Joint Embedding-Classifier Learning for Interpretable Collaborative Filtering

Clémence Réda, Jill-Jênn Vie, Olaf Wolkenhauer

Joint Embedding–Classifier Learning for Interpretable Collaborative Filtering

Clémence Réda
Department of Systems Biology and Bioinformatics
University of Rostock
G-18051 Rostock, Germany
clemence.reda@uni-rostock.de

Jill-Jênn Vie
Soda
Inria Saclay
F-91120 Palaiseau, France
jill-jenn.vie@inria.fr

Olaf Wolkenhauer
Department of Systems Biology and Bioinformatics, Univ. Rostock, G-18051 Rostock, Germany
Leibniz-Institute for Food Systems Biology, Freising, G-85354, Germany
Stellenbosch Institute of Advanced Study, Wallenberg Research Centre
Stellenbosch, SA-7602, South Africa
olaf.wolkenhauer@uni-rostock.de

Abstract

Interpretability is a topical question in recommender systems, especially in healthcare applications. An interpretable classifier quantifies the importance of each input feature for the predicted item-user association in a non-ambiguous fashion. We introduce the novel Joint Embedding Learning-classifier for improved Interpretability (JELI). By combining the training of a structured collaborative-filtering classifier and an embedding learning task, JELI predicts new user-item associations based on jointly learned item and user embeddings while providing feature-wise importance scores. Therefore, JELI flexibly allows the introduction of priors on the connections between users, items, and features. In particular, JELI simultaneously (a) learns feature, item, and user embeddings; (b) predicts new item-user associations; (c) provides importance scores for each feature. Moreover, JELI instantiates a generic approach to training recommender systems by encoding generic graph-regularization constraints. We show that the joint training approach yields a gain in the predictive power of the downstream classifier, that JELI can recover feature-association dependencies, and induces a restriction in the number of parameters compared to baselines in synthetic and drug-repurposing data sets.

1 Introduction

The Netflix Challenge [6] popularized collaborative filtering, where connections between items and users are inferred based on the guilt-by-association principle and similarities. This approach is particularly suitable for use cases where information about known user-item associations is sparse –typically, close to 99% of all possible user-item associations are not labeled, such as in the MovieLens movie recommendation data set [22]– and when there is implicit feedback. For instance, in the case of movie recommendations on streaming platforms or online advertising, the algorithm often gets only access to clicks, that is, positive feedback. However, the reasons for ignoring an item can be numerous: either the item would straightforwardly receive negative feedback, or the item is too far from the user's usual exploration zone but could still be enjoyed. In some rare cases, true negative feedback might be accessible but in even smaller numbers than the positive associations, for instance, for drug repurposing data sets, by reporting failed Phase III clinical trials [41]. Collaborative filtering algorithms then enable the modeling of the user's behavior based on their similarity to other users and the similarity of the potential recommended item to other items positively graded by this cluster of users.

Several types of algorithms implement collaborative filtering. For instance, matrix factorizations [37, 13] such as Non-negative Matrix Factorization (NMF) [44] or Singular Value Decomposition (SVD) [20], decompose the matrix of item-user associations into a product of two low-rank tensors. Other types of algorithms are (deep) neural networks [23, 5, 26], which build item and user embeddings with convolutional or graph neural networks based on common associations and/or additional feature

Preprint. Under review.

values. On the one hand, among those last approaches, graph-based methods, which integrate and infer edges between features, items, and users, seem promising in performance [34], supported by establishing complex connections between those entities. Conversely, matrix factorizations incorporate explicit interpretability, as one can try to connect the inferred latent factors to specific user and item features. One example is the factorization machine (FM) [36], which combines a linear regression-like term and a feature pairwise interaction term to output a score for binary classification. The learned coefficients of the FM explicitly contribute to the score for each item and user feature set. This type of interpretability, called feature attribution in the literature [29, 38, 47, 18], allows further downstream statistical analysis of the feature interactions. For instance, in our motivating example of drug repurposing, where the objective is to identify novel drug-disease therapeutic associations, if features are genes mutated by the pathology or targeted by the chemical compound, the overrepresented biological pathways among those that are respectively affected or repaired can be retrieved based on the set of key repurposing genes. This, in turn, offers important points to argue in favor of the therapeutic value of a drug-disease indication and for further development towards marketing.

In this work, we aim to combine the performance and versatility (in terms of embeddings) of graph-based collaborative filtering and the explicit interpretability of factorization machines to derive a "best-of-both-worlds" approach for predicting user-item associations. To achieve this, we introduce a special class of factorization machines that leverages a strong hypothesis on the structure of item and user embeddings depending on feature embeddings. This classifier is then jointly trained with a knowledge graph completion task. This knowledge graph connects items, users, and features based on the similarity between them and users and potentially additional priors on their relationships with features. The embeddings used to compute the edge probability scores in the knowledge graph are shared with the factorization machine, which allows the distillation of generic priors into the classifier.

Our paper is structured as follows. In Section 2, we introduce and give an overview of the state-of-the-art on factorization machines and knowledge graphs and how their combination might be able to overcome some topical questions in the field. Section 3 introduces the JELI algorithm, which features our novel class of structured factorization machines and a joint training strategy with a knowledge graph. Eventually, Section 4 shows the performance and interpretability of the JELI approach on both synthetic data sets and drug repurposing applications.

Notation For any matrix M (in capital letters), we denote $M_{i,:}$, $M_{:,j}$ and $M_{i,j}$ respectively its i^{th} line, j^{th} column and coefficient at position (i,j). For any vector $\boldsymbol{v}$ (in bold type), $\boldsymbol{v}_i$ is its i^{th} coefficient. Moreover, $M^\dagger$ is the pseudo-inverse of matrix M.

2 Related work

As previously mentioned, our proposed approach, JELI, leverages a generic knowledge graph completion task and the interpretability of factorization machines to derive a novel, explainable collaborative filtering approach.

2.1 Knowledge graph embedding learning

A knowledge graph is a set of triplets of the form (h, r, t) such that the *head* entity h is linked to the *tail* entity t by the relation r [32]. Entity and relation embeddings learned on the graph allow us to capture the structure and connections in the graph in a numerical form, as embeddings are parameters of a function predicting the presence of a triplet in the graph. Those parameters are then learned based on the current set of edges in the graph. This approach encodes the graph structure into numerical representations, which can later be provided to a downstream regression model [50]. The edge prediction function is usually called the interaction model. Many exist [9, 54, 16, 46], among these, the Multi-Relational Euclidean (MuRE) model [4], defined for any triplet (h, r, t) of respective embeddings $\boldsymbol{e}^h, \boldsymbol{e}^r, \boldsymbol{e}^t$ of dimension d as

$$\text{MuRE}(\boldsymbol{e}^h, \boldsymbol{e}^r, \boldsymbol{e}^t) = -\|R^r \boldsymbol{e}^h - (\boldsymbol{e}^t + \boldsymbol{e}^r)\|_2^2 + b^h + b^t \ ,$$

where $d \times d$ matrix R^r, and scalars b^h and b^t are respectively relation-, head- and tail-specific parameters. Notably, this interaction model has exhibited good embedding engineering properties throughout the literature [1, 52].

Yet, many challenges are present in this field of research. Current representation learning algorithms (no matter the selected interaction model between a triplet and its embedding) infer representations directly on the nodes and relations of the graph. However, this approach does not make it possible to establish a relationship between the nodes other than a similarity at the level of the numerical representation for neighboring nodes for specific relations in the graph. That is, specific logical operations depending on the relation are often ignored: for instance, for a relation r and its opposite $\neg r$, we would like to ensure that the score p assigned to triplet (h, r, t) is proportional to $-\overline{p}$, where $\overline{p}$ is the score associated with triplet $(h, \neg r, t)$. Moreover, knowledge graphs are currently more suited to categorical information, where entities and relationships take discrete values rather than numerical values. Numerical values could describe a relation such as "users from this specific age group are twice more

interested in that movie genre". Some recent works focus on integrating numerical values into knowledge graph embeddings. In KEN embeddings [14], a single-layer neural network is trained for each numeric relation, taking the attribute as input and returning an embedding. Another approach, TransEA [53], aims to optimize a loss function that linearly combines, with a hyperparameter, a loss value on the categorical variables (the difference between the scores and the indicator of the presence of a triplet) and another loss value on numerical variables, which seeks to minimize the gap between the variable and a scalar product involving its embedding. However, these two approaches add several additional hyperparameters and do not deal with interpretability.

Resorting to knowledge-graph-infused embeddings allows us to integrate prior knowledge constraints generically into the representations of entities, both items and users. We aim to enforce a structure on those embeddings to guarantee the good prediction of user-item associations by incorporating those embeddings into a special type of factorization machine.

2.2 Factorization machines

Factorization machines are a type of collaborative filtering algorithms introduced by [36]. Their most common expression, the second-order factorization machine of dimension d, comprises a linear regression term of coefficient (with a possibly nonzero intercept) and a term that combines interactions from all distinct pairs of features by featuring a scalar product of their corresponding low-rank latent vectors of dimension d. This approach, particularly in the presence of sparse feature vectors, is computationally efficient while performant on a variety of recommendation tasks: for instance, knowledge tracing for education [48], click-through rate prediction [21]. Computationally tractable evaluation and training routines were first proposed by [8] for higher-order factorization machines (HOFMs), which were introduced as well in [36] and include interactions from all distinct K sets of features, where $K \geq 2$, opening the way to even finer classification models. The definition of HOFMs is recalled in Definition 1.

