Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Survey Track

Survey on Feature Transformation Techniques for Data Streams

Maroua Bahri[1] , **Albert Bifet**[1,2] , **Silviu Maniu**[3,4,5] and **Heitor Murilo Gomes**[2]
[1]LTCI, Télécom Paris, IP-Paris, Palaiseau, France
[2]University of Waikato, Hamilton, New Zealand
[3]Université Paris-Saclay, LRI, CNRS, Orsay, France
[4]ENS DI, ENS, Université PSL, Paris, France
[5]Inria, Paris, France
{maroua.bahri, albert.bifet}@telecom-paris.fr, silviu.maniu@lri.fr, heitor.gomes@waikato.ac.nz

Abstract

Mining high-dimensional data streams poses a fundamental challenge to machine learning as the presence of high numbers of attributes can remarkably degrade any mining task's performance. In the past several years, dimension reduction (DR) approaches have been successfully applied for different purposes (e.g., visualization). Due to their high-computational costs and numerous passes over large data, these approaches pose a hindrance when processing infinite data streams that are potentially high-dimensional. The latter increases the resource-usage of algorithms that could suffer from the curse of dimensionality. To cope with these issues, some techniques for incremental DR have been proposed. In this paper, we provide a survey on reduction approaches designed to handle data streams and highlight the key benefits of using these approaches for stream mining algorithms.

1 Introduction

In the era of Internet-of-Things (IoT) data streams, applications in different domains have seen an explosion of information generated from heterogeneous data sources every day. Because of their unbounded size and infinite nature, data streams cannot be stored entirely in memory or scanned multiple times [Gama *et al.*, 2009]. In addition to the overwhelming volume of data, its dimensionality is increasing considerably in many domains, such as biology, social media, and spams filters. Those high-dimensional data may contain redundant or irrelevant features that can be potentially reduced to a smaller set of relevant features without a significant loss of information.

A natural way to handle such massive high-dimensional data adequately is to apply the learning task, that may suffer from the *curse of dimensionality*[1] [Bellman, 2015], on a *compressed* representation of the data which eases the process by using less computational resources. Thus, a preprocessing step is imperative to filter relevant features and therefore improve the results of a later machine learning task. The latter could be data visualization, noise filtering, or to allow cost and resource savings with data stream mining algorithms. To do so, a synopsis can be constructed from data points (*instances*) in the stream using summarization techniques, such as *sketches* by keeping frequencies of data, selecting a part of incoming data without reducing the number of features, known as sampling, or by applying dimension reduction (DR)[2]. Naturally, the choice of a suitable technique depends on the problem being solved. In what follows, we focus the most common application, DR, which aims to decrease the size of the feature space of data while keeping or extracting a subset of the most relevant features [Sorzano *et al.*, 2014].

As mentioned before, DR is crucial to avoid the curse of dimensionality, which may increase the use of computational resources and negatively affect the predictive performance. Several reduction techniques have been proposed, and widely investigated, in the offline setting [Van Der Maaten *et al.*, 2009; Sorzano *et al.*, 2014] to cope with high-dimensional data. However, these techniques do not adhere to the strict computational resources requirements of the data stream learning framework [Gama *et al.*, 2009]. To overcome this problem, some of these techniques have been adapted to efficiently process streaming data, effectively perform one-pass processing, and adhere to memory and time constraints.

We distinguish two main different dimensionality reduction categories: (i) *feature selection* which consists in selecting a subset of the input features, i.e., the most relevant and non-redundant features, without operating any sort of data transformation; and (ii) *feature transformation*–also called *feature extraction*– which consists in constructing from a set of input features in high-dimensional space, a new set of features in a lower dimensional space [Liu and Motoda, 1998].

Some recent surveys on streaming feature selection have been proposed [Barddal *et al.*, 2017; Hu *et al.*, 2018; Al-Nuaimi *et al.*, 2019].To the best of our knowledge, surveys on streaming feature transformation for do not exist even though the past several years have seen new approaches in this framework. Nevertheless, previous works have provided a general overview limited to summarization techniques (e.g., sampling, sketches), such as [Aggarwal and Philip, 2007; Ikonomovska *et al.*, 2007]. We therefore believe that this re-

[1]It is very challenging to learn in high-dimensional spaces.

[2]Dimensionality reduction, embedding, and manifold learning are the names for similar tasks.

view on feature transformation can provide up-to-date knowledge about recent findings and developments in the field.

Our goal in this paper is to provide a brief overview of the most crucial feature transformation techniques that are–or can be– used in the stream framework and discuss their similarities and differences. We also briefly discuss some promising future research directions.

