Anomaly Detection from Multilinear Observations via Time-Series Analysis and 3DTPCA

Jackson Cates
Department of Electrical Engineering and Computer Science
South Dakota Mines
Rapid City, South Dakota

Randy C. Hoover
Department of Electrical Engineering and Computer Science
South Dakota Mines
Rapid City, South Dakota

Kyle Caudle
Department of Mathematics
South Dakota Mines
Rapid City, South Dakota

David Marchette
Naval Surface Warfare Center
Dahlgren Division
Dahlgren, Virginia

Cagri Ozdemir
Department of Electrical Engineering and Computer Science
South Dakota Mines
Rapid City, South Dakota

***Abstract*—In the era of big data, there is massive demand for new techniques to forecast and analyze multi-dimensional data. One task that has seen great interest in the community is anomaly detection of streaming data. Toward this end, the current research develops a novel approach to anomaly detection of streaming 2-dimensional observations via multilinear time-series analysis and 3-dimensional tensor principal component analysis (3DTPCA). We approach this problem utilizing dimensionality reduction and probabilistic inference in a low-dimensional space. We first propose a natural extension to 2-dimensional tensor principal component analysis (2DTPCA) to perform data dimensionality reduction on 4-dimensional tensor objects, aptly named 3DTPCA. We then represent the sub-sequences of our time-series observations as a 4-dimensional tensor utilizing a sliding window. Finally, we use 3DTPCA to compute reconstruction errors for inferring anomalous instances within the multilinear data stream. Experimental validation is presented on a synthetic multilinear time-series data, video streams via MovingMNIST data, and dynamic networks via the NYC Taxi Record. Results illustrate that the proposed approach has a significant speedup in training time compared with deep learning, while performing competitively in terms of accuracy.**

I. Introduction

Anomaly detection for streaming multi-dimensional observational data has been one of the most influential tasks for machine learning. The application of anomaly detection for 2-dimensional or 3-dimensional observations can be applied to a diverse set of domains, a subset of which include, detecting anomalies in video sequences, dynamic networks, and satellite imagery [1]–[3]. Time-series anomaly detection differs from traditional anomaly detection techniques due to its temporal component. While there can be global outliers within the time-series, there can also be local outliers to due abnormal temporal behaviors.

The current research was supported in part by the Department of the Navy, Naval Engineering Education Consortium under Grant No. (N00174-19-1-0014) and the National Science Foundation under Grant No. (2007367). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Naval Engineering Education Consortium or the National Science Foundation.

Historically, there are numerous anomaly detection methods for univariate and multivariate time-series [4]. Several methods are available for this approach, such as autoregressive integrated moving average (ARIMA) models and long-short term memory (LSTM) neural networks [5]–[7]. In general, these approaches utilizing time-series forecasting to evaluate anomalies based on their forecast error. However, these methods assume that the historical data we can be well represent via time-series modeling.

Another approach for detecting anomalies is utilizing dimensionality reduction and probabilistic inference in the reduced dimensional space [4], [8]. In general, this approach involves reducing the dimension of the time-series to a lower dimensional subspace, with the assumption that anomalies will be significantly different to normal observations in this lower space. Along these lines, there have been numerous techniques that use autoencoders for dimensionality reduction via the encoded latent space [1], [8]–[11]. These autoencoders come in several variations such as LSTM autoencoders and convolutional neural network (CNN) autoencoders [9]–[11]. Another set of methods use singular value decomposition (SVD) in order to represent temporal observations in a lower-dimensional space via principal component analysis (PCA) [12], [13]. For example, the authors in [12] utilized the SVD to perform latent semantic analysis for local sub-sequences.

For multilinear time-series, extensions to such methods have been pursued to where the historical observations are no longer scalars or vectors but can be represented as a lateral slice of a tensor (e.g. $\mathcal{X}_t \in \mathbb{R}^{\ell \times 1 \times m}$).[1] A tensor in this context is a multi-dimensional array, often referred to as a n-mode or n-way array as defined in Section II. Recently, new methods have been proposed to perform dimensionality reduction on tensors by capitalizing on the spatial correlations within each observation while simultaneously capitalizing on

[1]Note: It's customary in the literature to represent tensors with upper-case calligraphic letters.

the temporal correlations across observations. In particular, two-directional tensor principal component analysis, referred to as 2DTPCA [14], investigates the fusion of two different reduced dimensional subspaces, one to represent correlations in the "row"-data and another to represent correlations in the "column"-data. For image classification problems, 2DTPCA has outperformed many state-of-the-art methods related to deep learning, LSTMs, and alternate forms of tensor representations.

The current paper builds on the work in [14] for dimensionality reduction but extends the correlation models for both 3-dimensional data as well as anomalous observations. In particular, we represent a sub-sequence of size p from a multilinear time-series model as a 4-mode tensor (e.g. $\mathcal{S}_t \in \mathbb{R}^{\ell \times 1 \times m \times p}$), where ℓ and m capture the spatial correlations in each observation and p captures the temporal correlations from a historical moving window. We propose a natural extension to 2DTPCA to perform dimensionality reduction for our subsequence tensor $\mathcal{S}_t$, aptly called 3-directional tensor principal component analysis or 3DTPCA for short. We illustrate that 3DTPCA decomposes row, column, and temporal dimensions for anomaly detection. Utilizing reconstruction error from the low-dimensional 3DTPCA space, we evaluate if new multilinear observations are abnormal. Experimental results presented on synthetic multilinear time-series data, video streams via MovingMNIST, and dynamic networks via the NYC Taxi Record to illustrate the effectiveness of the proposed approach for detecting anomalous events in multilinear observations. We also demonstrate the significant speedup that is achieved over deep learning approaches.

The remainder of the paper is organized as follows: In Section II we provide some mathematical background for the tensor linear algebra used throughout the paper. In Section III we present our proposed extensions from 2DTPCA to 3DTPCA for unsupervised learning, dimensionality reduction, and observational reconstruction. We also illustrate how we use 3DTPCA for multilinear time-series anomaly detection. In Section IV we present some experimental results with our proposed method and compare/constrast with the state-of-the-art. We evaluate the methods in terms of their receiver operating characteristic (ROC) curves and their computational speed. Section V contains our conclusions and future directions.