Definition 1 Higher–Order Factorization Machines (HOFMs). *Let us denote the set of available item and user features $\mathcal{F} \subseteq \mathbb{N}^*$. The general expression for HOFM [36, 8] of order $m \geq 2$ and dimensions $d_2, \ldots, d_m$ that takes as input a single feature vector $\boldsymbol{x} \in \mathbb{R}^{|\mathcal{F}|}$ is a model such that $\boldsymbol{\theta} = (\omega^0, \boldsymbol{\omega}^1, \boldsymbol{\omega}^2, \ldots, \boldsymbol{\omega}^m)$ where $\omega^0, \boldsymbol{\omega}^1 \in \mathbb{R} \times \mathbb{R}^{|\mathcal{F}|}$ and for any $i \in \{2, \ldots, m\}$, $\boldsymbol{\omega}^i \in \mathbb{R}^{|\mathcal{F}| \times d_i}$*

$$\mathrm{HOFM}_\theta(\boldsymbol{x}) \triangleq \omega^0 + (\boldsymbol{\omega}^1)^\intercal \boldsymbol{x} + \sum_{2 \leq t \leq m} \sum_{\substack{f_1 < \cdots < f_t \\ f_1, \ldots, f_t \in \mathcal{F}}} \langle \boldsymbol{\omega}^t_{f_1,:}, \ldots, \boldsymbol{\omega}^t_{f_t,:} \rangle \boldsymbol{x}_{f_1} \cdot \boldsymbol{x}_{f_2} \cdot \cdots \cdot \boldsymbol{x}_{f_{t-1}} \cdot \boldsymbol{x}_{f_t} \ ,$$

where $\langle \boldsymbol{\omega}^t_{f_1,:}, \ldots, \boldsymbol{\omega}^t_{f_t,:} \rangle \triangleq \sum_{d \leq d_t} \boldsymbol{\omega}^t_{f_1,d} \cdot \boldsymbol{\omega}^t_{f_2,d} \cdot \cdots \cdot \boldsymbol{\omega}^t_{f_{t-1},d} \cdot \boldsymbol{\omega}^t_{f_t,d}$ for any t and indices $f_1, \ldots, f_t$. In particular, for $m = 2$

$$\mathrm{FM}_\theta(\boldsymbol{x}) \triangleq \overbrace{\omega^0 + (\boldsymbol{\omega}^1)^\intercal x}^{\textit{linear regression term}} + \overbrace{\sum_{f < f', f, f' \in \mathcal{F}} \langle \boldsymbol{\omega}^2_{f,:}, \boldsymbol{\omega}^2_{f',:} \rangle \boldsymbol{x}_f \cdot \boldsymbol{x}_{f'}}^{\textit{pairwise interaction term}} \ .$$

Besides their good predictive power, factorization machines involve explicit coefficients that quantify the contribution of each K set of features to the final score associated with the positive class of associations. These coefficients offer a straightforward insight into the discriminating features for the recommendation problem, and this type of "white-box" explainability is related to a larger research field, feature attribution-based interpretability.

2.3 Feature attribution-based interpretability

Given a binary classifier C and a feature vector $\boldsymbol{x} \in \mathbb{R}^F$, a feature attribution function $\phi^C : \mathbb{R}^F \to \mathbb{R}^F$ returns importance scores for each feature contributing to the positive class score for the input vector $\boldsymbol{x}$. If the importance score associated with feature f is largely positive (resp., negative), it means that feature f drives the membership of $\boldsymbol{x}$ to the positive (resp., negative) class. In contrast, an importance score close to 0 indicates that feature f has little influence on the classification of data point $\boldsymbol{x}$. Albeit other types of interpretability approaches exist –based on decision rules given by single classifier trees or random forests [10, 31], counterfactual examples [49] or logic rules [42, 15]) – the importance score-based methods allow going beyond single feature influence, and to quantify the effect on classification of specific mechanisms, for instance by evaluating the overrepresentation of some ontologies among the most strongly-influencing features: for instance, functionally-consistent cell pathways based on the most discriminating biological features.

Some classifiers, as seen for factorization machines, readily include importance scores, whereas several approaches compute post-hoc importance scores. Importance scores are evaluated based on the outputs of an already trained "black-box" classifier, such as a neural network. Such approaches include Shapley values [29], LIME [38], DeepLIFT [27] (for image annotation) or sufficient explanations [3]. Yet, recent works show their lack of robustness and consistency across post-hoc feature attribution

methods, both empirically [47] and theoretically [18, 7]. However, the advantage of posthoc approaches is that they allow the explainability of any type of classifier and combine the richness of the model (predictive performance) and interpretability.

The approach described in our paper then aims to encompass any generic embedding model without losing the connection to the initial features of the input vectors to the classifier.

3 Joint Embedding–classifier Learning for improved Interpretability (JELI)

Let us define in formal terms the inputs to the associated recommendation problem of n_i items $i_1,i_2,\ldots,i_{n_i}$ to n_u users $u_1,u_2,\ldots,u_{n_u}$. The minimal input to the recommendation problem is the user-item association matrix $A \in \{-1,0,+1\}^{n_i \times n_u}$ which summarizes the known positive $(+1)$ –and possibly negative (-1)– associations and denotes unknown associations by zeroes. In simple terms, the recommender systems aim to replace zeroes by ± 1 while preserving the label of nonzero-valued associations. Second, in some cases, we also have access to the respective item and user feature matrices denoted $S \in \mathbb{R}^{F \times n_i}$ and $P \in \mathbb{R}^{F \times n_u}$. Without a loss of generality, we assume that the item and user feature matrices have the same F features $f_1, f_2, \ldots, f_F$. [1] Finally, there might be a partial graph on some of the items, users, features, and possibly other entities. For instance, such a graph might connect movies, users, and human emotions for movie recommendation [11], or drugs, diseases, pathways, and proteins or genes for drug repurposing [56, 12]. We denote this graph $\mathcal{G}(\mathcal{V}_\mathcal{G}, \mathcal{E}_\mathcal{G})$, where $\mathcal{V}_\mathcal{G}$ is the set of nodes in $\mathcal{G}$ and $\mathcal{E}_\mathcal{G}$ is its set of (undirected, labeled) edges.

We first introduce the class of higher-order factorization machines, called redundant structured HOFMs, which will classify user-item associations based on an assumption on the structure of item/user and feature embeddings.

3.1 Redundant structured HOFM (RHOFM)

This subtype of higher-order factorization machines features shared higher-order parameters across interaction orders, such that the corresponding dimensions of the HOFM satisfy $d_2 = \cdots = d_m = d$ in Definition 1. As such, RHOFMs are related to inhomogeneous ANOVA kernel HOFMs (iHOFMs) mentioned in [8]. This type of factorization machine is such that the higher-order dimensions are all equal (that is, $d_2 = \cdots = d_m = d$) and the corresponding higher-order coefficients are all proportionals to one another: for any $t,t' \geq 2$ and $f \leq F$. there exists $c \in \mathbb{R}$ such that $\boldsymbol{\omega}_f^t = c \cdot \boldsymbol{\omega}_f^{t'}$ in Definition 1. However, what distinguishes the RHOFM from a iHOFM is the following hypothesis on structure: it is assumed that every entity d-dimensional embedding $\boldsymbol{e} \in \mathbb{R}^d$ results from some function s_W with parameter $W \in \mathbb{R}^{F \times d}$ applied to the corresponding entity feature vector $\boldsymbol{x} \in \mathbb{R}^F$. For instance, an embedding $\boldsymbol{e}$ associated with feature vector $\boldsymbol{x}$ with a *linear structure function of dimension* d is defined as $\boldsymbol{e} = s_W(\boldsymbol{x}) = \boldsymbol{x}W$. Note that for completeness, we can define a feature vector for *features*, which is simply the result of the indicator function on features in F: for feature $f \in F$, its corresponding feature vector is $\boldsymbol{x}^f \triangleq (\delta_{(f_j=f)})_{j \leq F}$ where δ is the Kronecker symbol, such that the structure function s_W can be applied to any item, user or feature entity. Definition 2 gives the formal expression of RHOFMs for any order, dimension, and structure.

Definition 2 Redundant structured HOFMs (RHOFMs). *The RHOFM of structure s_W, order m and dimension d, with parameters $\boldsymbol{\theta} = (\omega^0, \boldsymbol{\omega}^1, \boldsymbol{\omega}^{2:m}, W) \in \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^{m-1} \times \mathbb{R}^{F \times d}$ on item and user of respective feature vectors $\boldsymbol{x}^i, \boldsymbol{x}^u \in \mathbb{R}^F$ is defined as*

$$\text{RHOFM}_{\boldsymbol{\theta}}(\boldsymbol{x}^i, \boldsymbol{x}^u) \triangleq \omega^0 + (\boldsymbol{\omega}^1)^\intercal (\boldsymbol{x}'^{iu})^\intercal \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix} + \sum_{2 \leq t \leq m} \boldsymbol{\omega}_{t-1}^{2:m} \sum_{\substack{f_1 < \cdots < f_t \\ f_1,\ldots,f_t \leq 2F}} \left\langle \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix}_{f_1,:}, \ldots, \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix}_{f_t,:} \right\rangle \boldsymbol{x}'^{iu}_{f_1} \boldsymbol{x}'^{iu}_{f_2} \ldots \boldsymbol{x}'^{iu}_{f_t},$$

where $\boldsymbol{x}'^{iu} \triangleq [(\boldsymbol{x}^i)^\intercal, (\boldsymbol{x}^u)^\intercal]^\intercal \in \mathbb{R}^{2F}$ is the concatenation of feature vectors along the row dimension, $\widetilde{\boldsymbol{x}}^{iu} \triangleq [\boldsymbol{x}^i, \boldsymbol{x}^u]^\intercal \in \mathbb{R}^{F \times 2}$ the concatenation along the column dimension, $\widetilde{W}_\lambda^{iu} \triangleq (\widetilde{\boldsymbol{x}}^{iu}(\widetilde{\boldsymbol{x}}^{iu})^\intercal + \lambda I_F)^\dagger (\widetilde{\boldsymbol{x}}^{iu})^\intercal [s_W(\boldsymbol{x}^i)^\intercal, s_W(\boldsymbol{x}^u)^\intercal] \in \mathbb{R}^{F \times d}$ is the λ-regularized approximate least squares estimator in the following equation in V: $s_W(\widetilde{\boldsymbol{x}}^{iu}) = \widetilde{\boldsymbol{x}}^{iu} V$, with $\lambda \geq 0$.

