KARYA ILHAM PUBLISHING
CONNECTING MINDS, BRIDGING IDEAS, INSPIRING READERS

Journal of Advanced Research in Computing and Applications

Journal homepage:
https://karyailham.com.my/index.php/arca
2462-1927

Enhancing Deep Learning Models through Tensorization: A Comprehensive Survey and Framework

Manal Helal[1,*]

[1] School of Physics, Engineering & Computer Science, Hertfordshire University, HATFIELD, United Kingdom

ARTICLE INFO

Article history:
Received 27 November 2025
Received in revised form 20 February 2026
Accepted 15 April 2026
Available online 16 June 2026

Keywords:
Artificial intelligence; blind source separation; image denoising; CP decomposition; neural network compression; Tensor Network (TN); Tensor Train (TT) decomposition; tensorization; Tucker decomposition; Singular Value Decomposition (SVD)

ABSTRACT

In modern machine learning, data are often simplified to 2-dimensional matrices for ease of application in linear algebra-based algorithms despite being inherently high-dimensional. However, applying multiway analysis through multilinear algebra to these multidimensional datasets provides more expressive models, reduces parameters, and accelerates processing, defying the expected dimensionality curse. This paper surveys the theoretical background necessary for understanding these methods, outlines the process of tensorizing matrix-form datasets, and reviews current methods and applications of multiway analysis in compressing deep learning models. It includes a framework for tensorization, a case study on Blind Source Separation, and a comprehensive review of tensorized machine learning and deep learning applications, concluding with key insights and future research directions.

1. Introduction

In modern machine learning, data are often simplified to 2-dimensional matrices for ease of application in linear algebra-based algorithms despite being inherently high-dimensional. However, applying multiway analysis through multilinear algebra to these multidimensional datasets provides more expressive models, reduces parameters, and accelerates processing, defying the expected dimensionality curse. This paper surveys the theoretical background necessary for understanding these methods, outlines the process of tensorizing matrix-form datasets, and reviews current methods and applications of multiway analysis in compressing deep learning models. It includes a framework for tensorization, a case study on Blind Source Separation, and a comprehensive review of tensorized machine learning and deep learning applications, concluding with key insights and future research directions.

* *Corresponding author.*
E-mail address: m.helal@herts.ac.uk

https://doi.org/10.37934/arca.43.1.161173

2. Fundamentals of Tensorization

Traditional machine learning (ML) models, including support vector machines (SVMs), regression models, decision trees, and various deep neural network (DNN) architectures like fully connected layers (FCNs), convolutional layers (CNNs), LSTMs, and transformers, are predominantly grounded in linear algebra. This typically necessitates representing data in a 2-dimensional matrix form, which is a simplified view of inherently high-dimensional data. For instance, image data are often represented with two spatial dimensions (width and height), with additional colour channels forming a third dimension and video data introducing a fourth temporal dimension. While standard ML and DNN algorithms manage these dimensions within the constraints of their design, such as using 1D-CNNs for sequential data and 2D-CNNs for spatial data, they often limit the potential to capture higher-order interactions within the data due to their reliance on lower-dimensional representations.

Tensorization, however, offers a more sophisticated approach by leveraging multilinear algebra to handle data in its full multidimensional form, thereby enabling more expressive models that can capture complex interactions among data dimensions. For instance, while traditional linear methods like PCA and SVD project high-dimensional data onto a lower-dimensional space, tensor methods maintain the intrinsic multiway structure of the data. This is particularly useful in deep learning, where tensor decomposition techniques can be employed to optimise model performance, reduce the number of parameters, and enhance computational efficiency, as seen in applications like Blind Source Separation.

The mathematical foundation of tensorization is rooted in spaces that extend beyond basic Euclidean geometry, such as Riemannian and Hilbert spaces, which facilitate advanced geometric and algebraic operations on data. **Table 1** summarises the foundation of multilinear geometric spaces, ML applications, and metrics for analysis. These concepts are crucial for developing and applying manifold learning algorithms like t-SNE and UMAP and understanding complex datasets' underlying structures. Moreover, tensor methods are pivotal in modern machine learning applications, including graph neural networks, variational autoencoders, and generative models like Wasserstein GANs, where they enable more accurate and efficient data processing by preserving the inherent multidimensional relationships within the data.

Table 1
Multilinear Algebra and tensors

Mathematical Space	Geometric Space Description	ML Applications	Similarity or Distance Measure
Euclidean Space	Flat space is defined by standard Cartesian coordinates.	Feature space for many algorithms, such as k-NN, SVM, and linear models.	Euclidean distance: $d(x,y) = \sqrt{\sum(x_i - y_i)^2}$
Manifolds	Generalised spaces that locally resemble Euclidean space but can have complex global structures.	Manifold learning, t-SNE, UMAP, LLE.	Geodesic distance (shortest path on the manifold).
Hilbert Space	Complete, infinite-dimensional inner product space.	Kernel methods in SVM, PCA in high-dimensional spaces, quantum computing.	Inner product: $\langle x, y\rangle = \sum x_i y_i$.
Curves using Hyperbolic	Hyperbolic: negatively curved space. Elliptic: positively curved space.	Hyperbolic embeddings for hierarchical data,	Hyperbolic 2d distance using the Poincaré disk model: $d(u,v) = 2\ arsinh(\frac{\|v-u\|}{2\sqrt{y_1 y_2}})$ where Euclidean

and Elliptic Geometry		elliptic geometry for spherical data.	distance $\|v-u\| = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$ or for 3D $d(u,v) = 2\,arsinh(\frac{\|v-u\|}{2\sqrt{z_1 z_2}})$, or Elliptic distance on a sphere of radius R such as $d_E(u,v) = R \cdot \cos^{-1}\left(\frac{u \cdot v}{R^2}\right)$.
Riemannian Geometry	Study of smooth manifolds with Riemannian metrics, describing how distances and angles are measured.	Riemannian manifold optimization, GCNs.	Riemannian distance: involves integrating the metric tensor along a curve between points u, v on Manifold M with g Riemannian metric as the infimum of the lengths of all smooth curves connecting u and v along parameter t as γ(0)=u, γ(1)=v, and $\dot{\gamma}(t)$ is the tangent vector to the curve γ at t: $d_g(u,v) = \int_u^v \sqrt{g_{\gamma(t)}(\dot{\gamma}(t).\dot{\gamma}(t))}\,dt$
Differential Geometry on Manifolds	Study of curves, surfaces, and their higher-dimensional analogues using calculus.	Advanced manifold learning, optimisation on curved spaces.	Geodesic distance, curvature-based measures.

