Semantically Invariant Tensor Factorization

Raphaël Bailly, Antoine Bordes, Nicolas Usunier

Semantically Invariant Tensor Factorization

Raphaël Bailly, baillyra@utc.fr [*]

Antoine Bordes, abordes@fb.com [†]

Nicolas Usunier, usunier@hds.utc.fr [‡]

Abstract

Multi-relational data can usually be represented as three-mode *tensors* with each slice (matrix) representing one relation (not necessarily symmetric) between two nodes. In this paper, we study factorization algorithms for such tensors that are semantically invariant, which means that they commute with the transposition of their frontal slices. We describe why this property is crucial for conveniently approximating such data and we demonstrate what are the necessary and sufficient conditions that any algorithm should have to fulfill it. Then, we introduce SITAR, a convex and semantic invariant algorithm, which produces low-rank approximations of tensors. We show empirically on three benchmarks that this well-defined algorithm outperforms previously presented low-rank factorization algorithm like RESCAL [11].

1 Introduction

This paper concerns tensor factorization motivated by the problem of link prediction in Knowledge Bases (KBs). KBs are popular nowadays to store and organize information in a structured fashion but suffer from incompleteness; the problem of KB completion is crucial. Information extraction can add data to KBs using external sources but other processes also attempt to add information using the KB itself, by performing (endogenous) link prediction.

A convenient way to formalize the concept of KB is to consider a set of entities $\mathcal{E}$ and a set of binary relations $\mathcal{R}$, and make a list of all triplets e_i, e_j and R_k such that the relation $R_k(e_i, e_j)$ holds. Such data can be represented by a adjacency 3-modes tensor, each slice (a binary non-symmetric matrix) representing a particular relation between entities; this tensor has the same dimension along the first two axis (the number of entities). False or missing assertions are represented by a 0, while a true and known assertion is represented by a 1. The goal of a learning algorithm for link prediction is

[*] Sorbonne universités, Université de technologie de Compiègne, CNRS, Heudiasyc UMR 7253, Compiègne, France

[†] Facebook AI Research 770 Broadway, New York, NY 10003. USA

[‡] Sorbonne universités, Université de technologie de Compiègne, CNRS, Heudiasyc UMR 7253, Compiègne, France

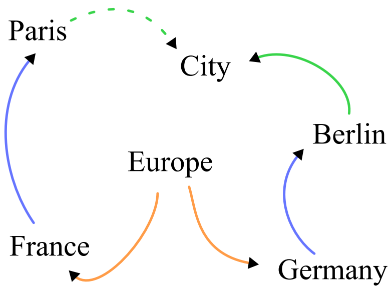

Figure 1: Example of Knowledge Base, with three relations *has-member*, *has-capital* and *has-type*

to discover new links, i.e. predict missing 1. Consider the KB of the example of Fig.1 which depicts the KB made of the following triplets: (`europe`, *has-member*, `france`), (`europe`, *has-member*, `germany`), (`france`, *has-capital*, `paris`), (`germany`, *has-capital*, `berlin`), (`berlin`, *has-type*, `city`). A link prediction algorithm algorithm should able to answer the question: (`paris`, *has-type*, `?`).

Many approaches have been proposed to perform link prediction in KBs. Most of them operate with latent representations (or embeddings) of the constituents (entities and relations) that are learnt under different formalisms. [9] proposed an extension of stochastic block models and [14] a Bayesian clustering framework. Energy-based models for learning embeddings using stochastic gradient descent have also been successfully adapted to this problem with variants using linear [3], bilinear formulations [1] or both [8]. Still, due to the tensor representation of KBs, a particular effort has concerned link prediction through tensor factorization or collective matrix factorization. Hence, standard methods like CANDECOMP/PARAFAC [7] or Tucker decomposition [17] have been applied. Yet, due to the particular form of the tensor in such cases (two dimensions represent the entities), methods derived from collective matrix factorization [13, 6] have shown to perform best. The most successful application of such approaches is RESCAL [11], which we will discuss at length in Section 3 and Section 5. Even if such collective matrix factorization methods have been conceived for the particular features of tensors representing KBs, we believe that some key properties are missing.

The work of this paper started from the observation that two relations in a KB can be inverse of each other, e.g. the relations *has-capital* and *is-capital-of*. This means that the adjacency matrix $\boldsymbol{X}_1$ associated to *has-capital* and the matrix $\boldsymbol{X}_2$ associated to *is-capital-of* are transpose of each others, that is $\boldsymbol{X}_1 = \boldsymbol{X}_2^\top$. Usually, among two inverse relationships, only one is expressed in the KB, mainly to ensure data consistency and avoid the existence of conflicting data; but this does not mean that one relation is more valid than the other. For instance, the choice, in the example, of the relation *has-capital* instead of *is-capital-of* is arbitrary. The order of the slices representing the different relations in the tensor is also arbitrary.

Hence, we would like the methods factorizing tensors representing KB data to be robust to such variations, which are semantically equivalent given the information of the KB. We term this property of factorization models *semantic invariance*. As we show in Section 2, semantic invariance requires factorization algorithms that commute with transposition and permutation of the frontal slices of the tensors. As we shall see, while this is verified by data-fitting terms of usual methods, this imposes strong constraints on the type of regularization that should be used.

This paper is organized as follows. In Section 2, we precisely define the property of commuting with transposition required by a tensor factorization method to be semantic invariant and present a theorem on the necessary and sufficient conditions under which an algorithm has it. Then, in Section 3, we demonstrate that, to fulfill these conditions, a method based on low-rank factorization must take a certain form. Sections 4 and 5 present SITAR, our convex algorithm for semantic invariant tensor factorization. We finally show in Section 6 that SITAR can achieve very promising empirical performance.

2 Semantic Invariance

The semantic invariance of a tensor factorization algorithm in the context of Knowledge Bases means that if one or several of the slices of the data matrix were transposed (i.e. we observe one or several relations in the reverse direction), then the result of the factorization should be transposed as well (so that the predicted fact is the same as before). In other words, semantic invariance means that the factorization of the "transposed tensor" should be the transposed of the factorization, or, equivalently, that the factorization algorithm should commute with the transposition operator.

In this section, we prove a necessary and sufficient condition under which a factorization algorithm for (n, n, p) tensors commutes with the transposition of the frontal slices.

2.1 Tensor Factorization Algorithms

We study algorithms that perform regularized (e.g. low-rank) approximations of tensors, which have the same dimensions along the first two modes. This incudes adjacency tensors representing KBs but is not restricted to them. Given a third-order tensor $\boldsymbol{X}$ of dimensions $\texttt{dim}(\boldsymbol{X}) = (n, n, p)$ and denoting $\mathcal{T}_{\texttt{dim}(\boldsymbol{X})}$ the set of all third-order tensors with the same dimensions as $\boldsymbol{X}$, we are primarily interested in algorithms of the form

$$\underset{\boldsymbol{W} \in \mathcal{T}_{\texttt{dim}(\boldsymbol{X})}}{\operatorname{argmin}} \ \|\boldsymbol{X} - \boldsymbol{W}\|_F^2 + \Omega(\boldsymbol{W}) \,,$$

where $\|\boldsymbol{X}\|_F^2$ is the squared Frobenius norm of $\boldsymbol{X}$, and Ω is the regularizer. In this section, we provide a necessary and sufficient condition on Ω such that if we arbitrarily transpose some of the frontal slices of $\boldsymbol{X}$ and approximate the resulting tensor, then we recover the approximation of $\boldsymbol{X}$ up to the appropriate transpositions of the frontal slices.

