Knowledge-guided Tensor Decomposition for Baselining and Anomaly Detection

Dimitri Leggas  
Reservoir Labs  
New York, NY  
leggas@reservoir.com

Christopher Coley  
Department of Aeronautics  
United States Air Force Academy  
USAFA, CO  
christopher.coley@usafa.edu

Teresa Ranadive  
Laboratory for Physical Sciences  
College Park, MD  
tranadive@lps.umd.edu

Abstract—We introduce a flexible knowledge-guided penalty for incorporating known or expected patterns of activity into tensor decomposition. Our modified tensor decomposition both enables efficient identification of semantically-related patterns across data sets and provides a means for identifying anomalous patterns. Specifically, we modify the loss function for a CP tensor decomposition such that a subset of the columns of the factor matrices are guided to align with the provided knowledge. We validate the effectiveness of the method for separating baseline patterns from anomalous patterns in the context of cyber network traffic logs. After performing daily decompositions of tensors formed from network logs, we derive a set of expected components describing baseline behavior. We then decompose a new tensor, created from network logs and containing a known anomaly, providing the baseline as knowledge guidance. Notably, the anomalous behavior appears among the unguided components, resulting in drastic reduction in the search space for anomalies. Additionally, we show that our knowledge-guided decomposition is robust to incorrect knowledge in that any enforced components that are not found in the original data have low weight. Our method, implemented in the tensor software package ENSIGN, is an efficient routine that reduces the post-processing needed for an anomaly detection workflow.

Index Terms—Tensor Decomposition, Anomalies, Anomaly Detection, Cyber Networks, Knowledge Guidance, Domain Knowledge

I. INTRODUCTION

Tensor decomposition is an unsupervised learning method for finding low-rank, latent structure in high-dimensional data sets and is a powerful technique for multi-way data analysis [1]–[4]. It has been shown to be effective for data discovery in diverse domains and data, such as spatiotemporal analysis [5], cybersecurity [6], [7], chemistry [9], [10], machine learning [11]–[13], precision healthcare [14], [15], genomics [16], neuroscience [17]–[19], and others. The CANDECOMP/PARAFAC (canonical decomposition/parallel factors) or CP tensor decomposition reduces the complexity of data and identifies patterns resulting from interactions between the various data fields, corresponding to dimensions in the tensor. Each component usually isolates a pattern of activity in the data. We develop a knowledge-guided tensor decomposition method that incorporates domain knowledge in order to guide the solution so that it is partially expressed in terms of known components, hence patterns. By providing a large number of expected components and leaving relatively few unguided, we are able to easily separate baseline patterns from anomalous behavior. Following existing work, we derive our baseline from a topic model [20]. Our approach addresses two main problems with using tensor decomposition for data analysis and knowledge discovery: 1) efficiently finding components that are semantically similar across multiple decompositions and 2) failing to identify which components in a decomposition are unseen in any prior decomposition, i.e. anomalous.

With respect to the first problem, multiple decompositions may share semantically-equivalent components that are numerically different, so identifying these shared components currently requires considerable effort from the data analyst. Consider, for example, a computer network analysis workflow that constructs and decomposes a tensor from daily connection logs. Each day, the decomposition will include certain components that describe expected user behavior (e.g., DNS requests, accessing a VPN). However, due to day-to-day variations in which users connect and at what time, these components may be numerically distinct even though they are semantically equivalent. Interpreting the meaning of components is often a manual process that requires a significant cognitive effort from the analyst, and such semantically-equivalent yet numerically-distinct components result in increased effort when analyzing tensor decompositions. Our knowledge-guided tensor decomposition finds the most accurate representation of the data in terms of known components, so it reduces the number of novel components that need to be analyzed. This allows an analyst to immediately identify the significance of the few new components.

The second problem stated above is also the result of the overhead required to manually interpret and label tensor decomposition components and the difficulty in automating the process. One challenge is the curse of dimensionality. When applied to large data, tensor decomposition yields components residing in a high-dimensional space, making it difficult to apply machine learning methods for anomaly detection, as one of the features of a set of points in a high-dimensional space is that their inter-point distances are roughly the same. While signature-based methods that filter and flag components based on specific signals are one way to circumvent this problem, they require that the user has initial knowledge about how anomalies should look. This approach is challenging as not all
anomalies can be classified \textit{a priori}. In our method, because known behaviors are guided to be represented in specific components, all anomalous activity is represented in a few components.

To summarize, the contributions of our work are as follows:

- We introduce a knowledge-guided tensor decomposition method in order to find semantically-equivalent components across multiple decompositions, thus improving the ease of analysis and interpretability of tensor decompositions.