II. Mathematical Preliminaries

In this section, we provide an overview of the notation and operations used throughout the paper for order-n tensors. A much more detailed treatment can be found in [15]–[20], however, to keep the current paper self-contained an overview is provided here.

A. Notation

First, we will review the basic definitions of a tensor from [15], [16]. In the context of this paper, we refer a multi-dimensional array as a *tensor*. The tensor's *order* is the number of dimensions or modes. For example, a tensor of order-n is denoted by $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$. Therefore, vectors and matrices are first- and second-order tensors, respectively.

It will be convenient to break a tensor $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$ up into various slices and tubal elements. A frontal slice of tensor with order-n will be notated with subscript indexing such as $\mathcal{A}_{(i_1 \ldots i_n)}$. For example, let $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times m_3}$ be a third-order tensor. The frontal slice corresponding to mode-3 is notated as $\mathcal{A}_{(i_3)}$ with *Python* slicing $\mathcal{A}[:, :, i_3]$. We will notate $\mathbf{a}^{(k)}_{i_1 \ldots i_n}$ as the mode-k fiber corresponding to the $i_1^{\text{th}} \cdots i_n^{\text{th}}$ index. For example, $\mathbf{a}^{(1)}_{i_2 i_3}$ is the mode-1 fiber corresponding to the i_2^{th} and i_3^{th} index, assuming *Python* indexing would yield $\mathbf{a}^{(1)}_{i_2 i_3} = \mathcal{A}[:, i_2, i_3]$.

Algorithm 1 tensor-tensor product induced by $*_L$

Input: Input tensors $\mathcal{A} \in \mathbb{R}^{m_1 \times \ell \times \cdots \times m_n}$ and $\mathcal{B} \in \mathbb{R}^{\ell \times m_2 \times \cdots \times m_n}$
Output: $\mathcal{C} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$
for $i = 3$ to n **do**
  $\tilde{\mathcal{A}} \leftarrow L(\mathcal{A}, [\,], i)$
  $\tilde{\mathcal{B}} \leftarrow L(\mathcal{B}, [\,], i)$
end for
for $i = 1$ to m_3 **do**
  for $j = 1$ to m_4 **do**
    $\vdots$
    for $k = 1$ to m_n **do**
      $\tilde{\mathcal{C}}(:, :, i, \ldots, k) = \tilde{\mathcal{A}}(:, :, i, \ldots, k) \cdot \tilde{\mathcal{B}}(:, :, i, \ldots, k)$
    end for
    $\vdots$
  end for
end for
for $i = n$ to 3 **do**
  $\mathcal{C} \leftarrow L^{-1}(\tilde{\mathcal{C}}, [\,], i)$
end for

B. Order-n Tensor Operations

An operation that is fundamental to the results of current work is the multiplication of two order-n tensors. A family of tensor-tensor products for third-order tensors has been formulated in the transform domain for any invertible linear transformation [15]. This work has been extended for order-n tensors in [17]. Utilizing the notation outlined in [15] and [17], we define the following operators:

Definition 1. Let $\mathcal{A} \in \mathbb{R}^{m_1 \times \ell \times \cdots \times m_n}$ and $\mathcal{B} \in \mathbb{R}^{\ell \times m_2 \times \cdots \times m_n}$ be order-n tensors. The tensor-tensor product based on L transform $\mathcal{A} *_L \mathcal{B} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$ is computed by applying matrix multiplication to the frontal slices of $\mathcal{A}$ and $\mathcal{B}$ in the transform domain. This is shown in an algorithmic fashion in Algorithm 1.[2]

[2]Note: For all algorithms shown, the vertical dots indicate nested for-loops. The nested for-loops continue from mode-3 to mode-n

Algorithm 2 tensor transpose

Input: Input tensor $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$
Output: $\mathcal{B} \in \mathbb{R}^{m_2 \times m_1 \times \cdots \times m_n}$
for $i = 3$ to n **do**
  $\tilde{\mathcal{A}} \leftarrow L(\mathcal{A}, [\,], i)$
end for
for $i = 1$ to m_3 **do**
  for $j = 1$ to m_4 **do**
    $\vdots$
    for $k = 1$ to m_n **do**
      $\tilde{\mathcal{B}}(:, :, i, \ldots, k) = \tilde{\mathcal{A}}(:, :, i, \ldots, k)^T$
    end for
    $\vdots$
  end for
end for
for $i = n$ to 3 **do**
  $\mathcal{B} \leftarrow L^{-1}(\tilde{\mathcal{B}}, [\,], i)$
end for

Definition 2. The identity tensor $\mathcal{I} \in \mathbb{R}^{m \times m \times m_3 \times \cdots \times m_n}$ is the tensor whose first frontal slice is the $m \times m$ identity matrix in the transform domain.

Definition 3. If $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$, then the tensor transpose $\texttt{ttrans}_L(\mathcal{A}) \in \mathbb{R}^{m_2 \times m_1 \times \cdots \times m_n}$ is computed by transposing the frontal slices of $\mathcal{A}$ in the transform domain. This is shown in an algorithmic fashion in Algorithm 2.

Definition 4. A tensor $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$ is called *orthogonal* if $\mathcal{A} *_L \texttt{ttrans}_L(\mathcal{A}) = \texttt{ttrans}_L(\mathcal{A}) * \mathcal{A} = \mathcal{I}$.

A tensor rotation operator for order-3 tensors is defined in [14], we extend this to order-n tensors here.

Definition 5. The **tensor rotation operator** $\texttt{rol}(\mathcal{A}, i, j)$ is the tensor obtained by swaping the axis in the i^{th} and j^{th} mode. For example, let tensor $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times m_3 \times \cdots \times m_n}$, then the tensor rotation along mode-1 and mode-2 results in $\texttt{rol}(\mathcal{A}, 1, 2) \in \mathbb{R}^{m_2 \times m_1 \times m_3 \times \cdots \times m_n}$.

