	**34.51**	35.00	**15.03**	**22.34**	**23.54**	**27.72**	**29.16**

Table 3 shows the TD-LR and TD-SVM models' performance comparison of HFS, HFH, Baseline, and the proposed MHDR. The highest scores are face bolded, and the values in brackets are the maximum improvement against the baseline. We can see from the table that the proposed MHDR gives the best performance across all the datasets (except TD-SVM's macro-f1 score of the NG dataset, which is known to have very few leaf nodes or categories). A significant improvement of approximately 3% is achieved with the proposed method against all features on all the datasets. In comparison with single-stage (HFS and HFH) an improvement of approximately 1% to 2% is noticed on all the datasets except the micro-f1 score of the DMOZ dataset, where a more comprehensive improvement of 4% to 5% approximately is noticed. We also observe that HFS achieves the worst performance on all the datasets except the micro-f1 score of 20NG and IPC datasets. Therefore, the proposed multi-stage method outperforms the single-stage methods because of its ability to blend the advantages of the two techniques (FS and FH technique). These advantages include: (i) HFS has the advantage of selecting optimal informative features in every node of the taxonomy, (ii) the redundancy introduced by HFH drastically reduces information loss due to hash collision. Lastly, by carefully studying the performance improvement over all features, we can see that dataset with rare categories (such as IPC and DMOZ-Small) benefit significantly from dimensional reduction methods, especially with the proposed multi-stage method.

Table 3
Performance comparison of proposed method, its counterparts, and baseline based on micro-f1 and macro-f1 using TD-LR and TD-SVM

Dataset	**Metric**	**HFS**		**HFH**		**Proposed MHDR**		**Baseline (All features)**	
		TD-LR	**TD-SVM**	**TD-LR**	**TD-SVM**	**TD-LR**	**TD-SVM**	**TD-LR**	**TD-SVM**
20NG	Micro-F1	80.56	73.21	81.06	74.77	**82.84 (+2.84)**	**75.61 (+2.59)**	80.00	73.02
	Macro-F1	79.98	72.80	80.65	**75.23 (+2.37)**	**82.20 (+2.03)**	75.19	80.17	72.68
IPC	Micro-F1	51.66	51.74	52.13	51.64	**53.11 (+0.87)**	**52.86 (+1.59)**	52.24	51.27
	Macro-F1	40.49	41.00	41.38	41.70	**42.18 (+1.06)**	**42.13 (+1.31)**	41.12	40.82
DMOZ-small	Micro-F1	34.51	34.44	37.26	34.29	**39.48 (0.62)**	**35.58 (+1.54)**	38.86	34.04
	Macro-F1	27.72	27.04	28.17	27.18	**29.03 (+1.26)**	**28.79 (+2.18)**	27.77	26.61

Figs. (2-4) show the comparisons of pre-processing time, learning time, and total running time. It should be noticed that Pre-processing time means the time taken for feature vectorization plus the time required to filter out irrelevant features. Learning time is the training and testing time required for the learning model. In contrast, total running time means the combined time for the pre-processing and learning time. From Fig. 2, we can see that BM (all features) takes the lowest amount of time since it does not need any additional time for computing features' importance, it only requires time for feature vectorization, which is very minimal. On the other hand, HFS takes the most extended amount of time since it requires additional time to filter out irrelevant features apart from feature vectorization time. Compared with HFS, the proposed method takes less time despite needing extra time to filter out irrelevant features. This is because it is sufficient to vectorize by computing TF and IDF using the feature hashing trick. The hashing vectorization time, which is constant per non-zero input feature, is approximately zero (≈0). Moreover, FH does not only reduce features dimension but also speeds up pre-processing time. From Fig. 3, the proposed MHDR takes shorter training and testing time due to the smaller features dimension, which needs to be considered during learning. However, there is no noticeable improvement for the datasets with a small number of instances, such as 20NG. Moreover, improvement is noticeable for datasets with a larger number of instances, such as IPC. For example, the proposed method reduces the model learning time of IPC from 4143 seconds to 1362 seconds. Fig. 4 shows the proposed method archives the shortest time on IPC datasets, whereas the BM method takes the shortest total time on the 20NG and DMOZ-small datasets. This shows that we can only observe significant improvement if the datasets are sparse and high-dimensional.

