applied sciences

MDPI

Article

UFODMV: Unsupervised Feature Selection for Online Dynamic Multi-Views

Fawaz Alarfaj [1], Naif Almusallam [1,*], Abdulatif Alabdulatif [2], Mohammed Ahmed Alomair [3], Abdulaziz Khalid Alsharidi [4] and Tarek Moulahi [5]

1 Department of Management Information Systems (MIS), College of Business Administration, King Faisal University (KFU), Al-Ahsa 31982, Saudi Arabia
2 Department of Computer Science, College of Computer, Qassim University, Buraydah 52571, Saudi Arabia
3 Department of Quantitative Methods, School of Business, King Faisal University, Al-Ahsa 31982, Saudi Arabia
4 Department of Mathematics and Statistics, College of Science, King Faisal University, Al-Ahsa 31982, Saudi Arabia
5 Department of Information Technology, College of Computer, Qassim University, Buraydah 52571, Saudi Arabia
* Correspondence: nalmuslem@kfu.edu.sa

Abstract: In most machine learning (ML) applications, data that arrive from heterogeneous views (i.e., multiple heterogeneous sources of data) are more likely to provide complementary information than does a single view. Hence, these are known as *multi-view data*. In real-world applications, such as web clustering, data arrive from diverse groups (i.e., sets of features) and therefore have heterogeneous properties. Each feature group is referred to as a particular view. Although multi-view learning provides complementary information for machine learning algorithms, it results in high dimensionality. However, to reduce the dimensionality, feature selection is an efficient method that can be used to select only the representative features of the views so to reduce the dimensionality. In this paper, an unsupervised feature selection for online dynamic multi-views (UFODMV) is developed, which is a novel and efficient mechanism for the dynamic selection of features from multi-views in an unsupervised stream. UFODMV consists of a clustering-based feature selection mechanism enabling the dynamic selection of representative features and a merging process whereby both features and views are received incrementally in a streamed fashion over time. The experimental evaluation demonstrates that the UFODMV model has the best classification accuracy with values of 20% and 50% compared with well-known single-view and multi-view unsupervised feature selection methods, namely OMVFS, USSSF, and SPEC.

Keywords: dynamic multi-view; unsupervised feature selection; model

check for updates

Citation: Alarfaj, F.; Almusallam, N.; Alabdulatif, A.; Alomair, M.A.; Alsharidi, A.K.; Moulahi, T. UFODMV: Unsupervised Feature Selection for Online Dynamic Multi-Views. *Appl. Sci.* **2023**, *13*, 4310. https://doi.org/10.3390/app13074310

Academic Editors: Chunhua Su, Keping Yu, Celestine Iwendi and Thippa Reddy Gadekallu

Received: 5 February 2023
Revised: 22 March 2023
Accepted: 26 March 2023
Published: 28 March 2023

1. Introduction

Data that arrive from heterogeneous views (i.e., multiple heterogeneous sources of data) are more likely to provide complementary information for machine learning (ML) models than does a single view [1]. Each instance in multi-view data has different groups of features, meaning that different views have different representations of the same set of instances, and each view comprises a group of features. For example, in medical applications, it is better to look at patients' healthcare conditions (i.e., instances) from different groups (i.e., views) of laboratory tests (i.e., features) for a more precise medical diagnosis. However, as not all features in different views are representative, this exacerbates the problem of high dimensionality. Indeed, the non-representative features not only increase the time complexity of the ML algorithms; they also decrease the accuracy of the classification [2]. Additionally, these non-representative features also waste storage space [3]. *Feature selection* (FS) is an efficient method that can reduce the dimensionality of the original feature set. This is achieved through the selection of a reduced set of features from the original feature space that can represent the entire feature set.

Traditional unsupervised feature selection methods [4–7] assume that data are independent and identically distributed (iid). However, this assumption does not apply when there are heterogeneous families/views of features, as multi-view data has a dependency/correlation between the views due to the representation of the same instance. Additionally, these traditional methods are unable to exploit the correlation between views that are concatenated, as data could lose their meaning if concatenated. For these two reasons, they are inappropriate for multi-view learning [8]. There are three main approaches for the feature clustering of multi-view data: *concatenation*, *distribution*, and *centralisation* [9]. With the concatenation method suggested in [4–7,10,11], all the data views are combined into a single matrix, ignoring the heterogeneous nature of the multi-view data. However, data lose their actual meaning if combined, resulting in poor feature selection. With the distribution method, features are selected from each view independently [12,13]. However, this method focuses on local feature selection at each view and it does not correlate the features of all the views, thereby possibly resulting in redundant features. Finally, the centralisation method [14] simultaneously considers the correlation of the features of all the views, which produces a better selection of representative features.

Few unsupervised feature selection methods are available for multi-view learning [8,15–18]. However, all the existing methods assume that the number of the views is static, meaning that they are complete and pre-existing and no new additional views can be added. However, this assumption is not appropriate for real applications where new views and features can be added at any given time. Additionally, the number of instances can increase. To the best of our knowledge, this is still an open problem that has not been properly addressed, which motivated us to investigate this problem further and contribute to the development of innovative methods. This paper proposes a method that overcomes the limitations of existing unsupervised feature selection methods for multi-view applications for dynamic multi-view applications in an online environment so that the number of both instances and views can increase overtime and be incrementally clustered.

1.1. Problem Statement

The problem we aim to solve here is the selection of a set of representative features that can appropriately approximate the original feature space. The challenge is associated with the incremental updating of the set of selected representative features amongst all of the views clustered to date so that: (i) the number of views, which are different sets of features, increases over time so that existing instances can be represented by new additional views; (ii) the number of instances can also increase (i.e., online learning), and the new instances are shown in all the current views; and (iii) different instances are shown by different combinations of views. This paper contributes to (i) and (ii), and (iii) will be investigated in future work.

There are two possible applications that we can describe to illustrate the problem we are addressing here. A healthcare application that deals with the prediction of early symptoms of a heart attack provides a good illustrative example. We could have a group of patients (i.e., instances) with different groups/views of medical tests (i.e., features). Let us assume that we initially have two groups/views of medical tests, namely medical tests belonging to a heart view and medical tests belonging to a physical therapy view (Figure 1a). We then have three possible scenarios: (1) new patients undergo all the medical tests of existing views (Figure 1a); (2) the current patients undergone additional medical tests, which relate to diabetes (additional view) (Figure 1b); and (3) different patients can undergo different medical tests presented in different views, but not all of them. In this paper, we have addressed the first and second scenarios, while the third one will be addressed in future work. This application example shows that both the number of views and the number of instances can increase at any time.

The other application we consider is web page clustering, where a set of pages (i.e., instances) is presented in both text view and image view. The set of pages might be updated later on, in terms of views, with an additional video view. Additionally, the number of web pages can increase.

The proposed method is not limited to the above-mentioned examples but is also applicable to broad applications of multi-view data where the instances can be represented by different feature spaces. Example applications could be web text translation by different languages [17], visual concept recognition [16], and gene selection [19].

Formally, we consider $\{X^{(v)}, v = 1, 2, \ldots\}$ a dataset of different views such that views can increase over time. Following the online learning, $N = \{inst_1, inst_2, \ldots\}$ is a group of instances in n_v views, where $X^{(v)} \in R^{(N_{inst} \times D_v)}$ and D_v is the dimension of the instances in the v^{th} view.