C. Order-n Tensor Singular Value Decomposition

The final tool required for the current research is a tensor singular value decomposition defined for order-n tensors (referred to as the *t-SVD*$_L$). We note that the *t-SVD*$_L$ can be defined using any inevitable linear transformation. L denotes the linear transformation used for computing the decomposition. For $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$, then there exists tensors U, S, V such that

$$\mathcal{A} = \mathcal{U} *_L \mathcal{S} *_L \texttt{ttrans}_L(\mathcal{V}),$$

where $\mathcal{U} \in \mathbb{R}^{m_1 \times m_1 \times m_3 \times \cdots \times m_n}$ is an orthogonal tensor of left-singular matrices (analogous to left-singular vectors in order-2 tensors), $\texttt{ttrans}_L(\mathcal{V}) \in \mathbb{R}^{m_2 \times m_2 \times m_3 \times \cdots \times m_n}$ is an orthogonal tensor of right-singular matricies (analogous to right-singular vectors in order-2 tensors), and $\mathcal{S} \in \mathbb{R}^{m_1 \times m_2 \times m_3 \times \cdots \times m_n}$ is an f-diagonal tensor (analogous to singular values in order-2 tensors). The *t-SVD*$_L(\mathcal{A})$ is computed by applying the matrix singular value decomposition to the frontal slices of $\mathcal{A}$ in the transform domain. This is shown in an algorithmic fashion in Algorithm 3

Algorithm 3 t-SVD$_L$

Input: Input tensors $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$
Outputs:
$\mathcal{U} \in \mathbb{R}^{m_1 \times m_1 \times m_3 \times \cdots \times m_n}$
$\mathcal{S} \in \mathbb{R}^{m_1 \times m_2 \times m_3 \times \cdots \times m_n}$
$\mathcal{V} \in \mathbb{R}^{m_2 \times m_2 \times m_3 \times \cdots \times m_n}$
for $i = 3$ to n **do**
  $\tilde{\mathcal{A}} \leftarrow L(\mathcal{A}, [\,], i)$
end for
for $i = 1$ to m_3 **do**
  for $j = 1$ to m_4 **do**
    $\vdots$
    for $k = 1$ to m_n **do**
      Set $\tilde{\mathcal{U}}(:, :, i, \ldots, k)$, $\tilde{\mathcal{S}}(:, :, i, \ldots, k)$, $\tilde{\mathcal{V}}(:, :, i, \ldots, k)$ to the result of the matrix SVD of $\tilde{\mathcal{A}}(:, :, i, \ldots, k)$
    end for
    $\vdots$
  end for
end for
for $i = n$ to 3 **do**
  $\mathcal{U} \leftarrow L^{-1}(\tilde{\mathcal{U}}, [\,], i)$
  $\mathcal{S} \leftarrow L^{-1}(\tilde{\mathcal{S}}, [\,], i)$
  $\mathcal{V} \leftarrow L^{-1}(\tilde{\mathcal{V}}, [\,], i)$
end for

III. Proposed Approach to Multilinear Time-Series Anomaly Detection

In this section we present our extension of 2DTPCA to 3DTPCA to perform dimensionality reduction for fourth order tensors. We then proceed to use 3DTPCA for multilinear time-series anomaly detection.

A. Proposed 3DTPCA Method

1) ***Unsupervised Training****:* 2DTPCA outlined in [14] performs dimensionality reduction on third order tensors. The authors accomplish this by utilizing the *t-SVD*$_L$ along mode-1 and mode-3 to capture the correlations across the row and column dimensions, respectively. They are able to fuse the row and column subspaces by applying the **tensor rotation operator** `rol` before data projection. We propose an extension to 2DTPCA to perform dimensionality reduction on a fourth-order tensor $\mathcal{A} \in \mathbb{R}^{\ell \times n \times m \times p}$, aptly called 3DTPCA. We extend 2DTPCA by following a similar process for mode-1 and mode-3, but now compute the *t-SVD*$_L$ for mode-4. Therefore, we propose a process to capitalize on the correlations along mode-1 (column-space), mode-3 (row-space), and mode-4 (temporal-space). The ultimate goal is to reduce the dimensionality of a subset of the multilinear data along

these modes in an effort to perform inference in a reduced dimensional space.

Because we are going to be computing the *t-SVD*$_L$ multiple times, we denote ${}^i\mathcal{U}_k$ as the first k lateral slices of the left singular tensor $\mathcal{U}$ computed for mode-i. We also denote ${}^i\mathcal{S}$ and ${}^i\mathcal{V}$ as the right singular tensor and f-diagonal tensor for mode-i, respectively. To outline the process of 3DTPCA, we compute *t-SVD*$_L(\mathcal{A}) = {}^1\mathcal{U} *_L {}^1\mathcal{S} *_L \texttt{ttrans}_L({}^1\mathcal{V})$. ${}^1\mathcal{U}$ now contains the left singular tensors associated with the correlations in mode-1. We then perform dimensionality reduction in mode-1 by projecting the tensor $\mathcal{A}$ onto the first k left singular tensors as

$$\mathcal{Y} = \texttt{ttrans}_L({}^1\mathcal{U}_k) *_L \mathcal{A} \in \mathbb{R}^{k\times n\times m\times p}.$$

To capture the correlations on mode-3, we first compute the tensor rotation operator $\bar{\mathcal{Y}} = \texttt{rol}(\mathcal{Y}, 1, 3) \in \mathbb{R}^{m\times n\times k\times p}$. We then compute *t-SVD*$_L(\bar{\mathcal{Y}}) = {}^3\mathcal{U} *_L {}^3\mathcal{S} *_L \texttt{ttrans}_L({}^3\mathcal{V})$. ${}^3\mathcal{U}$ now contains the left singular tensors associated with the correlations in mode-3. We then proceed with dimensionality reduction for mode-3 by projecting the tensor $\tilde{\mathcal{Y}}$ onto the first q left singular tensors via

$$\mathcal{W} = \texttt{ttrans}_L({}^3\mathcal{U}_q) *_L \bar{\mathcal{Y}} \in \mathbb{R}^{q\times n\times k\times p}.$$

Finally, to capture the correlations in mode-4, we compute the tensor rotation operator $\bar{\mathcal{W}} = \texttt{rol}(\mathcal{W}, 1, 4) \in \mathbb{R}^{p\times n\times k\times q}$ and then compute *t-SVD*$_L(\bar{\mathcal{W}}) = {}^4\mathcal{U} *_L {}^4\mathcal{S} *_L \texttt{ttrans}_L({}^4\mathcal{V})$. From there, training is complete and we are able to perform dimensionality reduction utilizing our collection of left singular tensors $\Theta = \{{}^1\mathcal{U}_k, {}^3\mathcal{U}_q, {}^4\mathcal{U}\}$. To define this process, we create the following definition:

Definition 6. If $\mathcal{A} \in \mathbb{R}^{\ell\times n\times m\times p}$, then training 3DTPCA is defined as $\texttt{3DTPCA}(\mathcal{A}) = \Theta = \{{}^1\mathcal{U}_k, {}^3\mathcal{U}_q, {}^4\mathcal{U}\}$ where Θ is the collection of our left singular tensors.