2 Dimensionality Reduction

DR is a critical part of improving machine learning algorithms' performance. It is defined as the projection of high-dimensional data into a low-dimensional space by reducing the input features to the most relevant ones. As mentioned before, feature transformation and feature selection process data differently. Although both reductions are used to minimize an input feature space size, feature transformation is a DR that creates a new subset – or combinations – of features by exploiting the redundancy and noise of the input set of features. In contrast, feature selection is characterized by keeping the most relevant attributes from the original set of features present in the data without changing them. In what follows, we refer to feature transformation as DR.

Formally, DR consists in finding some transformation function (or *map*) $A : \mathbb{R}^d \rightarrow \mathbb{R}^p$, where $p \ll d$, to be applied on each instance X_i of a data set S.

3 A DR Taxonomy for Data Streams

In this section, we introduce DR techniques that have been widely used in machine learning algorithms. These techniques operate by transforming and using the most relevant feature combinations, in turn reducing space and time demands; this can be crucial for applications such as classification and visualization. Note that DR is in general a very active field recently; however, many such methods have been proposed and used for offline purposes– and for static datasets. Hence, they cannot be used in streaming frameworks, or they have to be adapted.

Figure 1 shows a taxonomy that subdivides the DR techniques as follows: *data-dependent*, *data-independent*, and *graph-based* transformation. The data-dependent techniques are derived from the whole data to achieve the transformation, whereas the data-independent techniques are based on random projections and do not use the input data to perform the projection. Graph-based techniques are also data-dependent that build a neighborhood graph to maintain the data structure (i.e., preserves the neighborhood after projection).

3.1 Data-Dependent Techniques

Data-dependent techniques construct a projection function – or matrix – from the data. This requires the presence of the entirety – or at least a part of – the dataset. In the streaming context, where data are potentially infinite, the classical techniques from this category are therefore limited, since keeping the entire data stream in memory is impractical.

Principal Components Analysis (PCA)

PCA is the most popular and straightforward unsupervised technique that seeks to reduce the space dimension by finding a lower-dimensional basis in which the sum of squared distances between the original data and their projections is minimized, i.e. being as close as possible to zero while maximizing the variances between the first components. Mathematically, PCA aims to find a linear mapping formed by a few orthogonal linear combinations, also called eigenvectors or PCs, from the original data that maximizes a certain cost function. However, PCA computes eigenvectors and eigenvalues from a computed covariance matrix, relying on the whole dataset. This is ineffective for streaming data since a re-estimation of the covariance matrix from scratch for new observations is unavoidable.

In this context, different variations of component analysis have been proposed and adapted to the stream setting. For instance, Incremental PCA (IPCA) [Artac *et al.*, 2002] focuses on how to update the eigenvectors of images (called *eigenimages*) based on the previous coefficients. Candid Covariance-free Incremental PCA (CCIPCA) [Weng *et al.*, 2003] is another extension that updates the eigenvectors incrementally and does not need to compute the covariance matrix for each new incoming instance (each instance being an image) which makes it very fast. The main difference among these techniques arises is in how eigenvectors are updated. On the other hand, the common limitation concerns their application domain since both techniques deal with images as high-dimensional vectors and have not been tested on different types of data.

Ross *et al.* proposed a batch-incremental PCA that deals with a set of new instances each time a batch is complete. However, this approach is not suited for instance-incremental learning (i.e., processing instances one by one incrementally). Mitliagkas *et al.* proposed a memory-limited streaming PCA that attempts to make vanilla PCA incremental and computation-efficient with high-dimensional data. To achieve this, samples are drawn from a Gaussian spiked covariance model. A more recent work [Yu *et al.*, 2017] proposes a single-pass randomized PCA technique that iteratively updates the subspace's orthonormal basis matrix within an accuracy-performance trade-off. Yu *et al.* claim that this technique works well in many applications, albeit it has been evaluated only on a single image dataset.

The above PCA techniques apply to data stream mining algorithms to alleviate their computation costs. For instance, Feng *et al.* proposed an efficient online classification algorithm, FIKOCFrame, that uses a PCA variant, fast iterative kernel PCA [Günter *et al.*, 2007], to incrementally reduce the dimensionality before classification.

Cardot and Degras proposed recently a comparative review of the incremental PCA approaches where they provide guidance for selecting the appropriate approach based on their accuracy and computation resources (time and memory).