By reordering terms and by definition of $\widetilde{W}_\lambda^{iu}$ (full details in Appendix A), if we denote $f\%F$ the remainder of the Euclidean division of f by F, we can notice that

$$\text{RHOFM}_{\boldsymbol{\theta}}(\boldsymbol{x}^i, \boldsymbol{x}^u) \approx \omega^0 + (\boldsymbol{\omega}^1)^\intercal (s_W(\boldsymbol{x}^i) + s_W(\boldsymbol{x}^u)) + \sum_{2 \leq t \leq m} \boldsymbol{\omega}_{t-1}^{2:m} \sum_{\substack{f_1 < \cdots < f_t \\ f_1,\ldots,f_t \leq 2F}} \left\langle \boldsymbol{x}'^{iu}_{f_1} s_W(\boldsymbol{x}^{f_1\%F}), \ldots, \boldsymbol{x}'^{iu}_{f_t} s_W(\boldsymbol{x}^{f_t\%F}) \right\rangle \quad (1)$$

The RHOFM then comprises a term linear in the item/user embeddings and a product of feature embeddings weighted by the corresponding values in the item and user initial feature vectors. Moreover, if we assume a linear structure on the RHOFM, the

[1] Otherwise, one can join the two feature matrices and replace missing feature values by zeroes.

embedding vector for feature f_j is exactly $W_{f_j,:}$ and the embeddings for items and users are the sum of feature embeddings weighted by their corresponding values in the item and user vectors. The expression in Definition 2 is computationally efficient when combined with the routines described in [8], and the redundancy in the RHOFM allows it to benefit from the same type of computational speedup as inhomogeneous ANOVA kernels or iHOFMs.

Knowing that HOFMs (in Definition 1) and iHOFMs would take as input the concatenation along the row dimension of $(\boldsymbol{x}^i, \boldsymbol{x}^u)$, assuming that the dimensions across subsets are the same, *i.e.*, $d_2 = \cdots = d_m = d$, HOFMs comprise $1 + 2F + 2Fd(m-1)$ parameters, which can account for a prohibitive computation cost in practice. Similarly, iHOFMs would require the training of $1 + m + 2Fd$ parameters, whereas RHOFMs (in Definition 2) only feature $1 + m + (F+1)d$, hence removing the multiplicative constant on the number of features F, which has an impact for highly-dimensional data sets such as the TRANSCRIPT drug repurposing data set [40] which gathers values on $12{,}000$ genes across the human genome.

Regarding interpretability, as evidenced by Equation (1), the coefficients involved in the expression of the RHOFM are straightforwardly connected to the input embeddings. In the case of the linear structure and when $\boldsymbol{\omega}^1 = \mathbf{1}_d$, $\boldsymbol{\omega}^{2:m} = \mathbf{1}_{m-1}$ (or any other constant), the contributions from features on the one hand and the item/user values on the other can easily be disentangled. In that case, $\widetilde{W}^{iu}_\lambda \approx W$ and then for any feature f, the intrinsic (*i.e.*, independent from users or items) importance score is $\sum_{k \leq d} W_{f,k}$. When associated with an entity (item or user) of feature vector $\boldsymbol{x} \in \mathbb{R}^F$, its importance score is simply $\boldsymbol{x}_f \sum_{k \leq d} W_{f,k}$. Using $\widetilde{\boldsymbol{x}}^{iu} \widetilde{W}^{iu}_\lambda \approx s_W(\widetilde{\boldsymbol{x}}^{iu})$ in non-linear structures, we can extrapolate this result to obtain the following intrinsic feature importance score

Result 1 Feature importance scores in a RHOFM. *When $\boldsymbol{\omega}^1 = \mathbf{1}_d$, $\boldsymbol{\omega}^{2:m} = \mathbf{1}_{m-1}$ (or any other constant), the intrinsic (entity-independent) feature importance score for feature $f \leq F$ in a RHOFM (Definition 2) is $\sum_{k \leq d} (\widetilde{W}^{iu}_\lambda)_{f,k}$. As a consequence, the feature attribution function associated with feature vector $\boldsymbol{x} \in \mathbb{R}^F$ is $\phi^{\text{RHOFM}}(\boldsymbol{x}) \triangleq (\boldsymbol{x}_f \sum_{k \leq d} (\widetilde{W}^{iu}_\lambda)_{f,k})_{f \leq F}$.*

One could infer the RHOFM parameters by directly minimizing a loss function. However, as mentioned in the introduction, we would like to distill some prior knowledge information into the RHOFM, for instance, via a knowledge graph specific to the recommendation use case. By seeing the feature embeddings in the RHOFM as node embeddings in a knowledge graph, the next section describes how to jointly train the RHOFM and the feature embeddings on a knowledge graph completion task.

3.2 Joint training of the RHOFM and the knowledge graph embeddings

We will leverage the information from the partial graph $\mathcal{G}(\mathcal{V}_\mathcal{G}, \mathcal{E}_\mathcal{G})$ to fit the RHOFM, by reducing the problem of classification to the prediction of a subset of edges in a knowledge graph completion problem. To do so, we first extend the partial graph $\mathcal{G}$ based on the respective user-item association, item feature and user feature matrices A, S and P to build a knowledge graph $\mathcal{K}(\mathcal{V}, \mathcal{T})$ with nine types of relations.

Definition 3 Similarity-based knowledge graph augmented with prior edges. *Considering a similarity threshold $\tau \in [0,1]$ associated with a similarity function $\texttt{sim}\colon \mathbb{R}^F \times \mathbb{R}^F \to [-1,1]$, JELI builds a knowledge graph from the data set A, P and S and partial graph $\mathcal{G}(\mathcal{V}_\mathcal{G}, \mathcal{E}_\mathcal{G})$ as follows*

$$
\begin{aligned}
\mathcal{V} &\triangleq \{i_1, i_2, \ldots, i_{n_i}\} \cup \{u_1, u_2, \ldots, u_{n_u}\} \cup \{f_1, f_2, \ldots, f_F\} \ , \\
\mathcal{T} &\triangleq \{(s, \text{prior}, t) \mid (s,t) \in \mathcal{E}_\mathcal{G}, s, t \in \mathcal{V}\} \\
&\cup \ \{(i_j, -, u_k) \mid A_{i_j,u_k} = -1, j \leq n_i, k \leq n_u\} \cup \{(i_j, +, u_k) \mid A_{i_j,u_k} = +1, \ j \leq n_i, \ k \leq n_u\} \\
&\cup \ \{(u_j, \text{user-sim}, u_k) \mid \texttt{sim}(P_{:,u_j}, P_{:,u_k}) > \tau, \ j,k \leq n_u\} \cup \{(i_j, \text{item-sim}, i_k) \mid \texttt{sim}(S_{:,i_j}, S_{:,i_k}) > \tau, \ j,k \leq n_i\} \\
&\cup \ \{(i_j, \text{item-feat-pos}, f_k) \mid S_{f_k,i_j} > 0, \ k \leq F, \ j \leq n_i\} \cup \{(i_j, \text{item-feat-neg}, f_k) \mid S_{f_k,i_j} < 0, \ k \leq F, \ j \leq n_i\} \\
&\cup \ \{(u_j, \text{user-feat-pos}, f_k) \mid P_{f_k,u_j} > 0, \ k \leq F, \ j \leq n_u\} \cup \{(u_j, \text{user-feat-neg}, f_k) \mid P_{f_k,u_j} < 0, \ k \leq F, \ j \leq n_u\} \ .
\end{aligned}
$$

The objective of knowledge graph completion is to fit a model predictive of the probability of the presence of a triplets in the knowledge graph. In particular, computing the score associated with triplets of the form $(h, +, t)$, for (h, t) a user-item pair, boils down to fitting a classifier of user-item interactions. Conversely, a straightforward assumption is that the score associated with triplets $(h, +, t)$ should be opposite to the score assigned to triplets $(h, -, t)$. With that in mind, denoting the set of RHOFM parameters $\boldsymbol{\theta}$ and $\boldsymbol{\theta}^{\text{JELI}} \triangleq (\boldsymbol{\theta}, \{R_r, \ r \text{ relation}\}, \{\boldsymbol{e}_r, \ r \text{ relation}\}, \{b_h, \ h \in \mathcal{V}\})$ as the total set of parameters to estimate, we define in Equation (2) the edge score to be maximized for present tripletss in the knowledge graph $\mathcal{K}$

$$
\text{score}_{\boldsymbol{\theta}^{\text{JELI}}}(h, r, t) \triangleq \begin{cases} \text{MuRE}(s_W(\boldsymbol{x}^h), \boldsymbol{e}_r, s_W(\boldsymbol{x}^t); R_r, b_h, b_t) & \text{if } r \notin \{+, -\} \\ \text{RHOFM}_{\boldsymbol{\theta}}(\boldsymbol{x}^h, \boldsymbol{x}^t) & \text{if } r = + \\ -\text{RHOFM}_{\boldsymbol{\theta}}(\boldsymbol{x}^h, \boldsymbol{x}^t) & \text{if } r = - \end{cases} . \tag{2}
$$