Our result, presented in Theorem 2, does not only apply to Frobenius-norm approximations of tensors. To express the result in its full generality, we consider a more general form of algorithms which take two tensors as input, $\boldsymbol{X}$ and $\boldsymbol{A}$ such that

$\mathtt{dim}(\boldsymbol{X}) = \mathtt{dim}(\boldsymbol{A})$, where $\boldsymbol{X}$ is the target tensor and $\boldsymbol{A}$ represents weights to individual entries of the target tensor or captures the information of whether some entries of the tensor are unknown. Denoting by $\mathcal{T}$ the set of all tensors with the same dimension along the first two modes, we study algorithms $F : \mathcal{T} \times \mathcal{T} \to \mathcal{T}$ of the form:

$$F_{\boldsymbol{A}}(\boldsymbol{X}) = \operatorname*{argmin}_{\boldsymbol{W} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})}} f_{\boldsymbol{A}}(\boldsymbol{X}, \boldsymbol{W}) + \Omega(\boldsymbol{W}) \,, \tag{1}$$

where $f_{\boldsymbol{A}} : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$ is the data-fitting term (or loss) and $\Omega : \mathcal{T} \to \mathbb{R} \cup \{+\infty\}$ is the regularizer.

Examples of regularizers An example of regularization is to impose constraints on the multilinear rank of $\boldsymbol{W}$, which depends on the rank of the unfoldings of the tensor: given an (n, n, p) tensor $\boldsymbol{X}$, the mode-1 unfolding of the tensor $\boldsymbol{W}_{(1)}$ is the $(n, n \times p)$ matrix obtained by stacking horizontally the frontal slices of the tensor (unfolding along the other modes are defined similarly). Then, a constraint on the multilinear rank, as performed in truncated Tucker decompositions [10] of $\boldsymbol{W}$ is a regularizer of the form $\Omega(\boldsymbol{W}) = \sum_{k=1}^{3} \Omega_k(\boldsymbol{W}_{(k)})$ where Ω_k is a hard constraint on the rank of the mode-k unfolding of $\boldsymbol{W}$ (i.e. $\Omega_k(\boldsymbol{W}_{(k)}) = +\infty$ if the rank of matrix $\boldsymbol{W}_{(k)}$ is greater than a threshold, and 0 otherwise). This kind of regularization on the unfoldings of the tensors is also widely used in convex tensor factorization methods, using for instance $\Omega_k(\boldsymbol{W}) = \lambda_k \left\| \boldsymbol{W}_{(k)} \right\|_*$ where $\lambda_k \in \mathbb{R}_+$ and $\left\| \boldsymbol{W}_{(k)} \right\|_*$ is the nuclear norm (sum of singular values) of matrix $\boldsymbol{W}_{(k)}$ [16, 15]. As we shall see, even though these regularizers fall into our framework of study, they are usually heavily affected by the transposition of one or several frontal slices of the tensors (see Section 2.4).

Examples of data-fitting terms An example of f used in tensor completion or tensor approximation is the Frobenius norm of $\boldsymbol{X} - \boldsymbol{W}$ weighted by $\boldsymbol{A}$ (see e.g. [5]):

$$\begin{aligned} f_{\boldsymbol{A}}(\boldsymbol{X}, \boldsymbol{W}) &= \|\boldsymbol{X} - \boldsymbol{W}\|_{F,\boldsymbol{A}} \qquad (2) \\ &= \sum_{i,j,k} \boldsymbol{A}_{i,j,k} (\boldsymbol{X}_{i,j,k} - \boldsymbol{W}_{i,j,k})^2 \,, \end{aligned}$$

where $\boldsymbol{X}_{i,j,k}$ is the entry at coordinates (i, j, k) of tensor $\boldsymbol{X}$. Another example is the ranking criterion of e.g. [3], used for link prediction in the case of binary adjacency tensors (this loss does not depend on $\boldsymbol{A}$):

$$\begin{aligned} & f_{\boldsymbol{A}}(\boldsymbol{X}, \boldsymbol{W}) = \qquad (3) \\ & \quad \sum_{i,j,k} \boldsymbol{X}_{i,j,k} \sum_{i' \neq i} \max(0, 1 - \boldsymbol{W}_{i,j,k} + \boldsymbol{W}_{i',j,k}) \\ & \quad + \sum_{i,j,k} \boldsymbol{X}_{i,j,k} \sum_{j' \neq j} \max(0, 1 - \boldsymbol{W}_{i,j,k} + \boldsymbol{W}_{i,j',k}) \,. \end{aligned}$$

2.2 Commuting with Transposition

We now introduce the main property of tensor factorization algorithms that we study: factorization commutes with the transposition of frontal slices. Such transpositions are formalized by the operator $\top$: given a tensor $\boldsymbol{X}$ of dimensions (n, n, p) and a vector $\boldsymbol{b} = (b_1, ..., b_p) \in \{0,1\}^p$, $\boldsymbol{X}^{\top_{\boldsymbol{b}}}$ transposes $\boldsymbol{X}_{::k}$, the k-th frontal slice of $\boldsymbol{X}$, when $b_k = 1$, and leaves $\boldsymbol{X}_{::k}$ unchanged when $b_k = 0$.

$$\boldsymbol{X}^{\top_{\boldsymbol{b}}} = \tilde{\boldsymbol{X}} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})} \text{ with } \tilde{\boldsymbol{X}}_{::k} = \begin{cases} \boldsymbol{X}_{::k}^T & \text{if } b_k = 1 \\ \boldsymbol{X}_{::k} & \text{if } b_k = 0 \end{cases} .$$

Hence the following property:

Definition 1 (Factorization commutes with transposition)**.** *We say that F commutes with transposition if, for any $\boldsymbol{X}, \boldsymbol{A} \in \mathcal{T}_{n,n,p}$ and any $\boldsymbol{b} \in \{0,1\}^p$, we have:*

$$F_{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}\left(\boldsymbol{X}^{\top_{\boldsymbol{b}}}\right) = F_{\boldsymbol{A}}(\boldsymbol{X})^{\top_{\boldsymbol{b}}} .$$

In other words, should we transpose any frontal slice(s) of the original tensor $\boldsymbol{X}$ (together with the weight tensor $\boldsymbol{A}$), the factorization should grant the transposition of the result.

2.3 Assumptions

The result we prove in Theorem 2 is a necessary and sufficient condition that Ω should satisfy so that F satisfies the property of commutation with transposition of Definition 1. The equivalence is bound to three underlying assumptions on on F, f and Ω that we describe below.

Transposition invariance of the cost function The first and most important one is that the data-fitting term f should be invariant under any joint transposition of the frontal slices of $\boldsymbol{A}$, $\boldsymbol{X}$ and $\boldsymbol{W}$.