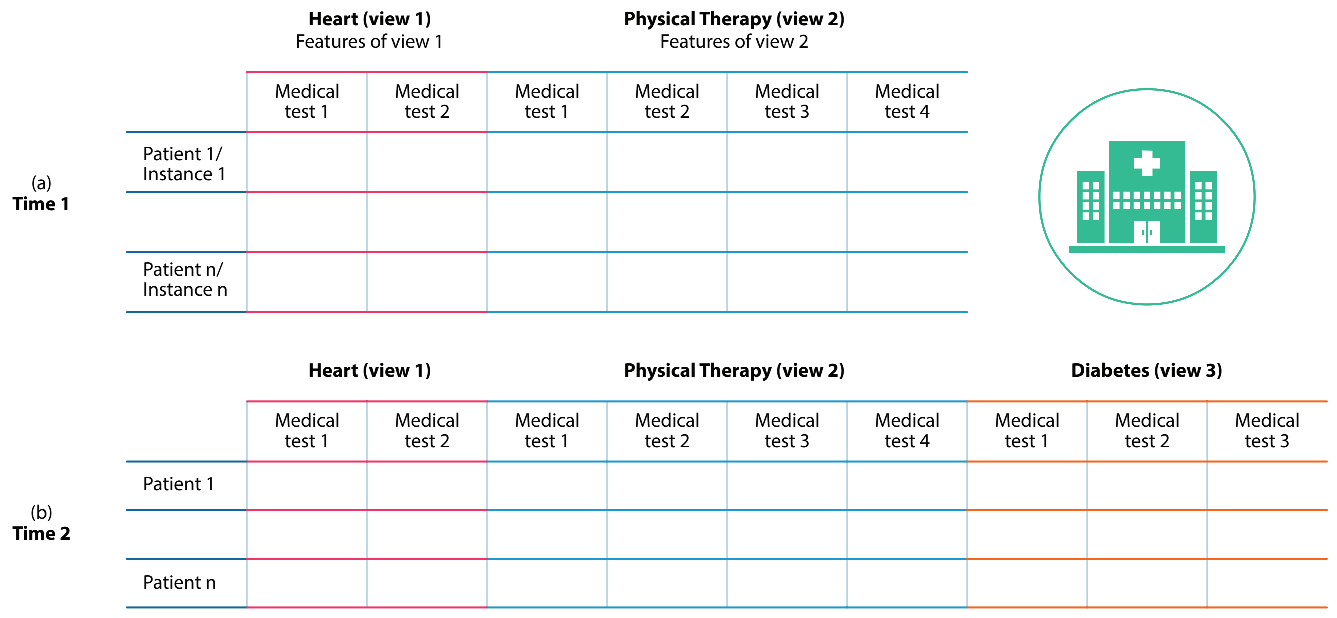


Figure 1. Healthcare application that (**a**) demonstrates two clinics, namely heart and physical therapy, with their own features and (**b**) demonstrates a new healthcare clinic, namely diabetes, which has an additional group of features (i.e., view).

1.2. Contribution

The main contributions of this paper are:

- An unsupervised feature selection for online dynamic multi-views (UFODMV) method is proposed that includes clustering-based technique along with an innovative merging mechanism that provides an efficient stream dynamic feature selection process with multi-views. UFODMV proposes a novel merging process that rapidly improves the dynamic feature selection process while significantly reducing the number of selected representative features in a hierarchical manner with a predefined reduction size.
- UFODMV is evaluated and compared with three well-known unsupervised feature selection methods, namely OMVFS, UFSSF, and SPEC. In this evaluation, three real and well-known multi-view datasets were used, including the Fox News, Caltech-7, and Multiple Features Handwritten datasets. UFODMV significantly outperformed the benchmarked methods when selecting a set of representative features that was able to classify the data accurately. UFODMV shows a significant classification accuracy of around 50% compared with the benchmarked methods.

The rest of the paper is organized as follows. Section 2 discusses the related works. The proposed method is presented in detail in Section 3. Experimental results and performance evaluations are shown in Sections 4–6. Finally, Section 7 concludes the paper.

2. Background

In the era of big data, the dimensions of data have increased significantly. In particular, the number of features can increase such that not all features are *representative* for learning machines. In addition, feature *redundancy* is more likely to occur. Various researchers have proposed feature selection as an efficient technique that can help to address the aforementioned challenges. The feature selection process comprises (i) subset generation, (ii) subset evaluation, (iii) stopping criterion, and (iv) result validation [20]. Subset generation searches for a set of features based on a particular strategy in readiness for the evaluation at the next step. Subset evaluation is the second step of the feature selection process, where every generated candidate feature is evaluated for its quality based on a specific evaluation criterion [21]. Evaluation criteria are broadly classified into *filter* and *wrapper* approaches based on whether or not the data mining algorithms are to be applied in the evaluation of the selected features [22]. The filter approach [23,24] relies on the general characteristics of the data to evaluate the quality of the generated candidate features without involving any data mining algorithm. This includes, but is not limited to, distance, information, correlation, and consistency measures. Filter-based algorithms have faster processing time than wrapper-based algorithms, as they do not include any data mining algorithm [25]. Conversely, the wrapper-based algorithms [26,27] require the use of specific data mining algorithms, such as clustering, in the process of evaluating the generated candidate features [28]. Despite the fact that the wrapper approach can discover better-quality candidate features than does the filter approach, this incurs high computational overheads [29]. Subset generation and evaluation of the feature selection process is iteratively repeated until they meet the requirement of the stopping criterion. The stopping criterion is activated by the completeness of the search, a pre-set maximum iteration time, or when the classification error rate is less than the pre-set threshold [30]. Then, the selected best candidate features are validated by conducting pre- and post-experimental testing of different aspects such as the classification error rate, number of selected features, the existence of redundant/non-representative features, and the time complexity [20].

3. Related Work

Feature selection (FS) is an efficient method used to reduce the high dimensionality of multi-view data. The current FS methods applied to multi-view data can be either *indirect* or *direct*. In the indirect approach, all the views are concatenated into one single matrix. Then, traditional/single-view FS methods (e.g., Fisher score [31], sparse multi-output regression [6], Laplacian score [32], SPEC [4], and multi-cluster feature selection [33]), which are designed for homogeneous data, can be applied on the single matrix. UFSSF [34] addressed the FS problem with streaming features by dynamically updating the set of selected features. However, the indirect approach of the aforementioned methods is not efficient for multi-view learning. By concatenating the views, these methods cannot fully exploit the correlation among the views, as the data lose their actual meaning due to the concatenation. Therefore, these methods produce an inaccurate representation of the selected features.

In the direct approach, FS methods are designed to select features from multi-view data. The methods proposed in [35–37] require data class labels and therefore cannot be used efficiently with real-world applications where data labels do not exist. There are few well-known unsupervised FS methods for multi-view data [38–43]. Table 1 shows the characteristics of current unsupervised feature selection methods where they were designed to tackle the problem of multi-view learning. In addition, none require data class labels in order select features. Additionally, OMVFS and UFODMV are the only methods that work in an online environment. Moreover, UFSSF (indirect approach) and UFODMV are the only methods that have addressed the problem of streaming features, where complete features do not exist in advance but arrive sequentially. However, other than our proposed UFODMV method, no current FS method has attempted to tackle the problem of dynamic views.

Below, we demonstrate the workings of the methods shown in Table 1.

Table 1. Characteristics of existing multi-view feature selection methods.

Method/Characteristic	Multi-View	Dynamic Multi-Views	Online	Streaming Features	Unsupervised
AUMFS	Yes	No	No	No	Yes
MVFS	Yes	No	No	No	Yes
Wang et al.	Yes	No	No	No	Yes
LUFS	Yes	No	No	No	Yes
SRRS	Yes	No	No	No	Yes
ASVW	Yes	No	No	No	Yes
OMVFS	Yes	No	Yes	No	Yes
UFODMV	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

3.1. *Adaptive Unsupervised Multi-View Feature Selection (AUMFS)*

AUMFS [16] selects the representative features of all the views simultaneously. It was proposed to tackle the problem of multi-view learning without the need for data class labels. To select representative features, AUMFS benefits from three information resources: the correlation, the data cluster structure, and the similarity between the views. A regression model was improved to predict cluster labels based on the $l_{2,1}$ Snorm as it imposes joint sparsity on all the views. To do so, AUMFS uses a data cluster structure. Additionally, a graph regularization is built based on the similarity of data across all views. Then, a spectral clustering is performed to produce pseudo-labels. To create the objective function, the learned graphs from each view are united with the weight vector of all the views. Tang et al. proposed a similar method, namely multi-view feature selection (MVFS) [44]. However, the main difference is that MVFS learns only one feature weight matrix of each view and not those of all the views. Then, it utilises the feature weight matrix to fit the pseudo-class labels by the least-square and the norm regulariser.

3.2. *Multi-View Clustering and Feature Learning via Structured Sparsity*

Wang et al. [15] assumed that previous studies assigned the same weight to all data from a single source. However, they believed that some views in some applications have more representative features than others. They illustrated this with an image processing application: they assumed that colour features are more discriminative in the identification of stop signs than are other features. Therefore, they addressed this problem by coming up with an unsupervised feature selection method that assigns weight to features. To do so, they used the structured-sparsity reqularizer to select representative features. They first applied a l_1 norm reqularizer on the feature weight matrix to select more useful views. They later applied a l_2 norm regulariser to the features within the selected views in order find the representative features. Therefore, the selected features should be able to discriminate the cluster structure.