Remember that the vector $\boldsymbol{x}^h$ is well-defined for any item, user, or feature h. Then we fit parameter $\boldsymbol{\theta}^{\text{JELI}}$ by minimizing the soft margin ranking loss with margin $\lambda^0 = 1$, which expression is recalled below

$$\forall \boldsymbol{\theta}' \ , \ \mathrm{L}^{\text{margin}}(\boldsymbol{\theta}') \triangleq \sum_{(h,r,t)\in\mathcal{T}} \sum_{(\overline{h},r,t)\notin\mathcal{T}} \log\Big(1 + \exp\big(\lambda^0 + \text{score}_{\boldsymbol{\theta}'}(h,r,t) - \text{score}_{\boldsymbol{\theta}'}(\overline{h},r,t)\big)\Big) \ .$$

Further implementation details and numerical considerations for the training pipeline are available in Appendix B.

3.3 Downstream tasks with JELI

Interestingly, not only does JELI build embeddings for items and users available at training time, but it can also be used to produce embeddings for new entities without requiring any retraining step. Given a feature vector $\boldsymbol{x} \in \mathbb{R}^F$ –padding with zeroes if needed on unavailable features– the corresponding embedding is $s_W(\boldsymbol{x})$. However, the main objective of the trained JELI model is to predict new (positive) user-item associations, possibly on items and users not observed at training time. In that case, for any pair of item and user feature vectors $(\boldsymbol{x}^i, \boldsymbol{x}^u) \in \mathbb{R}^F \times \mathbb{R}^F$, the label predicted by JELI with RHOFM parameter $\boldsymbol{\theta}$ is

$$\hat{y}^{\text{JELI}}(\boldsymbol{x}^i, \boldsymbol{x}^u) \triangleq \begin{cases} +1 & \text{if } \sigma(\text{RHOFM}_{\boldsymbol{\theta}}(\boldsymbol{x}^i, \boldsymbol{x}^u)) > 0.5 \\ -1 & \text{otherwise} \end{cases} ,$$

where σ is the standard sigmoid function.

Note that the JELI approach could be even more generic. Besides any knowledge graph, this joint training approach could feature any classifier – not necessarily an RHOFM, as long as the classifier remains interpretable– any knowledge graph completion loss function or any edge score function.

4 Experimental results

We first validate the performance, the interpretability, and the different components of JELI on synthetic data sets, for which the ground truth on feature importance is available. Then, we apply JELI to drug repurposing, our main motivating example for interpretability in recommendation. Further information about the generation of the synthetic data sets and numerical details is available in Appendix C.

4.1 Synthetic data sets

We consider two types of "interpretable" synthetic recommendation data, called "linear first-order" and "linear second-order", for which the ground truth feature importance scores are known. At fixed values of dimension d, feature number F, and numbers of items and users n_i and n_u, both item and user feature vectors are drawn at random from a standard Gaussian distribution, along with a matrix $W^\star \in \mathbb{R}^{F\times d}$. The algorithm cannot access the full feature values in most practical cases in recommendation tasks. Reasons for missing values can be diverse [45], but most likely follow a *not missing at random* mechanism, meaning that the probability of a missing value depends on the features. To implement such a mechanism, we applied a slightly adapted Gaussian self-masking [25] to the corresponding item and user feature matrices, such that we expect around 10% of missing feature values.

The complete set of user-item scores is obtained by a generating model $g_0 : \mathbb{R}^F \times \mathbb{R}^F \to [0,1]$. For "first-order" synthetic data sets, g_0 is defined as $(\boldsymbol{x}^i, \boldsymbol{x}^u) \mapsto \sigma(\sum_{k\leq d}(\boldsymbol{x}^i + \boldsymbol{x}^u)W^\star_{:,k}) = \sigma(\text{RHOFM}_{(0,\mathbf{1}_d,\mathbf{0}_{m-1},W^\star)}(\boldsymbol{x}^i, \boldsymbol{x}^u))$ where x^i and x^u are respectively the item and user feature vectors. For the "second-order" type, g_0 is simply $(\boldsymbol{x}^i, \boldsymbol{x}^u) \mapsto \sigma(\text{RHOFM}_{(1,\mathbf{1}_d,\mathbf{1}_{m-1},W^\star)}(\boldsymbol{x}^i, \boldsymbol{x}^u))$ where the order is $m = 2$. In both cases, the corresponding structure function $s_{W^\star}$ is linear, that is, $s_{W^\star}(\boldsymbol{x}) = \boldsymbol{x}W^\star$ and $\lambda = 0$.

Finally, since in practice, most of the user-item associations are inaccessible at training time, we label user-item pairs with -1, 0, and $+1$ depending on their score, such that the *sparsity number* –that is, the percentage of unknown values in the association matrix– is equal to a prespecified value greater than 50%.

4.1.1 JELI is performant for various validation metrics and reliably retrieves ground truth importance scores

We generate 10 synthetic datasets of each type ($F = 10$, $d = 2$, $n_i = n_u = 173$) and run JELI 100 times with different random seeds corresponding to different training/testing splits. Table 1 shows the numerical results across those 10×100 runs for several validation metrics on the predicted item-user associations and feature importance scores. The Area Under the Curve (AUC) is computed on all user-item pairs. In contrast, the Negative-Sampling AUC (NS-AUC) [55] is a ranking measure akin to an average of user-wise AUCs, giving a more refined quantification of prediction quality across users. The Spearman's rank correlation [43] is computed on ground truth importance scores $(\sum_{k\leq d} W^\star_{f,k})_{f\leq F}$ and predicted ones $(\sum_{k\leq d} \widehat{W}_{f,k})_{f\leq F}$ with $\widehat{W}$ the inferred embedding parameter.

Data set type	AUC	NS-AUC	Spearman's ρ
First-order	0.99 ± 0.013	0.89 ± 0.124	0.83 ± 0.279
Second-order	0.98 ± 0.019	0.86 ± 0.167	0.75 ± 0.363

Table 1: Average validation metrics with standard deviations across 100 iterations and 10 synthetic data sets of each type (total number of values: 1,000). Average (respectively, standard deviation) values are rounded to the closest second (resp., third) decimal place. AUC: Area Under the Curve. NS-AUC: Negative-Sampling AUC [55]. Spearman's ρ: Spearman's rank correlation.

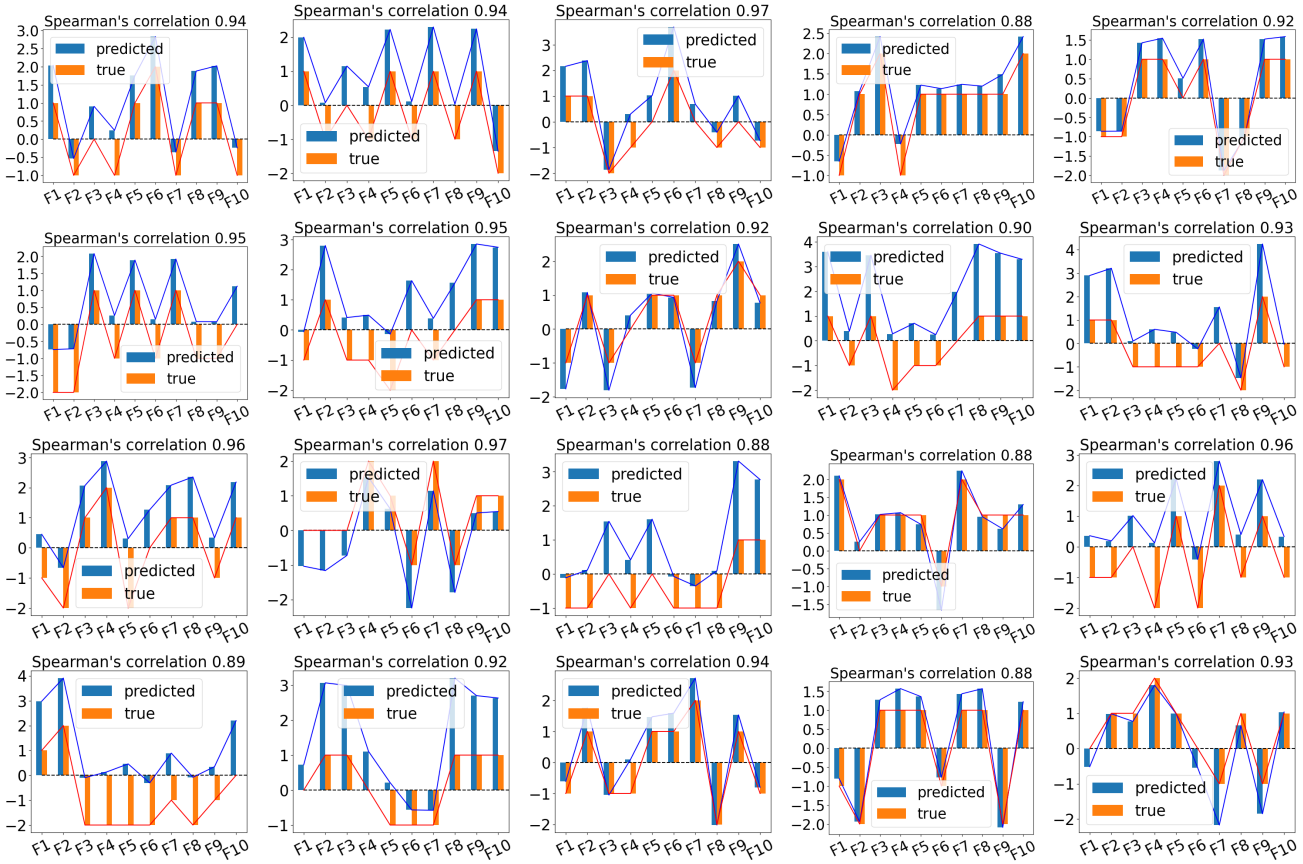

Figure 1: Barplots of the true and predicted feature importance scores for $F = 10$ features in each synthetic data set for the best-performing model across 100 iterations. Top-2 lines: on "first-order" synthetic data. Bottom-2 lines: on "second-order" synthetic data.