Multi-Dimensional Scaling (MDS)

MDS [Wickelmaier, 2003] is a well-known unsupervised technique used for embedding. It projects a given distance matrix into a non-linear lower-dimensional space while preserving the similarity among instances. Nevertheless, this technique is computationally expensive with large datasets and non-scalable because it requires the entire data distance matrix. Incremental versions have been proposed to alleviate

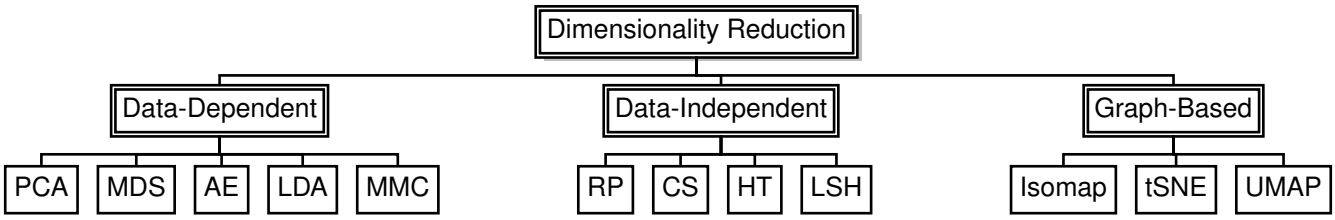


Figure 1: Taxonomy of dimensionality reduction techniques.

the computational requirements.

Incremental MDS (iMDS) technique, proposed by Agarwal *et al.*, keeps some distance preservation using the so-called *out of sample* mapping without the need of reconstructing the whole matrix. A more recent work by Zhang *et al.* proposed a new version of MDS for high-dimensional data, named *sc*MDS. It is a batch-incremental technique that introduces a realignment matrix for each batch to overcome the concept drift that may occur due to different batches that have different feature bases. Nevertheless, the efficiency of this batch-incremental technique depends on the size of the batch.

Auto-Encoder (AE)

AEs are a family of Neural Networks (NNs) which are designed for unsupervised learning, for learning a low-dimensional representation of a high-dimensional dataset, where the input is the same as the output. An AE has two main components, (i) the *encoder* step, during which the input data are compressed into a latent space representation; and (ii) the *decoder* step where the input data are reproduced from this new representation. Vincent *et al.* introduced the denoising AE (DAE), a variant of AE, that extracts features by adding perturbations to the input data and then attempts to reconstruct the original data. Zhou *et al.* proposed an online DAE that adaptively uses incremental feature augmentation, depending on the already existing features, to track drifts. However, this work does not address the convergence properties of the training task (the hyperparameters configuration used to construct the network, e.g., the number of epochs) that are crucial in the stream setting.

Unlike other algorithms, NNs naturally handle incremental learning tasks [Dong and Japkowicz, 2016]. While dealing with data streams, NNs learn by passing the data in smaller chunks (Mini-Batch Gradient Descent) or an instance at a time (Stochastic Gradient Descent). Using this way, each instance is going to be processed only once without being stored. The advantage of using this kind of technique is that it is not limited to linear transformations. Non-linearities are introduced using non-linear activation functions, NNs are therefore more flexible. Nevertheless, this high-quality results that this family of learners offers come at the price of slow learning speed due to the infinite nature of data and the large parameter space needed.

Linear Discriminant Analysis (LDA)

LDA [McLachlan, 2004], also known as Fisher Discriminant Analysis (FDA), is a linear transformation technique. Contrary to the techniques mentioned earlier, LDA performs a supervised reduction that takes into account the class labels of instances by looking for efficient discrimination of data in a way to maximize the separation of the existing categories (class labels), while other techniques, e.g. PCA, aim at an efficient representation. However, when dealing with evolving data streams, the set of labels of instances may be unknown at each learning stage because new classes may appear (concept evolution) [Haque *et al.*, 2016].

One way to cope with this issue is to update the discriminant eigenspace when a new class arrives, as introduced in the Incremental LDA (ILDA) approach [Pang *et al.*, 2005]. Another streaming extension of LDA has been proposed, called IDR/QR [Ye *et al.*, 2005]. It applies a singular value decomposition suitable for large datasets that uses less computational cost than ILDA. Kim *et al.* proposed an ILDA that incrementally updates the discriminant components using a different criterion. They claim to be more efficient in terms of time and memory than the previous approaches.

Maximum Margin Criterion (MMC)

MMC [Li *et al.*, 2004] is a supervised feature extractor technique based on the same representation of LDA while maximizing a different objective function. To overcome the limitations of MMC with streaming data, Yan *et al.* proposed an Incremental MMC (IMMC) approach, which infers an online adaptive supervised subspace from data streams by optimizing the MMC and updating the eigenvectors of the criterion matrix incrementally. Hence, the computation of IMMC is very fast since it does not need to reconstruct the criterion matrix when new instances arrive.

The incremental formulation of the proposed algorithm is mentioned in [Yan *et al.*, 2004] with the proof. A major drawback of this approach is its sensitivity to parameter setting.

3.2 Data-Independent Techniques

Data-independent techniques are mainly based on the principle of random projections. These techniques are therefore appropriate for evolving streams because they generate the projection matrices (or functions), and transform data into a low-dimensional space, independently from the input data.