3.3. *Unsupervised Feature Selection for Linked Social Media Data (LUFS)*

LUFS [45] targets applications where the data are linked. In particular, it selects representative features in social media where the data are linked. The algorithm does not require class labels in order to select features. It extracts information from the linked data in the form of link information and attribute value information. Then, it utilizes the extracted constraints for unsupervised feature selection. It selects features from each view and then combines them for learning.

3.4. Online Unsupervised Multi-View Feature Selection (OMVFS)

OMVFS [17] addresses the FS problem for multi-view online data. Specifically, the number of instances increases while the number of the views is fixed. It deploys FS for a clustering objective function via a non-negative matrix factorization with sparse learning. It processes the streaming data in the form of chunks and aggregates all the previous data to date into small matrices. These matrices are updated as new data arrive to learn the FS matrices. Therefore, the selected features are updated correspondingly.

For each view, OMVFS selects a subset of features to represent that view. Moreover, OMVFS combines the data of all the views into a consensus cluster indicator matrix. The derived objective function is as follows:

$$\min_{U,\{V^{(v)}\}} \sum_{v=1}^{n_v} (|| \; X^{(v)} - UV^{(v)^T} \; || + \beta_v \; || \; V^v \; ||_{2,1}) \tag{1}$$

such that $U^T U = I, U \geq 0, V^{(v)} \geq 0, v = 1, 2, \ldots, n_v$. $X^{(v)}$ is the dataset with different views. U is the cluster indicator, $V^{(v)}$, is the features matrix, and β_v is the sparsity parameter. OMVFS applies $l_{2,1}$ to V in order to sort the features and select the ones with greater weight.

3.5. Sparse Low-Rank Representation through Multi-View Subspace Learning (SRRS)

Many existing dimension reduction methods assume that the data (i.e., instances) are complete (i.e., there are no missing values). However, it is more likely in real applications that data samples are not complete for various reasons. For example, there may be sensor failure or restricted access to data. Therefore, SRRS [18] was proposed to address the problem of incomplete or missing multi-view data (i.e., instances or samples) in one or multiple views. To impute the missing values, SRRS jointly computes: (1) the intra-view relation by the sparse low-rank representation; (2) interview relations by global subspace representation. Then, a sparse feature selection via rank minimisation was proposed to find a set of representative features.

3.6. Multi-View Unsupervised Feature Selection with Adaptive Similarity and View Weight (ASVW)

MVFS and AUMFS measure the similarity of each view independently. They build a fixed Laplacian graph for each view individually. Therefore, they ignore the correlation between the views. ASVW [8] addressed this problem by adaptively exploiting correlations between views. The objective function is formulated by a global Laplacian graph for all the views. Additionally, the objective function is united with a sparse norm constraint in order to select representative features. The objective function is given in Equation (2):

$$\min \mathcal{L}\left(W_1,\ldots,W_{V,\alpha,S}\right) = \sum_{v=1}^{V} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_v^{r1} ||W_v^t x_i^{(v)} - W_v^t x_j^{(v)}||^2 \left(S_{ij}\right)^{r2} + \lambda \sum_{v=1}^{V} ||W_v||_{2,p}^{p} \tag{2}$$

such that

$$W_v^T W_v = I$$

$$\sum_{v=1}^{v} \alpha_v = 1, \alpha_v \geq 0$$

$$\sum_{j=1}^{n} S_{ij} = 1, S_{ij} \geq 0, || \; S_i \; ||_0 = k$$

where W_v is the projection matrix of each view; S_{ij} is the similarity matrix; k is the number of close neighbours; $|| \; W_v \; ||_{2,p}$ is the $L_{2,p}$ norm; λ is the non-negative control parameter; r_1 is a balance parameter in order to avoid trivial solution; and $X^{(v)}$ is the data in different views.

In summary, all the direct approach methods discussed above are designed for multi-view learning. However, they assume that the number of views is static, meaning that the views exist in advance, and there are no new views that can be added. As mentioned earlier, this assumption is not appropriate for real-world applications where new views can be added at any given time. Additionally, the number of instances can increase (i.e., online). To the best of our knowledge, this is still an open problem that has not been adequately addressed. This paper addresses the limitations of existing unsupervised

FS methods for multi-view applications by developing a feature selection method for dynamic views applications so that both instances and views can increase overtime and be clustered incrementally.

4. The Proposed UFODMV Method

This section provides the details of the proposed method. First, we explain the concepts of the dynamic views, the online learning, and the representative features.

- *Dynamic multi-views:* The complete views, which are heterogeneous sets of features, do not exist in advance. The views arrive sequentially and individually and are incrementally processed as they arrive. It is inefficient to wait for all the views to be collected before starting the clustering process as they are not static.
- *Online learning:* A complete set of instances do not exist in advance. They arrive sequentially and are incrementally processed upon their arrival, as shown in Figure 2.

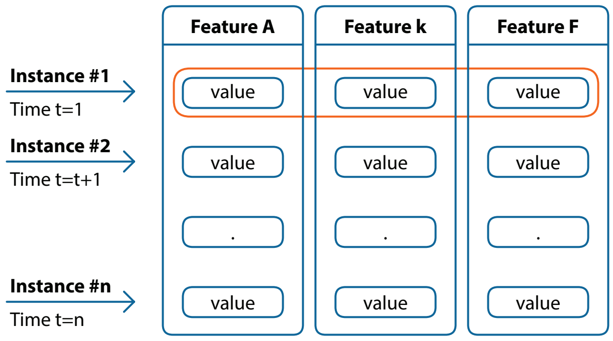


Figure 2. Online learning: The number of features is fixed; the number of instances is dynamic (i.e., increases over time).

- *Representative feature:* A feature assigned to a cluster is considered to be representative if it is closer to its cluster's centroid than are all other features assigned to the same cluster. Formally, $F = \{f_1, f_2, \ldots, f_n\}$ is the set of features such that $f_i = \{value_1, value_2, \ldots, value_n\}$. $C = \{c_1, c_2, \ldots, c_n\}$ is the set of cluster centroids. Let f_i be a feature vector (i.e., $f_i \subset F$) in a cluster with the centroid $c_i \in C$. Let Φ be a subset of features in a cluster. Let R_i be a representative feature so far (i.e., $R_i \subset F$) in a cluster with the centroid $c_i \in C$. The feature $f_i \subset \Phi$ is a representative feature in a cluster with centroid c_i if and only if:

$$dist(f_i, c_i) < dist(\Phi, c_i)$$

Any feature f_i that is not representative is simply considered as *non-representative*.

Figure 3 gives an overview of the proposed UFODMV method. The UFODMV method consists of two parts, namely clustering and feature selection. The chunk can have new instances or new views. If the chunk has new instances only, UFODMV incrementally clusters the new instances with the clusters resulting from the previous chunk. Similarly, when the chunk has a new view (i.e., a set of features), UFODMV incrementally clusters it with the clusters resulting from the previous chunk. With this method, clustering is used for the selection of features as it does not require the data class labels in order to group the data. Specifically, the UFODMV uses hierarchical clustering to merge the clusters, as detailed below. The selected set of representative features is updated at each clustering step.

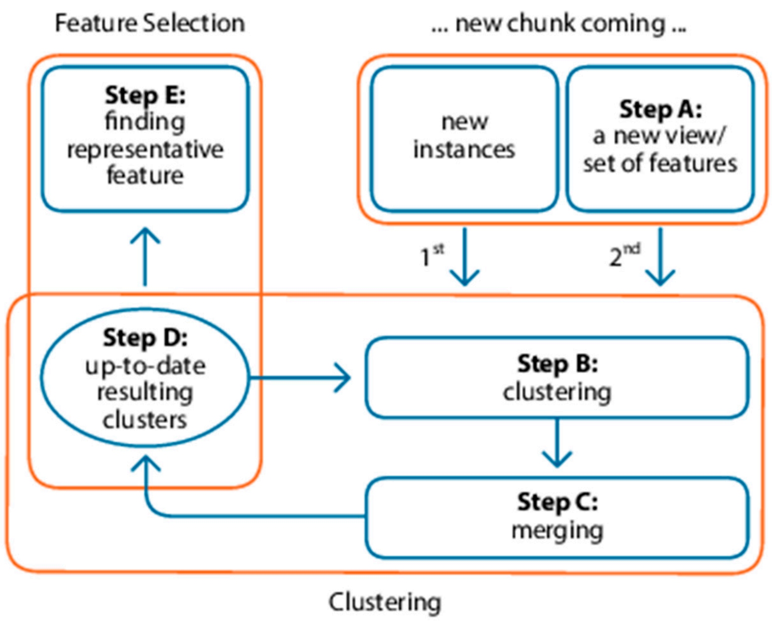

Feature Selection
... new chunk coming ...
Step E: finding representative feature
new instances
Step A: a new view/ set of features
1st
2nd
Step D: up-to-date resulting clusters
Step B: clustering
Step C: merging
Clustering

Figure 3. An overview of UFODMV method. In step A, a new chunk received from a data stream that has new instances and a view. In step B, the new instance and features set are clustered. In step C, the resulting clusters from step B are merged to the predefined reduction size. In step D, clusters are incrementally updated with new corresponding values. In step E, the representative features are selected from up-to-date clusters.