The invariance by transposition of f is formally stated by:

$$\begin{aligned} &\forall \boldsymbol{X}, \boldsymbol{A}, \boldsymbol{W} \in \mathcal{T}_{n,n,p}, \forall \boldsymbol{b} \in \{0,1\}^p, \\ &f_{\boldsymbol{A}}(\boldsymbol{X}, \boldsymbol{W}) = f_{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}\left(\boldsymbol{X}^{\top_{\boldsymbol{b}}}, \boldsymbol{W}^{\top_{\boldsymbol{b}}}\right) . \end{aligned} \tag{4}$$

This is a natural assumption and one can easily check that both data-fitting terms of (2) and (3) satisfy this invariance for instance.

Duplication invariance This property concerns the learning function F and means that when the input data is duplicated, i.e. two copies of each relationship are considered, then the result is a duplication of the initial result.

Let n, p, p' be three integers, and let $\boldsymbol{X} \in \mathcal{T}_{n,n,p}$ and $\boldsymbol{X}' \in \mathcal{T}_{n,n,p'}$. We denote by $(\boldsymbol{X}|_3\boldsymbol{X}')$ the $(n, n, p + p')$ tensor obtained by stacking along the third mode $\boldsymbol{X}$ and $\boldsymbol{X}'$:

$$(\boldsymbol{X}|_3\boldsymbol{X}') = \tilde{\boldsymbol{X}} \text{ with } \tilde{\boldsymbol{X}}_{::k} = \begin{cases} \boldsymbol{X}_{::k} & \text{if } 1 \leq k \leq p \\ \boldsymbol{X}'_{::k-p} & \text{if } 1 \leq k - p \leq p' \end{cases} .$$

Then, we assume that F satisfies the following invariance: given a target tensor $\boldsymbol{X}$ and a weighting tensor $\boldsymbol{A}$, the result of appliying F to $\boldsymbol{X}$ and $\boldsymbol{A}$ stacked with themselves is the stacking of the result of $F_{\boldsymbol{A}}(\boldsymbol{X})$:

$$\forall \boldsymbol{X} \in \mathcal{T}, \forall \boldsymbol{A} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})}, F_{(\boldsymbol{A}|_3\boldsymbol{A})}((\boldsymbol{X}|_3\boldsymbol{X})) = \left(F_{\boldsymbol{A}}(\boldsymbol{X})\,|_3 F_{\boldsymbol{A}}(\boldsymbol{X})\,\right). \tag{5}$$

This requirement is natural: by jointly approximating twice the same dataset, we obtain twice the same result.

Slice permutation invariance A third property that we may need id for f and Ω to be invariant by a permutation of the frontal slices. This requirement is met in most methods since the indexing of the slices rarely conveys any semantics. To that end, let $\boldsymbol{X} \in \mathcal{T}_{n,n,p}$ and $\boldsymbol{\sigma}$ a permutation of $\{1, \ldots, p\}$. Then, the permutation operator is:

$$\mathtt{P}_{\boldsymbol{\sigma}}(\boldsymbol{X}) = \tilde{\boldsymbol{X}} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})} \text{ with } \tilde{\boldsymbol{X}}_{::k} = \boldsymbol{X}_{::\boldsymbol{\sigma}(k)}$$

and the invariance by permutation of frontal slices of f and Ω can be written as:

$$\begin{aligned} &f_{\mathtt{P}_{\boldsymbol{\sigma}}(\boldsymbol{A})}(\mathtt{P}_{\boldsymbol{\sigma}}(\boldsymbol{X})\,, \mathtt{P}_{\boldsymbol{\sigma}}(\boldsymbol{W})) = f_{\boldsymbol{A}}(\boldsymbol{X}, \boldsymbol{W}) \text{ and} \\ &\Omega(\mathtt{P}_{\boldsymbol{\sigma}}(\boldsymbol{W})) = \Omega(\boldsymbol{W})\,. \end{aligned} \tag{6}$$

We notice at his point that a direct consequence of (6) is that F commutes with permutations of frontal slices:

$$F_{\mathtt{P}_{\boldsymbol{\sigma}}(\boldsymbol{A})}(\mathtt{P}_{\boldsymbol{\sigma}}(\boldsymbol{X})) = \mathtt{P}_{\boldsymbol{\sigma}}(F_{\boldsymbol{A}}(\boldsymbol{X}))\ . \tag{7}$$

2.4 Fundamental Counterexample

We show in this section why typical tensor approximation or completion algorithms relying on usual constraints on the rank of the unfoldings of the tensor *do not lead to algorithms that commute with transposition* by studying a counterexample algorithm.

Before that, we give here a simple lemma that simplifies the analysis. To simplify notation, given a tensor $\boldsymbol{X}$, we write $\boldsymbol{X}^{\top_{12}}$ the tensor obtained by transposing all frontal slices, i.e.

$$\boldsymbol{X}^{\top_{12}} = \boldsymbol{X}^{\top_{(1,\ldots,1)}}\ .$$

Lemma 1. *Assume F is of the form* (1) *and that f, F and Ω satisfy the conditions of Equations 4, 5 and 6.*

Then F commutes with transposition if and only if
$\forall \boldsymbol{X} \in \mathcal{T}, \forall \boldsymbol{A} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})}$,

$$F_{(\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}})\Big) = \left(F_{\boldsymbol{A}}(\boldsymbol{X})\,|_3 F_{\boldsymbol{A}}(\boldsymbol{X})^{\top_{12}}\right). \tag{8}$$

(Condition (6) *is not required for the only if direction.)*

Proof. only if direction: By the commutation with transposition and the definition of $\top$, we have $F_{(\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}})\Big) = F_{\boldsymbol{A}|_3\boldsymbol{A}}((\boldsymbol{X}|_3\boldsymbol{X}))^{\top_{\boldsymbol{b}}}$ where $\boldsymbol{b}$ contains p zeros followed by p ones. The duplication invariance (5) gives us

$$\begin{aligned} F_{(\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}})\Big) &= \big(F_{\boldsymbol{A}}(\boldsymbol{X})\,|_3 F_{\boldsymbol{A}}(\boldsymbol{X})\,\big)^{\top_{\boldsymbol{b}}} \\ &= \big(F_{\boldsymbol{A}}(\boldsymbol{X})\,|_3 F_{\boldsymbol{A}}(\boldsymbol{X})^{\top_{12}}\big)\,. \end{aligned}$$

if direction: Let $\boldsymbol{X} \in \mathcal{T}$, $(n,n,p) = \texttt{dim}(tens)$, $\boldsymbol{A} \in \mathcal{T}_{n,n,p}$, and $\boldsymbol{b} \in \{0,1\}^p$. Considering the tensor of size $2p$ $(\boldsymbol{A}^{\top_{\boldsymbol{b}}}|_3{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}^{\top_{12}})$, let us define the permutation $\boldsymbol{\sigma}$ of $\{1, ..., 2p\}$ such that:

$$\boldsymbol{\sigma}(k) = \begin{cases} k & \text{if and } b_{((k-1)\mathrm{mod}\, p)+1} = 0 \\ p+k & \text{if } k \leq p \text{ and } b_k = 1 \\ k-p & \text{if } k > p \text{ and } b_k = 1 \end{cases} \tag{9}$$

such that $(\boldsymbol{X}^{\top_{\boldsymbol{b}}}|_3{\boldsymbol{X}^{\top_{\boldsymbol{b}}}}^{\top_{12}}) = \mathtt{P}_{\boldsymbol{\sigma}}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}})\Big)$. Using the fact that F commutes with permutations of frontal slices (7), and the assumption (8), we obtain:

$$\big(F_{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}\Big(\boldsymbol{X}^{\top_{\boldsymbol{b}}}\Big)\,|_3 F_{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}\Big(\boldsymbol{X}^{\top_{\boldsymbol{b}}}\Big)^{\top_{12}}\big) = \mathtt{P}_{\boldsymbol{\sigma}}\Big(\big(F_{\boldsymbol{A}}(\boldsymbol{X})\,|_3 F_{\boldsymbol{A}}(\boldsymbol{X})^{\top_{12}}\big)\Big)\ .$$