*2) **Dimensionality Reduction**:* For the current research, we also need to be able to take a new data tensor $\mathcal{B} \in \mathbb{R}^{\ell\times w\times m\times p}$ and reduce the dimensionality of $\mathcal{B}$ utilizing Θ. Note that mode-2 of $\mathcal{B}$ can have a different size w, but the size of mode-1, mode-2, and mode-4 must be the same as $\mathcal{A}$. This is done in a similar fashion to the training process, however we do not recompute the *t-SVD*$_L$. Instead, we utilize the set of left singular tensors Θ for projection. We first project along mode-1 via

$$\mathcal{Y} = \texttt{ttrans}_L({}^1\mathcal{U}_k) *_L \mathcal{B} \in \mathbb{R}^{k\times w\times m\times p}.$$

We proceed to consider correlations in mode-3 by applying the tensor rotation operator $\bar{\mathcal{Y}} = \texttt{rol}(\mathcal{Y}, 1, 3) \in \mathbb{R}^{m\times w\times k\times p}$ and project along mode-3 via

$$\mathcal{W} = \texttt{ttrans}_L({}^3\mathcal{U}_q) *_L \bar{\mathcal{Y}} \in \mathbb{R}^{q\times w\times k\times p}.$$

Finally, we proceed to consider correlations in mode-4 by applying the tensor rotation operator $\bar{\mathcal{W}} = \texttt{rol}(\mathcal{W}, 1, 4) \in \mathbb{R}^{p\times w\times k\times q}$ and project along mode-4 via

$$\mathcal{Z} = \texttt{ttrans}_L({}^4\mathcal{U}_r) *_L \bar{\mathcal{W}} \in \mathbb{R}^{r\times w\times k\times q}.$$

We define this process of performing data dimensionality with the set of left singular tensors Θ with the following definition:

Definition 7. If $\mathcal{B} \in \mathbb{R}^{\ell\times w\times m\times p}$, then performing dimensionality reduction with 3DTPCA utilizing the set of left singular tensors Θ is defined as $\texttt{Reduce}(\mathcal{A}, \Theta) = \mathcal{Z} \in \mathbb{R}^{r\times w\times k\times q}$ where r, k, q are the reduced dimensions.

*3) **Reconstruction**:* We also desire to reconstruct our reduced tensor $\mathcal{Z}$ back to its original form $\mathcal{B}$. We can perform this by reversing the order of the previous process. We first reconstruct along mode-4 on $\bar{\mathcal{W}}$ as

$$\bar{\mathcal{W}} = {}^4\mathcal{U}_r *_L \mathcal{Z} \in \mathbb{R}^{p\times w\times k\times q}.$$

We then proceed inverse the tensor rotation operator by $\mathcal{W} = \texttt{rol}(\bar{\mathcal{W}}, 1, 4) \in \mathbb{R}^{q\times w\times k\times p}$. This process now allows us to reconstruct along mode-3 as

$$\bar{\mathcal{Y}} = {}^3\mathcal{U}_q *_L \mathcal{W} \in \mathbb{R}^{m\times w\times k\times q},$$

and perform the inverse of the tensor rotation opertator by $\mathcal{Y} = \texttt{rol}(\bar{\mathcal{Y}}, 1, 3) \in \mathbb{R}^{k\times n\times m\times p}$ to complete the reconstruction along mode-3. Finally, we finish reconstruction for mode-1 as

$$\hat{\mathcal{B}} = {}^1\mathcal{U}_k *_L \mathcal{Y} \in \mathbb{R}^{m\times w\times k\times q},$$

where $\hat{\mathcal{B}}$ signifies the reconstruction of $\mathcal{B}$. To define this process, we create the following definition:

Definition 8. If $\mathcal{Z} \in \mathbb{R}^{r\times n\times k\times q}$, then reconstructing $\mathcal{Z}$ utilizing the set of left singular tensors Θ is defined as $\texttt{Reconstruct}(\mathcal{Z}, \Theta) = \hat{\mathcal{B}} \in \mathbb{R}^{\ell\times n\times m\times p}$ where $\hat{\mathcal{B}}$ is approximate to the original tensor $\mathcal{B}$.

*4) **Fast t-SVD**:* Recently, the authors in [21] showed that for datasets with highly correlated observations, there is a method to increase computational speed of the *t-SVD*. They showed that for $\mathcal{A} \in \mathbb{R}^{\ell\times n\times m}$, we can reduce the dimension of mode-2 by applying the fast Fourier transform (FFT). Note that n is the number of observations, which is generally large. After computing the FFT for mode-2, observations that are highly correlated will have higher magnitude in low frequencies. Therefore, most of the information in mode-2 can be saved by selecting the first r lateral slices, where in general $r << n$. Then applying the inverse FFT in mode-2 results in $\bar{\mathcal{A}} \in \mathbb{R}^{\ell\times r\times m}$.

If we apply this same result to 3DTPCA, we can get a competitive advantage in computational speed. We refer this as *fast 3DTPCA*. Our process is that for tensor $\mathcal{A} \in \mathbb{R}^{\ell\times n\times m\times p}$, we will compute *fast 3DTPCA* by first reducing the tensor in mode-2 using the process outlined earlier. Using the new tensor $\bar{\mathcal{A}} \in \mathbb{R}^{\ell\times r\times m\times p}$, we can proceed to compute 3DTPCA as $\texttt{3DTPCA}(\bar{\mathcal{A}})$ to receive our set of left singular tensors Θ. We can then use Θ for future dimensionality reduction and reconstruction.

B. Proposed Anomaly Detection Model

Our aim is to detect anomalies for a time-series with multilinear observations $\mathcal{X}_t \in \mathbb{R}^{\ell\times 1\times m}$ for $t = 1, 2, \cdots, n$.