The steps for selecting a set of representative features from dynamic views in an online environment are:

1. **Initialising phase (Step A).**For the first chunk of data, UFODMV initialises a cluster by assigning its first feature as a centroid and a member of that cluster (see Algorithm 1 Line 2). Then, UFODMV clusters all remaining features in the chunk, as illustrated in the subsequent clustering phase. Every feature is strictly assigned to one cluster so that there is no overlapping of clusters (i.e., hard clustering).
2. **Clustering phase (Step B).** For each feature, UFODMV either assigns a feature f_j to a cluster or creates a new cluster with f_j as a centroid and a member of it. To do so, (a) UFODMV finds the closest cluster to f_j by computing the Euclidean distance $\sqrt{\sum_{i=1}^{n}(f_j - c_i)^2}$ between the feature f_j and each cluster centroid; (b) UFODMV decides the inclusion of f_j in the closest cluster by computing the new radius of the closest cluster including the feature f_j. If the new radius does not exceed the predefined input threshold T, UFODMV confirms the inclusion of f_j in its closest cluster. Otherwise, UFODMV creates a new cluster with f_j as a centroid and a member of it (see Algorithm 1 Line 5). The radius of a cluster is the sum of the squares of the distances of all its features to the cluster centroid. Formally, cluster radius = $\sum dist(\epsilon - c_i)^2$. The clustering phase does not initially force the features to be grouped into a limited k number of clusters. Instead, it optimises the features to be clustered naturally based on their distribution by allowing the number of clusters to increase. Then, the merging phase limits the clusters by merging them with the predefined reduction size k, as provided in the next phase.
3. **Merging phase (Step C).** As a result of the clustering phase, one could end up with too many clusters that exceed the required reduction size k number of clusters. Therefore, in this phase, UFODMV hierarchically merges clusters until the number of clusters = k. UFODMV merges (lets say cluster #1 and cluster #2) if the distance between the centroid of cluster #1 and the centroid of cluster #2 is the minimum compared to all other cluster centroids. Indeed, clusters with centroids that are close-distance

centroids are more likely to share the same characteristics, and therefore, they are merged. Formally, let $c_i = \{c_1, \ldots, c_n\}$ be a set of cluster centroids.

$$merging = minimum\_distance(clustercentroid1 - clustercentroid2)$$

Every time that we merge clusters, we compute a new centroid for the new cluster (see Algorithm 1, Line 6).

4. **Whenever another chunk arrives (Step D).** The new chunk either has new values (i.e., new instances) for exiting clustered features or additionally has new features in the new view. In the first case (see Algorithm 1, Lines 9–16), UFODMV first finds the corresponding features in the *k* clusters resulting from the merging phase. This is executed based on indexing the features. Then, UFODMV adds the new values to corresponding features in the *k* clusters. For each *k* cluster, UFODMV calculates the new radius after updating the centroid. Formally,

$$centroid\_update = \frac{newfeatures + existingfeatures}{number\_of\_features}; radius > T$$

If the new radius exceeds the threshold T, the clustering and merging phases are repeated for only those clusters. By limiting the clustering and merging phases to the clusters with radii greater than the parameter T, the computational complexity of the UFODMV is reduced. For example, if k = 5 and we have two clusters with the radius exceeding the predefined threshold T, we repeat the clustering and merging phases for $k-3$ clusters so that the total number of clusters is five. Otherwise, we go to the finding representative feature phase. In the second case (i.e., Algorithm 1, Lines 9–19), UFODMV performs the same process as for the first case in addition to conducting the clustering and merging phases with the new features of the new view.

5. **Finding representative features (Step E).** UFODMV selects a single feature from each cluster as a representative feature. The selected features are those that are closest to their clusters' centroids. The selected features from each cluster comprise the set of representative features (see Algorithm 1, Lines 20–26).

The UFODMV algorithm is given below. It can work in three scenarios based on the chunk structure: (1) streaming features, where every new chunk comprises only new features, and the number of instances is fixed (Lines 2–7); (2) online learning, where every new chunk arrives with new instances, while the number of features is fixed (Lines 9–16); and (3) dynamic views with online learning, where every new chunk comes with both new views or sets of features and new instances (Lines 9–19). This makes the algorithm suitable for a wide range of application requirements. In this paper, we experimentally investigate the dynamic views in an online learning environment as it covers all aforementioned scenarios and is flexible for real applications such as health care.

Algorithm 1 The pseudocode of UFODMV algorithm

Input: $X^{(v)}$: data matrices from different views, K: the number of desired clusters, T: the radius threshold
Output: R, the set of representative features

```
1: Initialise the chunks following the required scenario (streaming features, online learning, or
   online learning with dynamic views)
2: Initialise a cluster with a single feature $f_i \subset chunk_1$ as a centroid and a member
3: //For all remaining features in $chunk_1$
4: for f = 1:n do
5:     $[clusters]$ = clustering(features, T)
6:     $[desired\_clusters]$ = merging(clusters, K)
7: resulted_clusters = desired _clusters
8: for chunk = 2:n do
9:     //For new instances
10:    idx = find(features(resulted_clusters))
11:    Add the new values to the corresponding features(resulted_clusters)
12:    updated_clusters = (update radius(resulted_clusters), update centroid(resulted_clusters))
13:    clusters_exceed_T = (find(updated_clusters) = = radius $>$ T))
14:    $[clusters]$ = clustering(clusters_exceed_T)
15: //Merge clusters such that clusters_exceed_T $+$ updated_clusters = k
16: $[desired\_clusters]$ = merging(clusters)
17: //for new view
18: $[clusters]$ = clustering(new_view,desired _clusters)
19: $[desired\_clusters]$ = merging(clusters)
20: //Select a representative feature of each cluster
21: for desired_clusters = 1:n do
22:    for j = 1:n do
23:        selected_feature = min(distance(feature(j), centroid(desired_cluster)))
24:    set_of_selected_features(1,desired_clusters) = (selected_feature)
25: R = set_of_selected_features
26: resulted_clusters = desired _clusters
```

5. Experimental Evaluation

This section describes the experimental setup of the proposed UFODMV and the benchmark methods for real datasets. The experiments were conducted to investigate the following:

- How the selected set of representative features affects the classification accuracy on online dynamic multi-view data for various feature sets;
- The speed of UFODMV when selecting a set of representative features in the aforementioned environment.

To simulate the online learning and the dynamic views, the datasets were structured as shown in Figure 4, where every new chunk has new instances and a view. The number of chunks was set to be equal to the number of the views, and the chunk size was the number of data instances divided by the number of views. The chunks were structured this way as it matched the settings of real applications where both views and instances can increase over time. In a hospital, new patients might have already undergone various medical tests. Additionally, existing patients may undergo a new set of medical tests.

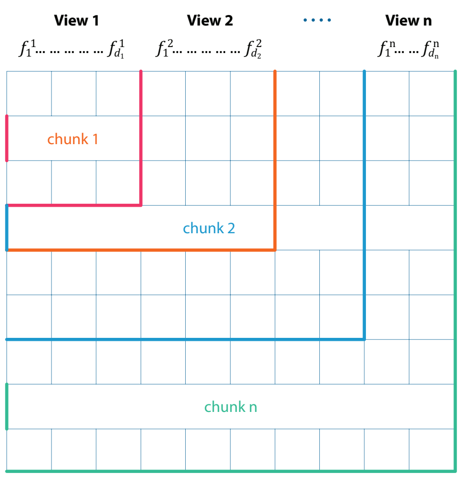


Figure 4. Illustration of proposed dynamic view structure where instances are rows and features are columns. Each new chunk consists of new instances and an additional view.