The result comes by comparing the first p frontal slices of the two sides of the equation, and noticing that the first p slices of the right-hand side are exactly $F_{\boldsymbol{A}}(\boldsymbol{X})^{\top_{\boldsymbol{b}}}$. □

Before giving the example, we introduce a notation on the stacking of matrices that will be used in subsequent sections. Given two integers n and m, we denote by $\mathcal{M}_{n,m}$ the set of matrices of dimensions (n,m) and by $\texttt{rk}(\boldsymbol{M})$ the rank of matrix $\boldsymbol{M}$. Given two matrices $\boldsymbol{M} \in \mathcal{M}_{n,m}$ and $\boldsymbol{M'} \in \mathcal{M}_{n,m'}$, we denote by $(\boldsymbol{M}|\boldsymbol{M'})$ the $(n, m+m')$ matrix formed by stacking the columns of $\boldsymbol{M}$ and $\boldsymbol{M'}$:

$$\begin{aligned} &(\boldsymbol{M}|\boldsymbol{M'}) = \tilde{M} \in \mathcal{M}_{n,m+m'} \\ &\text{with } \tilde{\boldsymbol{M}}_{i,j} = \begin{cases} \boldsymbol{M}_{i,j} & \text{if } 1 \leq j \leq m' \\ \boldsymbol{M'}_{i,j-m} & \text{if } 1 \leq j-m \leq m' \end{cases}\ . \end{aligned}$$

Now let us consider the following standard tensor factorization algorithm, which minimizes the Frobenius norm under a rank contraint on the mode-1 unfolding (ignoring the weight tensor $\boldsymbol{A}$):

$$\begin{aligned} &\underset{\boldsymbol{W} \in \mathcal{T}_{\texttt{dim}(\boldsymbol{X})}}{\operatorname{argmin}} \ \|\boldsymbol{X} - \boldsymbol{W}\|_F^2 \\ &\quad \text{u.c. } \texttt{rk}\big(\boldsymbol{W}_{(1)}\big) \leq d\,. \end{aligned} \tag{10}$$

Given some (n,n) matrix $\boldsymbol{M}$ construct the $(n,n,2)$ tensor $(\boldsymbol{M}|_3\boldsymbol{M}^T)$. The solution to (10) is given by the truncated SVD of $(\boldsymbol{M}|_3\boldsymbol{M}^T)_{(1)} = (\boldsymbol{M}|\boldsymbol{M}^T)$ at rank d. However, if we denote by $(\boldsymbol{S_1}|\boldsymbol{S_2})$ the truncated SVD of $(\boldsymbol{M}|\boldsymbol{M}^T)$ with $\boldsymbol{S}_1, \boldsymbol{S}_2 \in \mathcal{M}_{n,n}$, we cannot guarantee $\boldsymbol{S}_1 = \boldsymbol{S}_2^T$ in general. Hence, using Lemma 1, we have that tensor approximations with constraints on the rank of the unfodings *do not commute* with transposition in general.

2.5 Properties of Regularizers

Our main result is that a tensor approximation algorithm commutes with the transposition of frontal slices if and only if it is equivalent to approximate $\boldsymbol{X}$ or to approximate $(\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}})$ by enforcing the constraint of Lemma 1. Before proving the main result, we first give a mean to transform any approximation algorithm of the form (1) to an approximation algorithm that commutes with the transposition of frontal slices.

Theorem 1. *Assume that f and Ω are invariant by permutation of frontal slices (Equation 6).*

Let us define $\tilde{F} : \mathcal{T} \times \mathcal{T} \to \mathcal{T}$ as follows:

$$\tilde{F}_{\boldsymbol{A}}(\boldsymbol{X}) = \underset{\boldsymbol{W} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})}}{\operatorname{argmin}} f_{(\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}), (\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})\Big) \\ + \Omega((\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}}))\,.$$

Then $\tilde{F}$ commutes with the transposition of frontal slices.

Proof. Let $(n, n, p) = \mathtt{dim}(\boldsymbol{X})$, $\boldsymbol{b} \in \{0, ..., 1\}^p$, and $\bar{\boldsymbol{b}} \in \{0, ..., 1\}^p$ such that $\bar{b}_k = 1 - b_k$. Then $\tilde{F}_{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}\left(\boldsymbol{X}^{\top_{\boldsymbol{b}}}\right)$ equals

$$\underset{\boldsymbol{W} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})}}{\operatorname{argmin}} f_{(\boldsymbol{A}^{\top_{\boldsymbol{b}}}|_3\boldsymbol{A}^{\top_{\bar{\boldsymbol{b}}}})}\Big((\boldsymbol{X}^{\top_{\boldsymbol{b}}}|_3\boldsymbol{X}^{\top_{\bar{\boldsymbol{b}}}}), (\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})\Big) \\ + \Omega((\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}}))\,.$$

With the change of variable $\boldsymbol{W} \leftarrow \boldsymbol{W}^{\top_{\boldsymbol{b}}}$, if we consider

$$\boldsymbol{Y} = \underset{\boldsymbol{W} \in \mathcal{T}_{\mathtt{dim}(\boldsymbol{X})}}{\operatorname{argmin}} f_{(\boldsymbol{A}^{\top_{\boldsymbol{b}}}|_3\boldsymbol{A}^{\top_{\bar{\boldsymbol{b}}}})}\Big((\boldsymbol{X}^{\top_{\boldsymbol{b}}}|_3\boldsymbol{X}^{\top_{\bar{\boldsymbol{b}}}}), (\boldsymbol{W}^{\top_{\boldsymbol{b}}}|_3\boldsymbol{W}^{\top_{\bar{\boldsymbol{b}}}})\Big) \\ + \Omega((\boldsymbol{W}^{\top_{\boldsymbol{b}}}|_3\boldsymbol{W}^{\top_{\bar{\boldsymbol{b}}}}))\,,$$

then, by noticing that ${\boldsymbol{W}^{\top_{\boldsymbol{b}}}}^{\top_{\boldsymbol{b}}} = \boldsymbol{W}$, we have

$$\boldsymbol{Y}^{\top_{\boldsymbol{b}}} = \tilde{F}_{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}\left(\boldsymbol{X}^{\top_{\boldsymbol{b}}}\right)\,. \tag{11}$$