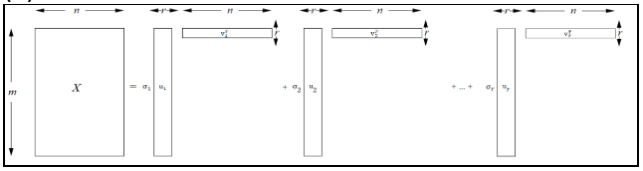
3. Survey of Existing Approaches

This section delves into various methodologies for multiway analysis and their applicability to tensorized datasets. The first subsection provides an overview of multiway analysis methods, drawing from foundational research and notable surveys, including [21] and Chapter 4 of [2]. Following this, we explore tensorization techniques for both 2-dimensional and multiway datasets.

3.1 Multiway Analysis Methods

Multiway analysis methods encompass a variety of tensor-based algorithms—such as factorisation, regression, clustering, and completion—that analyse data across multiple dimensions (modes). These algorithms offer insights similar to those of conventional machine learning methods like PCA and SVD. Still, they can also serve as a pre-processing step for tensorized or non-tensorized machine learning (ML) and deep learning (DL) models.

Tensor decompositions are essential in reducing the dimensionality of tensors and identifying dominant factors within them. While methods like SVD and PCA are widely used, they often fail to capture the non-linear structure of data. In contrast, techniques such as Multidimensional Scaling (MDS), Isomap, Locally Linear Embedding, and Spectral Clustering preserve or learn the non-linear manifold of the dataset. The following tensor decomposition methods excel at capturing complex interactions within high-dimensional datasets:

Tensor decomposition method	Illustration
Candecomp/Parafac (CP) Decomposition: As a multiway extension of SVD, CP decomposition factorises a tensor into a sum of rank-one tensors, enabling the reconstruction of the original tensor from its dominant components. This method generalises the SVD approach to N-dimensional	(a)  $X = USV^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$

tensors, capturing the essential structure of the data.

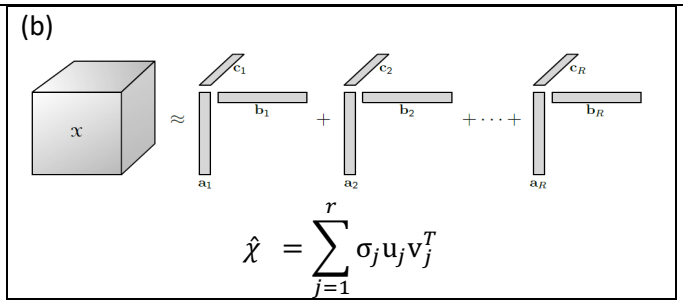


Fig. 1 **:** (a) 2-D SVD, (b) 3-D SVD / CP generalises to the higher dimension [1]

Tucker Decomposition: Known as a higher-order PCA, Tucker decomposition decomposes a tensor into a core tensor, which is then multiplied by factor matrices along each mode. Unlike CP decomposition, Tucker retains a richer structure within the core tensor, allowing for more nuanced data representation.

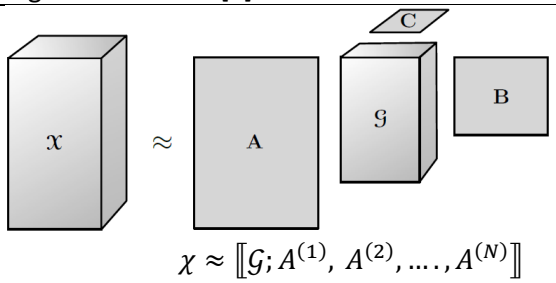


Fig. 2: Tucker Decomposition

Tensor Networks: Tensor networks hierarchically represent large-scale tensors using lower-rank core tensors. Common approaches include Tensor Train (TT), Tensor Ring (TR), and Matrix Product States (MPS), among others. These methods are particularly effective for handling large-scale data by reducing the computational complexity associated with high-dimensional tensors.

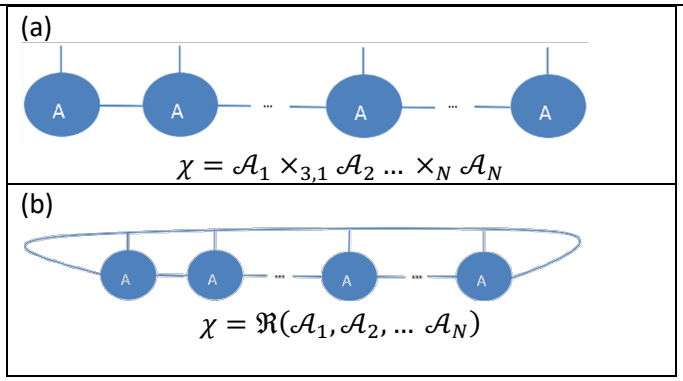


Fig. 3. (a) TT decomposition, (b) TR decomposition

Tensor Completion: Tensor completion extends matrix completion techniques to multi-dimensional data, aiming to interpolate missing values within a tensor. Methods like Tensor Decomposition with Relational Constraints (TDRC) enhance traditional tensor decomposition by incorporating auxiliary data, such as similarity matrices, to improve prediction accuracy in applications like miRNA-disease association studies.

Tensor Regression: Tensor regression models generalise linear regression to Nth-order tensors as $\mathcal{Y} = f(\mathcal{X}) + \epsilon$, allowing for the mapping of high-dimensional predictors to target variables. Techniques like CP and Tucker regression reduce the number of parameters needed, making it feasible to work with large, multi-dimensional datasets such as MRI scans while preserving the data's inherent structure.