To properly answer the two questions above, UFODMV was benchmarked against three well-known unsupervised FS methods, namely OMVFS [17], UFSSF [34], and SPEC [4]. To the best of our knowledge, to date, there are no unsupervised FS methods that have been developed for dynamic-view applications. Therefore, these three methods have been selected as they comprise a combination of streaming features, batch, online, and multiple view applications. A summary of these methods is provided in Table 2. SPEC and UFSSF cannot be applied directly to dynamic views, as SPEC is designed only for batched datasets, and UFSSF is designed for streaming features. Therefore, the views of each dataset were concatenated in a single matrix. Then, these two methods were applied to that data matrix. For OMVFS, we followed the settings given in [17], and we set the chunk size of each view to be equal to the chunk size used in UFODMV in order to make a fair comparison. An exception is the dynamic views, as OMVFS works with multiple but not dynamic views. α_v and β_v remained the same for all different views. We conducted a grid search in $\{10^{-2}, 10^{-1}, 10^{1}, 10^{2}\}$ and selected the one with better results. γ was set to 10^7. We used the code (https://github.com/software-shao/Online-Unsupervised-Multiview-Feature-Selection/blob/master/OMVFS.m) (accessed on 20 January 2023) as directed in [17].

For UFODMV, we reported the selected features of the last chunk in order to make a fair comparison with the benchmark methods. The radius was set such that we could obtain k number of clusters or selected features. k was set to the required number of representative features to be selected.

Table 2. Summary of the proposed and benchmarked methods.

Method/Category	Multi-View	Dynamic Views	Online	Streaming Features	Unsupervised
SPEC	No	No	No	No	Yes
UFSSF	No	No	No	Yes	Yes
OMVFS	Yes	No	Yes	No	Yes
UFODMV	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

For all the evaluated methods, the number of selected features was varied to ensure the reliability of the results. The representative features selected by UFODMV and the other benchmark methods were evaluated using two well-known classifiers: naive Bayes [46] and lazy nearest neighbor [47] (also called IB1). In addition to the classifiers, k-fold cross-validation was applied to all selected features to avoid the problem of overfitting the data and to produce accurate results. First, the set of selected features was divided into subsets of equal size depending on the selected k folds. One subset was retained as the testing subset, and the remainder were used as training subsets. Finally, the average value of all folds was set to be the average result. In the evaluation, k was set to ten (10) as suggested in [48]. All four algorithms were implemented in the MATLAB programming language. They were executed using the Mac operating system macOS High Sierra with a 2.9 GHz Intel Core i7 and 16 GB RAM.

5.1. Datasets

Three datasets with various dimensionalities were used to evaluate the the proposed UFODMV and the benchmark methods. We selected these datasets because they are commonly used for multi-view learning. The three adopted datasets were collected mainly for the purpose of classification and clustering as clustering is a part of the proposed approach for the selection of representative features. Data were randomly shuffled to avoid order-dependency between the instances and the features and to accurately evaluate the UFODMV and the benchmark methods. A brief description of each dataset is given below:

- *Fox News*: This is a news article dataset. Each article is represented in two views, namely a text view and an image view. Text or words in titles, abstracts, and bodies comprise the text view data. The images associated with each article comprise the image view data. The image view has 996 features, and the text view has 27,072 features. The total number of instances is 1523, and the data fall into four classes. https://sites.google.com/site/qianmingjie/home/datasets/cnn-and-fox-news (accessed on 20 January 2023).
- *Caltech-7*: This is an image dataset where pictures of objects belong to seven classes. It has six views/group of features, namely Gabor (48 features), wavelet moments (40 features), CENTRIST (254 features), histogram of oriented gradients (1984 features), GIST (512 features) and local binary patterns (928). In total, there are 1474 instances. https://github.com/yeqinglee/mvdata (accessed on 20 January 2023)
- *Handwritten/multiple features*: This dataset consists of features of written numerals ('0' to '9'). These numerals were written manually by hands. There are six groups/views of features, namely Fourier coefficients of the character shapes (76 features), pixel averages (240 features), Profile correlations (216 features), Zernike moment (46 features), Karhunen–Love coefficients (64 features), and morphological (6 features). The total number of instances is 2000. https://archive.ics.uci.edu/ml/datasets/Multiple+Features (accessed on 20 January 2023).

5.2. Evaluation Metrics

The aim of the proposed UFODMV was to select a set of representative features from dynamic views in online mode. The selected features should improve the accuracy of the ML algorithms when performing their tasks. We selected classifiers, as the datasets had class labels. The classifiers were trained with the class labels. Then, the test data, which did not have class labels, were tested by the trained classifiers. The process of selecting the features from the dynamic views should be carried out within an acceptable running time. Therefore, the metrics used for the evaluation were grouped according to classification accuracy metrics and running time.

(A) *Classification accuracy metrics.* Three metrics were adopted to determine the classification accuracy: recall, precision, and F-measure, and they are computed as below:

$$Recall = \frac{TruePositive(TP)}{TruePositive(TP) + FalseNegative(FN)} \tag{3}$$

$$Precision = \frac{TruePositive(TP)}{TruePositive(TP) + FalsePositive(FP)} \tag{4}$$

$$F - measure = \frac{Recall \times Precision}{Recall + Precision} \tag{5}$$

These metrics determine whether the selected features significantly improve the classification accuracy of the classifiers. These particular metrics were adopted as they are widely used for measuring classification accuracy.

(B) *Running time.* The major reason for developing UFODMV was to remove those non-representative features from dynamic view applications to produce better data representations. Therefore, this improves the classification accuracy of the classifiers when this set of representative features is used as input for the classifiers. However, this target should be achieved within an acceptable running time. To do so, the time taken by UFODMV, UFSSF, SPECk and OMVFS to select representative features were computed in seconds.

6. Results and Discussion

This section introduces and discusses the results of the conducted experiments.

Classification Accuracy

The classification accuracy was measured based on three evaluation metrics: recall, precision, and F-measure. Two classifiers were used, namely naive Bayes and IB1. Then, we present the results relating to the running time. The time taken by each benchmark method to select features is reported in seconds.

The experimental results regarding the classification accuracy of UFODMV and the benchmark methods on the three datasets are presented in Tables 3–8. The following observations can be made from the results:

- With all methods, the accuracy (recall, precision, and f-measure) increases as the number of the selected features increases, thus indicating that all methods selected good representative features.
- UFODMV significantly outperformed the traditional/single view feature selection methods as it incrementally exploited the correlation between the views. Therefore, it selected better representative features.
- UFODMV was also compared with a multi-view feature selection method, namely OMVFS. However, UFODMV consistently has the highest recall, precision, and f-measure with all different numbers of selected features. This is because UFODMV incrementally updates the means of the clusters to evaluate the representativeness of the selected features, as these might lose their representativeness over time due to the arrival of new features.
- For example, Tables 3 and 4 show the classification accuracy of UFODMV and the benchmark methods on the Handwritten dataset. For both the naive Bayes and IB1 classifiers, UFODMV has around 5–7% better accuracy compared to the traditional/single-view methods. Additionally, it has around 50% better accuracy compared to OMVFS regardless of whether the classification metric is recall, precision, or f-measure. Table 7 shows the classification accuracy of UFODMV when applied to the Fox News dataset using the naive Bayes classifier. UFODMV has around 15–20% better accuracy when compared with traditional/single-view methods. Additionally, it achieves around 40–60% better accuracy than OMVFS.
- The method with the second-best accuracy is UFSSF. This is valid for almost all datasets with all evaluation classifiers. Indeed, although UFSSF was not developed for multi-view data, it incrementally clusters the features in a streaming feature manner.
- For the Fox News dataset, all the methods have an overall reduction in accuracy when they are evaluated using the IB1 classifier. This might be due to the Fox News data being sparse, since the IB1 classifier is efficient when classifying dense data.

Table 3. Classification accuracy using Handwritten dataset with naive Bayes.