Random Projection (RP)

Random projection is a powerful technique for dimensionality reduction that has been widely applied with several mining algorithms for solving numerous problems [Vempala, 2005]. RP is based on the Johnson-Lindenstrauss (JL) lemma 1 which asserts that N instances from a Euclidean space can be projected into a lower-dimensional space of $\mathcal{O}(\log(N/\epsilon^2))$ dimensions under which pairwise distances are preserved within a multiplicative factor of $1 \pm \epsilon$.

Lemma 1 Let $\epsilon \in [0,1]$, $X = \{x_1, ..., x_N\} \in \mathbb{R}^d$. Given a number $p \geq \log(N/\epsilon^2)$, $\forall x_i, x_j \in X$ there is a linear map $A : \mathbb{R}^d \rightarrow \mathbb{R}^p$ such that:

$$(1-\epsilon)\|x_i - x_j\|_2^2 \leq \|Ax_i - Ax_j\|_2^2 \leq (1+\epsilon)\|x_i - x_j\|_2^2, \quad (1)$$

where A is a random matrix that can be generated using, e.g., a Gaussian distribution.

Hence, RP offers a computationally-efficient and straightforward way to compress the dimension of input data rapidly while approximately preserving the pairwise distances between any two instances. However, this technique sometimes leads to a slight loss in accuracy.

Compressed Sensing (CS)

Compressed sensing, also called compressed sampling, technique based on the principle that a data compression method has to deal with redundancy while transforming and reconstructing data [Donoho, 2006]. Given a sparse and high-dimensional vector $x \in \mathbb{R}^d$, the goal of CS is to measure $y \in \mathbb{R}^p$ and then reconstruct x, for $p \ll d$, as $y = Ax$, where $A \in \mathbb{R}^{p \times d}$ is called a *sampling* or *sensing* matrix.

The technique has been widely studied and used throughout different domains in the offline framework, such as image processing [Qiu *et al.*, 2009]. The basic idea is to use orthogonal features to provably and properly represent sparse and high-dimensional vectors $x \in \mathbb{R}^d$ as well as reconstruct them from a small number of feature vectors $y \in \mathbb{R}^p$, where $p \ll d$. Two main concepts are crucial to the recovery of the stream with high probability, (i) the *sparsity:* CS exploits the fact that the input data x may be s-sparse; and (ii) the *Restricted Isometry Property (RIP):* the matrix A is said to respect the RIP for all s-sparse data if there exists $\epsilon \in [0,1]$ such that:

$$(1-\epsilon)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1+\epsilon)\|x\|_2^2, \quad (2)$$

where $x \in X$.

RP and CS are closely related. Random matrices (e.g., Bernoulli, Binomial, Gaussian) are also known to satisfy the RIP with high probability if $p = \mathcal{O}(s\log(d))$ [Achlioptas, 2003], which is essentially a JL type condition on projections using the sensing matrix A. The difference is mainly in terms of how big the matrix A has to be. On the other hand, CS could be data-dependent when we use Fourier transform to obtain the sensing matrix from data. In [Freris *et al.*, 2013], authors proposed a recursive scheme for using Fourier matrices with data streams which constructs successive windows and uses the measurement in the previous window to obtain the next one. However, this approach is expensive in terms of memory since it keeps data on windows and it is still not as accurate as using a Gaussian matrix [Arjoune *et al.*, 2018].

Hashing Trick (HT)

Hashing trick [Weinberger *et al.*, 2009], also known as feature hashing, is a fact and space-efficient technique that projects sparse instances or vectors into a lower feature space using a hash function. Given a list of keys that represents a set of features from the input instances, it computes then the hash function for each key, which will ensure its mapping to a specific cell of a fixed size vector that constitutes the new compressed instance. The HT technique has the appealing properties of being very fast, simple, and sparsity preserving. A significant advantage to point out is that this technique is very memory-efficient because the output feature vector size is limited, making it a clear candidate for using, especially for online learning on streams. This technique has been used in conjunction with stream mining algorithms. For instance, Bahri *et al.* proposed recently a naive Bayes approach that uses HT technique to alleviate its computational resources on sparse high-dimensional data streams.

Locality Sensitive Hashing (LSH)

The basic idea behind LSH technique [Datar *et al.*, 2004] is the application of hashing functions which map, with high probability, similar instances (in the high-dimensional d-space) that have the same hash code to the same bucket. I.e., if instances are phrases that are very similar to each other, they might be different by only one or a couple of words or even one character; hence, LSH will generate very similar, ideally, identical hash codes to increase the probability of collision for those instances. LSH operates by partitioning the space with hyperplanes into disjoint regions, which are spatially proximate. A particular hyperplane is going to cut the space into two half-spaces, and arbitrarily one side is called positive "1" and the other side negative "0"; this will help in classifying the instances for that dimension. The process is iterative: the first bit in the hash code of an instance is assigned with respect to its position. Then, the process keeps cutting the space and assigning bits the same way. Therefore, we obtain the hash codes based on the bits assigned after each hyperplane.