Albeit there is a large variation in the quality of the prediction due to the random training/testing split when considering the average best value across 100 iterations, the metrics in Table 1 show a high predictive power for JELI, along with a consistently high correlation between true and predicted feature importance scores: the average Spearman's rank correlation for the best-trained models across all 10 data sets is 0.932 for "first-order" sets and 0.932 for "second-order" ones. The bar plots representing the ground truth and predicted importance scores for each of these 10 sets and each type of synthetic data in Figure 1 show that JELI can preserve the global trend in importance scores across data sets.

4.1.2 JELI is robust in synthetic data sets across sparsity numbers

We also compare the predictive performance of JELI compared to embedding-based recommender systems from the state-of-the-art, namely Fast.ai collaborative learner [23], the heterogeneous attention network (HAN) algorithm [51] and the neural inductive matrix completion with graph convolutional network (NIMCGCN) [26]. We set, whenever appropriate, the same hyperparameter values for all algorithms (with $d = 2$). We run each algorithm on 100 different random seeds on 5 "first-order" synthetic data sets generated with sparsity numbers in $\{50\%, 65\%, 80\%\}$, for 500 tests. Figure 2 reports the boxplots and the

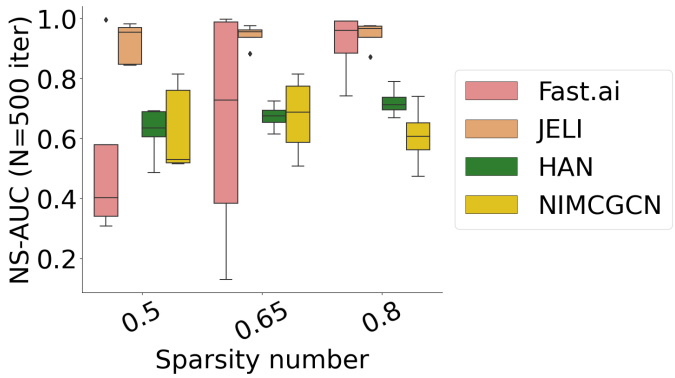

Figure 2: NS-AUC values across "first-order" synthetic data sets for sparsity numbers and 500 iterations for JELI and state-of-the-art embedding-based recommender systems.

		AUC	NS-AUC	NDCG
50%	Fast.ai	0.99± 0.0	0.52± 0.3	0.85± 0.1
	HAN	0.93± 0.0	0.62± 0.1	0.18± 0.1
	NIM	0.93± 0.0	0.63± 0.1	0.39± 0.1
	JELI	0.99± 0.0	0.92± 0.1	0.96± 0.1
65%	Fast.ai	0.99± 0.0	0.64± 0.4	0.78± 0.3
	HAN	0.93± 0.0	0.67± 0.0	0.12± 0.1
	NIM	0.94± 0.0	0.67± 0.1	0.42± 0.1
	JELI	0.99± 0.0	0.94± 0.0	0.94± 0.1
80%	Fast.ai	0.99± 0.0	0.91± 0.1	0.77± 0.2
	HAN	0.96± 0.0	0.72± 0.0	0.20± 0.1
	NIM	0.93± 0.0	0.61± 0.1	0.19± 0.0
	JELI	0.99± 0.0	0.94± 0.0	0.85± 0.2

Table 2: Average metrics with standard deviations across 100 iterations and 5 "first-order" sets. The NDCG at rank n_i is averaged across users. NIM is NIMCGCN.

confidence intervals on corresponding validation metrics. In addition to the AUC and NS-AUC, we include the Non-Discounted Cumulative Gain (NDCG) computed for each user at rank n_i (number of items) and averaged across users as a counterpart to the NS-AUC measure.

As illustrated by Figure 2, JELI consistently outperforms the state-of-the-art on all metrics and remains robust to the sparsity number.

4.1.3 Ablation study: both the structure and the joint learning are crucial to the performance

We perform the same type of experiments as in Section 4.1.2 on several ablated versions of JELI to estimate the contribution of each part to the predictive performance. We introduce several JELI variants. First, we remove the structured and embedding part of the RHOFM classifier. FM is the regular second-order factorization machine of dimension d on $2F$-dimensional input vectors, without structure on the coefficients (see Definition 1), whereas CrossFM2 is a more refined non-structured second-order factorization machine, where the feature pairwise interaction terms only comprise pairs of features on both the item and user vectors, that is, with notation from Definition 1

$$\text{CrossFM}_{(\omega^0,\boldsymbol{\omega}^1,\boldsymbol{\omega}^2)}(\boldsymbol{x}^i,\boldsymbol{x}^u) \triangleq \omega^0 + (\boldsymbol{\omega}^1)^\intercal \begin{bmatrix}\boldsymbol{x}^i\\ \boldsymbol{x}^u\end{bmatrix} + \sum_{f\leq F, f'>F} \langle \boldsymbol{\omega}^2_f, \boldsymbol{\omega}^2_{f'}\rangle \boldsymbol{x}^i_f \boldsymbol{x}^u_{f'-F} \ .$$

Next, we also study methods featuring separate learning of the embeddings and the RHOFM classifier, named Separate Embedding Learning and Training algorithms (SELT). We consider different feature embedding types. SELT-PCAf uses the d principal component analysis (PCA) run on the concatenation of the item and user matrices along the column dimension, resulting in a $F \times (n_i + n_u)$ matrix. SELT-PCAf then infers feature embeddings based on each feature's d first principal components. Another PCA-based baseline, SELT-PCAiu, applies the learned PCA transformation directly on item and user feature vectors to obtain item and user embeddings. Finally, the SELT-KGE approach completes the knowledge graph task to obtain item and user embeddings –without enforcing the feature-dependent structure– on the knowledge graph described in Definition 3 with an empty partial graph. Then, SELT-KGE uses those item and user embeddings to train the RHOFM classifier.

The final results in Figure 3 show that the most crucial part for predictive performance across sparsity numbers is the factorization machine, which is unsurprising given the literature on factorization machines applied to sparse data. One can observe that separate embedding learning and factorization machine training leads to mediocre performance. The combination of a structured factorization machine and jointly learned embeddings, that is, JELI, gives the best performance and is even more significant as the set of known associations gets smaller (and the sparsity number is larger).

4.2 Application to drug repurposing

We aim to predict new therapeutic indications, that is, novel associations between chemical compounds and diseases. The interpretability of the model for predicting associations between molecules and pathologies is crucial to encourage its use for health. In that case, higher-order factorization machines are very interesting models due to their inherent interpretability. However, particularly for the most recent drug repurposing datasets (*e.g.*, TRANSCRIPT [40] and PREDICT [39]), the number

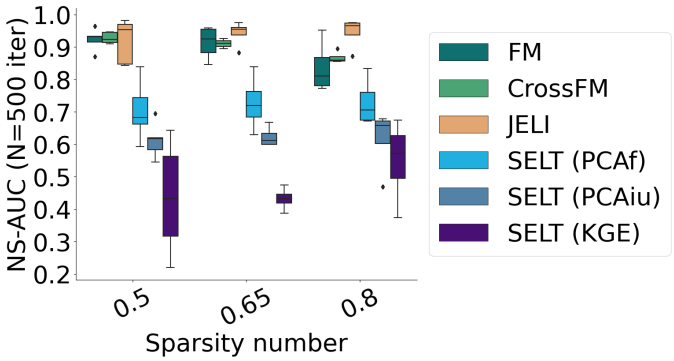

Figure 3: NS-AUC values across "first-order" synthetic data sets for sparsity numbers and 500 iterations for JELI and ablated variants.