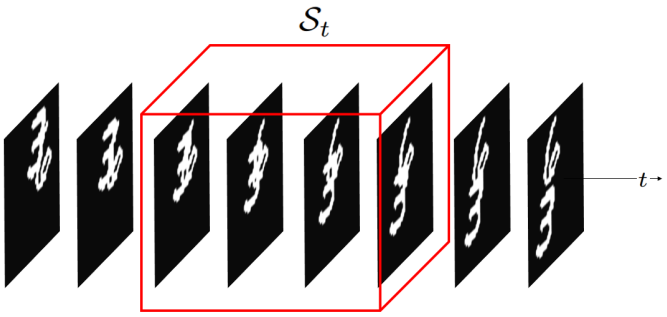


Fig. 1: Graphical illustration of the sliding window. The red box illustrates the sub-sequence $\mathcal{S}_t$ of multilinear time-series X_t with a sliding window size of $p = 4$. Multilinear observations originate from the MovingMNIST dataset presented in Section IV.

We assume that the multilinear observations $\mathcal{X}_t$ contains no anomalies during unsupervised training. We accomplish anomaly detection by representing our multilinear time-series as a tensor where mode-1 is rows, mode-3 is columns, and mode-4 is temporal. We then perform 3DTPCA with dimensionality reduction and reconstruction along our time-series. Then we proceed to detect anomalies based on their reconstruction error. Anomalies should be represented poorly in PCA space and therefore should have a higher reconstruction error.

*1) **Unsupervised Training**:* To captured temporal patterns, we construct our time-series into the tensor $\mathcal{A} \in \mathbb{R}^{\ell \times n-p \times m \times p}$ utilizing a *sliding window* of size p, a technique shown in [22]. This technique creates a sub-sequence of size p for each multilinear observations $\mathcal{X}_t$. An illustrative example of the sliding window is shown in Fig. 1. We create a sub-sequence of our time-series by stacking our multilinear observations $\mathcal{X}_t$ along mode-4 as

$$\mathcal{S}_t = \begin{pmatrix} \mathcal{X}_t & \mathcal{X}_{t-1} & \cdots & \mathcal{X}_{t-p} \end{pmatrix} \in \mathbb{R}^{\ell \times 1 \times m \times p} \tag{1}$$

where $\mathcal{S}_t$ is the sub-sequence at time t, and we have a window of size p. We then proceed to stack our sub-sequences S_t along mode-2 as

$$\mathcal{A} = \begin{pmatrix} \mathcal{S}_p & \mathcal{S}_{p+1} & \cdots & \mathcal{S}_n \end{pmatrix} \in \mathbb{R}^{\ell \times n-p \times m \times p}. \tag{2}$$

The resulting tensor $\mathcal{A} \in \mathbb{R}^{\ell \times n-p \times m \times p}$ contains our rows in mode-1, our columns in mode-3, and our temporal sub-sequences in mode-4. To define this process of using equations 1 and 2 we create the following definition:

Definition 9. Utilizing a *sliding window* for multilinear observations $\mathcal{X}_t \in \mathbb{R}^{\ell \times 1 \times m}$ for $t = 1, 2, \cdots, n$ is defined as $\texttt{Window}(\mathcal{X}_t) = \mathcal{A} \in \mathbb{R}^{\ell \times n-p \times m \times p}$ where p is the size of the window.

Algorithm 4 `Window`

Input: Multilinear observations $\mathcal{X}_t \in \mathbb{R}^{\ell \times 1 \times m}$ for $t = 1, 2, \cdots, n$ and window size p.
Output: $\mathcal{A} \in \mathbb{R}^{\ell \times n-p \times m \times p}$
for $i = p$ to n **do**
  for $j = 1$ to p **do**
    $\mathcal{A}(:, i, :, j) = X_{i-j}$
  end for
end for

To represent the tensor $\mathcal{A}$ in PCA space, we now compute

$$\texttt{3DTPCA}(\mathcal{A}) = \Theta = \{{}^1\mathcal{U}_k, {}^3\mathcal{U}_q, {}^4\mathcal{U}_r\},$$

resulting in a set of left singular tensors Θ. We proceed to perform dimensionality reduction and reconstruction utilizing 3DTPCA via

$$\hat{\mathcal{A}} = \texttt{Reconstruct}(\texttt{Reduce}(\mathcal{A}, \Theta), \Theta),$$

where $\hat{\mathcal{A}}$ is the reconstructed tensor from $\mathcal{A}$. Note that we contain approximates for the sub-sequences $\hat{\mathcal{A}} = \begin{pmatrix} \hat{\mathcal{S}}_p & \hat{\mathcal{S}}_{p+1} & \cdots & \hat{\mathcal{S}}_n \end{pmatrix} \in \mathbb{R}^{\ell \times n-p \times m \times p}$. We proceed compute the reconstruction error $\epsilon_t \in \mathbb{R}$ as

$$\epsilon_t = ||\mathcal{S}_t - \hat{\mathcal{S}}_t||_F, \tag{3}$$

where F notates the Forbenius norm. To perform anomaly detection, we create a threshold from our reconstruction errors ϵ_t based on a chosen percentile. As shown in the next section, we will use this threshold for anomaly detection.

*2) **Anomaly Detection**:* Once we receive w new multilinear observations $\mathcal{X}_v \in \mathbb{R}^{\ell \times 1 \times m}$ for $v = n+1, n+2, \cdots, w$, we perform anomaly detection by first utilizing the sliding window for the new observations via

$$\texttt{Window}(\mathcal{X}_v) = \mathcal{B} \in \mathbb{R}^{\ell \times w-p \times m \times p}$$

where $\mathcal{B}$ is our new data tensor. We then perform dimensionality reduction and reconstruction utilizing 3DTPCA via

$$\hat{\mathcal{B}} = \texttt{Reconstruct}(\texttt{Reduce}(\mathcal{B}, \Theta), \Theta),$$

and compute the reconstruction error via

$$\epsilon_v = ||\mathcal{S}_v - \hat{\mathcal{S}}_v||_F.$$

Utilizing the threshold we previously received during training, we label anomalies if they are above the threshold from training as

$$\texttt{Anomaly}(\mathcal{X}_v) = \begin{cases} \text{Anomaly} & \epsilon_v \geq \text{threshold} \\ \text{Normal} & \epsilon_v < \text{threshold} \end{cases}$$

IV. Experimental Results

In this section, we will compare our proposed 3DTPCA method with other methods in the literature. We will evaluate the methods quantitatively based on their receiver operating characteristic (ROC) curve and computational speed. Deep learning has several techniques to perform multilinear time-series anomaly detection [1], [8]. Specifically, we utilize an

Fig. 2: Illustration of the MovingMNIST dataset. The left frame contains a normal observation before our anomaly. At time-step 4500 an anomalous digit enters the frame as shown in the middle frame. The anomalous digit leaves the frame after time-step 4800 as shown in the right frame.

long-short term memory neural network (LSTM) autoencoder to compare our technique with another reconstruction based method. We also compare our results to principle component analysis (PCA) to examine the advantage we receive when we represent the multilinear time-series spatially.