There is an efficiency-resource tradeoff with LSH. To achieve good accuracy, it requires the use of several hash functions; consequently, the memory increases, which will slow down the reduction process and make it less suitable with large data. LSH is used in several interesting real-word applications: Netflix users with similar tastes in movies for recommendation systems, plagiarism based on a body of documents, or finding similar text. It is also used in classification tasks, e.g., topic classification of pages, by exploiting the fact that pages on the same topic will contain similar words.

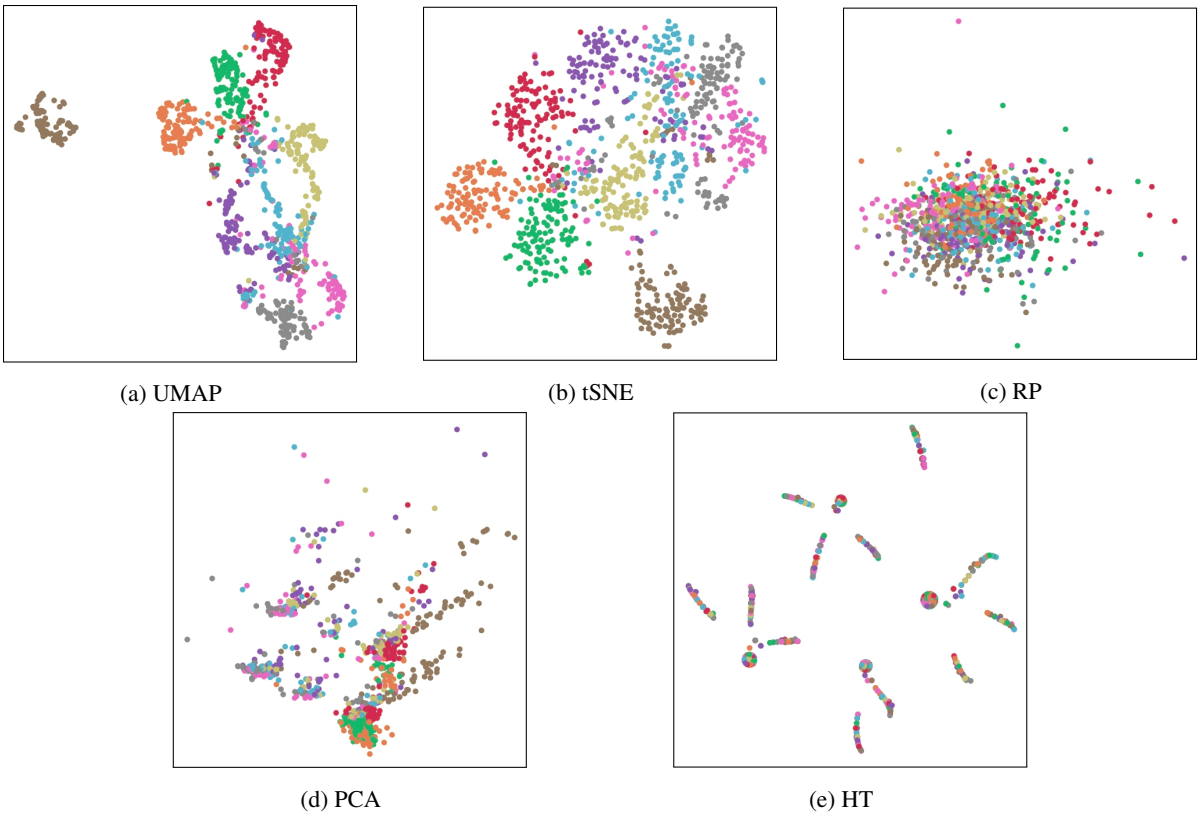
3.3 Graph-Based Techniques

Graph-based techniques are data-dependent techniques that start by constructing a graph based on instance similarities and then operate on this representation.

Isometric Mapping (Isomap)

Isomap is a manifold learning technique that can be viewed as a combination of the principles of PCA and MDS. It starts by building a neighborhood graph on the manifold from which a geodesic distance matrix is constructed. Isomap assumes that pairwise geodesic distances are equal to Euclidean ones (obtained by applying the MDS on the resulting geodesic distance matrix) in the low-dimensional space. Since it requires the computation of pairwise distances, Isomap is thus not appropriate for the incremental setting with large datasets.