Now, using the invariance by permutation of frontal slices of f and Ω (6), and defining $\boldsymbol{\sigma}$ given $\boldsymbol{b}$ as in (9) in the proof of Lemma 1, we have $\mathtt{P}_{\boldsymbol{\sigma}}\Big((\boldsymbol{A}^{\top_{\boldsymbol{b}}}|_3\boldsymbol{A}^{\top_{\bar{\boldsymbol{b}}}})\Big) = (\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})$, and the same holds for $\boldsymbol{X}$ and $\boldsymbol{W}$. We then have $\boldsymbol{Y} = \tilde{F}_{\boldsymbol{A}}(\boldsymbol{X})$ and consequently $\tilde{F}_{\boldsymbol{A}}(\boldsymbol{X})^{\top_{\boldsymbol{b}}} = \tilde{F}_{\boldsymbol{A}^{\top_{\boldsymbol{b}}}}\left(\boldsymbol{X}^{\top_{\boldsymbol{b}}}\right)$ by (11), which is the desired result. □

We can finally state our main theorem:

Theorem 2. *Assume F is of the form* (1) *and that f, F and Ω satisfy the conditions of Equations 4, 5 and 6.*

Then F commutes with the transposition of frontal slices if and only if $\forall \boldsymbol{X} \in \mathcal{T}, \forall \boldsymbol{A} \in \mathcal{T}_{\texttt{dim}(\boldsymbol{X})}$,

$$\begin{aligned} F_{\boldsymbol{A}}(\boldsymbol{X}) = \operatorname*{argmin}_{\boldsymbol{W}\in\mathcal{T}_{\texttt{dim}(\boldsymbol{X})}} \; & f_{(\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}),(\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})\Big) \\ & + \Omega((\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}}))\,, \end{aligned} \tag{12}$$

(Condition (6) *is not required for the only if direction.)*

Proof. only if direction: Let $(n,n,p) = \texttt{dim}(\boldsymbol{X})$. If we denote by $(\boldsymbol{Y}|_3\boldsymbol{Y}') = F_{(\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}})\Big)$, where $\boldsymbol{Y}$ and $\boldsymbol{Y}'$ have dimensions (n,n,p), Lemma 1 implies that $\boldsymbol{Y}' = \boldsymbol{Y}^{\top_{12}}$. We can thus obtain the same result by restricting the argmin of F in (1) to consider only tensors of the form $(\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})$ with $\boldsymbol{W} \in \mathcal{T}_{n,n,p}$. Thus, $F_{\boldsymbol{A}}(\boldsymbol{X})$ is equal to the first p frontal slices of

$$\begin{aligned} \operatorname*{argmin}_{\substack{(\boldsymbol{W}|_3\boldsymbol{W}') \\ \boldsymbol{W}\in\mathcal{T}_{\texttt{dim}(\boldsymbol{X})} \\ \boldsymbol{W}'\in\mathcal{T}_{\texttt{dim}(\boldsymbol{X})}}} \; & f_{(\boldsymbol{A}|_3\boldsymbol{A}^{\top_{12}})}\Big((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}),(\boldsymbol{W}|_3\boldsymbol{W}')\Big) \\ & + \Omega((\boldsymbol{W}|_3\boldsymbol{W}')) \\ \text{u.c.}\;\; & \boldsymbol{W}^{\top_{12}} = \boldsymbol{W}' \end{aligned}$$

which is equivalent to the desired result (12).

if direction: It is a direct consequence of Theorem 1. If we define $\tilde{F}$ as in Theorem 1, Eq. 12 says that $F = \tilde{F}$ and thus F commutes with the transposition of frontal slices. □

When the data-fitting term f is invariant by transposition, and if we want that factorization commutes with transposition, then it is intuitive to say that the regularizer itself should be invariant by transposition. In fact, Theorem 1 precisely gives a means to make the regularizer invariant by transposition: it is sufficient to regularize the stacking of the parameter tensor with its transpose. We present in the next section an algorithm that uses this transformation in a low-rank tensor factorization setting.

3 Rank Constraint and Shared Embeddings

In this section, we now apply the results of Theorem 1 to the case of low-rank tensor factorization. More precisely, we will consider the case where the Ω regularizer is a constraint on the rank of the different unfoldings of the tensor. This type of constraint is a very common regularization; see for instance [6, 12, 11].

The main result of this section demonstrates an equivalence between the regularization promoted by Theorem 1 and a factorization of the form $\boldsymbol{W}_k = \boldsymbol{U}\boldsymbol{R}_k\boldsymbol{U}^\top$, where $\boldsymbol{U}$ is a low-rank matrix of embeddings, shared by both sides of triplets, and where $\boldsymbol{R}_k$ are factorized matrices corresponding to the approximation $\boldsymbol{W}_k$ of the slice $\boldsymbol{X}_k$. Thus, each entity is represented by a single vector, which is shared by all relations, regardless of its position. In other words, we show that, in order to commute with transpositions and hence be semantic invariant, a tensor factorization constraining the rank of the unfoldings should take the form of a $\boldsymbol{U}\boldsymbol{R}_k\boldsymbol{U}^\top$ factorization.

Constraining the rank of the unfoldings Let us consider the following approximation of a tensor $\boldsymbol{X}$

$$F(\boldsymbol{X}) = \operatorname{argmin} f(\boldsymbol{X}, \boldsymbol{W}) + \Omega(\boldsymbol{W})$$

where the regularization $\Omega(\boldsymbol{W})$ acts as a hard constraint on the rank of the different unfoldings. Hence, the minimization problem can be written:

$$\begin{aligned} F(\boldsymbol{X}) = &\operatorname{argmin} f(\boldsymbol{X}, \boldsymbol{W}) \\ &\text{u.c. } \mathtt{rk}\big(\boldsymbol{W}_{(i)}\big) \leq d_i \text{ for } i = 1, 2, 3 \ . \end{aligned} \tag{13}$$

Theorem 2 shows that, to commute with transposition, the minimization problem (13) has to be equivalent to

$$\begin{aligned} F(\boldsymbol{X}) = &\operatorname{argmin} f((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}), (\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})) \\ &\text{u.c. } \mathtt{rk}\Big((\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(i)}\Big) \leq d_i \text{ for } i = 1, 2, 3 \ . \end{aligned} \tag{14}$$

One can check that, for the two first modes, the rank contraints are both equivalent to

$$\mathtt{rk}\big((\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)})\big) \leq d$$

where $(\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)})$ is the stacking of the two unfoldings of $\boldsymbol{W}$ along the two first modes. Finally, the minimization problem can be written

$$\begin{aligned} F(\boldsymbol{X}) = &\operatorname{argmin} f((\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}), (\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})) \\ &\text{u.c. } \mathtt{rk}\big(\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)}\big) \leq d \\ &\quad\ \mathtt{rk}\Big((\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(3)}\Big) \leq d' \ . \end{aligned} \tag{15}$$

The next result shows the equivalence between using the constraint $\mathtt{rk}\big(\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)}\big) \leq d$ and a factorization along the two first modes with shared embeddings $\boldsymbol{U}$:

Theorem 3. *For any (n, n, p) tensor $\boldsymbol{W}$, the two conditions are equivalent:*

$$\mathtt{rk}\big((\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)})\big) \leq d \tag{16}$$

and

$$\boldsymbol{W}_k = \boldsymbol{U}\boldsymbol{R}_k\boldsymbol{U}^\top, \ \textit{with}\ \mathtt{rk}(\boldsymbol{U}) \leq d \tag{17}$$