A. Datasets

We utilize datasets that are designed for unsupervised training. Specifically, the training set will contain no anomalies while the testing set will contain both normal and abnormal observations. Namely: (a) a synthetic transform-based tensor autoregression ($\mathcal{L}$-TAR) process [23]; (b) an adapted version of the MovingMNIST dataset that contains a grid of bouncing MNIST digits [24]; (c) the NYC Trip Record dataset of taxicab trips [25].

*1) **Synthetic** $\mathcal{L}-$**TAR**:* We generate a 20×20 synthetic time-series utilizing a $\mathcal{L}$-TAR process described in [23]. We generate 3000 normal observations as

$$\mathcal{X}_t = \mathcal{A} *_L \mathcal{X}_{t-1} + \mathcal{E}_{t,1},$$

where $\mathcal{A}_1 \in \mathbb{R}^{20\times20\times20}$ is a coefficient tensor that was arbitrarily selected as

$$\mathcal{A}_{(1)} = \begin{pmatrix} 0.5 & 0 & \cdots & 0 \\ 0 & 0.5 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0.5 \end{pmatrix},$$

$$\mathcal{A}_{(2)} = \mathcal{A}_{(3)} = \mathbf{0},$$

and $\mathcal{E}_{t,1}$ is white noise generated under a uniform distribution $\mathcal{E}_{t,1} \sim U(-0.1, 0.1)$. We proceed to generate 1000 abnormal observations by increasing the noise. We do this by increasing uniform distribution's range by 0.0001 for each proceeding time step. This results in similar $\mathcal{L}$-TAR process as

$$\mathcal{X}_t = \mathcal{A}_1 * \mathcal{X}_{t-1} + \mathcal{E}_{t,2},$$

where $\mathcal{E}_{t,2} \sim U(-0.1 - 0.0001t, 0.1 + 0.0001t)$ is our new white noise that is dependent on time. We proceed to use the first 2500 observations for training and the last 1500 observations for testing.

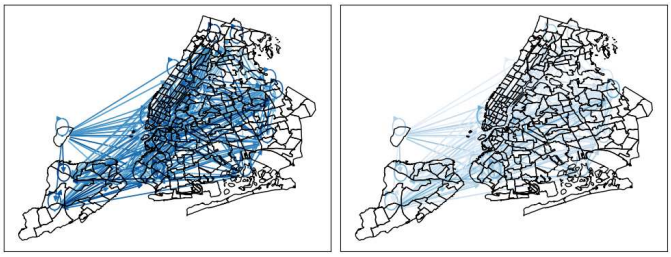


Fig. 3: Graph representation of the NYC Trip Record dataset. The nodes are zones in the Manhattan area. The edge's weight are the daily number of taxicab trips from the pickup zone to the dropoff zone. Edge weights are shown as the edge's opacity. **Left:** Before COVID-19 on January 1st, 2020. **Right:** COVID-19 on March 14th, 2020.

*2) **MovingMNIST**:* The MovingMNIST dataset presented in [24] is a dataset of bouncing MNIST digits. In the literature, this dataset has been used for forecasting a multilinear time-series [24], [26], [27]. We adapt this dataset for anomaly detection by first generating 5000 normal frames of size 50×50 with just two bouncing digits. We create an anomaly by inserting a third digit at time-step 4500. The anomalous digit leaves the frame at time-step 4800. The frames of normal and abnormal observations are shown in Fig. 2. We proceed to use the first 4000 observations for training and the last 1000 observations for testing.

*3) **NYC Trip Record**:* The NYC Trip Record dataset contains taxicab pickup and dropoff locations for the New York Manhattan area [25]. We use this dataset for anomaly detection because COVID-19 caused a dramatic decrease in the amount of trips. The impact of COVID-19 can be seen in Fig. 3. The original dataset splits the area into 263 pickup and dropoff zones. These zones can be represented as nodes in a dynamic graph. We picked 22 of the most frequently visited zones and represent the dynamic graph with a 22×22 adjacency matrix. We represent the edges of the dynamic graph as daily number of trips for the corresponding pickup and dropoff zones. We noticed that the decrease in the amount of trips happened around March 14$^{\text{th}}$, 2020. Therefore, we consider any date after March 14$^{\text{th}}$, 2020 as abnormal and the rest as normal. We use Janurary 1$^{\text{st}}$, 2015 to December 31$^{\text{st}}$, 2019 as our training set (1826 observations) and the entire year of 2020 as our testing set (365 observations).