Two major observations were drawn from this experiment. Firstly, the effectiveness of the proposed approach can be noticed when the dataset is extremely large. Secondly, a significant improvement is noticed with DMOZ-small, although it has fewer instances and features than 20NG. This noticeable improvement is due to the larger number of categories present in DMOZ-small, resulting in larger parameter vectors (≈300M) than 20NG (≈900k). Therefore, exploiting the proposed HMDR in LSHTC benefits in achieving higher performance and lower running time.

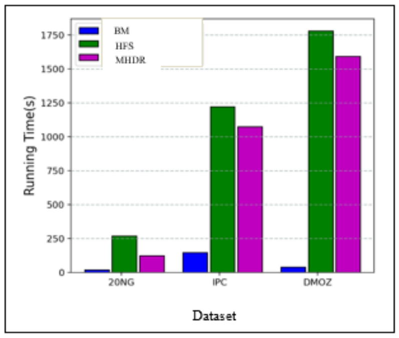

Fig. 2. Pre-processing time comparison based on TD-LR model using 20NG, IPC, and DMOZ-small Dataset

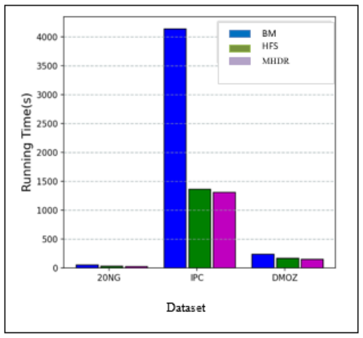

Fig. 3. Learning time comparison based on TD-LR model using 20NG, IPC, and DMOZ-small dataset

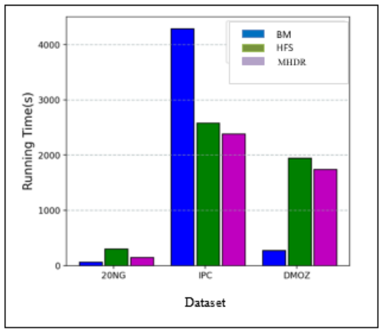

Fig. 4. Total time comparison based on TD-LR model using 20NG, IPC, and DMOZ-small dataset

6. Conclusion and future work

To conclude this study, we have proposed a Hierarchical Multi-stage Dimensional Reduction approach for LSHTC problems that exploits the advantages of feature hashing and feature selection techniques. In the process of achieving this, we have provided solutions to address the two well-known issues associated with the existing dimensional reduction methods, namely, bad collisions and results discrepancy. Firstly, we present an MFH approach based on term weight to minimize the rate of bad collisions. Secondly, for solving the results discrepancy issue, we present HBFA that selects the most important features based on IG and T-test. Lastly, we present an integrated approach to handle the compound issues together faced by HC models. Experimental results show that a significant improvement of approximately 3% and 2% is achieved with the proposed HMDR against BM and existing single-stage (HFS and HFH) approaches, respectively. It also records the lowest running time with a difference of about 300s and 1500s.

In further work, the following task is recommended for further investigation: (i) the approach should be investigated in Multi-Task Learning for Large-scale Problems**.** (ii) Investigating a distributed approach to extend the proposed method in order to improve the running speed and scalability will be beneficial.

Acknowledgment

This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia (Grant No. KFU241143).

References

Ado, A., Deris, M. M., Samsudin, N. A., & Aliyu, A. (2021). A new feature filtering approach by integrating IG and T-test evaluation metrics for text classification. *International Journal of Advanced Computer Science and Applications*, *12*(6).

Ado, A., Samsudin, N. A., & Deris, M. M. (2021, July). A new feature hashing approach based on term weight for dimensional reduction. In *2021 International Congress of Advanced Technology and Engineering (ICOTEN)* (pp. 1-7). IEEE.

Azeez, N. A., Lawal, A. O., Misra, S., & Oluranti, J. (2022). Machine learning approach for identifying suspicious uniform resource locators (URLs) on Reddit social network. African Journal of Science, Technology, Innovation and Development, 14(6), 1618-1626.

Babbar, R., Partalas, I., Gaussier, E., Amini, M. R., & Amblard, C. (2016). Learning taxonomy adaptation in large-scale classification. *Journal of Machine Learning Research*, *17*(98), 1-37.

Charuvaka, A., & Rangwala, H. (2015). Hiercost: Improving large scale hierarchical classification with cost sensitive learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part I 15* (pp. 675-690). Springer International Publishing.