Proof. First, one can check that the constraint (16) is equivalent to

$$\mathtt{rk}\Big((\boldsymbol{W}_1|\ldots|\boldsymbol{W}_p|\boldsymbol{W}_1^\top|\ldots,|\boldsymbol{W}_p^\top)\Big) \leq d$$

Indeed, the fibers of $\boldsymbol{W}$ along the mode 1 (resp. 2) are the rows (resp. columns) of the slices $\boldsymbol{W}_k$, so $\boldsymbol{W}_{(1)}$ (resp. $\boldsymbol{W}_{(2)}$) is equal to $(\boldsymbol{W}_1^\top|\ldots,|\boldsymbol{W}_p^\top)$ (resp. $(\boldsymbol{W}_1|\ldots,|\boldsymbol{W}_p)$) (more or less the columns order).

$$\boldsymbol{X} = \begin{array}{c} \\ \texttt{berlin} \\ \texttt{france} \\ \texttt{city} \\ \texttt{europe} \\ \texttt{germany} \\ \texttt{paris} \end{array} \begin{array}{cccccc} \texttt{berlin} & \texttt{france} & \texttt{city} & \texttt{europe} & \texttt{germany} & \texttt{paris} \\ \end{array}$$

$$\boldsymbol{X} = \begin{array}{c|cccccc} & \texttt{berlin} & \texttt{france} & \texttt{city} & \texttt{europe} & \texttt{germany} & \texttt{paris} \\ \texttt{berlin} & 0 & 0 & 1 & 0 & 0 & 0 \\ \texttt{france} & 0 & 0 & 0 & 0 & 0 & 1 \\ \texttt{city} & 0 & 0 & 0 & 0 & 0 & 0 \\ \texttt{europe} & 0 & 1 & 0 & 0 & 1 & 0 \\ \texttt{germany} & 1 & 0 & 0 & 0 & 0 & 0 \\ \texttt{paris} & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Figure 2: Example of a single-relation knowledge base.

Now, it is clear that if $\boldsymbol{W}_k$ has the form $\boldsymbol{U}\boldsymbol{R_k}\boldsymbol{U}^\top$ with $\texttt{rk}(\boldsymbol{U}) \leq d$, then

$$\texttt{rk}\Big((\boldsymbol{W}_1|\dots|\boldsymbol{W}_p|\boldsymbol{W}_1^\top|\dots,\boldsymbol{W}_p^\top)\Big) \leq d$$

and the tensor $\boldsymbol{W}$ satisfies the constraints of Equation (17).

Conversely, if the matrices $\boldsymbol{W}_k$ satisfy constraints of Equation (17), let

$$U\boldsymbol{D}[\boldsymbol{V}_1^\top|\dots|\boldsymbol{V}_p^\top|{\boldsymbol{V}'_1}^\top|\dots,{\boldsymbol{V}'_p}^\top]$$

be the truncated SVD of $(\boldsymbol{W}_1|\dots|\boldsymbol{W}_p|\boldsymbol{W}_1^\top|\dots,\boldsymbol{W}_p^\top)$ to the rank d (i.e. $\boldsymbol{U}$ is an (n,d) matrix).

One has, for all k, $\boldsymbol{W}_k = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}_k^\top = \boldsymbol{V}'_k\boldsymbol{D}\boldsymbol{U}^\top$. As $\boldsymbol{U}$ is a unitary matrix, one has $\boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I}_r$, and $\boldsymbol{U}\boldsymbol{U}^\top\boldsymbol{W}_k = \boldsymbol{U}(\boldsymbol{U}^\top\boldsymbol{U})\boldsymbol{D}\boldsymbol{V}_k^\top = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}_k^\top = \boldsymbol{W}_k$, thus $\boldsymbol{W}_k = \boldsymbol{U}\boldsymbol{U}^\top\boldsymbol{V}'_k\boldsymbol{D}\boldsymbol{U}^\top = \boldsymbol{U}\boldsymbol{R}_k\boldsymbol{U}^\top$ with $\boldsymbol{R}_k = \boldsymbol{U}^\top\boldsymbol{V}'_k\boldsymbol{D}$. Hence, $\forall k, \boldsymbol{W}_k$ has the form $\boldsymbol{U}\boldsymbol{R_k}\boldsymbol{U}^\top$ with $\boldsymbol{U}$ being an (n,d) matrix. □

4 A Convex Relaxation of RESCAL

This section focuses on the case where the data-fitting function f is given by

$$f(\boldsymbol{X},\boldsymbol{W}) = \|\boldsymbol{X} - \boldsymbol{W}\|_F^2 \,.$$

The previous section proved that a rank constraint, in order to commute with transposition, should take the form of a constraint on a shared representation of entities. We show now that transforming the optimization problem indicated by Theorem 2 leads to a RESCAL-like algorithm [11], but also that the hard constraint on the rank embedding matrix $\boldsymbol{U}$ is too coarse to solve adequately the compromise between generalization capacity and consistency with the training data.

In a second step, we establish that by replacing the rank constraint by a nuclear norm penalization, which is a standard way of relaxing a rank constraint for tensor factorization problems, Theorem 2 can actually provide a convex relaxation of RESCAL.

4.1 Failure of an Hard Rank Constraint

Let us consider the simple single-relation KB of Fig. 2. This KB is built from the merging of the three relations of the introduction example of Fig. 1 with the only relation considered being '*has-member* OR *has-capital* OR *has-type*'. Our goal here is to predict the link from `paris` to `city`, as stated in Fig. 3.

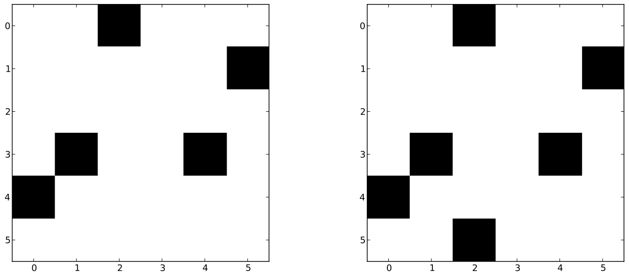

Figure 3: Input matrix $\boldsymbol{X}$ (left) and expected matrix (right) with the triplet (`paris`, *has-type*,`city`). (*black*=1, *white*=0)

Considering the optimization problem as in the fundamental counterexample 2.4, i.e. minimization of the Frobenius norm with a rank constraint on the mode-1 unfolding, is hopeless if one wants to be able to predict the `paris`-`city` link because this regularization would not update a null vector (that of `paris`). We now see that considering the commutativity with transposition leads to a regularization that helps to solve this link prediction task.