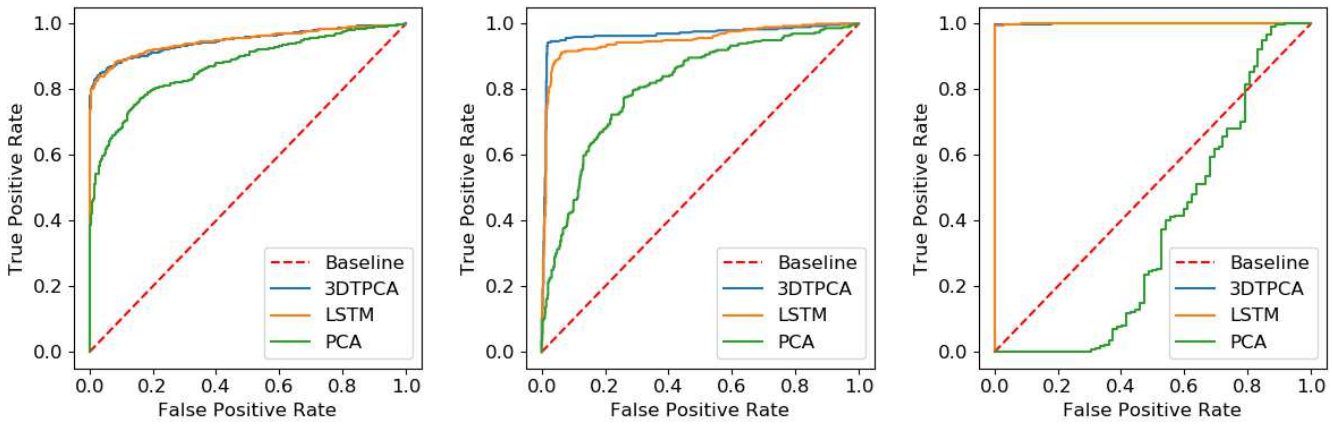
B. Performance in Anomaly Detection

We evaluate the method's effectiveness of anomaly detection utilizing ROC curves. ROC curves are produced by changing the threshold of the reconstruction errors and calculating the true positive rate (TPR) vs the false positive rate (FPR). We calculate the area under the ROC curve (AUC) for comparison. The ROC curves are shown in Fig. 4, and the AUC is shown in Table II. We also show how we configured the methods in Table I.

As can be seen from Table II, the proposed method is a top performer for these datasets. 3DTPCA is very competitive

TABLE I: Model Configurations

Methods	$\mathcal{L}-$TAR	MovingMNIST	NYC Trip Record
fast 3DTPCA	$p = 5$, reduced tensor size $= 3 \times 7 \times 1$	$p = 10$, reduced tensor size $= 1 \times 1 \times 1$, utilizes *fast t-SVD* with $r = 100$	$p = 7$, reduced tensor size $= 1 \times 2 \times 2$, utilizes the *fast t-SVD* with $r = 50$
LSTM	LSTM layers of size (128, 32, 32, 128) with relu activation	LSTM layers of size (1024, 256, 64, 64, 256, 1024) with relu activation	LSTM layers of size (1024, 256, 64, 64, 256, 1024) with relu activation
PCA	6 principal components	25 principal components	20 principal components



(a) $\mathcal{L}-$TAR (b) MovingMNIST (c) NYC Trip Record

Fig. 4: Receiver operating characteristic (ROC) curves for quantitative evaluation.

with the deep learning approach. We also see that we receive a large advantage over PCA when we represent our observations spatially.

We also show the reconstruction errors produced with each method in Fig. 5. The red highlight shows where anomalous observations are present and the orange dashed line shows the 90% threshold based on the training reconstruction errors. As we can see, 3DTPCA and the LSTM perform very similar in terms of detecting anomalous observations, however, as illustrated below, 3DTPCA is significantly better in terms of computational speedup.

TABLE II: AUC for Experiments

Methods	$\mathcal{L}-$TAR	MovingMNIST	NYC Trip Record
fast-3DTPCA	0.947	**0.966**	**0.9994**
LSTM	**0.967**	0.961	0.9992
PCA	0.856	0.810	0.365

TABLE III: Computational Speed (in seconds)

Methods	$\mathcal{L}-$TAR	MovingMNIST	NYC Trip Record
3DTPCA	3.71 ± 0.23	13.68 ± 0.59	2.22 ± 0.13
LSTM	6.01 ± 0.14	61.15 ± 0.70	73.16 ± 0.53
PCA	0.16 ± 0.03	1.11 ± 0.12	0.37 ± 0.09
3DTPCA Speedup vs. LSTM	1.62	4.47	32.95

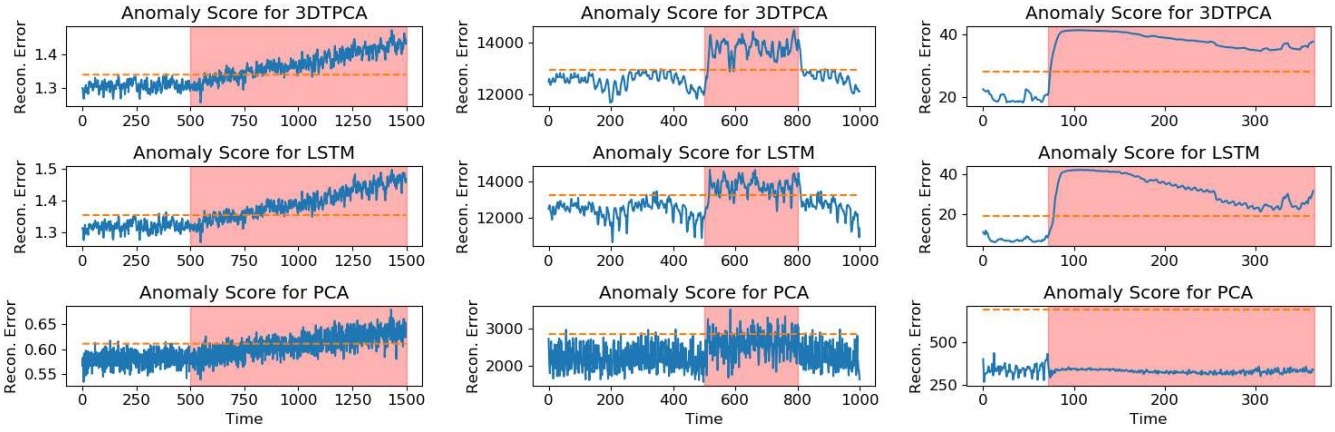
TABLE IV: AUC for Fast Experiments

Methods	MovingMNIST	NYC Trip Record
fast-3DTPCA	**0.966**	**0.999**
fast-LSTM	0.818	0.368
fast-PCA	0.231	0.692

C. Performance in Computational Speed

To evaluate the computational speed, we measured the running time of training 20 times and calculated the average running time $\pm$ the standard deviation. The results shown in seconds can be seen in Table III. We can see that 3DTPCA has a significant speedup compared to the LSTM while performing nearly the same in anomaly detection.