Tensor Clustering: Tensor clustering is an unsupervised learning approach that identifies clusters within tensor data X by factorising the data matrix into a canonical basis vector A, in which each row selects a row in B, which contains the clustering vectors, $X \approx AB^T$. Methods exist to estimate the two unknowns (A and B) from X, such as Independent Component Analysis (ICA) and dictionary learning algorithms adapted to handle tensor data. These algorithms facilitate the discovery of underlying patterns within high-dimensional datasets.

3.2 Multiway (Tensorized) Dataset Sources

This section explores sources of tensorized data, including traditional datasets that can be transformed into tensor form, as well as naturally occurring multiway datasets.

Traditional Datasets: Public datasets from platforms like Kaggle and UCI can be tensorized through data fusion techniques. Understanding the different modes within these datasets allows for the integration or segmentation of data to meet specific application requirements, such as combining outcomes from different hospital trials.

Graphs and Networks Datasets: High-dimensional datasets such as Wireless Sensor Networks (WSN) and Knowledge Graphs are naturally suited for tensor representation. For example, WSN data can be represented as a tensor capturing sensors, base stations, clusters, and messages, while Knowledge Graphs can be modelled as tensors with modes representing entities and relationships.

Image and Video Datasets: Image and video datasets, like MNIST and video files, benefit significantly from tensor representation. Tensorizing these datasets preserves spatial and temporal information, enabling more efficient processing in convolutional neural networks (CNNs) and other tensor-based models. **Fig**. 4 presents an example of a 3D grey-shade video of 128 x 88 image size and 20 frames. Multiplying the shape vector for pair-wise Linear Subspace Learning (LSL) vectorisation in (a) produced a massive 189 GB large covariance matrix in memory and the required computational time. Applying a tensorial Multi-linear Subspace Learning (MSL) in (b) required summing three smaller covariance matrices, producing 95.8KB of memory and is both memory and computational time efficient [2].

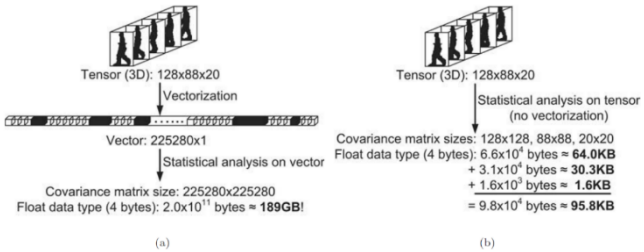


Fig. 4. (a) pair-wise approaches flatten datasets vs (b) tensorised data approaches compression example [2]

Health and Biomedical Datasets: Biomedical datasets, such as EEG signals and MRI scans, are inherently multi-dimensional. These datasets can be effectively analysed using multiway data analysis techniques, which capture the complex interactions between modes, such as time, subjects, and experimental conditions.

3.3 Tensorization Methods

Tensorization is the process of transforming traditional matrix-form datasets into multi-dimensional tensors. This section discusses various deterministic and stochastic methods for tensorizing data, depending on the analysis objectives and the nature of the original data.

Reshaping a dataset involves converting its structure, often from a matrix form into a tensor, to capture the underlying relationships among its variables better. This process can include multiple indexing, pivot table transformations, or tensorisation, which involves mapping data from lower to higher dimensions. For instance, student grades across various subjects, years, and exams can be transformed from a matrix into a 4th-order tensor, enabling more sophisticated analysis of trajectories across students, subjects, or time periods. The fusion of multiple data sources is another reshaping method. The process integrates data from multiple sources/modalities to create a comprehensive representation, leveraging the multi-way nature of tensors.

Sparse Representation: Not all data fills every possible combination of indices in a tensor. Sparse tensors, where most elements are zero, are essential for handling high-dimensional data efficiently. Consider a 4th-order tensor where modes represent users, items, interaction types (e.g., view, click, purchase), and timestamps. In this tensor, most entries are likely to be zeros or nulls because most users interact with only a small subset of the available items, engage in a limited number of interaction types, and only at specific times. Sparse tensor structures can drastically reduce memory usage and computational requirements compared to dense representations. Sparsity can be addressed by incorporating regularisation techniques like dropouts, using sparse tensor representations from scratch, and building ML packages that accept these representations. Various libraries, such as the 2-dimensional scipy.sparse Python package does not scale to sparse nd-arrays arrays. A dictionary data structure can capture the n-dimensional indices tuple as key and values as a list of all aggregated features.

Quantisation: Quantisation reduces the precision of continuous variables, which can simplify tensor computations without significantly affecting accuracy. For instance, instead of using precise dates, temperature data might be aggregated by year or month to reduce the tensor size, making processing faster and less memory intensive.

Parallelisation: When tensors become large, parallelisation techniques are employed to distribute the computational load across multiple processors. By partitioning tensors and distributing these partitions across a cluster of nodes, large datasets can be processed more efficiently. Each node works on a specific partition, enabling simultaneous computation, which is crucial in large-scale tensor operations. For example, parallelisation that is invariant of shape and dimension is applied for distributed processing on a cluster of computing nodes using a high-dimensional wavefront, which was applied to the Multiple Sequence Alignment problem [3-5]. Tensors were dynamically created for various sequences in a linear memory array. Each computing node accesses its assigned dense partitions from a specific index up to the partition size, applying wave-front parallelism with the dependency-aware distribution.

Deterministic tensorisation refers to systematic methods that convert data into higher dimensions in a predictable manner, facilitating the application of reverse processes, like detensorisation, to reduce dimensions when needed. Techniques such as Hankelization and Löwnerization are examples, useful in fields like signal processing and telecommunication for harmonic retrieval and direction-of-arrival estimation [6].

Statistical tensorisation leverages statistical measures like covariance matrices to structure data along specific modes. Higher-order statistics, such as cumulants and moments, provide a deeper analysis of non-Gaussian datasets with independent variables. These methods are particularly useful in applications like Blind Source Separation (BSS), where they help identify and separate latent variables in the data.