Commutativity with transposition as regularization Applying the recipe from Theorem 2, the minimization problem to solve is the following:

$$\begin{aligned} F(\boldsymbol{X}) = &\operatorname{argmin} \|(\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}) - (\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})\|_F^2 \\ &\text{u.c. } \mathtt{rk}\big(\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)}\big) \leq d \\ &\qquad \mathtt{rk}\Big((\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(3)}\Big) \leq d' \ . \end{aligned} \tag{18}$$

On can check that the minimization of $\|(\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}) - (\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})\|_F^2$ is equivalent to that of $\|\boldsymbol{X} - \boldsymbol{W}\|_F^2$, and with Theorem 3, one can show that (18) is equivalent to

$$\begin{aligned} F(\boldsymbol{X}) = &\operatorname{argmin} \|\boldsymbol{X} - \boldsymbol{W}\|_F^2 \\ &\text{u.c. } \boldsymbol{W}_k = \boldsymbol{U}\boldsymbol{R}_k\boldsymbol{U}^\top, rk(\boldsymbol{U}) \leq d \\ &\qquad \mathtt{rk}\Big((\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(3)}\Big) \leq d' \end{aligned} \tag{19}$$

which is a variation of RESCAL (with a rank constraint on the third mode instead of a penalty on $\|U\|_F^2$ and $\|R\|_F^2$). Applying the minimization of (19) to the example of Fig. 2 gives the result represented Fig. 4. The link (`paris`, *has-type*, `city`) clearly appears for an embedding matrix of rank $d = 4$. However, the information (`france`,*has-capital*,`paris`) gets lost: the rank constraint is too strong, and enforces the embeddings of `france` and `germany` to be very similar.

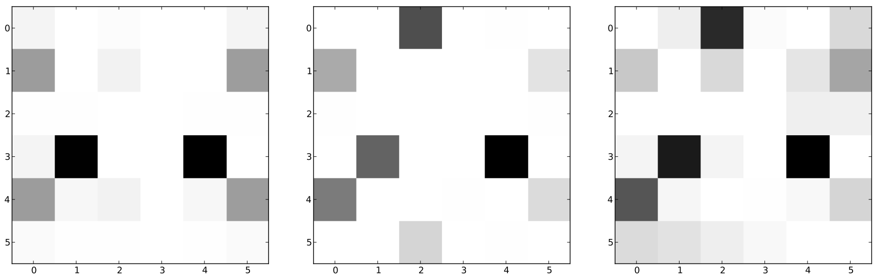

Figure 4: Solutions of the problem of Fig. 2, provided by minimization of (19) for $d = 3, 4, 5$ from left to right.

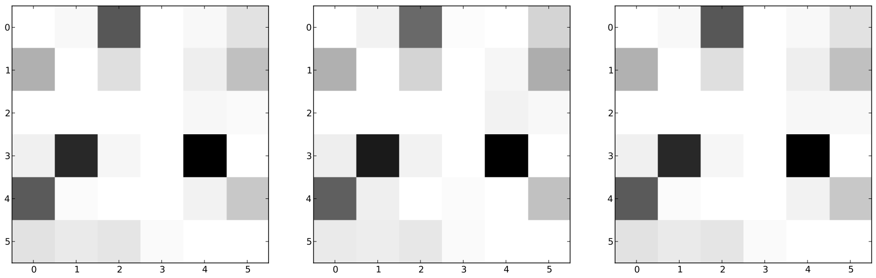

Figure 5: Solutions of RESCAL with $d = 5$, and (λ_U, λ_R) equal to $(0.9, 0.05), (1, 0.05), (1, 0.06)$ from left to right.

The algorithm RESCAL uses an extra regularization term $\frac{\lambda_U}{2}\|\boldsymbol{U}\|_F^2 + \frac{\lambda_R}{2}\|\boldsymbol{R}\|_F^2$ which can be tuned in order to get the correct prediction for the link, while preserving initial information. Some solutions provided by RESCAL are depicted in Fig. 5. The result is very sensitive to the parameter tuning: if the solution for $d = 5$, (λ_U, λ_R) equal to $(1, 0.05)$ seems to be consistent (it predicts correctly (`paris`, *has-type*, `city`’) without losing information), in the two other cases the triplet (`france`, *has-capital*, `paris`) has a lower score than (`france`, *has-capital*, `berlin`). This means in that there is few hope to find a correct parameter tuning with standard methods via cross-validation, because of the instability on the rank of the predicted triplets.

4.2 Convex Relaxation of the Rank Constraint

We saw with a simple example, that considering a minimization problem commuting with transposition – with a rank constraint as a regularizer – leads to an algorithm based on shared embeddings and that this algorithm can predict new valid links. However, the obtained regularization is not entirely satisfactory, because it does not allow for fine adjustments of the generalization trade-off. Even for RESCAL, the tuning of hyper-parameters seems too unstable to ensure good generalization ability.

Many works have been conducted to designe efficient convex tensor factorization,

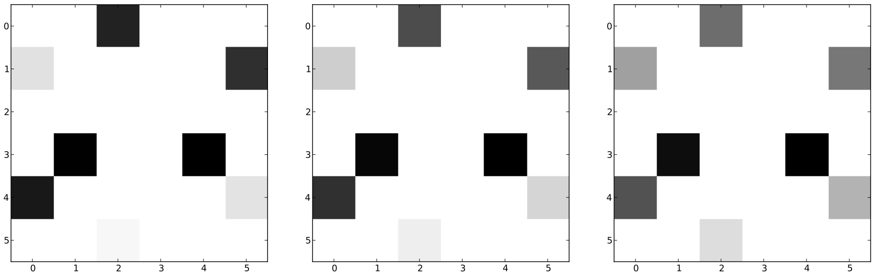

Figure 6: Solutions provided by minimizing (21), with $\lambda' = 0$ (single relation) and $\lambda = 1, 1.5, 2$ (from left to right). In each case, the link 'paris *has-type* city' is predicted with no loss of information.

primarily based on the use of nuclear norm penalty instead of a hard constraint on the rank e.g. [16, 4, 15]. We now propose a method for tensor factorization based on minimizing the Frobenius norm and a nuclear norm penalty as a regularizer. The design of the functional to be minimized is directly driven by the the property of commuting with transpositions.

Consider the following minimization problem:

$$\begin{aligned} F(\boldsymbol{X}) = &\operatorname{argmin} \|(\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}) - (\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})\|_F^2 \\ &+ \sum_{i=1}^{3} \lambda_i \|(\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(i)}\|_* \end{aligned} \tag{20}$$

One has that $\|(\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(1)}\|_* = \|(\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(2)}\|_* = \|(\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)})\|_*$, hence an equivalent minimization problem is

$$\begin{aligned} F(\boldsymbol{X}) = &\operatorname{argmin} \|\boldsymbol{X} - \boldsymbol{W}\|_F^2 \\ &+ \lambda\|(\boldsymbol{W}_{(1)}|\boldsymbol{W}_{(2)})\|_* + \lambda'\|(\boldsymbol{W}|_3\boldsymbol{W}^{\top_{12}})_{(3)}\|_* \end{aligned} \tag{21}$$

Applying the minimization of (21) to the example of Fig. 2 gives the results represented in Fig. 6. For a wide range of parameters, the link paris – city is correctly predicted, with no loss of information: the score of france-paris remains higher than that of france-berlin for all the configurations. The results are also less noisy than with a hard constraint of rank, or even than for RESCAL.