		AUC	NS-AUC	NDCG
50%	FM2	0.99± 0.0	0.92± 0.0	0.97± 0.0
	CrossFM2	0.99± 0.0	0.93± 0.0	1.00± 0.0
	S-PCAf	0.95± 0.0	0.70± 0.1	0.58± 0.2
	S-PCAiu	0.95± 0.0	0.61± 0.2	0.45± 0.2
	S-KGE	0.91± 0.0	0.43± 0.2	0.25± 0.2
	JELI	0.99± 0.0	0.92± 0.1	0.96± 0.0
65%	FM2	0.98± 0.0	0.91± 0.0	0.87± 0.1
	CrossFM2	0.99± 0.0	0.91± 0.0	0.95± 0.0
	S-PCAf	0.95± 0.0	0.73± 0.1	0.54± 0.2
	S-PCAiu	0.94± 0.0	0.62± 0.0	0.34± 0.1
	S-KGE	0.90± 0.0	0.43± 0.0	0.06± 0.0
	JELI	0.99± 0.0	0.94± 0.0	0.94± 0.1
80%	FM2	0.97± 0.0	0.84± 0.1	0.56± 0.1
	CrossFM2	0.98± 0.0	0.87± 0.0	0.74± 0.0
	S-PCAf	0.95± 0.0	0.73± 0.1	0.38± 0.1
	S-PCAiu	0.93± 0.0	0.62± 0.1	0.20± 0.0
	S-KGE	0.91± 0.0	0.55± 0.1	0.12± 0.1
	JELI	0.99± 0.0	0.94± 0.0	0.85± 0.2

Table 3: Average metrics with standard deviations across 100 iterations and 5 "first-order" sets. The NDCG at rank n_i is averaged across users. S indicates an instance of SELT.

of features ($F \approx 12{,}000$ and $F \approx 6{,}000$, respectively) is too large to effectively train a factorization machine due to the curse of dimensionality. Resorting to knowledge graphs then enables the construction of low-dimensional vector representations of these associations. Then, these representations are fed as input to the classifier during training instead of the initial feature vectors.

4.2.1 JELI is on par with state-of-the-art approaches on drug repurposing data sets

We now run JELI and the baseline algorithms tested in Section 4.1.2 on Gottlieb [30] (named Fdataset in the paper), LRSSL [28], PREDICT-Gottlieb [19] and TRANSCRIPT [40] drug repurposing data sets which feature a variety of data types and sizes. Please refer to Appendix C for more information. Figure 4 reports the validation metrics for each method's 100 different training/testing splits with $d = 15$. From those results, we can see that the performance of JELI is on par with the top algorithm, HAN, and sometimes outperforms it, while providing interpretability.

4.2.2 JELI can integrate any graph prior on the TRANSCRIPT data set

We now focus on the TRANSCRIPT data set, which involves gene activity measurements across $F = 12{,}096$ genes for $n_i = 204$ drugs and $n_u = 116$ diseases. We compare the predictive power of JELI on the TRANSCRIPT data set with the default knowledge graph created by JELI (named "Sim" network) and the knowledge graph augmented with a protein-protein interaction network, that we call "Sim+PPI" network. The "Sim" network corresponds to the knowledge graph in Definition 3 with an empty partial graph, whereas the "Sim+PPI" graph is built based on the partial graph listing protein-protein interactions (where proteins are matched one-to-one to their corresponding coding genes). The resulting performance for the AUC, average NDGC@n_i and NS-AUC metrics is shown on Figure 5. Adding biologically meaningful prior information to JELI yields a statistically significant improvement in all validation metrics except the NDCG.

5 Discussion

We propose in this work the JELI approach for integrating knowledge graph-based regularization into an interpretable recommender system. The structure incorporated into user and item embeddings take into account numerical feature values in a generic fashion, which allows to go beyond the categorical relations encoded in knowledge graphs without adding a large number of parameters. This method allows us to derive item and user representations of fixed dimensions and score a user-item association, even on previously unseen items and users. We have shown the performance and the explainability power of JELI on synthetic and real-life data sets. The Python package that implements the JELI approach is available at the following open-source

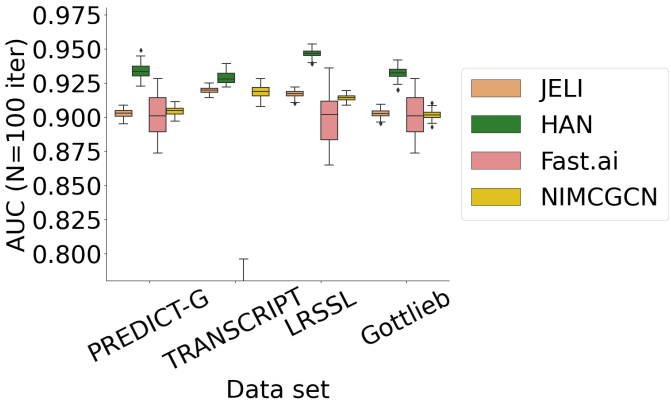

Figure 4: AUC values across drug repurposing data sets for sparsity numbers and 100 iterations for JELI and state-of-the-art embedding-based approaches.

		AUC	NS-AUC	NDCG
Gottlieb	Fast.ai	0.90± 0.0	0.50± 0.1	0.01± 0.0
	HAN	0.93± 0.0	0.67± 0.0	0.02± 0.0
	NIM	0.90± 0.0	0.51± 0.0	0.01± 0.0
	JELI	0.90± 0.0	0.52± 0.0	0.02± 0.0
LRSSL	Fast.ai	0.90± 0.0	0.49± 0.1	0.01± 0.0
	HAN	0.95± 0.0	0.69± 0.0	0.10± 0.0
	NIM	0.91± 0.0	0.53± 0.0	0.01± 0.0
	JELI	0.92± 0.0	0.51± 0.0	0.02± 0.0
PRED-G	Fast.ai	0.90± 0.0	0.50± 0.1	0.01± 0.0
	HAN	0.93± 0.0	0.68± 0.0	0.01± 0.0
	NIM	0.91± 0.0	0.49± 0.0	0.01± 0.0
	JELI	0.90± 0.0	0.47± 0.0	0.02± 0.0
TRANSC	Fast.ai	0.61± 0.1	0.57± 0.1	0.04± 0.0
	HAN	0.93± 0.0	0.61± 0.0	0.08± 0.0
	NIM	0.92± 0.0	0.57± 0.0	0.04± 0.0
	JELI	0.92± 0.0	0.56± 0.0	0.02± 0.0

Table 4: Average metrics with standard deviations across 100 iterations for each drug repurposing data set. The NDCG at rank n_i is averaged across users. NIM is the algorithm NIMCGCN, TRANSC refers to the data set TRANSCRIPT, and PRED-G to the data set PREDICT-Gottlieb.

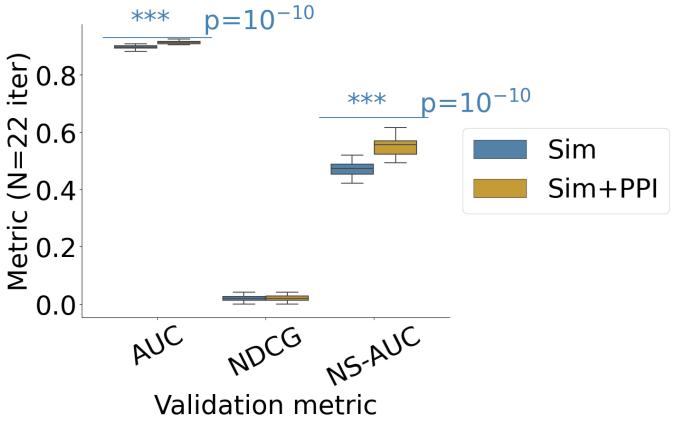

Figure 5: Predictive performance of JELI with different graph priors (the default knowledge graph "Sim" and the protein-protein interaction augmented network "Sim+PPI"). Statistical tests are one-way ANOVA tests with significance level $\alpha = 1\%$.

repository: `github.com/RECeSS-EU-Project/JELI/`. Experimental results can be reproduced using code which will be made available soon in a future repository.

There are a few limitations to the JELI approach, however. The first one is that JELI performs best on sparse user and item feature matrices and requires the input of the dimension value, which can tremendously impact the algorithm's performance. Moreover, this approach is quite slow compared to state-of-the-art algorithms since it simultaneously solves two tasks: the recommendation one on user-item pairs and the knowledge graph completion. However, this slowness is mitigated by the superior interpretability of JELI compared to the baselines. Furthermore, an interesting subsequent work would focus on integrating missing values into the recommendation problem. As it is, JELI ignores the missing features and potentially recovers qualitative item-feature –respectively, user-feature– links during the knowledge graph completion tasks. That is, provided an approach to quantify the strength of the link between an item and a feature, JELI might also be extended to perform an imputation of this item's corresponding missing feature value.

Acknowledgments and Disclosure of Funding

Competing interests The authors declare no competing interests.

Acknowledgments Part of this work was completed during a secondment of C.R. at Soda Team, Inria Saclay, F-91120 Palaiseau, France. The research leading to these results has received funding from the European Union's HORIZON 2020 Programme under grant agreement no. 101102016 (RECeSS, HORIZON TMA MSCA Postdoctoral Fellowships - European Fellowships, C.R.). The funding sources have played no role in the design, the execution nor the analyses performed in this study. The authors thank Félix Lefebvre for fruitful discussions about knowledge graphs.

Authors' contributions **C.R.**: Conceptualization, writing, implementation – original draft. **J.-J. V., O. W.**: Substantial edition of the manuscript.

References

[1] Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Mikhail Galkin, Sahand Sharifzadeh, Asja Fischer, Volker Tresp, and Jens Lehmann. Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):8825–8845, 2021.

[2] Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. Pykeen 1.0: a Python library for training and evaluating knowledge graph embeddings. *Journal of Machine Learning Research*, 22(82):1–6, 2021.

[3] Salim I Amoukou and Nicolas JB Brunel. Consistent sufficient explanations and minimal local rules for explaining regression and classification models. *arXiv preprint arXiv:2111.04658*, 2021.

[4] Ivana Balazevic, Carl Allen, and Timothy Hospedales. Multi-relational poincaré graph embeddings. *Advances in Neural Information Processing Systems*, 32, 2019.