Law *et al.* proposed a streaming version of Isomap that updates the geodesic distances and the coordinates incrementally. This technique is not fully incremental because a new instance can affect the neighborhood structure and, therefore, the geodesic matrix. Thus, there is a need to examine how this



(a) UMAP (b) tSNE (c) RP

(d) PCA (e) HT

Figure 2: Dimensionality reduction techniques while projecting CNAE dataset in 2-dimensional space.

new instance interacts with the existing ones before finding its coordinates. Another incremental Isomap, denoted S-Isomap, has been proposed lately by Schoeneman *et al.* which does not recompute the whole geodesic distance matrix when a new instance arrives, but only finds its nearest neighbors (that will be used to approximate the geodesic distance between this new observation and the others already available in the batch). This approach fails when used to process data because it assumes that the data are weakly correlated, and thus unable to detect when concept drift takes place.

t-distributed Stochastic Neighbor Embedding (tSNE)

tSNE [Maaten and Hinton, 2008] is one of the most prominent DR techniques in the state-of-the-art. It is a graph-based non-linear technique proposed to visualize high-dimensional data embedded in a lower-space (typically 2 or 3 dimensions) by using the insight that similar instances in the high-dimensional space should be represented by close instances in the low-dimensional space. tSNE uses a fixed parameter named perplexity similar to the number of neighbors that controls the neighborhood size of each node in the graph, which prevents it from preserving global data structure. The main weakness of tSNE in our context is about the scalability, i.e. to add more instances, we need to re-run tSNE from scratch.

Uniform Manifold Approximation & Projection (UMAP)

UMAP [McInnes *et al.*, 2018] is a new manifold technique, similar to tSNE, that has attracted attention recently and is built upon rigorous mathematical foundation through the Riemannian geometry. UMAP starts by constructing open balls over all instances and building simplicial complexes. The space reduction is obtained by finding a representation, in a lower-space, that closely resembles the topological structure in the original space. Given the new dimension, an equivalent fuzzy topological representation is then constructed. Afterward, UMAP optimizes it by minimizing the cross-entropy between these two fuzzy representations. In addition to being faster than tSNE, UMAP offers also a better visualization quality by preserving more of the global structure. Unlike tSNE, UMAP has no restriction on the projected space size making it useful not only for visualization, but also as a general DR technique for mining algorithms.

A batch-incremental strategy has been proposed recently to build manifolds on small chunks then used for classification through a streaming lazy algorithm [Bahri *et al.*, 2020]. However, this technique is still not fully incremental and the batch-incremental manner slows down the whole process.

4 Research Challenges and Open Directions

This survey aims at providing a literature review for progress in dimensionality reduction techniques. In this section, we briefly provide some promising future research directions. To the best of our knowledge, tSNE and UMAP do not have any fully incremental version, ultimately, both techniques are essentially transductive[3] and do not learn a mapping function from the input space. Hence, they need to process all the

[3]Transductive learning consists on learning on a dataset but predicting on a known set of unlabeled instances from the same dataset.

instances for each new unseen observation, which prevents them from being applicable within a data stream framework.

Figure 2 shows the performance of different DR techniques on CNAE dataset[4] while projecting into a 2-dimension space in an offline fashion. This dataset consists of 9 classes, where each color represents a class label. We notice that UMAP in Figure (a) offers the most interesting visualization while separating classes (already proved in [McInnes *et al.*, 2018] with different data). tSNE (Figure (b)) offers a less visible separation than UMAP because it does not preserve the global structure of data. On the other hand, we see a lot of overlapping with RP, PCA, and HT in Figures (c), (d), and (e), respectively, because of their linear nature. Figure 2 (e) shows the projection with HT, we notice kind of points because HT maps features to the same cells (collisions) depending on the hash function being used. Besides, the overlap after the transformation can potentially affect any later learning task, notably distance or neighborhood-based algorithms. We see that in contrast to linear techniques, nonlinear techniques have the ability to offer an advantage when dealing with complex data.

5 Discussion

DR plays a significant role in the data stream mining area since it aims at keeping the most relevant features to reduce the computational cost of stream mining algorithms and make the structure effective for visualization (see Figure 3).

Techniques such as PCA, LDA, and MDS are the most classical ones for DR. As we mentioned before in Section 3, some versions of the data-dependent techniques have been proposed to deal with evolving data streams. Nevertheless, this category of techniques usually provides good accuracy when combined with stream data mining algorithms. On the other hand, data-independent techniques are naturally adapted to the evolving environment of data streams and do not suffer from the scalability problem. Moreover, using data-independent techniques is extremely fast because it is performed without including the input data content. This transformation performs as well, if not better, as data-dependent transformation because it is less sensitive to new unseen instances and could benefit from the infinite nature of streams. Sometimes, data-independent schemes could destroy any interpretability in the case of visualization (e.g., Figures 2 (c) and (d)). Thus, the choice of the (data-dependent or data-independent) technique poses an accuracy-resource tradeoff that may depend on the problem being solved and the algorithm used (e.g., use a graph-based manner for visualization to preserve the neighborhood and the global structure of data).