Method\Number of Selected Features	**25**	**50**	**75**	**100**	**Metric**
UFODMV (proposed)	**0.861**	**0.886**	**0.965**	**0.961**	Recall
OMVFS	0.288	0.299	0.334	0.349	
SPEC	0.804	0.866	0.891	0.907	
UFSSF	0.81	0.867	0.912	0.922	
UFODMV (proposed)	**0.862**	**0.887**	**0.956**	**0.972**	Precision
OMVFS	0.284	0.329	0.342	0.349	
SPEC	0.804	0.869	0.892	0.908	
UFSSF	0.823	0.869	0.914	0.945	
UFODMV (proposed)	**0.861**	**0.886**	**0.955**	**0.961**	F-measure
OMVFS	0.269	0.270	0.307	0.342	
SPEC	0.803	0.867	0.891	0.906	
UFSSF	0.821	0.866	0.91	0.923	

UFODMV has better accuracy compared with the benchmark methods. This is due to the fact that UFODMV has an efficient clustering methodology. In fact, when partitioning the clusters, UFODMV allows the features to be clustered naturally based on their similarities, and later on, it merges the clusters into the required clusters. This clustering methodology ensures that features are well partitioned or grouped, and good clustering will definitely result in a good representation of the features. Additionally, UFODMV limited the representative feature of each cluster as the one with minimum distance to its cluster's centroid. This step ensures that the feature selected from each cluster is a good representative of other features in that cluster. Finally, because views are dynamic, the set of features were updated at each clustering step, as some features might be representative for a limited amount of time.

Although in a few cases, UFSSF achieved better accuracy than the proposed UFODMV, the latter addressed the problem of increasing instances (i.e., online learning) and increasing views (dynamic views). UFSSF only addressed the dynamic features problem (i.e., increasing features), and therefore, we can accept the tiny accuracy difference in a few cases while addressing the problem of both increasing features and increasing instances. Additionally, when UFODMV is compared with the online multi-view method (OMVFS), it has significantly better recall, precision, and f-measure for all the used datasets.

Table 4. Classification accuracy using Handwritten dataset with IB1.

Method\Number of Selected Features	**25**	**50**	**75**	**100**	**Metric**
UFODMV (proposed)	**0.926**	**0.936**	**0.976**	**0.979**	Recall
OMVFS	0.116	0.118	0.127	0.147	
SPEC	0.855	0.864	0.909	0.895	
UFSSF	0.857	0.884	0.916	0.938	
UFODMV (proposed)	**0.929**	**0.939**	**0.976**	**0.979**	Precision
OMVFS	0.119	0.123	0.133	0.152	
SPEC	0.857	0.856	0.909	0.896	
UFSSF	0.858	0.874	0.916	0.938	
UFODMV (proposed)	**0.926**	**0.939**	**0.976**	**0.979**	F-measure
OMVFS	0.116	0.117	0.128	0.148	
SPEC	0.853	0.854	0.909	0.895	
UFSSF	0.857	0.863	0.912	0.937	

Table 5. Classification accuracy using Caltech dataset with naive Bayes.

Method\Number of Selected Features	**25**	**50**	**75**	**100**	**Metric**
UFODMV (proposed)	**0.815**	**0.875**	**0.879**	**0.905**	Recall
OMVFS	0.039	0.04	0.044	0.484	
SPEC	0.72	0.786	0.811	0.827	
UFSSF	0.763	0.83	0.853	0.873	
UFODMV (proposed)	**0.848**	**0.9**	**0.906**	**0.915**	Precision
OMVFS	0.381	0.383	0.385	0.418	
SPEC	0.721	0.809	0.835	0.851	
UFSSF	0.775	0.853	0.879	0.898	
UFODMV (proposed)	**0.824**	**0.882**	**0.887**	**0.907**	F-measure
OMVFS	0.023	0.029	0.043	0.405	
SPEC	0.717	0.795	0.82	0.836	
UFSSF	0.765	0.837	0.862	0.881	

Table 6. Classification accuracy using Caltech dataset with IB1.

Method\Number of Selected Features	**25**	**50**	**75**	**100**	**Metric**
UFODMV (proposed)	**0.891**	**0.927**	**0.935**	**0.939**	Recall
OMVFS	0.371	0.379	0.382	0.387	
SPEC	0.739	0.77	0.778	0.78	
UFSSF	0.879	0.881	0.912	0.919	
UFODMV (proposed)	**0.88**	**0.926**	**0.931**	**0.937**	Precision
OMVFS	0.374	0.377	0.392	0.393	
SPEC	0.736	0.791	0.815	0.819	
UFSSF	0.872	0.873	0.906	0.912	
UFODMV (proposed)	**0.883**	**0.923**	**0.931**	**0.935**	F-measure
OMVFS	0.373	0.377	0.386	0.389	
SPEC	0.734	0.771	0.781	0.782	
UFSSF	0.868	0.869	0.906	0.912	

Table 7. Classification accuracy using Fox News dataset with naive Bayes.

Method\Number of Selected Features	**25**	**50**	**75**	**100**	**Metric**
UFODMV (proposed)	**0.561**	**0.687**	**0.746**	**0.764**	Recall
OMVFS	0.354	0.169	0.149	0.157	
SPEC	0.379	0.399	0.42	0.442	
UFSSF	0.389	0.433	0.51	0.539	
UFODMV (proposed)	**0.641**	**0.721**	**0.763**	**0.777**	Precision
OMVFS	0.304	0.254	0.359	0.324	
SPEC	0.428	0.462	0.473	0.514	
UFSSF	0.518	0.598	0.6278	0.624	
UFODMV (proposed)	**0.562**	**0.693**	**0.75**	**0.768**	F-measure
OMVFS	0.316	0.163	0.122	0.123	
SPEC	0.469	0.514			
UFSSF	0.517	0.592	0.618	0.613	

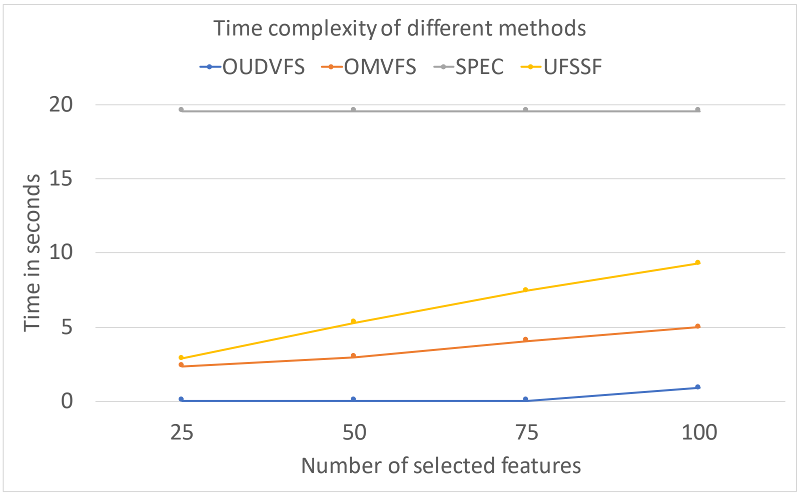
Table 8. Classification accuracy using Fox News dataset with IB1.

Method\Number of Selected Features	25	50	75	100	Metric
UFODMV (proposed)	**0.586**	**0.651**	**0.662**	**0.676**	Recall
OMVFS	0.287	0.302	0.302	0.309	
SPEC	0.503	0.573	0.599	0.612	
UFSSF	0.508	0.586	0.602	0.625	
UFODMV (proposed)	**0.598**	**0.664**	**0.677**	**0.688**	Precision
OMVFS	0.288	0.296	0.307	0.311	
SPEC	0.501	0.581	0.584	0.599	
UFSSF	0.508	0.585	0.617	0.627	
UFODMV (proposed)	**0.591**	**0.654**	**0.666**	**0.676**	F-measure
OMVFS	0.287	0.298	0.304	0.309	
SPEC	0.5	0.579	0.584	0.599	
UFSSF	0.508	0.585	0.606	0.625	