[5] Zeynep Batmaz, Ali Yurekli, Alper Bilge, and Cihan Kaleli. A review on deep learning for recommender systems: challenges and remedies. *Artificial Intelligence Review*, 52:1–37, 2019.

[6] Robert M Bell, Yehuda Koren, and Chris Volinsky. All together now: A perspective on the Netflix prize. *Chance*, 23(1):24–29, 2010.

[7] Blair Bilodeau, Natasha Jaques, Pang Wei Koh, and Been Kim. Impossibility theorems for feature attribution. *Proceedings of the National Academy of Sciences*, 121(2):e2304406120, 2024.

[8] Mathieu Blondel, Akinori Fujino, Naonori Ueda, and Masakazu Ishihata. Higher-order factorization machines. *Advances in Neural Information Processing Systems*, 29, 2016.

[9] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[10] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[11] Arno Breitfuss, Karen Errou, Anelia Kurteva, and Anna Fensel. Representing emotions with knowledge graphs for movie recommendations. *Future Generation Computer Systems*, 125:715–725, 2021.

[12] Payal Chandak, Kexin Huang, and Marinka Zitnik. Building a knowledge graph to enable precision medicine. *Scientific Data*, 10(1):67, 2023.

[13] Wei-Sheng Chin, Bo-Wen Yuan, Meng-Yuan Yang, Yong Zhuang, Yu-Chin Juan, and Chih-Jen Lin. Libmf: a library for parallel matrix factorization in shared-memory systems. *Journal of Machine Learning Research*, 17(86):1–5, 2016.

[14] Alexis Cvetkov-Iliev, Alexandre Allauzen, and Gaël Varoquaux. Relational data embeddings for feature enrichment with background information. *Machine Learning*, 112(2):687–720, 2023.

[15] Adnan Darwiche and Auguste Hirth. On the reasons behind decisions. *arXiv preprint arXiv:2002.09284*, 2020.

[16] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[17] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

[18] Hidde Fokkema, Rianne de Heide, and Tim van Erven. Attribution-based explanations that provide recourse cannot be robust. *Journal of Machine Learning Research*, 24(360):1–37, 2023.

[19] Chu-Qiao Gao, Yuan-Ke Zhou, Xiao-Hong Xin, Hui Min, and Pu-Feng Du. Dda-skf: predicting drug–disease associations using similarity kernel fusion. *Frontiers in Pharmacology*, 12:784171, 2022.

[20] Gene Golub and William Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2(2):205–224, 1965.

[21] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017.

[22] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *ACM Transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[23] Jeremy Howard et al. Fast.ai. `https://github.com/fastai/fastai`, 2018.

[24] Sagar Imambi, Kolla Bhanu Prakash, and GR Kanagachidambaresan. Pytorch. *Programming with TensorFlow: Solution for Edge Computing Applications*, pages 87–104, 2021.

[25] Marine Le Morvan, Julie Josse, Thomas Moreau, Erwan Scornet, and Gaël Varoquaux. Neumiss networks: differentiable programming for supervised learning with missing values. *Advances in Neural Information Processing Systems*, 33:5980–5990, 2020.

[26] Jin Li, Sai Zhang, Tao Liu, Chenxi Ning, Zhuoxuan Zhang, and Wei Zhou. Neural inductive matrix completion with graph convolutional networks for mirna-disease association prediction. *Bioinformatics*, 36(8):2538–2546, 2020.

[27] Junbing Li, Changqing Zhang, Joey Tianyi Zhou, Huazhu Fu, Shuyin Xia, and Qinghua Hu. Deep-lift: Deep label-specific feature learning for image annotation. *IEEE Transactions on Cybernetics*, 52(8):7732–7741, 2021.

[28] Xujun Liang, Pengfei Zhang, Lu Yan, Ying Fu, Fang Peng, Lingzhi Qu, Meiying Shao, Yongheng Chen, and Zhuchu Chen. Lrssl: predict and interpret drug–disease associations based on data integration using sparse subspace learning. *Bioinformatics*, 33(8):1187–1196, 2017.

[29] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[30] Huimin Luo, Jianxin Wang, Min Li, Junwei Luo, Xiaoqing Peng, Fang-Xiang Wu, and Yi Pan. Drug repositioning based on comprehensive similarity measures and bi-random walk algorithm. *Bioinformatics*, 32(17):2664–2671, 2016.

[31] Sascha Marton, Stefan Lüdtke, Christian Bartelt, and Heiner Stuckenschmidt. Gradtree: Learning axis-aligned decision trees with gradient descent. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 14323–14331, 2024.

[32] Maximilian Nickel, Volker Tresp, Hans-Peter Kriegel, et al. A three-way model for collective learning on multi-relational data. In *ICML*, volume 11, pages 3104482–3104584, 2011.

[33] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[34] Clémence Réda, Jill-Jênn Vie, and Olaf Wolkenhauer. Comprehensive evaluation of collaborative filtering in drug repurposing. *HAL preprint HAL-04626970*, 2024.

[35] Clémence Réda, Jill-Jênn Vie, and Olaf Wolkenhauer. stanscofi and benchscofi: a new standard for drug repurposing by collaborative filtering. *Journal of Open Source Software*, 9(93):5973, 2024.

[36] Steffen Rendle. Factorization machines. In *2010 IEEE International conference on data mining*, pages 995–1000. IEEE, 2010.

[37] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

[38] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[39] Clémence Réda. Predict drug repurposing dataset. doi: 10.5281/zenodo.7983090.

[40] Clémence Réda. Transcript drug repurposing dataset. doi: 10.5281/zenodo.7982976.

[41] Clémence Réda, Jill-Jênn Vie, and Olaf Wolkenhauer. stanscofi and benchscofi: a new standard for drug repurposing by collaborative filtering. *Journal of Open Source Software*, 9(93):5973, 2024.

[42] Andy Shih, Arthur Choi, and Adnan Darwiche. A symbolic approach to explaining bayesian network classifiers. *arXiv preprint arXiv:1805.03364*, 2018.

[43] Charles Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 100(3/4):441–471, 1987.

[44] Suvrit Sra and Inderjit Dhillon. Generalized nonnegative matrix approximations with bregman divergences. *Advances in neural information processing systems*, 18, 2005.

[45] EW Steyerberg and EW Steyerberg. Dealing with missing values. *Clinical Prediction Models: A Practical Approach to Development, Validation, and Updating*, pages 115–137, 2009.

[46] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[47] Vinitra Swamy, Bahar Radmehr, Natasa Krco, Mirko Marras, and Tanja Käser. Evaluating the explainers: black-box explainable machine learning for student success prediction in moocs. *arXiv preprint arXiv:2207.00551*, 2022.

[48] Jill-Jênn Vie and Hisashi Kashima. Knowledge tracing machines: Factorization machines for knowledge tracing. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 750–757, 2019.

[49] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31:841, 2017.

[50] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on knowledge and data engineering*, 29(12):2724–2743, 2017.

[51] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032, 2019.

[52] Junkang Wu, Wentao Shi, Xuezhi Cao, Jiawei Chen, Wenqiang Lei, Fuzheng Zhang, Wei Wu, and Xiangnan He. Disenkgat: knowledge graph embedding with disentangled graph attention network. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 2140–2149, 2021.

[53] Yanrong Wu and Zhichun Wang. Knowledge graph embedding with numeric attributes of entities. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 132–136, 2018.

[54] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[55] Hsiang-Fu Yu, Mikhail Bilenko, and Chih-Jen Lin. Selection of negative samples for one-class matrix factorization. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 363–371. SIAM, 2017.

[56] Shuangjia Zheng, Jiahua Rao, Ying Song, Jixian Zhang, Xianglu Xiao, Evandro Fei Fang, Yuedong Yang, and Zhangming Niu. Pharmkg: a dedicated knowledge graph benchmark for bomedical data mining. *Briefings in bioinformatics*, 22(4):bbaa344, 2021.

A Explicit structure-dependent approximation of a RHOFM

Starting from the notation and the expression of an RHOFM introduced by Definition 2

$$\text{RHOFM}_{\boldsymbol{\theta}}(\boldsymbol{x}^i, \boldsymbol{x}^u) \triangleq \omega^0 + (\boldsymbol{\omega}^1)^\intercal (\boldsymbol{x}'^{iu})^\intercal \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix} + \sum_{2 \leq t \leq m} \boldsymbol{\omega}_{t-1}^{2:m} \sum_{\substack{f_1 < \cdots < f_t \\ f_1, \ldots, f_t \leq 2F}} \left\langle \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix}_{f_1,:}, \ldots, \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix}_{f_t,:} \right\rangle \boldsymbol{x}'^{iu}_{f_1} \boldsymbol{x}'^{iu}_{f_2} \ldots \boldsymbol{x}'^{iu}_{f_t} .$$

Given the definition of $\widetilde{W}_\lambda^{iu}$, it is easy to see that

$$(\boldsymbol{x}'^{iu})^\intercal \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix} = (\boldsymbol{x}^i)^\intercal \widetilde{W}_\lambda^{iu} + (\boldsymbol{x}^u)^\intercal \widetilde{W}_\lambda^{iu} \approx s_W(\boldsymbol{x}^i) + s_W(\boldsymbol{x}^u) .$$