Domain-specific tensorisation techniques apply transformations tailored to particular types of data. For instance, a 3rd-order tensor might be used in signal processing to represent time-frequency data across multiple channels. Transformations like the Short-Time Fourier Transform (STFT) or wavelet transforms enable multi-scale, multi-orientation data representation, which is crucial for detailed signal analysis. Additionally, advanced methods, such as those involving Generalised Characteristic Functions (GCFs), can further enrich tensorisation by incorporating higher-order statistics, leading to more compact and expressive tensor representations [7,8]. Using tensor network representations, it is possible to super compress datasets with as many as 1050 entries down to 107 or even lower. These tensorisation techniques are vital for developing efficient machine learning algorithms and deep neural networks that handle complex, high-dimensional datasets with reduced computational and memory requirements.

3.4 Literature Review

Integrating tensorization techniques in machine and deep learning models has shown significant advancements across various applications. This section explores key case studies demonstrating the benefits of tensorization in enhancing model performance, reducing complexity, and improving generalisation in different domains.

Tensorization in Artificial Neural Networks (ANNs) has been recognised to reduce the models' complexity as it increases with the addition of layers and neurons, often leading to over-parameterization [9]. Tensorization offers a solution by compressing neural networks, thus reducing the number of parameters while maintaining or even improving model performance. For instance, tensorized activation functions, such as recursive neurons, can be used to compute weighted sums recursively within tree structures, effectively managing the hierarchical complexity of data [10]. Techniques like CP decomposition or Tensor Train (TT) decomposition further refine this process by breaking down tensor aggregations, allowing for a more efficient and structured approach to model learning. This compression is particularly evident when tensor decomposition algorithms, such as tensor networks, are applied to the weight tensors of an ANN, leading to shallower networks with fewer layers yet retaining high performance. Replacing specific layers in a model with tensor decomposition layers, such as TT layers, enables the model to capture latent variables effectively, optimising the learning process [11-13].

Tensorization in Machine Learning Models is outlined in a range of studies. In data warehousing and business intelligence, tensor decomposition methods have been utilised for processing large data cubes, significantly enhancing the efficiency of online analytical processing (OLAP) systems [14]. In signal processing, tensor methods have improved blind source separation (BSS) by enhancing the reconstruction of signals through methods like Hankelization [55] and Bayesian Tucker decomposition [15]. These tensor-based approaches outperform traditional methods, particularly in capturing the complex relationships inherent in multidimensional data.

In bioinformatics, tensorization aids in binary classification tasks, such as predicting miRNA-disease associations, by converting 2D datasets into multi-way tensors and applying tensor completion techniques [16]. Similarly, in social network analysis and semantic data mining, tensor models have been used to analyse the evolution of user interactions over time, allowing for identifying patterns and relationships that are not easily captured by conventional methods [17,18]. For instance, tensor-based methods like Tucker decomposition and CP decomposition have been employed to analyse temporal knowledge graphs, predicting new links and proposing ontological terms more accurately than traditional approaches [19,20].

Tensorization in Deep Learning Models, particularly in convolutional neural networks (CNNs), , and recurrent neural networks (RNNs), has significantly advanced model compression and performance. For example, TensorNet, a CNN model utilising TT layers, substantially reduced the number of parameters, compressing the network size by a factor of 7 without sacrificing accuracy [12]. Similarly, hierarchical Tucker (HT) tensor formats have been used to compress CNNs, maintaining high accuracy while significantly reducing model size [21].

In RNNs and LSTM networks, tensorization techniques like TT and HT have been shown to compress models effectively, though with varying impacts on accuracy. For instance, TT-LSTM models have demonstrated better suitability for CNN compression, while HT formats offer higher compression ratios for RNNs, albeit with some loss in accuracy [22].

In natural language processing (NLP), tensor-based models have outperformed traditional deep learning models in tasks such as sentiment analysis and event prediction. Recursive Neural Tensor

Networks (RNTNs) and tensor-based attention mechanisms have effectively modelled semantic relationships and context within language data [23]. Moreover, tensorization in transformers, mainly through block-term decomposition (BTD), has enhanced language modelling and neural translation tasks, achieving higher compression and performance than standard transformer models [24]. Multi-modal Visual Question Answering (VQA) used tensors to fuse visual and textual representations outperformed the bilinear models based on the outer product and its massive parameters [25]. Also, For graph transformation, graph tensors learn embeddings of time-varying graphs based on a tensor framework [26]. **More case studies are summarised in supplement A.**

Python Packages for Tensorization have been developed to facilitate the implementation of tensor-based methods in machine and deep learning. These packages offer various functionalities, from tensor decomposition and regression to neural network layers optimised for tensor operations. Notable examples include Tensorly [27], which supports multiple tensor decomposition techniques such as CP and Tucker, and TensorNetwork, a package developed by Google for advanced tensor algebra operations. Other packages like scikit-tt [28], scikit-tensor [29], and TensorNet-TF [30] provide specialised tools for implementing tensor methods in different domains, enabling researchers and practitioners to leverage the power of tensorization in their models. **More packages are listed in supplement B.**

In conclusion, tensorization has proven to be a powerful tool in both machine learning and deep learning, offering significant benefits in terms of model compression, performance enhancement, and the ability to capture complex relationships within data. The successful application of tensorization across various case studies underscores its potential to transform traditional approaches, leading to more efficient and effective models in various applications.

3.5 Proposed Framework

The proposed framework enhances deep learning models by leveraging tensorization to transform datasets into higher-dimensional tensors, capturing complex variable interactions. The framework's key components include initial tensorization, representation learning through methods like PCA, SVD, Tucker, or CPD, and dimensionality reduction to simplify the data. This refined, tensorized data is then integrated into machine learning and deep learning models, improving performance. Implementation involves tensorizing the data, applying representation learning and dimensionality reduction, and feeding the processed data into the models for enhanced analysis and predictions. The complete framework is depicted in Fig. 5.