- Our method utilizes a modified objective function that allows for guidance information to be incorporated into any mode of any component, whereas previous works are limited only to providing guidance information to one mode of a subset of components.

- We demonstrate that our method is effective at isolating anomalies into unguided components when many other components are guided. The resulting decompositions more readily discover patterns of interest, thus reducing post-processing requirements and alleviating analyst workload.

- We demonstrate the applicability and functionality of our method to large-scale sparse count tensors used to represent cyber network data.

The structure of this paper is as follows. First, in Section II, we discuss related work that incorporates domain knowledge into tensor decomposition and applications of tensor decompositions to computer network analysis. In Section III, we provide the necessary background and then give the details of our method, presenting the overall formulation and details on how to perform the resulting optimization problem. In Section IV we present the results of numerical experiments that validate the efficacy and robustness of our method, presenting the overall formulation and details on how to perform the resulting optimization problem. In Section V we make concluding remarks and discuss future work.

## II. RELATED WORK

Other works have incorporated domain knowledge into tensor decomposition in order to guide the solution toward desirable components. Our approach is different from these works in that it is not domain specific and it embeds prior knowledge in the resulting tensor decomposition, rather than relying on pairwise constraints or regularization to indirectly shape the decomposition. The Rubik factorization, proposed by Wang et al. [14], uses a bias term and pairwise constraints to produce components that represent medically-coherent computational phenotypes while incorporating clinical knowledge in a single factor matrix. Likewise, Henderson et al. [15] develop a semi-supervised tensor decomposition in order to encourage tensor components to associate patients with different disease statuses with different phenotypes. Our method is most closely related to these works, but we generalize these approaches by removing the domain-specific terms and increasing the flexibility and breadth of the guidance that can be provided.

Additionally, we allow for the use of any elementwise loss function, an approach pioneered by Hong et al. [21], and we use an all-at-once optimization method, whereas the other works use alternating minimization.

Guidance information has also been incorporated into other tensor algorithms. Xiong et al. [22] and Xiong et al. [23] employ guidance knowledge for tensor completion. Lee et al. [24] propose Guided and Interpretable Factorization for Tensors (GIFT), a knowledge-guided Tucker decomposition, for application to cancer analysis. In this work, we focus on a knowledge-guided decomposition framework for CP decompositions.

One practical problem that this work addresses, i.e. the difficulty in identifying sets of semantically-equivalent components across different decompositions, has been addressed in other unguided approaches. Gudibanda et al. [25] propose a coupled tensor decomposition such that different tensors share a subset of common information. While this decomposition could be used as a way of associating components between multiple decompositions, it is designed to decompose data from disparate sources rather than to enforce expected components. Also, it does not allow for a strictly-defined subset of information shared across tensors. Our framework permits any subset of components to be specified in order to guide components toward expected behavior. Several works approach the problem of finding similar components through post-processing of components. As one example, Ezick et al. [26] leverage heuristic-based detectors and graph analytics to automate the identification of patterns in large-scale decompositions. In contrast to these works, our approach relies only on a modified objective function and requires no post-processing.

MultiAspectForensics [27] and TensorSplat [28] have applied tensor decomposition to network traffic data in order to extract anomalies and malicious patterns. Unlike these works, we extend beyond theoretical research and algorithm development to emphasize operationally-practical results. We build upon Ezick et al. [26] by using tensor decomposition in the cyber domain, but our method uses a modified decomposition rather than post-processing.

For the problem of determining a baseline, which we use as the knowledge guidance for our decompositions, we follow the work of Henretty et al. [20], who treat decomposition components as distributions over the indices and learn a topic model from them in order to provide an effective clustering of the components. The topics have the same structure as components, and they show that they resemble the most common component classes (e.g., DNS, VPN, etc.). We learn a topic model over previously found components, and use the topics as knowledge.

## III. METHOD

### A. Background

Prior to presenting our method, we first describe the concepts related to tensor decomposition. For a full treatment on the topic, see [4]. A \textit{tensor} is a multidimensional array. The
order of a tensor is the number of dimensions, also known as modes. Thus, for any \( N \in \mathbb{N} \), the \( N \)-dimensional array \( X \in \mathbb{R}^{I_1 \times \cdots \times I_N} \) is an \( N \)-mode tensor. Hence, a vector is a 1-mode tensor and a matrix is a 2-mode tensor. We denote the \((i_1, \ldots, i_N)\)th entry of the \( N \)-mode tensor \( X \) as \( x_{i_1,\ldots,i_N} \).