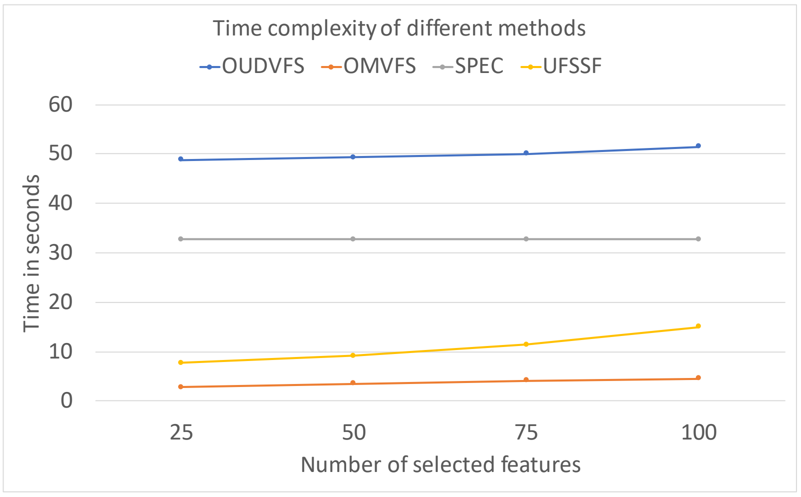
7. Running Time

Figure 5 depicts the results of the running time of UFODMV and the benchmark methods. The running time of all methods increased when the data were sparse. For example, all methods applied to the Fox News dataset required higher running time than when they were applied to the Caltech-7 and Handwritten datasets. Additionally, the running time of all the methods except for SPEC increased as the number of selected features increased. This is because SPEC returns a weighted vector of all features. Therefore, for each dataset, it has a fixed running time for different numbers of selected features. For the Handwritten dataset, as shown in Figure 5a, UFODMV has the best running time compared with OMVFS, UFSSF and SPEC. UFODMV selected different numbers of features with excellent running time. It selected 25, 50, and 75 features in approximately 0.023 s. Additionally, UFODMV took 0.875 s to select 100 features. This is because of two factors: (1) when a new chunk arrives, UFODMV re-partitions only those clusters whose features exceed the predefined threshold T, not all the clusters; and (2) in the merging step, distance is computed between centroids so as to merge the closest clusters. This is unlike other clustering methods that merge clusters based on the distance between their features. SPEC took the highest running time compared to the benchmarked methods, returning a weight vector of features in approximately 19.56 s. OMVFS and UFSSF were the second and third fastest methods, respectively. They were very competitive when 25 features were selected. However, they consistently recorded a difference of around 0.52% for all other numbers of selected features.

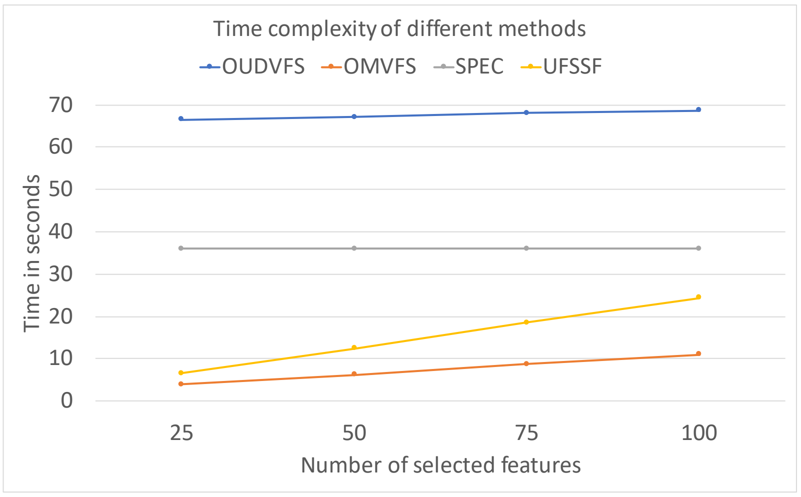
However, with the Caltech and Fox News datasets, UFODMV was not as fast as the benchmark methods (see Figure 5b,c). This is due to two factors: (1) the UFODMV algorithm has many for-loop functions in order to incrementally cluster the instances after clustering the views. Actually, it is well-known that the Matlab platform works very slowly with loops, which is why a cloud platform was developed for the loop function. (2) If a new chunk arrives, UFODMV assigns the new values of existing features into their corresponding clusters. Then, UFODMV re-clusters current clusters and merges them if they exceed a pre-set radius threshold. However, for sparse datasets, such as those of Caltech-7 and Fox News, we need to set a small radius value in order to obtain the required number of selected features. However, the negative side of this is that the smaller the radius, the more clusters need to be clustered and sequentially merged, which in turn requires more processing time. SPEC was the second-highest method in terms of running time for both datasets. It needed 32.68 and 36.04 s for the Caltech-7 and Fox datasets, respectively. OMVFS is the fastest in terms of running time for both datasets. This is because it uses matrix factorisation, which can process sparse data efficiently. UFSSF is the second method with the lowest running time for both datasets.



(**a**)



(**b**)



(**c**)

Figure 5. Time complexity of different methods on (**a**) Handwritten dataset, (**b**) Caltech-7 dataset, and (**c**) Fox News dataset.

Overall, UFODMV is not the best in terms of running time; hence, further investigation will be conducted to measure the running time of UFODMV using a different programming language. Although UFODMV did not achieve the best running time, it had the best classification accuracy, despite the challenge of solving unsupervised feature selection for dynamic multi-view and online learning.

8. Conclusions

In this paper, an online unsupervised feature selection method is proposed for dynamic views. In real-world applications, both new views and new instances can be added over time. UFODMV is different from existing single-view feature selection methods as it incrementally exploits any similarity of all the views presented so far. This results in the better representation of the selected features. Additionally, it is different from existing multi-view feature selection methods as it addresses the problem of dynamic views and online learning. UFODMV is proposed as a clustering-based algorithm that aims to cluster the views and the instances sequentially. Then, from each cluster, it selects as the representative feature the one closest to its cluster centroid. The set of selected features is updated at each clustering step. The performance of OUDVFD was tested on three multi-view datasets and compared with the performances of well-known single-view and multi-view feature selection methods. UFODMV has better accuracy when tested with several evaluation metrics, namely recall, precision and F-measure. However, UFODMV does not have the best running time because the incremental clustering of the views and instances adds to the complexity of the loop function. Further investigation will be undertaken, which involves implementing the code with a programming language other than MATLAB, which is very slow with loops. In future work, gene selection for cancer/patient classification is a crucial problem that is going to be investigated, along with software development for users.

Author Contributions: Conceptualization, N.A., A.A. and M.A.A.; Methodology, A.A.; Software, N.A. and A.K.A.; Validation, M.A.A.; Writing—original draft, N.A., A.A. and M.A.A.; Writing—review & editing, F.A., A.A. and A.K.A.; Project administration, T.M. All authors have read and agreed to the published version of the manuscript.

Funding: 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. 1925].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, J.; Cheng, K.; Wang, S.; Morstatter, F.; Trevino, R.P.; Tang, J.; Liu, H. Feature selection: A data perspective. *Comput. Surv. (CSUR)* **2017**, *50*, 94. [CrossRef]
2. Cao, B.; He, L.; Kong, X.; Philip, S.Y.; Hao, Z.; Ragin, A.B. Tensor-based multi-view feature selection with applications to brain diseases. In Proceedings of the IEEE International Conference on Data Mining (ICDM), Shenzhen, China, 14–17 December 2014; pp. 40–49.
3. Wangila, K.W.; Gao, K.; Zhu, P.; Hu, Q.; Zhang, C. Mixed sparsity regularized multi-view unsupervised feature selection. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Beijing, China, 1–17 September 2017; pp. 1930–1934.
4. Zhao, Z.; Liu, H. Spectral feature selection for supervised and unsupervised learning. In Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, USA 20–24 June 2007; ACM: New York, NY, USA, 2007; pp. 1151–1157.
5. Mitra, P.; Murthy, C.; Pal, S.K. Unsupervised feature selection using feature similarity. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 301–312. [CrossRef]
6. Zhao, Z.; Wang, L.; Liu, H. Efficient spectral feature selection with minimum redundancy. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 11–13 October 2010; pp. 673–678.
7. Hou, C.; Nie, F.; Yi, D.; Wu, Y. Feature selection via joint embedding learning and sparse regression. In Proceedings of the Proceedings-International Joint Conference on Artificial Intelligence (IJCAI), Catalonia, Spain, 16–22 July 2011; Volume 22, p. 1324.
8. Hou, C.; Nie, F.; Tao, H.; Yi, D. Multi-view unsupervised feature selection with adaptive similarity and view weight. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 1998–2011. [CrossRef]