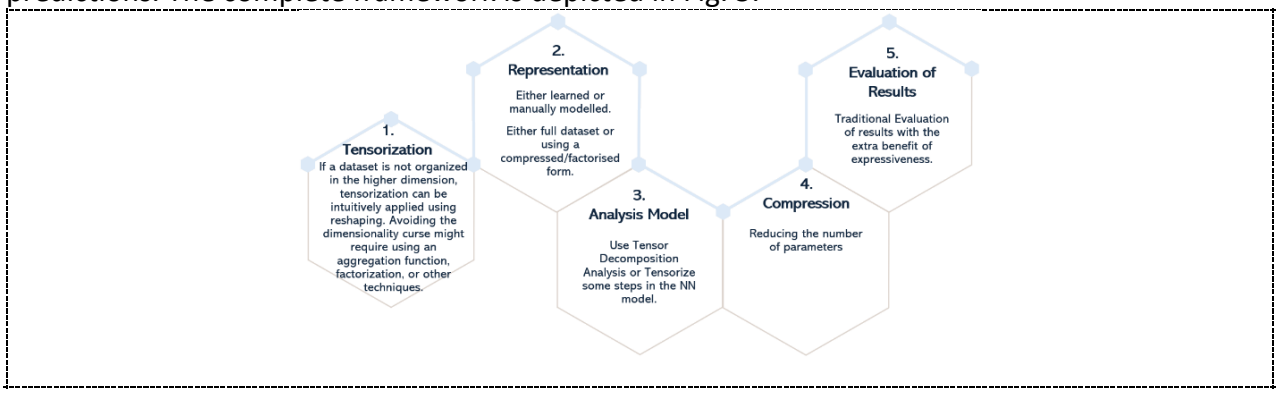
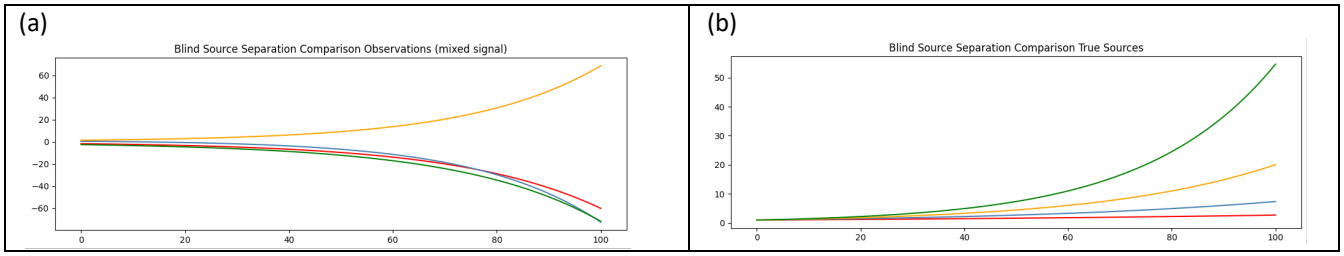


Fig. 5. Tensor computing framework

4. Experimental Setup and Results

To evaluate the proposed tensorization framework, various datasets from different domains, such as four synthetic signals of small lengths of 101 time steps, four audio of 117601 lengths, and MNIST images of 784 flat pixels were selected to ensure robust and comprehensive testing. The datasets underwent standardised preprocessing steps, including normalisation, denoising, and segmentation, to maintain consistency across experiments. The framework's effectiveness was tested under various conditions, including noise robustness, signal types, real-time processing, and parameter sensitivity. Controlled noise was introduced at varying levels and different types to assess the framework's robustness in signal reconstruction. Parameter sensitivity analysis included testing for different ranks. However, more fine-tuning for all involved parameters might enhance the results. Baseline comparisons with fundamental blind source separation approaches of Independent Component Analysis (ICA), Principal Component Analysis (PCA), Discrete Wave Transform (DWT), and Non-negative matrix factorisation (NMF) provided a comprehensive evaluation. The evaluation of the optimised tensor-based decomposition methods employed the Hankelization across signals to ensure uniformity. Experiments were carefully structured to include multiple repetitions, averaging the results to account for variability, and tailored adjustments were made for specific problem requirements. In quantitative evaluation metrics like root mean square error (RMSE), lower values indicate better reconstruction accuracy but do not measure the quality of the reconstruction. We evaluated using other metrics such as Structural Similarity Index (SSIM), correlation coefficients and Frequency Domain Analysis, where higher values indicate better structural quality of the reconstruction.

The results illustrated in **Fig. 6** show the shape of the observed mixtures in (a), and the true sources are in (b). The ICA are structurally the most distant (c), then PCA is closer (e), and the tensor-based is the closest (d) in reverse order of the reconstruction error. This might be due to each method's different scaling and signal permutations. Further tests for three noise levels and reduced fank vs full rank revealed more insights. All methods suffered from performance degradation due to noise levels, and increasing the rank did not consistently improve performance, particularly for PCA and NMF. ICA was the most effective technique for the lowest Root Mean Square Error (RMSE) of 2.93 on synthetic data and 0.088 for audio files maintaining structural integrity in low-noise environments. NMF showed promise in retaining structural features with an SSIM of 0.68 in synthetic data and -0.00029 for sound files but struggled with accuracy in high-noise conditions, yielding higher RMSE values among the methods. DWT showed poor performance in high noise levels, especially regarding frequency similarity and SSIM, indicating its limitations in noisy environments. Conversely, despite achieving the highest RMSE reconstruction errors, the Hankel and multiway methods attained the highest frequency similarity of 196168 in synthetic data and 6271.8 for sound files, leading to an overall positive correlation of 0.94 in synthetic data and -0.2 for audio files, suggesting that it preserves frequency characteristics well, which is crucial for specific applications like audio analysis. When structurally similar but with a negative correlation (e.g. -0.50), this suggests that while the method captures frequency features, it may not accurately reflect the overall trends of the original signal. This could indicate potential phase shifts or distortions in the reconstructed signal. **All results metrics are available in supplement C.**