Note that in Table I we utilize the fast *t-SVD* for 3DTPCA. This is the reason why we get massive speedups for the MovingMNIST and NYC Trip Record datasets. However, we could utilize the results from the fast *t-SVD* for the other methods, aptly called the *fast-LSTM* and *fast-PCA*. We found that for the other methods, utilizing the FFT to reduce the amount of observations does lead to worse results in anomaly detection. For comparison, we re-perform the experiments in Section IV-B, but we utilize the FFT to reduce the amount of observations in a similar fashion to 3DTPCA. The results of this are outlined in Table IV. Note that the synthetic $\mathcal{L}-$TAR process is not included because the observations are not highly correlated, so applying fast *t-SVD* for 3DTPCA



(a) $\mathcal{L}-$TAR (b) MovingMNIST (c) NYC Trip Record

Fig. 5: Anomaly Scores for quantitative evaluation. **Top row:** 3DTPCA. **Middle row:** LSTM. **Bottom row:** PCA. The red highlight areas indicates where anomalies are present. The orange dashed line is the 90% threshold based on the training reconstruction errors.

results in a similar set of observations rendering the fast-approach somewhat useless. As illustrated in Table IV, it's clear that the proposed approach performs significantly better than both the *fast-LSTM* and *fast-PCA*.slice

V. Conclusions and Future Directions

Overall, the proposed method is worthy for multilinear time-series anomaly detection. Representing our multilinear observations spatially, we also perform competitively with deep learning in terms of accuracy and computational speed. Future work includes extending 3DTPCA to nDTPCA so we can perform anomaly detection with $n-$order observations. Future work also includes injecting seasonal observations into the sliding window to observe seasonal anomalies, and extending other methods of dimensionality reduction from univariate and multivariate time-series, such as forecast-able component analysis and piece-wise vector quantized approximation.

References

[1] M. Ribeiro, A. E. Lazzaretti, and H. S. Lopes, "A study of deep convolutional auto-encoders for anomaly detection in videos," *Pattern Recognition Letters*, vol. 105, pp. 13–22, 2018.

[2] M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[3] F. Rembold, C. Atzberger, I. Savin, and O. Rojas, "Using low resolution satellite imagery for yield prediction and yield anomaly detection," *Remote Sensing*, vol. 5, no. 4, pp. 1704–1733, 2013.

[4] M. Braei and S. Wagner, "Anomaly detection in univariate time-series: A survey on the state-of-the-art," 2020.

[5] V. Kozitsin, I. Katser, and D. Lakontsev, "Online forecasting and anomaly detection based on the arima model," *Applied Sciences*, vol. 11, no. 7, 2021.

[6] M. Saqib, E. Şentürk, S. A. Sahu, and M. A. Adil, "Ionospheric anomalies detection using autoregressive integrated moving average (arima) model as an earthquake precursor," *Acta Geophysica*, vol. 69, no. 4, p. 1493–1507, 2021.

[7] Z. Zeng, G. Jin, C. Xu, S. Chen, Z. Zeng, and L. Zhang, "Satellite telemetry data anomaly detection using causal network and feature-attention-based LSTM," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–21, 2022.

[8] J. Ren, F. Xia, Y. Liu, and I. Lee, "Deep video anomaly detection: Opportunities and challenges," *2021 International Conference on Data Mining Workshops (ICDMW)*, pp. 959–966, 2021.

[9] P. Liu, X. Sun, Y. Han, Z. He, W. Zhang, and C. Wu, "Arrhythmia classification of lstm autoencoder based on time series anomaly detection," *Biomedical Signal Processing and Control*, vol. 71, p. 103228, 2022.

[10] Y. Wei, J. Jang-Jaccard, W. Xu, F. Sabrina, S. Camtepe, and M. Boulic, "LSTM-autoencoder based anomaly detection for indoor air quality time series data," 2022.

[11] B. Lahasan and H. Samma, "Optimized deep autoencoder model for internet of things intruder detection," *IEEE Access*, vol. 10, pp. 8434–8448, 2022.

[12] J. Camacho, A. Pérez-Villegas, P. García-Teodoro, and G. Maciá-Fernández, "Pca-based multivariate statistical network monitoring for anomaly detection," *Computers & Security*, vol. 59, pp. 118–137, 2016.

[13] B. Pilastre, L. Boussouf, S. D'Escrivan, and J.-Y. Tourneret, "Anomaly detection in mixed telemetry data using a sparse representation and dictionary learning," *Signal Processing*, vol. 168, p. 107320, 2020.

[14] C. Ozdemir, R. C. Hoover, and K. Caudle, "2DTPCA: A new framework for multilinear principal component analysis," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 344–348.

[15] E. Kernfeld, M. Kilmer, and S. Aeron, "Tensor–tensor products with invertible linear transforms," *Linear Algebra and its Applications*, vol. 485, pp. 545–570, 2015.

[16] N. Hao, M. E. Kilmer, K. S. Braman, and R. C. Hoover, "New tensor decompositions with applications in facial recognition," *SIAM Journal on Imaging Science (SIIMS)*, vol. 6, no. 1, pp. 437–463, Feb. 2013.

[17] C. Ozdemir, R. C. Hoover, K. Caudle, and K. Braman, "High-order multilinear discriminant analysis via order-*n* tensor eigendecomposition," 2022.

[18] M. E. Kilmer, C. D. Martin, and L. Perrone, "A third-order generalization of the matrix SVD as a product of third-order tensors," Tufts University, Department of Computer Science, Tech. Rep. TR-2008-4, October 2008.

[19] M. E. Kilmer and C. D. Moravitz Martin, "Factorization strategies for third-order tensors," *Linear Algebra and Its Applications*, no. Special Issue in Honer of G.W.Stewart's 75^{th} birthday, 2009.

[20] K. Braman, "Third-order tensors as linear operators on a space of matrices," *Linear Algebra and its Applications*, vol. 433, no. 7, pp. 1241 – 1253, 2010.

[21] C. Ozdemir, R. C. Hoover, and K. Caudle, "Fast tensor singular value decomposition using the low-resolution features of tensors," in *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2021, pp. 527–533.

[22] N. Takeishi and T. Yairi, "Anomaly detection from multivariate time-series with sparse representation," *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2651–2656, 2014.

[23] J. Cates, R. C. Hoover, and K. Caudle, "Transform-based tensor auto regression for multilinear time series forecasting," in *2021 IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2021.

[24] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using LSTMs," 2015.

[25] "Tlc trip record data," 2022. [Online]. Available: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[26] N. Kalchbrenner, A. van den Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu, "Video pixel networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70, 06–11 Aug 2017, pp. 1771–1779.

[27] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool, "Dynamic filter networks," in *Advances in Neural Information Processing Systems*, vol. 29. Curran Associates, Inc., 2016.