9. de Araújo, R.C.; de Carvalho, F.d.A.; Lechevallier, Y. Multi-view hard c-means with automated weighting of views and variables. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), HongKong, China, 1–8 June 2017; pp. 2792–2799.
10. Trivedi, A.; Rai, P.; Daumé III, H.; DuVall, S.L. Multiview clustering with incomplete views. In Proceedings of the NIPS 2010: Workshop on Machine Learning for Social Com- puting, Whistler, BC, Canada, 11 December 2010.
11. Yamanishi, Y.; Vert, J.P.; Kanehisa, M. Protein network inference from multiple genomic data: A supervised approach. *Bioinformatics* **2004**, *20*, i363–i370. [CrossRef] [PubMed]
12. Ghaemi, R.; Sulaiman, M.N.; Ibrahim, H.; Mustapha, N. A survey: Clustering ensembles techniques. *World Acad. Sci. Eng. Technol.* **2009**, *50*, 636–645.
13. Xie, X.; Sun, S. Multi-view clustering ensembles. In Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC), Adelaide, Australia, 4–5 December 2013; Volume 1, pp. 51–56.
14. Tzortzis, G.; Likas, A. Convex mixture models for multi-view clustering. In Proceedings of the International Conference on Artificial Neural Networks, Munich, Germany, 17–19 September 2019; Springer: Berlin/Heidelberg, Germany, 2009; pp. 205–214.
15. Wang, H.; Nie, F.; Huang, H. Multi-view clustering and feature learning via structured sparsity. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 352–360.
16. Feng, Y.; Xiao, J.; Zhuang, Y.; Liu, X. Adaptive unsupervised multi-view feature selection for visual concept recognition. In Proceedings of the Asian Conference on Computer Vision, Daejeon, Republic of Korea, 5–9 November 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 343–357.
17. Shao, W.; He, L.; Lu, C.T.; Wei, X.; Yu, P.S. Online unsupervised multi-view feature selection. *arXiv* **2016**, arXiv: 1609.08286.
18. Yang, W.; Shi, Y.; Gao, Y.; Wang, L.; Yang, M. Incomplete-data oriented multiview dimension reduction via sparse low-rank representation. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 6276–6291. [CrossRef]
19. Das, S.; Rai, S.N. Statistical approach for biologically relevant gene selection from high-throughput gene expression data. *Entropy* **2020**, *22*, 1205. [CrossRef]
20. Liu, H.; Motoda, H. *Comput. Methods Feature Sel*; CRC Press: Boca Raton, FL, USA, 2007.
21. Zhu, W.; Si, G.; Zhang, Y.; Wang, J. Neighborhood effective information ratio for hybrid feature subset evaluation and selection. *Neurocomputing* **2013**, *99*, 25–37. [CrossRef]
22. Wald, R.; Khoshgoftaar, T.M.; Napolitano, A. How the choice of wrapper learner and performance metric affects subset evaluation. In Proceedings of the IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI), Herndon, VA, USA, 2–6 November 2013; pp. 426–432.
23. Nandi, G. An enhanced approach to Las Vegas Filter (LVF) feature selection algorithm. In Proceedings of the 2nd National Conference on Emerging Trends and Applications in Computer Science (NCETACS), Shillong, India, 4–5 March 2011; pp. 1–3.
24. Suri, N.R.; Murty, M.N.; Athithan, G. Unsupervised feature selection for outlier detection in categorical data using mutual information. In Proceedings of the 12th International Conference on Hybrid Intelligent Systems (HIS), Pune, India, 4–7 December 2012; pp. 253–258.
25. Jiang, S.; Wang, L. Unsupervised feature selection based on clustering. In Proceedings of the IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), Changsha, China, 23–26 September 2010; pp. 263–270.
26. Hsu, C.N.; Huang, H.J.; Dietrich, S. The ANNIGMA-wrapper approach to fast feature selection for neural nets. *IEEE Trans. Syst. Man, Cybern. Part Cybern.* **2002**, *32*, 207–212.
27. Zhou, H.; Wu, J.; Wang, Y.; Tian, M. Wrapper approach for feature subset selection using GA. In Proceedings of the International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Hualien, Taiwan, 16–19 November 2021; pp. 188–191.
28. Freeman, C.; Kulić, D.; Basir, O. An evaluation of classifier-specific filter measure performance for feature selection. *Pattern Recognit.* **2015**, *48*, 1812–1826. [CrossRef]
29. Hong, Y.; Kwong, S.; Chang, Y.; Ren, Q. Consensus unsupervised feature ranking from multiple views. *Pattern Recognit. Lett.* **2008**, *29*, 595–602. [CrossRef]
30. Liu, H.; Yu, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 491–502.
31. Duda, R.O.; Hart, P.E.; Stork, D.G. *Pattern Classification*; John Wiley & Sons: Hoboken, NJ, USA, 2012.
32. He, X.; Cai, D.; Niyogi, P. Laplacian score for feature selection. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 4–7 December 2006; pp. 507–514.
33. Cai, D.; Zhang, C.; He, X. Unsupervised feature selection for multi-cluster data. In Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; ACM: New York, NY, USA, 2010; pp. 333–342.
34. Almusallam, N.; Tari, Z.; Chan, J.; AlHarthi, A. UFSSF-An Efficient Unsupervised Feature Selection for Streaming Features. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Kuala Lumpur, Malaysia, 16–18 April 2001; Springer: Berlin/Heidelberg, Germany, 2018; pp. 495–507.
35. Lin, Q.; Yang, L.; Zhong, P.; Zou, H. Robust supervised multi-view feature selection with weighted shared loss and maximum margin criterion. *Knowl.-Based Syst.* **2021**, *229*, 107331. [CrossRef]

36. Lin, Q.; Men, M.; Yang, L.; Zhong, P. A supervised multi-view feature selection method based on locally sparse regularization and block computing. *Inf. Sci.* **2022**, *582*, 146–166. [CrossRef]
37. Men, M.; Zhong, P.; Wang, Z.; Lin, Q. Distributed learning for supervised multiview feature selection. *Appl. Intell.* **2020**, *50*, 2749–2769. [CrossRef]
38. Acharya, S.; Cui, L.; Pan, Y. A consensus multi-view multi-objective gene selection approach for improved sample classification. *BMC Bioinform.* **2020**, *21*, 1–16. [CrossRef] [PubMed]
39. Tuan, D.L.T.; Hoang, V.T. Using Feature Selection Based on Multi-view for Rice Seed Images Classification. In *Research in Intelligent and Computing in Engineering*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 651–661.
40. Zhang, H.; Wu, D.; Nie, F.; Wang, R.; Li, X. Multilevel projections with adaptive neighbor graph for unsupervised multi-view feature selection. *Inf. Fusion* **2021**, *70*, 129–140. [CrossRef]
41. Wang, C.; Chen, X.; Chen, B.; Nie, F.; Wang, B.; Ming, Z. Learning unsupervised node representation from multi-view network. *Inf. Sci.* **2021**, *579*, 700–716. [CrossRef]
42. Wan, Y.; Sun, S.; Zeng, C. Adaptive similarity embedding for unsupervised multi-view feature selection. *IEEE Trans. Knowl. Data Eng.* **2020**, *33*, 3338–3350. [CrossRef]
43. Bai, X.; Zhu, L.; Liang, C.; Li, J.; Nie, X.; Chang, X. Multi-view feature selection via nonnegative structured graph learning. *Neurocomputing* **2020**, *387*, 110–122. [CrossRef]
44. Tang, J.; Hu, X.; Gao, H.; Liu, H. Unsupervised feature selection for multi-view data in social media. In Proceedings of the 2013 SIAM International Conference on Data Mining, Austin, TX, USA, 2–4 May 2013; SIAM: Philadelphia, PA, USA, 2013, pp. 270–278.
45. Tang, J.; Liu, H. An unsupervised feature selection framework for social media data. *IEEE Trans. Knowl. Data Eng.* **2014**, *26*, 2914–2927. [CrossRef]
46. John, G.H.; Langley, P. Estimating continuous distributions in Bayesian classifiers. In Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, 18 August 1995; Morgan Kaufmann Publishers Inc.: Burlington, MA, USA, 1995, pp. 338–345.
47. Aha, D.W.; Kibler, D.; Albert, M.K. Instance-based learning algorithms. *Mach. Learn.* **1991**, *6*, 37–66. [CrossRef]
48. Chen, H.L.; Yang, B.; Liu, J.; Liu, D.Y. A support vector machine classifier with rough set-based feature selection for breast cancer diagnosis. *Expert Syst. Appl.* **2011**, *38*, 9014–9022. [CrossRef]