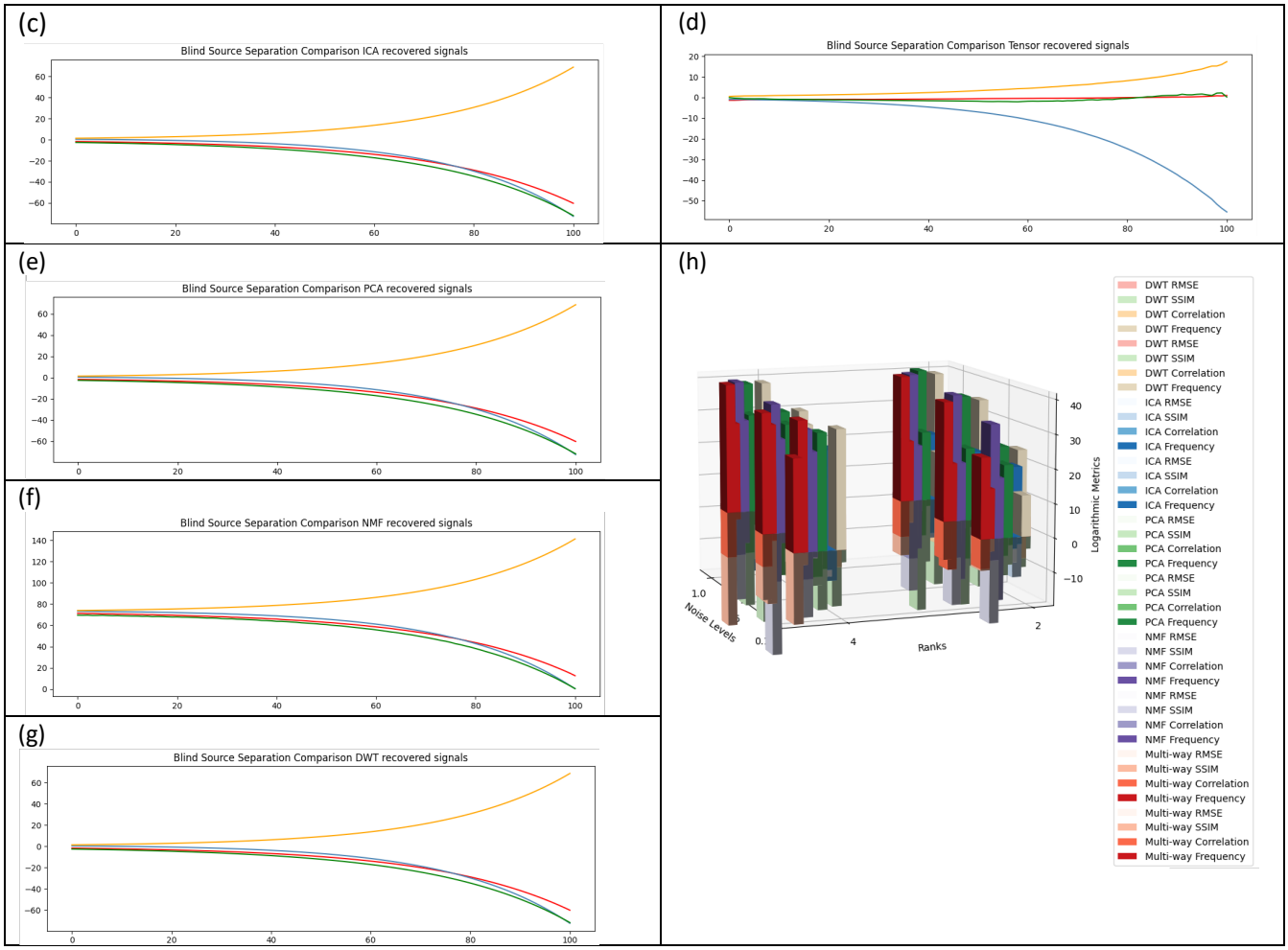


Fig. 6. (a) Observed Mixed Signals, (b) True Signals, (c) BSS reconstructed signals from ICA, (d) multiway compared, (e) PCA, (f) NMF, (g) DWT (h) RMSE, SSIM, Correlation and Frequency Similarity Metrics of all methods

Despite the high RMSE, the Hankel method maintained reasonable SSIM and correlation scores, especially in audio data under low noise conditions. This indicates its potential usefulness in specific scenarios, particularly where frequency retention is prioritised. Accurately capturing frequency content can be more important than perfectly reconstructing the time-domain signal if a specific task or analysis emphasises frequency preservation—such as in signal processing system identification in vibration analysis or fault detection. The multi-way methods could be precious despite their lower scores in other performance metrics.

5. Conclusion

This manuscript comprehensively surveyed multi-way analysis techniques compared to traditional linear algebra-based machine learning (ML) algorithms using matrix-form datasets. While multi-way reconstruction methods demonstrated some limitations in metrics such as RMSE, their notable performance in frequency similarity underscores their potential value in applications emphasising frequency characteristics, such as signal processing and system identification. In these scenarios, the trade-offs between frequency representation and other signal quality metrics must be carefully considered, ensuring that the method aligns with the specific goals of the analysis. The findings of this experiment have broader implications for the development of ML frameworks,

highlighting the need for generalisation to accommodate datasets with varying dimensions. As multi-way analysis techniques are integrated with deep neural networks (DNNs), tensorization methods are crucial for enhancing the representational capacity of complex relationships. Despite the challenges associated with preprocessing data into the required tensor form and the lack of standardisation across Python packages, the benefits—such as improved generalisation, interpretability, and scalability—make these approaches compelling. Future work should focus on optimising tensor operations within existing deep learning frameworks and exploring advanced tensor decomposition techniques (e.g., CP, Tucker) to balance performance and interpretability across diverse applications.

This experimental setup can be enhanced by adding more recent deep learning models like CNN, RNN or transformer-based blind source separation models. Additionally, exploring the integration of tensorized models with graph neural networks (GNNs) and dynamic graph networks (DGNs) presents an exciting avenue for advancing multi-way analysis. As the field evolves, a systematic approach to benchmarking and defining expressiveness metrics will facilitate meaningful comparisons of different methodologies. To demonstrate the versatility of tensorization in enhancing deep learning models, other objectives can be tested, such as signal denoising, feature extraction, image reconstruction, multimodal data fusion, time series forecasting, and recommendation systems.