A rank-\( R \) CP decomposition is an approximation of \( X \) as a sum of \( R \) outer products of \( N \) vectors, \( A^{(1)} \in \mathbb{R}^{I_1 \times R} \), \( A^{(N)} \in \mathbb{R}^{I_N \times R} \):

\[
X \approx \sum_{r=1}^{R} a^{(1)}_{i_1r} \cdots a^{(N)}_{i_Nr} \tag{1}
\]

It is convenient to express the decomposition as the outer product of its factor matrices \( A^{(1)} \in \mathbb{R}^{I_1 \times R} \), \( A^{(N)} \in \mathbb{R}^{I_N \times R} \), where the \( r \)th column of \( A^{(n)} \) is the vector \( a^{(n)}_{i_nr} \). We use the notation

\[
X \approx T = \left[ \lambda; A^{(1)}, \ldots, A^{(N)} \right] \tag{2}
\]

where the knowledge matrix is assumed to have its columns normalized to have length 1 (this is an easy pre-processing step and does not alter the evaluation of the penalty), \( || \cdot || \) denotes the 2-norm (although similar results are achieved with the 1-norm), and \( p^{(n)}_r \in \mathbb{R}_{\geq 0} \) determines the relative weight of the penalty terms. \( p^{(n)}_r \) is a user-selected value and is chosen such that \( p^{(n)}_r > 0 \) for every \( r \) corresponding to a non-zero column in the knowledge matrix and \( p^{(n)}_r = 0 \) otherwise. Only modes of components whose corresponding \( p^{(n)}_r > 0 \) are guided, so they serve as selectors for knowledge-guided components.

C. Optimization

We perform a knowledge-guided CP decomposition by finding the factor matrices \( A^{(n)} \in \mathbb{R}^{I_n \times R} \), \( n = 1, \ldots, N \), that solve

\[
\min F_g (T, X) \text{ subject to } T = \left[ A^{(1)}, \ldots, A^{(N)} \right]. \tag{6}
\]

Specifically, we solve the optimization problem in (6) using an all-at-once optimization method. To do this, we need the gradient of the loss function \( F_g \) described in (5). The gradient can be calculated as

\[
\frac{\partial F_g}{\partial a^{(n)}_{i_nr}} = \frac{\partial F}{\partial a^{(n)}_{i_nr}} + \frac{p^{(n)}_r}{\|a^{(n)}_r\|} \left( \hat{a}^{(n)}_r \cdot \hat{a}^{(n)}_r - \hat{a}^{(n)}_i \right), \tag{7}
\]

where \( \hat{a}^{(n)}_r = a^{(n)}_r / \|a^{(n)}_r\| \). The evaluation of \( \frac{\partial F}{\partial a^{(n)}_{i_nr}} \) may be found in the literature for many commonly used loss functions [21], and we present the remainder below.

If we let

\[
P = \sum_{n=1}^{N} \sum_{r=1}^{R} p^{(n)}_r \left( 1 - \frac{\hat{a}^{(n)}_r \cdot \hat{a}^{(n)}_r}{\|a^{(n)}_r\|^2} \right), \tag{8}
\]

then as every entry in a factor matrix only appears in one term (corresponding to its column),

\[
\frac{\partial P}{\partial a^{(n)}_{i_nr}} = \frac{\partial}{\partial a^{(n)}_{i_nr}} \left[ p^{(n)}_r \left( 1 - \frac{\hat{a}^{(n)}_r \cdot \hat{a}^{(n)}_r}{\|a^{(n)}_r\|^2} \right) \right]. \tag{9}
\]

This derivative is found easily by the quotient rule.

D. Anomaly Detection

We show that the knowledge-guided tensor decomposition is capable of performing anomaly detection by guiding a large number of baseline components, which are then separated automatically from unseen components. We use a topic model to learn a baseline from previously-found components. Our workflow begins by considering many historical decompositions of a certain data type (in our case, cyber network logs). Latent Dirichlet Allocation (LDA) [29] is a generative model that assumes observed distributions are generated from underlying topics. Following Henretty et al. [20], we treat the components as our observed distributions and learn a set of topics, which become our knowledge. We then decompose new...
data providing the topics as knowledge and leaving only a few components unguided. These few unguided components are checked for anomalies.

E. Implementation

We implement our knowledge-guided tensor decomposition in ENSIGN [30]–[36], a highly parallelized tensor decomposition package written in ANSI C. As our framework relies on all-at-once tensor decomposition, the implementations only require modifying the penalty and gradient calculation routines to account for the knowledge guidance penalty term. We choose ENSIGN due to its scalable performance on large datasets and its ability to deliver results for our small robustness experiments and our large anomaly detection experiments.