6 Concluding Remarks

In this survey, we provided a literature review that surveys the vast set of dimensionality reduction techniques for streaming – online or incremental – data. We proposed a simple and useful taxonomy of existing works where we differentiated between data-dependent and data-independent techniques and their impact on data stream mining algorithms. This can help the readers to choose suitable dimensionality reduction techniques for their tasks. We also discussed some promising future research directions which concern the adaptation of two powerful techniques, tSNE and UMAP, to the stream setting.

[4]https://archive.ics.uci.edu/ml/datasets/CNAE-9

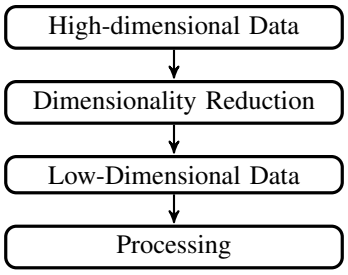


Figure 3: The goal of dimensionality reduction.

Acknowledgements

This work was done in the context of IoTA AAP Emergence DigiCosme Project and was funded by Labex DigiCosme.

References

[Achlioptas, 2003] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *JCSS*, 66(4):671–687, 2003.

[Agarwal *et al.*, 2010] Arvind Agarwal, Jeff M Phillips, Hal Daumé III, and Suresh Venkatasubramanian. Incremental multidimensional scaling. In *LW*, 2010.

[Aggarwal and Philip, 2007] Charu Aggarwal and S Yu Philip. A survey of synopsis construction in data streams. In *Data Streams*, pages 169–207. Springer, 2007.

[AlNuaimi *et al.*, 2019] Noura AlNuaimi, Mohammad Mehedy Masud, Mohamed Adel Serhani, and Nazar Zaki. Streaming feature selection algorithms for big data: A survey. *ACI*, 2019.

[Arjoune *et al.*, 2018] Youness Arjoune, Naima Kaabouch, Hassan El Ghazi, and Ahmed Tamtaoui. A performance comparison of measurement matrices in compressive sensing. *IJCS*, 31(10):e3576, 2018.

[Artac *et al.*, 2002] Matej Artac, Matjaz Jogan, and Ales Leonardis. Incremental pca for on-line visual learning and recognition. In *Object Recognition*, pages 781–784, 2002.

[Bahri *et al.*, 2018] Maroua Bahri, Silviu Maniu, and Albert Bifet. Sketch-based naive bayes algorithms for evolving data streams. In *ICBD*, pages 604–613. IEEE, 2018.

[Bahri *et al.*, 2020] Maroua Bahri, Bernhard Pfahringer, Albert Bifet, and Silviu Maniu. Efficient batch-incremental classification using umap for evolving data streams. In *IDA*, pages 40–53, 2020.

[Barddal *et al.*, 2017] Jean Paul Barddal, Heitor Murilo Gomes, Fabrício Enembreck, and Bernhard Pfahringer. A survey on feature drift adaptation: Definition, benchmark, challenges and future directions. *JSS*, 127:278–294, 2017.

[Bellman, 2015] Richard E Bellman. *Adaptive control processes: a guided tour*. Princeton university press, 2015.

[Cardot and Degras, 2018] Hervé Cardot and David Degras. Online principal component analysis in high dimension: which algorithm to choose? *ISR*, 86(1):29–50, 2018.

[Datar *et al.*, 2004] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *SoCG*, 2004.

[Dong and Japkowicz, 2016] Yue Dong and Nathalie Japkowicz. Threaded ensembles of supervised and unsupervised nns for stream learning. In *CAIAC*, 2016.

[Donoho, 2006] David L Donoho. Compressed sensing. *TIT*, 52(4):1289–1306, 2006.

[Feng *et al.*, 2009] Wu Feng, Zhong Yan, Li Ai-ping, and Wu Quan-yuan. Online classification algorithm for data streams based on fast iterative kernel pca. In *ICNC*, 2009.

[Freris *et al.*, 2013] Nikolaos M Freris, Orhan Oçal, and Martin Vetterli. Compressed sensing of streaming data. In *Allerton*, pages 1242–1249. IEEE, 2013.

[Gama *et al.*, 2009] João Gama, Raquel Sebastião, and Pedro Pereira Rodrigues. Issues in evaluation of stream learning algorithms. In *SIGKDD*. ACM, 2009.

[Günter *et al.*, 2007] Simon Günter, Nicol N Schraudolph, and SVN Vishwanathan. Fast iterative kernel principal component analysis. *JMLR*, 8(8):1893–1918, 2007.

[Haque *et al.*, 2016] Ahsanul Haque, Latifur Khan, Michael Baron, Bhavani Thuraisingham, and Charu Aggarwal. Efficient handling of concept drift and concept evolution over stream data. In *ICDE*, pages 481–492. IEEE, 2016.