Basic Linear Algebra Subprograms (BLAS) libraries have optimised many linear algebra computations and equation-solving algorithms. The BLAS program initially only supported vector operations, matrix operations, and matrix-matrix operations. Recently, tensor-tensor operations were implemented for fourth-order tensors [31], including tensor (Kronecker) products, KhatriRao products, Hadamard products, tensor contraction, t-products, or L-products. This optimisation level must also be compatible with different deep-learning frameworks and parallel hardware platforms to ensure tractable tensor-tensor operations on variable-order tensors. The Tensorised NN Frameworks need to provide optimised Tensorised Layer types, tensorised activation functions and tensorised forward and backward propagation algorithms, such as SGD with DMRG algorithms, AutoDiff [32] and DDSP (differentiable digital signal processing) [33]. Some considerations might be required in hardware design beyond the currently optimised matrix multiplication on GPUs and better on application-specific TPUs to enable faster processing of the tensorial deep learning layers. The ongoing research in machine learning in quantum computing [34] can enable quantum parallelism for ongoing research, such as a variational quantum algorithm for singular value decomposition (VQSVD) [35] and generally, the quantum ML algorithms that can be tensorized on the quantum platforms [36]. Several case studies on tensorizing neural networks at different stages of their architecture were presented in sections 5 and 6. However, ongoing research is required to determine how many and what kind of tensorizations are optimal and how they impact the performance.

To strike a balance between compression and performance enhancement, as well as interpretability across different models and applications, the choice of tensor decomposition techniques, such as CP, Tucker, HT, TT, or others, should be further explored. Standardised challenges and benchmarks with clear expressiveness metrics will make future proposals easier to compare [37]. This manuscript explores tensorization and its potential to revolutionise deep learning models and multi-way analysis. Further investigation of these methodologies will pave the way for future machine-learning systems that are more efficient, expressive, and adaptable.

In closing, the exploration of tensorization within multi-way analysis offers a transformative potential for machine learning applications, enabling more efficient, expressive, and adaptable models. Continued investigation into these methods promises to bridge the gap between theoretical advancements and practical implementations, paving the way for innovations in data representation,

processing efficiency, and signal reconstruction capabilities across diverse domains. This will help ultimately enhance their deployment in real-world applications, including those with limited computational resources, such as smart IoT devices. By establishing standardised benchmarks and expressiveness metrics, we can foster meaningful comparisons among future proposals, further driving innovation in the integration of multi-way analysis with machine learning methodologies

The results revealed a nuanced trade-off between RMSE and signal expressiveness, where traditional error metrics like RMSE might not fully capture the qualitative aspects of signal reconstruction. While ICA and PCA showed lower RMSE, the tensor-based methods preserved the original signals' structural characteristics, emphasising the need for new evaluation metrics that balance quantitative accuracy and qualitative expressiveness. This discussion highlights the importance of considering both error and expressiveness in model evaluation, especially in real-world applications where the integrity of reconstructed signals is paramount.

References

[1] Kolda, Tamara G., and Brett W. Bader. "Tensor decompositions and applications." *SIAM review* 51, no. 3 (2009): 455-500. https://doi.org/10.1137/07070111X

[2] Lu, Haiping, Konstantinos N. Plataniotis, and Anastasios N. Venetsanopoulos. "A survey of multilinear subspace learning for tensor data." *Pattern Recognition* 44, no. 7 (2011): 1540-1551. https://doi.org/10.1016/j.patcog.2011.01.004

[3] M. Helal, H. El-Gindy, B. Gaeta, and V. Sinchenko, “High Performance Multiple Sequence Alignment Algorithms for Comparison of Microbial Genomes,” presented at the 19th International Conference on Genome Informatics, Gold Coast, Australia, 2008.

[4] Helal, Manal, Lenore Mullin, John Potter, and Vitali Sintchenko. "Search space reduction technique for distributed multiple sequence alignment." In *2009 Sixth IFIP International Conference on Network and Parallel Computing*, pp. 219-226. IEEE, 2009. https://doi.org/10.1109/NPC.2009.43

[5] Helal, Manal. "Indexing and partitioning schemes for distributed tensor computing with application to multiple sequence alignment." PhD diss., University of New South Wales, Sydney, Australia, 2009.

[6] Debals, Otto, and Lieven De Lathauwer. "Stochastic and deterministic tensorization for blind signal separation." In *International Conference on Latent Variable Analysis and Signal Separation*, pp. 3-13. Cham: Springer International Publishing, 2015. https://doi.org/10.1007/978-3-319-22482-4_1

[7] Cichocki, Andrzej, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao, and Danilo P. Mandic. "Tensor networks for dimensionality reduction and large-scale optimization part 1 low-rank tensor decompositions." *Foundations and Trends in Machine Learning* 9, no. 4-5 (2016): 249-429. https://doi.org/10.1561/2200000059

[8] Cichocki, Andrzej, Anh-Huy Phan, Qibin Zhao, Namgil Lee, Ivan Oseledets, Masashi Sugiyama, and Danilo Mandic. "Tensor networks for dimensionality reduction and large-scale optimizations part 2 applications and future perspectives." *Foundations and Trends® in Finance* 9, no. 6 (2017): 431-673. https://doi.org/10.1561/2200000067

[9] Denil, Misha, Babak Shakibi, Laurent Dinh, Marc'Aurelio Ranzato, and Nando De Freitas. "Predicting parameters in deep learning." *Advances in neural information processing systems* 26 (2013). [Online]. Available: http://arxiv.org/abs/1306.0543

[10] Bacciu, Davide, and Danilo P. Mandic. "Tensor decompositions in deep learning." *arXiv preprint arXiv:2002.11835* (2020).

[11] Yang, Yongxin, and Timothy Hospedales. "Deep multi-task representation learning: A tensor factorisation approach." *arXiv preprint arXiv:1605.06391* (2016).

[12] Novikov, Alexander, Dmitrii Podoprikhin, Anton Osokin, and Dmitry P. Vetrov. "Tensorizing neural networks." *Advances in neural information processing systems* 28 (2015).