IV. RESULTS

A. Robustness to Incorrect Guidance Information

Because our method, by design, will align columns of the factor matrices with the provided guidance, it is possible to force components in the tensor decomposition to represent completely arbitrary information. To prevent this type of misuse from corrupting the results of the decomposition, it is important that the method is robust to incorrect guidance information and does not represent user-provided knowledge as being part of the data when it is not. In this section, we demonstrate the validity of our method by showing simultaneously that knowledge is enforced and that the method is robust in the face of incorrect guidance information. We take a tensor with a known decomposition and construct a component that does not appear in the solution. Then, we gradually perturb one of the known components until it becomes this “bogus” component. For each of components between the correct and incorrect components, we perform a knowledge-guided decomposition and provide it as knowledge. In each decomposition, the guidance component does appear in the factor matrices, but its corresponding weight decreases as the knowledge becomes increasingly far from the true solution. The weight becomes small enough that it is clear that it is not part of the solution.

Specifically, we construct a rank-4 tensor \( X \in \mathbb{R}^{5 \times 5 \times 5} \) with a known CP decomposition. A sequence of rank-4 CP decompositions of \( X \) are performed using the least-squares loss function. Components interpolating a known component and a bogus component are provided as knowledge. Each decomposition is performed with the same initial guess so that the only difference in results is dependent on the particular guidance information. We consider the last component of the known solution represented by \( a_{1}^{(1)}, a_{2}^{(2)}, a_{3}^{(3)} \). The bogus guidance information is defined as a single component where the first mode of the component is represented by a linear function, the second mode of the component by a quadratic, and the third mode of the component by a cubic. We then interpolate the components with a parameter \( \alpha \in [0, 1] \) so that when \( \alpha = 0 \) we provide a correct component as guidance and when \( \alpha = 1 \) we provide a completely bogus component as guidance. That is, for this problem, the parameters in Equation 7 are:

\[
\begin{align*}
\hat{a}_{1}^{(1)} &= (1 - \alpha)a_{1}^{(1)} + \alpha [1, 2, 3, 4, 5]^T \\
\hat{a}_{2}^{(2)} &= (1 - \alpha)a_{2}^{(2)} + \alpha [1, 4, 9, 16, 25]^T \\
\hat{a}_{3}^{(3)} &= (1 - \alpha)a_{3}^{(3)} + \alpha [1, 8, 27, 64, 125]^T \\
p_{1}^{(k)} &= 15 \text{ for } k = 1, 2, 3
\end{align*}
\]

The results of the decompositions are depicted in Fig. 1. The robustness of our method is demonstrated by the fact that the weight of the knowledge-guided component decreases as \( \alpha \) increases, indicating that the guidance information is not expected to be prominent in the data tensor.

![Fig. 1: We demonstrate the robustness of our knowledge-guided decomposition to bogus guidance information. We perform a sequence of 11 knowledge-guided decompositions in which the knowledge-guided component lies on the line between a component that is part of the known solution and a completely fabricated component not part of the data tensor. When \( \alpha = 0 \), the knowledge-guided component is the correct solution, and when \( \alpha = 1 \), the knowledge-guided component has its first mode represented by a linear function, the second mode by a quadratic, and the third mode by a cubic. When the knowledge-guidance is close to the known solution, the weight of the guided component in the final solution is nontrivial. As the guided component approaches the bogus component, the weight decreases.](image)

B. Application to Cyber Network Traffic

Finally, we apply the knowledge-guided tensor decomposition to network data in order to demonstrate how the technique aligns semantically-similar components across multiple decompositions and separates anomalous components. We first learn a topic model of components from five network logs. We then use these as knowledge when decomposing a new log, while leaving five components unguided. We discover a port
We consider five weekday Bro/Zeek logs from a small business network, and for each log we construct a 4-dimensional tensor. The features chosen are timestamp, source IP address, destination IP address, and destination port, which indicate when connections occur, who is involved in those connections, and often the purpose of the connection. As discussed by Leggas et al. [37], in order to obtain coherent results, we apply a process called binning so that the indices in each mode correspond to different subsets of the feature space after suitable discretization. Specifically, the timestamp is binned to the minute while the other modes are not binned as they represent discrete entities. The value at a given index is the count of the number of records that fall into the corresponding bin. After synchronizing the modes so that indices across the different tensors refer to the same entity or time, the tensors had size 1,440 × 90 × 105 × 2,351 and 64,827; 64,601; 63,154; 63,251; and 63,374 non-zeros, respectively. As the tensors are large, sparse tensors consisting of count data, we use the Kullback-Leibler divergence loss function and perform the decomposition using ENSIGN [30], [31].