Cunningham, J. P., & Ghahramani, Z. (2015). Linear dimensionality reduction: Survey, insights, and generalizations. *The Journal of Machine Learning Research*, *16*(1), 2859-2900.

Dhillon, I. S., Mallela, S., & Kumar, R. (2003). A divisive information theoretic feature clustering algorithm for text classification. The Journal of machine learning research, 3, 1265-1287.

Fall, C. J., Törcsvári, A., Benzineb, K., & Karetka, G. (2003, April). Automated categorization in the international patent classification. In Acm Sigir Forum (Vol. 37, No. 1, pp. 10-25). New York, NY, USA: ACM.

Gopal, S., & Yang, Y. (2013, August). Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 257-265).

Guo, Y., Liu, Y., Bakker, E. M., Guo, Y., & Lew, M. S. (2018). CNN-RNN: a large-scale hierarchical image classification framework. *Multimedia tools and applications*, *77*(8), 10251-10271.

Krishnan, R., Samaranayake, V. A., & Jagannathan, S. (2019). A hierarchical dimension reduction approach for big data with application to fault diagnostics. *Big Data Research*, *18*, 100121.

Mladenić, D., & Grobelnik, M. (2003). Feature selection on hierarchy of web documents. *Decision support systems*, *35*(1), 45-87.

Naik, A., & Rangwala, H. (2016a). Filter based taxonomy modification for improving hierarchical classification. *arXiv preprint arXiv:1603.00772*.

Naik, A., & Rangwala, H. (2016b). Embedding feature selection for large-scale hierarchical classification. In *2016 IEEE International Conference on Big Data (Big Data)* (pp. 1212-1221). IEEE.

Naik, A., & Rangwala, H. (2018). *Large scale hierarchical classification: state of the art*. Springer International Publishing.

Partalas, I., Kosmopoulos, A., Baskiotis, N., Artieres, T., Paliouras, G., Gaussier, E., ... & Galinari, P. (2015). Lshtc: A benchmark for large-scale text classification. *arXiv preprint arXiv:1503.08581*.

Pavia, S., Piraino, N., Islam, K., Pyayt, A., & Gubanov, M. N. (2022). Hybrid Metadata Classification in Large-scale Structured Datasets. *J. Data Intell.*, *3*(4), 460-473.

Pilnenskiy, N., & Smetannikov, I. (2020). Feature selection algorithms as one of the python data analytical tools. *Future Internet*, *12*(3), 54.

Ramírez-Corona, M., Sucar, L. E., & Morales, E. F. (2016). Hierarchical multilabel classification based on path evaluation. *International Journal of Approximate Reasoning*, *68*, 179-193.

Ristoski, P., & Paulheim, H. (2014). Feature selection in hierarchical feature spaces. In *Discovery Science: 17th International Conference, DS 2014, Bled, Slovenia, October 8-10, 2014. Proceedings 17* (pp. 288-300). Springer International Publishing.

Rong, M., Gong, D., & Gao, X. (2019). Feature selection and its use in big data: challenges, methods, and trends. Ieee Access, 7, 19709-19725.

Roul, R. K., & Sahoo, J. K. (2018). Text categorization using a novel feature selection technique combined with ELM. In *Recent Findings in Intelligent Computing Techniques: Proceedings of the 5th ICACNI 2017, Volume 3* (pp. 217-228). Springer Singapore.

Silla, C. N., & Freitas, A. A. (2011). A survey of hierarchical classification across different application domains. *Data mining and knowledge discovery*, *22*, 31-72.

Stein, R. A., Jaques, P. A., & Valiati, J. F. (2019). An analysis of hierarchical text classification using word embeddings. *Information Sciences*, *471*, 216-232.

Vora, S., & Yang, H. (2017, July). A comprehensive study of eleven feature selection algorithms and their impact on text classification. In 2017 Computing Conference (pp. 440-449). IEEE.

Weinberger, K., Dasgupta, A., Langford, J., Smola, A., & Attenberg, J. (2009, June). Feature hashing for large scale multitask learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 1113-1120).

Wibowo, W., & Williams, H. E. (2002, November). Strategies for minimising errors in hierarchical web categorisation. In *Proceedings of the eleventh international conference on Information and knowledge management* (pp. 525-531).

Zhao, F., Huang, Y., Wang, L., & Tan, T. (2015). Deep semantic ranking based hashing for multi-label image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1556-1564).

Zhou, D., Xiao, L., & Wu, M. (2011). Hierarchical classification via orthogonal transfer.