[13] G. G. Calvi, A. Moniri, M. Mahfouz, Q. Zhao, and D. P. Mandic, “v,” *arXiv:1903.06133 [cs, eess]*, Jan. 2020, Accessed: Feb. 14, 2021. [Online]. Available: http://arxiv.org/abs/1903.06133

[14] Spelta, Alessandro. "Financial market predictability with tensor decomposition and links forecast." *Applied network science* 2, no. 1 (2017): 7. https://doi.org/10.1007/s41109-017-0028-1

[15] Böttcher, Alexander, Wieland Brendel, Bernhard Englitz, and Matthias Bethge. "Trace your sources in large-scale data: one ring to find them all." *arXiv preprint arXiv:1803.08882* (2018).

[16] Huang, Feng, Xiang Yue, Zhankun Xiong, Zhouxin Yu, Shichao Liu, and Wen Zhang. "Tensor decomposition with relational constraints for predicting multiple types of microRNA-disease associations." *Briefings in bioinformatics* 22, no. 3 (2021): bbaa140. https://doi.org/10.1093/bib/bbaa140

[17] Acar, Evrim, Seyit A. Camtepe, Mukkai S. Krishnamoorthy, and Bülent Yener. "Modeling and multiway analysis of chatroom tensors." In *International Conference on Intelligence and Security Informatics*, pp. 256-268. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. https://doi.org/10.1007/11427995_21

[18] E. Acar, S. A. Çamtepe, and B. Yener, "Collective Sampling and Analysis of High Order Tensors for Chatroom Communications," in *Intelligence and Security Informatics*, vol. 3975, S. Mehrotra, D. D. Zeng, H. Chen, B. Thuraisingham, and F.-Y. Wang, Eds., in Lecture Notes in Computer Science, vol. 3975. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 213–224. https://doi.org/10.1007/11760146_19

[19] T. Lacroix, G. Obozinski, and N. Usunier, "Tensor Decompositions for temporal knowledge base completion," presented at the The International Conference on Learning Representations (ICLR), Apr. 2020. Accessed: Aug. 25, 2022. [Online]. Available: http://arxiv.org/abs/2004.04926

[20] Nickel, Maximilian, Volker Tresp, and Hans-Peter Kriegel. "A three-way model for collective learning on multi-relational data." In *Icml*, vol. 11, no. 10.5555, pp. 3104482-3104584. 2011.

[21] Gabor, Mateusz, and Rafał Zdunek. "Compressing convolutional neural networks with hierarchical Tucker-2 decomposition." *Applied Soft Computing* 132 (2023): 109856. https://doi.org/10.1016/j.asoc.2022.109856

[22] Yang, Yinchong, Denis Krompass, and Volker Tresp. "Tensor-train recurrent neural networks for video classification." In *International conference on machine learning*, pp. 3891-3900. PMLR, 2017.

[23] Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. "Recursive deep models for semantic compositionality over a sentiment treebank." In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631-1642. 2013. https://doi.org/10.18653/v1/D13-1170

[24] Ma, Xindian, Peng Zhang, Shuai Zhang, Nan Duan, Yuexian Hou, Ming Zhou, and Dawei Song. "A tensorized transformer for language modeling." *Advances in neural information processing systems* 32 (2019).

[25] Ben-Younes, Hedi, Rémi Cadene, Matthieu Cord, and Nicolas Thome. "Mutan: Multimodal tucker fusion for visual question answering." In *Proceedings of the IEEE international conference on computer vision*, pp. 2612-2620. 2017. https://doi.org/10.1109/ICCV.2017.285

[26] Malik, Osman Asif, Shashanka Ubaru, Lior Horesh, Misha E. Kilmer, and Haim Avron. "Tensor graph neural networks for learning on time varying graphs." In *Proceedings of NIPS Workshop*. 2019.

[27] Kossaifi, Jean, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. "Tensorly: Tensor learning in python." *Journal of Machine Learning Research* 20, no. 26 (2019): 1-6.

[28] P. Gelß, *Scikit-TT*. (2022). [Online]. Available: https://github.com/PGelss/scikit_tt

[29] M. Nickel, *scikit-tensor Library*. (Nov. 2013). [Online]. Available: https://pypi.org/project/scikit-tensor/

[30] T. Garipov, D. Podoprikhin, A. Novikov, and D. Vetrov, "Ultimate tensorization: compressing convolutional and FC layers alike." [Online]. Available: https://github.com/timgaripov/TensorNet-TF

[31] X.-Y. Liu and X. Wang, "Fourth-order Tensors with Multidimensional Discrete Transforms," May 03, 2017, *arXiv*: arXiv:1705.01576. Accessed: Sep. 13, 2022. [Online]. Available: http://arxiv.org/abs/1705.01576

[32] A. Paszke *et al.*, "Automatic differentiation in PyTorch," presented at the 31st Conference on Neural Information Processing Systems, CA, USA, 2017, p. 4.

[33] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," presented at the International Conference on Learning Representations, arXiv, Jan. 2020. Accessed: Sep. 13, 2022. [Online]. Available: http://arxiv.org/abs/2001.04643

[34] V. Bergholm *et al.*, "PennyLane: Automatic differentiation of hybrid quantum-classical computations," Jul. 29, 2022, *arXiv*: arXiv:1811.04968. Accessed: May 27, 2024. [Online]. Available: http://arxiv.org/abs/1811.04968

[35] X. Wang, Z. Song, and Y. Wang, "Variational Quantum Singular Value Decomposition," *Quantum*, vol. 5, p. 483, Jun. 2021. https://doi.org/10.22331/q-2021-06-29-483

[36] Zaman, Kamila, Alberto Marchisio, Muhammad Abdullah Hanif, and Muhammad Shafique. "A survey on quantum machine learning: Current trends, challenges, opportunities, and the road ahead." *arXiv preprint arXiv:2310.10315* (2023).

[37] Errica, Federico, Davide Bacciu, and Alessio Micheli. "Theoretically expressive and edge-aware graph learning." *arXiv preprint arXiv:2001.09005* (2020).

[38]P. Kantor, G. Muresan, F. Roberts, D. D. Zeng, F.-Y. Wang, H. Chen, and R. C. Merkle, Eds., in Lecture Notes in Computer Science, vol. 3495. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 256–268. https://doi.org/10.1007/11427995_21