First, we perform a standard, unguided rank-100 decomposition of the five tensors and inspect the components. Each unguided decomposition has components describing various expected activities occurring on the network. For example, each decomposition has two components describing DNS requests. The semantics of these components are determined by the fact that their dominant destination IP address is the network’s DNS server and the dominant destination port is 53. The subtle difference in the components is in the timestamp mode: the first component describes continuous baseline behavior relating to DNS, while the second shows hourly spikes corresponding to a scheduled periodic job on one source machine. While the ability to separate such subtle patterns is one strength of tensor decomposition, it is difficult to correctly align the components automatically by comparing scores in the component. When the components’ modes are concatenated, the components reside in a high-dimensional vector space (in this case, \( \mathbb{R}^{3.986} \)), so inter-component distances do not differentiate components with subtly-different semantics. Indeed, in this example, aligning components between the decompositions by checking distances misidentifies the two DNS components. Given a new decomposition, a data analyst would have to duplicate their effort in studying components in order to make the correct alignment between baseline components. Moreover, this manual baselining would be necessary in order to ensure that no anomalous components are missed.

To lessen the cognitive load required to sift through components, we perform LDA inference with 100 topics, using the components as training observations. Note that treating a component as a distribution is as simple as concatenating the vectors in its outer product. The learned topics generate components, and for every class of component seen in distributions, there is a corresponding topic. Taking a new Bro/Zeek log from a different weekday, we perform a rank-105 knowledge-guided decomposition in which the 100 topics are provided as knowledge for the first 100 components and five components are left unguided. By performing a knowledge-guided decomposition of the new tensor, we are able to automatically and correctly align the different expected components with the known baseline. Moreover, as 100 of 105 components of the decomposition are accounted for in terms of known behaviors, novel, unexpected, and potentially malicious behaviors are isolated in the remaining five components, which are much easier to check than an entire decomposition. Indeed, we isolate the malicious behavior of one machine into a single component among the five that are unguided. The source machine isolated in this component periodically scans another machine on the network and attempts to connect to it by SSH. Thus our method effectively allows an analyst to easily identify novel components without spending time parsing each of the components, many of which are expected.

V. CONCLUSION

In this work, we present a novel method for knowledge-guided tensor decomposition that 1) enables efficient identification of semantically-related patterns across data sets and 2) provides a means for performing anomaly detection using tensor decomposition. This method relies on a modification of the loss function used in a CP tensor decomposition such that specific prior knowledge guides a subset of the modes in the resulting factor matrices. We validate the method’s robustness to bogus guidance information and its effectiveness in extracting coherent patterns. We also apply the method to knowledge discovery tasks for cyber network data. Our method improves decomposition quality and makes interpretation easier by guiding baseline components using the provided knowledge while leaving relatively few components to capture novel behaviors.

There are several avenues of future work. First, there is a need to better understand how selection of the penalty parameter affects the quality of the decompositions, the robustness of the decompositions with respect to bogus guidance information, and convergence performance of the optimization. Mathematical analysis of the methodology would contribute to these efforts. Second, practical techniques for consistent application of the methodology to various problems of interest and its use in large-scale deployment need to be developed. Finally, we plan to further explore the use of knowledge-guided tensor decomposition for anomaly detection.

ACKNOWLEDGMENTS

The authors thank James Ezick, Muthu Baskaran, and Tom Henretty for their conversations and feedback that helped develop this work. They also thank the anonymous reviewers for their comments, which helped improve this paper.

REFERENCES

(a) First DNS component in unguided decomposition of day 1

(b) Second DNS component in unguided decomposition of day 1

(c) Topic, reshaped as a component, corresponding to the second type of DNS component in the unguided decompositions

(d) Anomalous port scan found in guided decomposition

Fig. 2: These plots of components are taken from different decompositions. In each subplot, each bar chart corresponds to a mode in the represented component, and the bars indicate the scores of each index in the mode. The text at right is the top scoring indices in each mode. The two components in (a) and (b) are taken from decompositions of baseline traffic and describe behavior relating to DNS. As expected, the dominant destination IP address is a DNS server and the dominant port is 53. These components reflect the fact that each day contains two components describing DNS requests with subtly different interpretations. The “component” in (c) is a topic learned with LDA. Because semantically-equivalent components appear across the unguided decompositions of all network logs, topics that look like these components are learned after performing LDA inference. Finally, by providing the topics as knowledge to a guided decomposition of another network log, all baseline components can be ignored. The novel and potentially anomalous activities are isolated in the unguided components. One of these five components, shown in (d), reveals one machine periodically performing a port scan on and trying to access another machine.