5 SITAR algorithm

This section presents our final algorithm SITAR (for Semantic Invariant Tensor Approximation through Regularization), which takes up the idea of tensor factorization by minimizing the Frobenius norm with a nuclear norm regularization of (21) with the addition of the ability to relax the equality constraint corresponding to the property (8):

Algorithm 1 *SITAR* $_{12}$

```
Input: tensor $\boldsymbol{X}$, parameters $\mu$ and $\lambda$, precision $\epsilon$
$\boldsymbol{Y}, \boldsymbol{Y}' = \boldsymbol{0}$, $L_{new} = \|\boldsymbol{X}\|_F^2$, $\delta = \frac{1}{2(\mu+1)}$
repeat
    $\nabla_1 = (\boldsymbol{Y} - \boldsymbol{X}) + \mu(\boldsymbol{Y} - \boldsymbol{Y}')$
    $\nabla_2 = \mu(\boldsymbol{Y}' - \boldsymbol{Y})$
    $\boldsymbol{u}, \boldsymbol{s}, \boldsymbol{vh} = SVD(((\boldsymbol{Y} - \delta\nabla_1)_{(1)} | (\boldsymbol{Y}' - \delta\nabla_2)_{(2)}))$
    $\boldsymbol{s}' = max(\boldsymbol{s} - \delta\lambda, 0)$, $\boldsymbol{vh} = (\boldsymbol{vh}_1 | \boldsymbol{vh}_2)$
    $\boldsymbol{Y} = unfold_{(1)}(\boldsymbol{us}'\boldsymbol{vh}_1)$
    $\boldsymbol{Y}' = unfold_{(2)}(\boldsymbol{us}'\boldsymbol{vh}_2)$
    $L_{old} = L_{new}$
    $L_{new} = \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{Y}\|_F^2 + \frac{\mu}{2}\|\boldsymbol{Y} - \boldsymbol{Y}'\|_F^2$
        $+\lambda\|(\boldsymbol{Y}_{(1)} | \boldsymbol{Y}'_{(2)})\|_*$
until $(L_{old} - L_{new})/L_{old} < \epsilon$
Return: $\boldsymbol{Y}$
```

$$
\begin{aligned}
F(\boldsymbol{X}) = \operatorname{argmin} & \frac{1}{2}\|(\boldsymbol{X}|_3\boldsymbol{X}^{\top_{12}}) - (\boldsymbol{W}|_3\boldsymbol{W}')\|_F^2 \\
& + \sum_{i=1}^{3} \lambda_i \|(\boldsymbol{W}|_3\boldsymbol{W}')_{(i)}\|_* + \frac{\mu}{2}\|\boldsymbol{W}' - \boldsymbol{W}^{\top_{12}}\|_F^2
\end{aligned} \tag{22}
$$

The μ parameter allows to control of the propagation of similarities between rows and columns, and hence the sharing of embeddings. With $\mu = 0$, the similarities between entities will spread to a distance of 1 in the graph of a relationship, whereas considering $\mu > 0$ will induce a recursion in the propagation of similarities.

Algorithm 1 describes an implementation of SITAR for $\lambda_3 = 0$, which is closer to a convex relaxation of RESCAL.

6 Experiments

We tested SITAR on three benchmarks in the field of learning multi-relational data: *UMLS*, *Nations* and *kinships*. (see [9] for full description of the datasets). *Kinships* depicts kinship relations between members of the Alyawarra tribe in central Australia (26 relations, 104 entities, 10790 observations). *UMLS* is a small part of the semantic network Unified Medical Language System (49 relations, 135 entities, 6752 observations). And *Nations* describes political interactions within countries between 1950 and 1965 (56 relations, 14 entities, 2024 observations).

Table 1 presents the results of SITAR as well as that of the best performing methods of the literature for these datasets: SME [2], CP [7], LFM [8] and RESCAL [11]. As in those previous work, we use the precision-recall AUC score as evaluation metrics and a 10-fold cross-validation scheme.

We compare two versions of our algorithm: SITAR $_{123}$ and SITAR $_{12}$, the latter has $\lambda_3 = 0$. SITAR performs constantly better than RESCAL. On the *kinships* benchmark,

Table 1: PR-AUC scores of SITAR and other standard algorithms.

METHOD	UMLS	NATIONS	KINSHIPS
SME(LIN.)	0.979 ± 0.003	0.777 ± 0.025	0.149 ± 0.003
SME(BIL.)	***0.985*** ± 0.003	0.865 ± 0.015	0.894 ± 0.011
CP	0.95	0.83	0.94
LFM	***0.990*** ± 0.003	***0.909*** ± 0.009	0.946 ± 0.005
RESCAL	0.976 ± 0.003	0.82 ± 0.02	0.952 ± 0.006
SITAR $_{12}$	0.977 ± 0.003	0.838 ± 0.017	***0.964*** ± 0.005
SITAR $_{123}$	0.976 ± 0.003	***0.890*** ± 0.019	***0.969*** ± 0.004
μ	$\mu \sim 2.2$	$\mu \sim 0.1$	$\mu \sim 100$

SITAR reaches the best performance. The SITAR algorithm (as RESCAL) appears to be efficient when a large μ is requested, i.e. when sharing embeddings is important for the discovery of new links.

7 Conclusion

This paper introduced the concept of semantic invariance, a set of properties that can be required from a tensor factorization algorithm seeking to approximate a KB. We described the general form that must have such a semantically invariant algorithm, and proposed a new convex tensor factorization algorithm following this framework.

Acknowledgments

Raphael Bailly is funded through a Google Faculty Research Award. This work was carried out in the framework of the Labex MS2T (ANR-11-IDEX-0004-02), and was funded by the French National Agency for Research (EVEREST-12-JS02-005-01).

References

[1] A. Bordes, X. Glorot, J. Weston, and Y. Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, April 21-23, 2012*, pages 127–135, 2012.

[2] A. Bordes, X. Glorot, J. Weston, and Y. Bengio. A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation. *Machine Learning*, 94(2):233–259, 2014.

[3] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In C. Burges, L. Bottou,

M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. 2013.

[4] G. Bouchard, D. Yin, and S. Guo. Convex collective matrix factorization. In *AISTATS*, volume 31 of *JMLR Proceedings*, pages 144–152. JMLR.org, 2013.

[5] S. Gandy, B. Recht, and I. Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 27(2):025010, 2011.

[6] R. A. Harshman. Models for analysis of asymmetrical relation- ships among n objects or stimuli. In *First Joint Meeting of the Psychometric Society and the Society for Mathematical Psychology*, 1978.

[7] R. A. Harshman and M. E. Lundy. Parafac: Parallel factor analysis. In *Computational Statistics and Data Analysis*, 1994.

[8] R. Jenatton, N. L. Roux, A. Bordes, and G. Obozinski. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 3176–3184, 2012.

[9] C. Kemp, J. Tenenbaum, T. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the National Conference on Artificial Intelligence*, 2006.

[10] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[11] M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, pages 809–816, 2011.

[12] A. Paccanaro and G. E. Hinton. Learning distributed representations of concepts using linear relational embedding. *IEEE Trans. Knowl. Data Eng.*, 13(2):232–244, 2001.

[13] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, 2008.

[14] I. Sutskever, J. B. Tenenbaum, and R. R. Salakhutdinov. Modelling relational data using bayesian clustered tensor factorization. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1821–1828. 2009.

[15] R. Tomioka and T. Suzuki. Convex tensor decomposition via structured schatten norm regularization. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1331–1339. 2013.

[16] R. Tomioka, T. Suzuki, K. Hayashi, and H. Kashima. Statistical performance of convex tensor decomposition. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 972–980, 2011.

[17] L. R. Tucker. Some mathematical notes on three-mode factor analysis. In *Psychometrika*, 1966.