Let us consider now the t-interaction term, for $t \geq 2$. For any set of t features $f_1, f_2, \ldots, f_t$, using the notation $\boldsymbol{x}^f \triangleq (\delta_{f_j = f})_{j \leq |F|}$ and $f \% F$ as the remainder of the Euclidean division of f by F

$$\begin{aligned}
\left\langle \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix}_{f_1,:}, \ldots, \begin{bmatrix} \widetilde{W}_\lambda^{iu} \\ \widetilde{W}_\lambda^{iu} \end{bmatrix}_{f_t,:} \right\rangle \boldsymbol{x}'^{iu}_{f_1} \boldsymbol{x}'^{iu}_{f_2} \ldots \boldsymbol{x}'^{iu}_{f_t} &= \sum_{k \leq d} \left(\Pi_{l \leq t} \, (\widetilde{W}_\lambda^{iu})_{f_l \% F, k} \right) \left(\Pi_{j \leq t} \, \boldsymbol{x}'^{iu}_{f_j} \right) \\
&= \sum_{k \leq d} \left(\Pi_{l \leq t} \, \boldsymbol{x}^{f_l \% F} (\widetilde{W}_\lambda^{iu})_{:,k} \right) \left(\Pi_{j \leq t} \, \boldsymbol{x}'^{iu}_{f_j} \right) \\
&\approx \sum_{k \leq d} \left(\Pi_{l \leq t} \, s_W(\boldsymbol{x}^{f_l \% F})_k \right) \left(\Pi_{j \leq t} \, \boldsymbol{x}'^{iu}_{f_j} \right) \\
&= \sum_{k \leq d} \Pi_{j \leq t} \, \boldsymbol{x}'^{iu}_{f_j} s_W(\boldsymbol{x}^{f_j \% F})_k \\
&= \left\langle \boldsymbol{x}'^{iu}_{f_1} s_W(\boldsymbol{x}^{f_1 \% F}), \ldots, \boldsymbol{x}'^{iu}_{f_t} s_W(\boldsymbol{x}^{f_t \% F}) \right\rangle .
\end{aligned}$$

This leads to Equation (1) in the main text.

B Implementation of the joint training procedure in JELI

The training procedure iteratively updates across epochs and batches of triplets the feature embeddings $W \in \mathbb{R}^{F \times d}$, the MuRE-specific hyperparameters $R_r \in \mathbb{R}^{d \times d}$ for each relation r (9 in total by Definition 3), the biases $b \in \mathbb{R}^{|\mathcal{V}|}$, and the hyperparameters of the RHOFM $(\omega^0, \boldsymbol{\omega}^1, \boldsymbol{\omega}^{2:m}) \in \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^{m-1}$, for a total of $(9d^2 + |\mathcal{V}|) + (1 + m + 2Fd) = d(9d + 2F) + |\mathcal{V}| + m + 1$ parameter values. In practice, we implement this procedure using the PyKeen Python package [2], an Adam optimizer and the PseudoTypedNegativeSampler class in PyKeen for the negative sampling to switch the head of triplets and then compute the soft margin ranking loss L^{margin}. The MuRE interaction class in PyKeen is modified to allow the computation of structured embeddings for items and users in the score.

Before the training phase, we normalize the item and user feature matrices to cope with heterogeneous feature values. We replace missing values with zeroes, quantile normalize each feature and then normalize to $[-1,1]$ (with the function normalize$(\cdot, \text{norm} = \ell_1)$ from the Python package scikit-learn [33]).

We also force sparsity in feature values by adding a supplementary preprocessing layer which removes all "weak-signal" normalized values v such that $|v| < t \in (0,1)$ (thresholding with value $t = 0.001$) or such that

$$\inf_{v'} \left\{ \text{freq}(v') \leq \frac{q}{2} \right\} < v < \inf_{v'} \left\{ \text{freq}(v') \geq \frac{1-q}{2} \right\} , \; q \in (0,1) .$$

We use the latter method throughout the experimental study, with $q = 0.9$. Note that those two approaches are equivalent for normally distributed frequencies of values.

Feature type	Data set	Source	n_i	$\lvert\mathcal{F}_i\rvert$	n_u	$\lvert\mathcal{F}_u\rvert$	Nb. positive	Nb. negative	sparsity (%)
Text-mining	Gottlieb	[30]	593	593	313	313	1,933	0	99.0
Biological	PREDICT-Gottlieb	[19]	593	1,779	313	313	1,933	0	99.0
	LRSSL	[28]	763	2,049	681	681	3,051	0	99.4
	TRANSCRIPT	[40]	204	12,096	116	12,096	401	11	98.3

Table 5: Overview of the drug repurposing data sets in the experimental study in Section 4, with the number of items (drugs), item features, users (diseases), user features, positive and negative associations along with the corresponding sparsity number. Text-mining data sets involve similarity scores on chemical structures (for drugs) and on medical descriptions (diseases) as features. In contrast, biological data sets incorporate similarity scores on drug and disease annotations (PREDICT-Gottlieb, LRSSL) and gene activity data (TRANSCRIPT).

C Experimental details

C.1 Drug repurposing data sets

The drug repurposing data sets were retrieved using the stanscofi Python package [35]. Table 5 shows their size and an overview of their contents. When items and users did not use the same set of features $\mathcal{F}_i$ and $\mathcal{F}_u$, we considered the disjoint union of the item and user feature sets $\mathcal{F}_i \cup \mathcal{F}_u$ by padding with zeroes whenever a feature was missing.

C.2 Synthetic dataset

In both considered types of synthetic data sets, as described in the main text at Section 4, we first draw at random the item and user feature matrices $\widetilde{S}$ and $\widetilde{P}$ and feature embeddings $W^\star$ from a standard Gaussian distribution and set a fixed generating model g_0 for computing ground truth association scores based on item and user feature vectors. We now lay out how the sparsity in feature values (Subsection C.2.1) and in associations (Subsection C.2.2) is implemented.

C.2.1 Adapted Gaussian self-masking procedure

As mentioned in the main text, we implemented a slightly modified version of the Gaussian self-masking procedure introduced in [25, Assumption 4] to generate not missing at random values. For each entity (item or user) $j \leq n$, where $n \in \{n_i, n_u\}$ the number of entities, we denote $M_{j,f}$ the binary value which indicates whether the feature value $\boldsymbol{x}_f^j$ is missing. For fixed feature-specific coefficients $K_f \in (0,1)$ for any feature $f \leq F$, we then recall that that the Gaussian self-masking mechanism is defined as

$$\mathbb{P}(M_{j,1},...,M_{j,F} \mid (\boldsymbol{x}^j)_{j\leq n}) = \Pi_{f=1}^F K_f \exp\left(-\frac{1}{2}\frac{(\boldsymbol{x}_f^j - \widetilde{\mu}_f)^2}{\widetilde{\sigma}_f^2}\right) \text{ where } \widetilde{\mu}_f \triangleq \frac{1}{n}\sum_{l\leq n}\boldsymbol{x}_f^l \text{ and } \widetilde{\sigma}_f^2 \triangleq \frac{1}{n-1}\sum_{l\leq n}(\boldsymbol{x}_f^l - \widetilde{\mu}_f)^2 \ .$$

We want to ensure that the sparsity (*i.e.*, the percentage of feature values set to zero) is at most at 10%. We then modify the Gaussian self-masking procedure as follows: after drawing at random the coefficients $(K_f)_{f\leq F}$ and min-max normalizing them, we define the probability of the feature value associated with feature f of being missing as

$$\mathbb{P}(M_{j,f} \mid (\boldsymbol{x}_f^j)_{j\leq n}) = 0.2K_f \exp\left(-\frac{1}{2}\frac{(\boldsymbol{x}_f^j - \widetilde{\mu}_f)^2}{\widetilde{\sigma}_f^2}\right) \ .$$

We define then the final item and user feature matrices as$S \triangleq M^i \otimes \widetilde{S}$ and $P \triangleq M^u \otimes \widetilde{P}$, where M^i and M^u are drawn from a Gaussian self-masking procedure with respective input matrices S and P, and $\otimes$ is the element-wise matrix multiplication.

C.2.2 Enforcing the sparsity in associations

Given the final item and user feature matrices as defined in the last paragraph, we set association score matrix $\widetilde{A}$ such that for each item i and user u, $\widetilde{A}_{i,u} \triangleq g_0(S_{:,i},P_{:,u})$. Then, we would like to ensure that the sparsity number –that is, the percentage of unknown user-item associations– is equal to $s \in (0.5, 1)$. If $t(s)$ and $t'(s)$ are respectively the $\frac{100(1+s)}{2}^{\text{th}}$ and $\frac{100(1-s)}{2}^{\text{th}}$

quantiles of values in $\widetilde{A}$, then we define the final association matrix $A \in \{-1,0,+1\}^{n_i \times n_u}$ as

$$\forall i \leq n_i,\ \forall u \leq n_u,\ A_{i,u} \leftarrow \begin{cases} +1 & \text{if } \widetilde{A}_{i,u} \geq t(s) \\ -1 & \text{if } \widetilde{A}_{i,u} \leq t'(s) \\ 0 & \text{otherwise} \end{cases} .$$

C.3 Training in separate embedding/RHOFM learning approaches (SELT or factorization machines)

To train the corresponding baseline models, we reimplement a training procedure with Python package PyTorch [24] equivalent to the PyKeen fit function (that is, using the margin ranking loss, the same parameters to the Adam optimizer, and a negative sampler which associates 3 negative samples to each positive sample in a batch. The negative sampler uses negative_sampling from the Python package PyTorch-Geometric [17].

In SELT-KGE –that is, learning embeddings based on a knowledge graph completion task and then feed them to the RHOFM solo training procedure– we use the same parameters as in JELI to a PyKeen training procedure to learn the embeddings.