[Hu *et al.*, 2018] Xuegang Hu, Peng Zhou, Peipei Li, Jing Wang, and Xindong Wu. A survey on online feature selection with streaming features. *FCS*, 12(3):479–493, 2018.

[Ikonomovska *et al.*, 2007] Elena Ikonomovska, Suzana Loskovska, and Dejan Gjorgjevik. A survey of stream data mining. In *ETAI*, pages 19–21, 2007.

[Kim *et al.*, 2011] Tae-Kyun Kim, Björn Stenger, Josef Kittler, and Roberto Cipolla. Incremental linear discriminant analysis using sufficient spanning sets and its applications. *IJCV*, 91(2), 2011.

[Law *et al.*, 2004] Martin HC Law, Nan Zhang, and Anil K Jain. Nonlinear manifold learning for data stream. In *ICDM*, 2004.

[Li *et al.*, 2004] Haifeng Li, Tao Jiang, and Keshu Zhang. Efficient and robust feature extraction by maximum margin criterion. In *NIPS*, pages 97–104, 2004.

[Liu and Motoda, 1998] Huan Liu and Hiroshi Motoda. *Feature extraction, construction and selection: A data mining perspective*, volume 453. Springer, 1998.

[Maaten and Hinton, 2008] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008.

[McInnes *et al.*, 2018] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. 2018.

[McLachlan, 2004] Geoffrey J McLachlan. *Discriminant analysis and statistical pattern recognition*. Wiley, 2004.

[Mitliagkas *et al.*, 2013] Ioannis Mitliagkas, Constantine Caramanis, and Prateek Jain. Memory limited, streaming pca. In *NIPS*, pages 2886–2894, 2013.

[Pang *et al.*, 2005] Shaoning Pang, Seiichi Ozawa, and Nikola Kasabov. Incremental linear discriminant analysis for classification of data streams. *TSMC*, 35(5), 2005.

[Qiu *et al.*, 2009] Chenlu Qiu, Wei Lu, and Namrata Vaswani. Real-time dynamic mr image reconstruction using kalman filtered compressed sensing. In *ICASSP*, 2009.

[Ross *et al.*, 2008] David A Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–141, 2008.

[Schoeneman *et al.*, 2017] Frank Schoeneman, Suchismit Mahapatra, Varun Chandola, Nils Napp, and Jaroslaw Zola. Error metrics for learning reliable manifolds from streaming data. In *ICDM*, pages 750–758. SIAM, 2017.

[Sorzano *et al.*, 2014] Carlos Oscar Sánchez Sorzano, Javier Vargas, and A Pascual Montano. A survey of dimensionality reduction techniques. 2014.

[Van Der Maaten *et al.*, 2009] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative. *JMLR*, 10(66-71):13, 2009.

[Vempala, 2005] Santosh S Vempala. *The random projection method*, volume 65. American Mathematical Soc, 2005.

[Vincent *et al.*, 2008] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103, 2008.

[Weinberger *et al.*, 2009] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *ICML*, pages 1113–1120, 2009.

[Weng *et al.*, 2003] Juyang Weng, Yilu Zhang, and Wey-Shiuan Hwang. Candid covariance-free incremental principal component analysis. *TPAMI*, 2003.

[Wickelmaier, 2003] Florian Wickelmaier. An introduction to mds. *Sound Quality Research Unit*, 46(5):1–26, 2003.

[Yan *et al.*, 2004] Jun Yan, Benyu Zhang, Shuicheng Yan, Qiang Yang, Hua Li, Zheng Chen, Wensi Xi, Weiguo Fan, Wei-Ying Ma, and Qiansheng Cheng. Immc: incremental maximum margin criterion. In *SIGKDD*, 2004.

[Ye *et al.*, 2005] Jieping Ye, Qi Li, Hui Xiong, Haesun Park, Ravi Janardan, and Vipin Kumar. Idr/qr: An incremental dimension reduction algorithm via qr decomposition. *TKDE*, pages 1208–1222, 2005.

[Yu *et al.*, 2017] Wenjian Yu, Yu Gu, Jian Li, Shenghua Liu, and Yaohang Li. Single-pass pca of large high-dimensional data. *IJCAI*, pages 1453–1461, 2017.

[Zhang *et al.*, 2018] Xi Zhang, Hao Huang, Klaus Mueller, and Shinjae Yoo. Streaming classical multidimensional scaling. In *NYSDS*, pages 1–2. IEEE, 2018.

[Zhou *et al.*, 2012] Guanyu Zhou, Kihyuk Sohn, and Honglak Lee. Online incremental feature learning with denoising autoencoders. In *AISTATS*, 2012.